Leitfaden

AWS CodeBuild

API-Version 2016-10-06

AWS CodeBuild: Leitfaden

Table of Contents

Was ist AWS CodeBuild?

AWS CodeBuild ist ein vollständig verwalteter Build-Service in der Cloud. CodeBuild kompiliert Ihren Quellcode, führt Einheitentests durch und erzeugt Artefakte, die bereitgestellt werden können. CodeBuild macht es überflüssig, Ihre eigenen Build-Server bereitzustellen, zu verwalten und zu skalieren. Es bietet vorgefertigte Build-Umgebungen für gängige Programmiersprachen und Build-Tools wie Apache Maven, Gradle und mehr. Sie können Build-Umgebungen auch in anpassen CodeBuild , um Ihre eigenen Build-Tools zu verwenden. CodeBuild skaliert automatisch, um Spitzenanforderungen an Builds zu erfüllen.

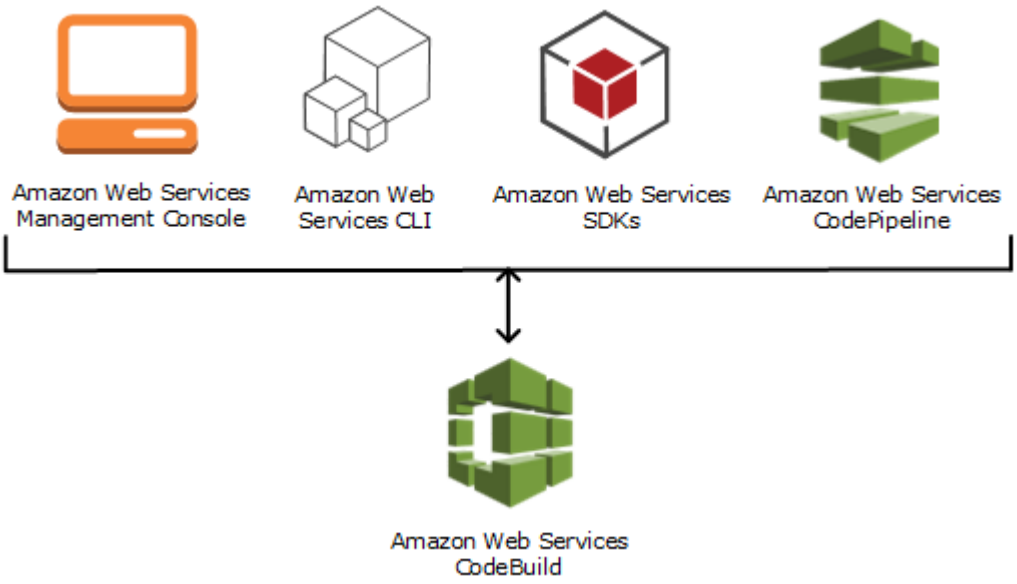
CodeBuild bietet folgende Vorteile:

- Vollständig verwaltet – CodeBuild Sie müssen keine eigenen Build-Server einrichten, patchen, aktualisieren und verwalten.
- On-Demand – CodeBuild skaliert On-Demand, um Ihre Build-Anforderungen zu erfüllen. Sie zahlen nur für die Anzahl der Build-Minuten, die Sie wirklich nutzen.
- Standardmäßig CodeBuild bietet vorkonfigurierte Build-Umgebungen für die gängigsten Programmiersprachen. Sie müssen lediglich auf Ihr Build-Skript verweisen, um den ersten Build zu starten.

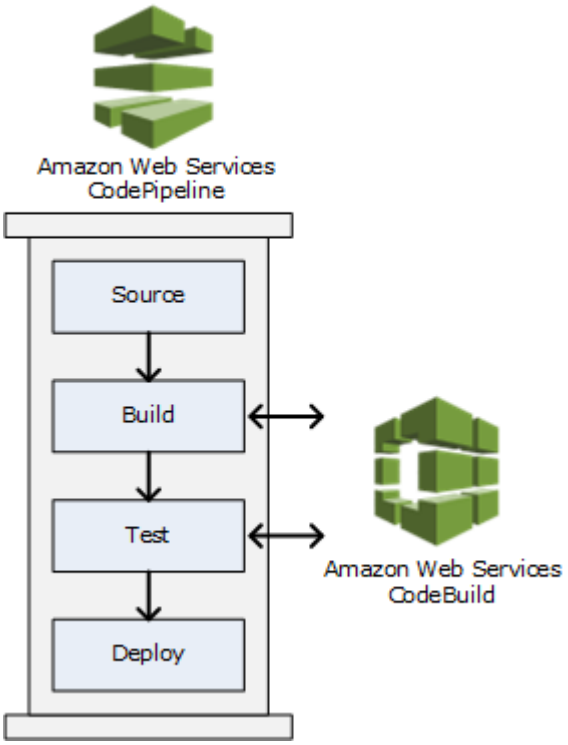
Weitere Informationen finden Sie unter AWS CodeBuild.

So führen Sie aus CodeBuild

Zum Ausführen von CodeBuild können Sie die AWS CodeBuild- oder die AWS CodePipeline-Konsole verwenden. Sie können die Ausführung von auch CodeBuild mithilfe der AWS Command Line Interface (AWS CLI) oder der AWS SDKs automatisieren. SDKs



Wie das folgende Diagramm zeigt, können Sie CodeBuild als Build- oder Testaktion zur Build- oder Testphase einer Pipeline in hinzufügenAWS CodePipeline. AWS CodePipeline ist ein kontinuierlicher Bereitstellungsservice, mit dem Sie die Schritte modellieren, visualisieren und automatisieren können, die zum Freigeben Ihres Codes erforderlich sind. Dazu gehört auch der Code-Build. Eine Pipeline ist ein Workflow, der beschreibt, wie Codeänderungen das Freigabeverfahren durchlaufen.



Informationen CodePipeline zur Verwendung von zum Erstellen einer Pipeline und zum Hinzufügen einer CodeBuild Build- oder Testaktion finden Sie unter Verwenden von CodePipeline mit CodeBuild. Weitere Informationen zu CodePipelinefinden Sie im AWS CodePipeline -Benutzerhandbuch.

Die CodeBuild Konsole bietet auch eine Möglichkeit, schnell nach Ihren Ressourcen zu suchen, z. B. Repositorys, Build-Projekte, Bereitstellungsanwendungen und Pipelines. Wählen Sie Go to Ressource (Zur Ressource) oder drücken Sie die Taste / und geben Sie dann den Namen der Ressource ein. Alle Übereinstimmungen werden in der Liste angezeigt. Bei der Suche wird nicht zwischen Groß- und Kleinschreibung unterschieden. Sie sehen nur die Ressourcen, für die Sie die Berechtigung besitzen. Weitere Informationen finden Sie unter Anzeigen von Ressourcen in der Konsole.

Preise für CodeBuild

Weitere Informationen finden Sie unter -CodeBuild Preise.

Wie beginne ich mit CodeBuild?

Wir empfehlen, dass Sie zuerst die folgenden Schritte ausführen:

1. Weitere Informationen zu finden Sie CodeBuild in den Informationen unter Konzepte.
2. Experiment ierenSie mit CodeBuild in einem Beispielszenario, indem Sie den Anweisungen unter folgenErste Schritte mit der Konsole.
3. Verwenden Sie CodeBuild in Ihren eigenen Szenarien, indem Sie den Anweisungen unter folgenPlanen eines Builds.

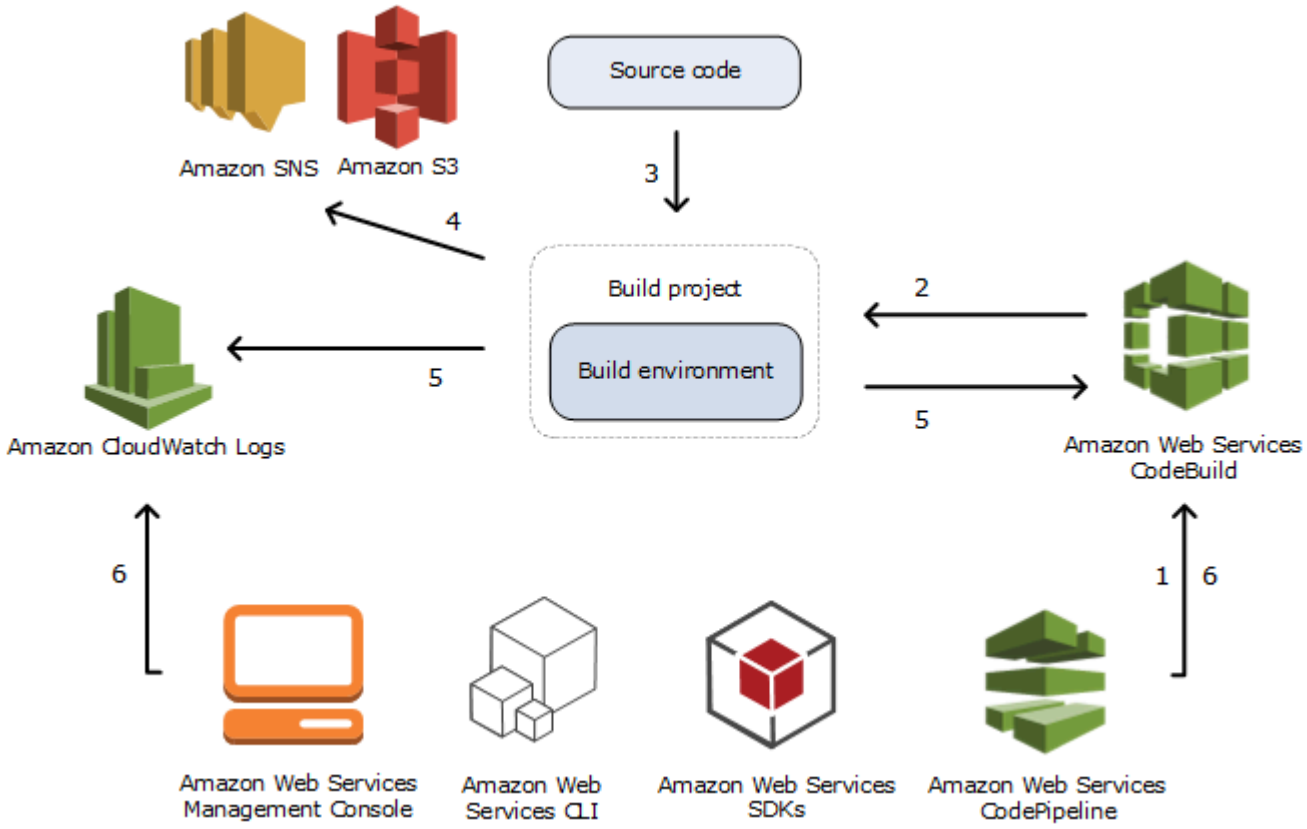
AWS CodeBuild-Konzepte

Die folgenden Konzepte sind wichtig, um zu verstehen, wie CodeBuild funktioniert.

Themen

Funktionsweise von CodeBuild

Im folgenden Diagramm sind die Vorgänge bei der Ausführung eines Builds in CodeBuild dargestellt:



1. Als Eingabe müssen Sie CodeBuild ein Build-Projekt zur Verfügung stellen. EINBuild-Projektenthält Informationen zum Ausführen eines Builds, einschließlich wo der Quellcode erhalten ist, welche Build-Umgebung verwendet werden soll, welche Build-Befehle ausgeführt werden sollen und wo die Build-Ausgabe gespeichert werden soll. EINBuild-Umgebungstellt eine Kombination von Betriebssystem-, Programmiersprachen-Laufzeit- und Tools dar, die CodeBuild zum Ausführen eines Builds verwendet. Weitere Informationen finden Sie unter:

   - Erstellen eines Build-Projekts
   - Build-Umgebungsreferenz

2. CodeBuild verwendet das Build-Projekt zum Erstellen der Build-Umgebung.
3. CodeBuild lädt den Quellcode in die Build-Umgebung und verwendet anschließend die Build-Spezifikation (buildspec), wie diese im Build-Projekt definiert wurde oder im Quellcode direkt enthalten ist. EINbuildspecist eine Sammlung von Build-Befehlen und zugehörigen Einstellungen im YAML-Format, die CodeBuild verwendet, um einen Build auszuführen. Weitere Informationen hierzu finden Sie unter Build-Spezifikationsreferenz.
4. Wenn eine Build-Ausgabe vorhanden ist, lädt die Build-Umgebung diese Ausgabe in einen S3-Bucket. Die Build-Umgebung kann auch Aufgaben ausführen, die Sie in den Build-

Spezifikationen festlegen (wie z. B. das Senden von Build-Benachrichtigungen an ein Amazon SNS SNS-Thema). Ein Beispiel finden Sie unter Build-Benachrichtigungsbeispiel.

5. Während der Ausführung des Builds sendet die Build-Umgebung Informationen an CodeBuild und Amazon CloudWatch Logs.
6. Während der Ausführung des Builds können Sie dasAWS CodeBuild-KonsoleAWS CLI, oderAWS-SDKs zum Abrufen von Build-Informationen und ausführliche Build-Informationen von Amazon CloudWatch Logs. Bei Verwendung vonAWS CodePipelineUm Builds auszuführen, erhalten Sie nur eingeschränkte Build-Informationen von CodePipeline.

Nächste Schritte

Nun, da Sie mehr über AWS CodeBuild wissen, empfehlen wir Ihnen diese nächsten Schritte:

1. Experimentmit CodeBuild in einem Beispielszenario, indem Sie den Anleitungen unter folgenErste Schritte mit der Konsoleaus.
2. Verwenden vonCodeBuild in Ihren eigenen Szenarien, indem Sie den Anleitungen unter folgenPlanen eines Buildsaus.

Erste Schritte mit CodeBuild

In den folgenden Tutorials können Sie mit AWS CodeBuild aus einer Sammlung von Beispiel-Quellcodeeingabedateien eine bereitstellbare Version des Quellcodes erstellen.

Beide Tutorials haben die gleichen Eingaben und Ergebnisse, ein Tutorial nutzt jedoch die AWS CodeBuild-Konsole und das andere die AWS CLI.

> **Important**
>
> Die Verwendung des AWS-Root-Kontos für dieses Tutorial wird nicht empfohlen.

Erste Schritte mit AWS CodeBuild unter Verwendung der Konsole

In diesem Tutorial erstellen Sie mit AWS CodeBuild aus einer Sammlung von Beispielquellcode-Eingabedateien (Build-Eingabeartefakte oder Build-Eingabe) eine bereitstellbare Version des Quellcodes (Build-Ausgabeartefakt oder Build-Ausgabe). Insbesondere weisen Sie CodeBuild an, Apache Maven, ein gängiges Build-Tool, zu verwenden, um eine Reihe von Java-Klassendateien in einer Java Archive (JAR)-Datei zu erstellen. Dieses Tutorial setzt nicht voraus, dass Sie mit Apache Maven oder Java vertraut sind.

Sie können mit CodeBuild über die CodeBuild Konsole, AWS CodePipeline, die AWS CLIoder die AWS SDKs arbeiten. SDKs In diesem Tutorial wird gezeigt, wie Sie die - CodeBuild Konsole verwenden. Für weitere Informationen zur Nutzung von CodePipeline siehe Verwenden von CodePipeline mit CodeBuild.

> **Important**
>
> Die Schritte in diesem Tutorial setzen voraus, dass Sie Ressourcen (z. B. einen S3-Bucket) erstellen, die Kosten für das AWS-Konto verursachen können. Dazu gehören mögliche Gebühren für CodeBuild und für AWS Ressourcen und Aktionen im Zusammenhang mit Amazon S3AWS KMS, und - CloudWatch Protokollen. Weitere Informationen finden Sie unter -AWS CodeBuildPreise, Amazon S3-Preise, AWS Key Management Service -Preise und Amazon- CloudWatch Preise.

Schritte

Schritt 1: Erstellen des Quellcodes

(Teil von: Erste Schritte mit AWS CodeBuild unter Verwendung der Konsole)

In diesem Schritt erstellen Sie den Quellcode, den Sie CodeBuild im Ausgabe-Bucket erstellen möchten. Dieser Quellcode besteht aus zwei Java-Klassendateien und einer Apache Maven Project Object Model (POM)-Datei.

1. Erstellen Sie in einem leeren Verzeichnis auf Ihrem lokalen Computer oder der Instance folgende Verzeichnisstruktur.

```
(root directory name)
    `-- src
         |-- main
         |     `-- java
         `-- test
               `-- java
```

2. Erstellen Sie diese Datei mithilfe eines Texteditors, benennen Sie sie `MessageUtil.java` und speichern Sie sie im Verzeichnis `src/main/java`.

```
public class MessageUtil {
  private String message;
```

```
  public MessageUtil(String message) {
    this.message = message;
  }

  public String printMessage() {
    System.out.println(message);
    return message;
  }

  public String salutationMessage() {
    message = "Hi!" + message;
    System.out.println(message);
    return message;
  }
}
```

Diese Klassendatei erstellt die Zeichenfolge, die an sie übergeben wurde, als Ausgabe. Der `MessageUtil`-Konstruktor legt die Zeichenfolge fest. Die `printMessage`-Methode erstellt die Ausgabe. Die `salutationMessage`-Methode gibt `Hi!` gefolgt von der Zeichenfolge aus.

3. Erstellen Sie diese Datei, benennen Sie sie `TestMessageUtil.java` und speichern Sie sie im Verzeichnis `/src/test/java`.

```
import org.junit.Test;
import org.junit.Ignore;
import static org.junit.Assert.assertEquals;

public class TestMessageUtil {

  String message = "Robert";
  MessageUtil messageUtil = new MessageUtil(message);

  @Test
  public void testPrintMessage() {
    System.out.println("Inside testPrintMessage()");
    assertEquals(message,messageUtil.printMessage());
  }

  @Test
  public void testSalutationMessage() {
    System.out.println("Inside testSalutationMessage()");
    message = "Hi!" + "Robert";
```

```
    assertEquals(message,messageUtil.salutationMessage());
  }
}
```

Diese Klassendatei setzt die `message`-Variable in der Klasse `MessageUtil` auf `Robert`. Anschließend führt sie einen Test durch, um zu sehen, ob die `message`-Variable erfolgreich festgelegt wurde, indem sie überprüft, ob die Zeichenfolgen `Robert` und `Hi!Robert` in der Ausgabe vorhanden sind.

4. Erstellen Sie diese Datei, benennen Sie sie `pom.xml` und speichern Sie sie im Stammverzeichnis (oberste Ebene).

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/
maven-v4_0_0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>org.example</groupId>
  <artifactId>messageUtil</artifactId>
  <version>1.0</version>
  <packaging>jar</packaging>
  <name>Message Utility Java Sample App</name>
  <dependencies>
    <dependency>
      <groupId>junit</groupId>
      <artifactId>junit</artifactId>
      <version>4.11</version>
      <scope>test</scope>
    </dependency>
  </dependencies>
  <build>
    <plugins>
      <plugin>
        <groupId>org.apache.maven.plugins</groupId>
        <artifactId>maven-compiler-plugin</artifactId>
        <version>3.8.0</version>
      </plugin>
    </plugins>
  </build>
</project>
```

Apache Maven verwendet die Anweisungen in dieser Datei, um die Dateien `MessageUtil.java` und `TestMessageUtil.java` in eine Datei namens `messageUtil-1.0.jar` zu konvertieren und dann die angegebenen Tests auszuführen.

An diesem Punkt sollte Ihre Dateistruktur wie folgt aussehen:

```
(root directory name)
    |-- pom.xml
    `-- src
         |-- main
         |     `-- java
         |           `-- MessageUtil.java
         `-- test
               `-- java
                     `-- TestMessageUtil.java
```

Nächster Schritt

Schritt 2: Erstellen der buildspec-Datei

Schritt 2: Erstellen der buildspec-Datei

(Vorheriger Schritt: Schritt 1: Erstellen des Quellcodes)

In diesem Schritt erstellen Sie eine Build-Spezifikationsdatei. Eine Build-Spezifikation ist eine Sammlung von Build-Befehlen und zugehörigen Einstellungen im YAML-Format, die CodeBuild verwendet, um einen Build auszuführen. Ohne eine Build-Spezifikation CodeBuild kann Ihre Build-Eingabe nicht erfolgreich in die Build-Ausgabe konvertieren oder das Build-Ausgabeartefakt in der Build-Umgebung finden, um es in Ihren Ausgabe-Bucket hochzuladen.

Erstellen Sie diese Datei, benennen Sie sie `buildspec.yml` und speichern Sie sie im Stammverzeichnis (oberste Ebene).

```
version: 0.2

phases:
  install:
    runtime-versions:
      java: corretto11
  pre_build:
```

```
    commands:
      - echo Nothing to do in the pre_build phase...
  build:
    commands:
      - echo Build started on `date`
      - mvn install
  post_build:
    commands:
      - echo Build completed on `date`
artifacts:
  files:
    - target/messageUtil-1.0.jar
```

Important

Da es sich bei der Build-Spezifikationsdeklaration um eine gültige YAML handelt, ist die Formatierung in einer Build-Spezifikationsdeklaration wichtig. Wenn die Anzahl der Leerzeichen in der Build-Spezifikationsdeklaration nicht mit dieser übereinstimmt, schlägt der Build ggf. sofort fehl. Verwenden Sie eine YAML-Validierung, um zu testen, ob es sich bei der Build-Spezifikationsdeklaration um eine gültige YAML handelt.

Note

Statt eine Build-Spezifikationsdatei in Ihren Quellcode einzuschließen, können Sie Build-Befehle beim Erstellen eines Build-Projekts separat deklarieren. Das ist hilfreich, wenn Sie Ihren Quellcode mit unterschiedlichen Build-Befehlen erstellen möchten, ohne jedes Mal das Quellcode-Repository zu aktualisieren. Weitere Informationen finden Sie unter Syntax der Build-Spezifikation.

In dieser Build-Spezifikationsdeklaration gilt:

- `version` steht für die Version des verwendeten Build-Spezifikationsstandards. Diese Build-Spezifikationsdeklaration verwendet die aktuelle Version, `0.2`.
- `phases` steht für die Build-Phasen, in denen Sie CodeBuild anweisen können, Befehle auszuführen. Diese Build-Phasen sind hier als `install`, `pre_build`, `build` und `post_build` aufgelistet. Sie können die Schreibweise dieser Build-Phasennamen nicht ändern und Sie können keine zusätzlichen Build-Phasennamen erstellen.

In diesem Beispiel CodeBuild führt während der -`build`Phase den `mvn install` Befehl aus. Dieser Befehl weist Apache Maven an, die Java-Klassendateien zu kompilieren, zu testen und die kompilierten Dateien in ein Build-Ausgabeartefakt zu verpacken. Der Vollständigkeit halber sind in diesem Beispiel einige `echo`-Befehle in jeder Build-Phase platziert. Wenn Sie die detaillierten Build-Informationen später in diesem Tutorial anzeigen, können Sie der Ausgabe dieser `echo`-Befehle entnehmen, wie und in welcher Reihenfolge CodeBuild Befehle ausführt. (Auch wenn alle Build-Phasen in diesem Beispiel enthalten sind, müssen Sie nicht unbedingt eine Build-Phase einschließen, wenn Sie nicht vorhaben, während dieser Phase Befehle auszuführen.) Führt für jede Build CodeBuild -Phase jeden angegebenen Befehl nacheinander in der aufgeführten Reihenfolge aus, von Anfang bis Ende.

- `artifacts` stellt den Satz von Build-Ausgabeartefakten dar, CodeBuild die in den Ausgabe-Bucket hochlädt. `files` stellt die Dateien dar, die in die Build-Ausgabe aufgenommen werden sollen. CodeBuild lädt die einzelne `messageUtil-1.0.jar` Datei hoch, die im `target` relativen Verzeichnis in der Build-Umgebung gefunden wird. Der Dateiname `messageUtil-1.0.jar` und der Verzeichnisname `target`, unter denen Apache Maven Build-Ausgabeartefakte erstellt und speichert, gelten nur für dieses Beispiel. In Ihren eigenen Builds lauten diese Dateinamen und Verzeichnisse anders.

Weitere Informationen hierzu finden Sie unter Build-Spezifikationsreferenz.

An diesem Punkt sollte Ihre Dateistruktur wie folgt aussehen:

```
(root directory name)
    |-- pom.xml
    |-- buildspec.yml
    `-- src
         |-- main
         |     `-- java
         |           `-- MessageUtil.java
         `-- test
               `-- java
                     `-- TestMessageUtil.java
```

Nächster Schritt

Schritt 3: Erstellen von zwei S3-Buckets

Schritt 3: Erstellen von zwei S3-Buckets

(Vorheriger Schritt: Schritt 2: Erstellen der buildspec-Datei)

Auch wenn Sie einen einzelnen Bucket für dieses Tutorial verwenden können, ist mit zwei Buckets einfacher zu erkennen, woher die Build-Eingabe stammt und wohin die Build-Ausgabe geschrieben wird.

- Einer dieser Buckets (Eingangs-Bucket) speichert die Build-Eingabe. In diesem Tutorial lautet der Name dieses Eingabe-Buckets `codebuild-region-ID-account-ID-input-bucket`, wobei *`region-ID`* die AWS-Region des Buckets und *`account-ID`* die AWS-Konto-ID angibt.
- Der andere Bucket (Ausgabe-Bucket) nimmt die Build-Ausgabe auf. In diesem Tutorial ist der Name dieses Ausgabe-Buckets `codebuild-region-ID-account-ID-output-bucket`.

Wenn Sie andere Namen für diese Buckets gewählt haben, müssen Sie diese Namen im gesamten Tutorial verwenden.

Die beiden Buckets müssen sich in derselben AWS-Region wie Ihre Builds befinden. Wenn Sie beispielsweise CodeBuild anweisen, einen Build in der Region USA Ost (Ohio) auszuführen, müssen sich diese Buckets auch in der Region USA Ost (Ohio) befinden.

Weitere Informationen erhalten Sie unter Erstellen eines Buckets im Benutzerhandbuch für Amazon Simple Storage Service.

> **Note**
>
> Obwohl CodeBuild auch Build-Eingaben unterstützt GitHub, die in CodeCommit, und Bitbucket-Repositorys gespeichert sind, zeigt Ihnen dieses Tutorial nicht, wie Sie sie verwenden. Weitere Informationen finden Sie unter Planen eines Builds.

Nächster Schritt

Schritt 4: Hochladen des Quellcodes und der Build-Spezifikationsdatei

Schritt 4: Hochladen des Quellcodes und der Build-Spezifikationsdatei

(Vorheriger Schritt: Schritt 3: Erstellen von zwei S3-Buckets)

In diesem Schritt fügen Sie den Quellcode und die Build-Spezifikationsdatei zum Empfangs-Bucket hinzu.

Erstellen Sie mit dem ZIP-Dienstprogramm Ihres Betriebssystems eine Datei namens `MessageUtil.zip`, die `MessageUtil.java`, `TestMessageUtil.java`, `pom.xml` und `buildspec.yml` enthält.

Die Verzeichnisstruktur der Datei `MessageUtil.zip` muss wie folgt aussehen:

```
MessageUtil.zip
    |-- pom.xml
    |-- buildspec.yml
    `-- src
         |-- main
         |     `-- java
         |           `-- MessageUtil.java
         `-- test
               `-- java
                     `-- TestMessageUtil.java
```

> **Important**
>
> Schließen Sie nicht das Verzeichnis *(root directory name)* ein, sondern nur die Verzeichnisse und Dateien, die im Verzeichnis *(root directory name)* enthalten sind.

Laden Sie die Datei `MessageUtil.zip` in den Empfangs-Bucket namens `codebuild-`*region-ID*`-`*account-ID*`-input-bucket`.

> **Important**
>
> Für - CodeCommit GitHub, - und Bitbucket-Repositorys müssen Sie standardmäßig eine Build-Spezifikationsdatei mit dem Namen `buildspec.yml` im Stammverzeichnis (der obersten Ebene) jedes Repositorys speichern oder die Deklaration der Build-Spezifikation als Teil der Build-Projektdefinition einschließen. Erstellen Sie keine ZIP-Datei, die den Quellcode und die Build-Spezifikationsdatei des Repositorys enthält.
> Bei Build-Eingaben, die nur in S3-Buckets gespeichert sind, müssen Sie eine ZIP-Datei erstellen, die den Quellcode enthält, und – gemäß Konvention – eine Build-Spezifikationsdatei namens `buildspec.yml` im Stammverzeichnis (oberste Ebene) speichern oder die Build-Spezifikationsdeklaration in die Build-Projektdefinition einschließen.

Wenn Sie einen anderen Namen für Ihre Build-Spezifikationsdatei verwenden oder auf eine Build-Spezifikation verweisen möchten, die nicht im Stammverzeichnis gespeichert ist, können Sie eine Überschreibung der Build-Spezifikation im Rahmen der Build-Projektdefinition angeben. Weitere Informationen finden Sie unter Dateiname der Build-Spezifikation und Speicherort.

Nächster Schritt

Schritt 5: Erstellen des Build-Projekts

Schritt 5: Erstellen des Build-Projekts

(Vorheriger Schritt: Schritt 4: Hochladen des Quellcodes und der Build-Spezifikationsdatei)

In diesem Schritt erstellen Sie ein Build-Projekt, das AWS CodeBuild zum Ausführen des Builds verwendet. Ein Build-Projekt enthält Informationen zum Ausführen eines Builds, einschließlich des Quellcodes, der zu verwendenden Build-Umgebung, der auszuführenden Build-Befehle und des Speicherorts der Build-Ausgabe. Eine Build-Umgebung stellt eine Kombination aus Betriebssystem, Laufzeit der Programmiersprache und Tools dar, die zum Ausführen eines Builds CodeBuild verwendet. Die Build-Umgebung wird als Docker-Image ausgedrückt. Weitere Informationen finden Sie unter Docker Overview auf der Docker Docs-Website.

Für diese Build-Umgebung weisen Sie an, ein Docker-Image CodeBuild zu verwenden, das eine Version des Java Development Kit (JDK) und Apache Maven enthält.

So erstellen Sie ein Build-Projekt

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die -AWS CodeBuildKonsole unter https://console.aws.amazon.com/codesuite/codebuild/home.
2. Verwenden Sie die AWS Regionsauswahl, um eine -AWSRegion auszuwählen, in der unterstützt CodeBuild wird. Weitere Informationen finden Sie unter AWS CodeBuild-Endpunkte und -Kontingente in der Allgemeine Amazon Web Services-Referenz.
3. Wenn eine CodeBuild Informationsseite angezeigt wird, wählen Sie Build-Projekt erstellen aus. Erweitern Sie andernfalls im Navigationsbereich Build , wählen Sie Build-Projekte und dann Build-Projekt erstellen aus.
4. Geben Sie auf der Seite Create build project (Build-Projekt erstellen) unter Project configuration (Projektkonfiguration) für Project name (Projektname) einen Namen für dieses Build-Projekt

ein (in diesem Beispiel `codebuild-demo-project`). Build-Projektnamen müssen in allen AWS-Konten eindeutig sein. Wenn Sie einen anderen Namen vergeben, müssen Sie diesen im gesamten Tutorial verwenden.

> **Note**
>
> Auf der Seite Create build project (Build-Projekt erstellen) wird möglicherweise eine Fehlermeldung ähnlich der folgenden angezeigt: You are not authorized to perform this operation (Sie sind nicht berechtigt, diesen Vorgang auszuführen).. Dies liegt höchstwahrscheinlich daran, dass Sie sich bei der AWS Management Console als -Benutzer angemeldet haben, der keine Berechtigungen zum Erstellen eines Build-Projekts hat. Um dies zu beheben, melden Sie sich von der ab AWS Management Consoleund dann wieder mit Anmeldeinformationen einer der folgenden IAM-Entitäten:
>
> - Ein Administratorbenutzer in Ihrem AWS Konto. Weitere Informationen finden Sie unter Erstellen Ihres ersten AWS-Konto Stammbenutzers und Ihrer ersten Stammgruppe im -Benutzerhandbuch.
> - Ein -Benutzer in Ihrem AWS Konto mit den `IAMFullAccess` verwalteten Richtlinien `AWSCodeBuildAdminAccess`, und `AmazonS3ReadOnlyAccess`, die diesem Benutzer oder einer IAM-Gruppe zugeordnet sind, zu der der Benutzer gehört. Wenn Sie in Ihrem AWS Konto keinen -Benutzer oder keine -Gruppe mit diesen Berechtigungen haben und diese Berechtigungen Ihrem Benutzer oder Ihrer Gruppe nicht hinzufügen können, wenden Sie sich an Ihren AWS Kontoadministrator, um Unterstützung zu erhalten. Weitere Informationen finden Sie unter AWS verwaltete (vordefinierte) Richtlinien für AWS CodeBuild.
>
> Beide Optionen beinhalten Administratorrechte, die es Ihnen ermöglichen, ein Build-Projekt zu erstellen, damit Sie dieses Tutorial abschließen können. Es wird empfohlen, immer die Mindestberechtigungen zu verwenden, die für die Ausführung Ihrer Aufgabe erforderlich sind. Weitere Informationen finden Sie unter AWS CodeBuild Referenz zu Berechtigungen.

5. Wählen Sie in Quelle für Quellanbieter Amazon S3 aus.
6. Wählen Sie für Bucket die Option codebuild-***region-ID*** -***account-ID*** -input-bucket aus.
7. Geben Sie für S3 object key (S3-Objektschlüssel) **`MessageUtil.zip`** ein.
8. Wählen Sie unter Environment für Environment image weiterhin Managed Image aus.

9. Wählen Sie für Operating system (Betriebssystem) die Option Amazon Linux 2 aus.
10. Wählen Sie unter Runtime (Laufzeit) die Option Standard aus.
11. Wählen Sie für Image aws/codebuild/amazonlinux2-x86_64-standard :4.0 aus.
12. Wählen Sie unter Service role weiterhin New service role aus und lassen Sie Role name unverändert.
13. Belassen Sie für Buildspec die Option Use a buildspec file (Eine buildspec-Datei verwenden) aktiviert.
14. Wählen Sie unter Artefakte für Typ Amazon S3 aus.
15. Wählen Sie für Bucket-Name die Option codebuild-***region-ID*** -***account-ID*** -output-bucket aus.
16. Lassen Sie Name und Path (Pfad) leer.
17. Wählen Sie Create build project (Build-Projekt erstellen) aus.

Nächster Schritt

Schritt 6: Ausführen des Builds

Schritt 6: Ausführen des Builds

(Vorheriger Schritt: Schritt 5: Erstellen des Build-Projekts)

In diesem Schritt weisen Sie AWS CodeBuild an, den Build mit den Einstellungen des Build-Projekts auszuführen.

So führen Sie den Build aus

1. Öffnen Sie die -AWS CodeBuildKonsole unter https://console.aws.amazon.com/codesuite/codebuild/home.
2. Wählen Sie im linken Navigationsbereich Build projects aus.
3. Wählen Sie in der Liste der Build-Projekte codebuild-demo-projectund dann Build starten aus. Der Build beginnt sofort.

Nächster Schritt

Schritt 7: Anzeigen der Zusammenfassung der Build-Informationen

Schritt 7: Anzeigen der Zusammenfassung der Build-Informationen

(Vorheriger Schritt: Schritt 6: Ausführen des Builds)

In diesem Schritt zeigen Sie eine Zusammenfassung der Informationen über den Build-Status an.

So zeigen Sie eine Zusammenfassung der Build-Informationen an

1. Wenn die Seite codebuild-demo-project:***<build-ID>*** nicht angezeigt wird, wählen Sie in der Navigationsleiste Build history aus. Wählen Sie als Nächstes in der Liste der Build-Projekte für Projekt den Link Build-Ausführung für auscodebuild-demo-project. Es sollte nur ein passender Link vorhanden sein. (Wenn Sie dieses Tutorial bereits zuvor durchgearbeitet haben, wählen Sie in der Spalte Completed (Fertiggestellt) den Link mit dem neuesten Wert aus.)
2. Auf der Seite Build-Status sollten unter Phasendetails die folgenden Build-Phasen angezeigt werden, wobei in der Spalte Status Erfolgreich ist:

   - SUBMITTED
   - IN WARTESCHLANGE
   - PROVISIONING
   - DOWNLOAD_SOURCE
   - INSTALL
   - PRE_BUILD
   - BUILD
   - POST_BUILD
   - UPLOAD_ARTIFACTS
   - FINALIZING
   - COMPLETED

   Unter Build-Status sollte Succeeded angezeigt werden.

   Wenn stattdessen In Progress (Verarbeitung läuft...) angezeigt wird, wählen Sie die Schaltfläche zum Aktualisieren aus.
3. Neben jeder Build-Phase gibt der Wert Duration (Dauer) an, wie lange diese Build-Phase gedauert hat. Der Wert End time gibt an, wann die Build-Phase beendet wurde.

Nächster Schritt

Schritt 8: Anzeigen der detaillierten Build-Informationen

Schritt 8: Anzeigen der detaillierten Build-Informationen

(Vorheriger Schritt: Schritt 7: Anzeigen der Zusammenfassung der Build-Informationen)

In diesem Schritt zeigen Sie detaillierte Informationen zu Ihrem Build in - CloudWatch Protokollen an.

> **Note**
>
> Um vertrauliche Informationen zu schützen, werden die folgenden in CodeBuild Protokollen ausgeblendet:
>
> - AWS-Zugriffsschlüssel-IDs. Weitere Informationen finden Sie unter Verwalten von Zugriffsschlüsseln für IAM-Benutzer im AWS Identity and Access Management - Benutzerhandbuch.
> - Mit dem Parameter Store angegebene Zeichenfolgen. Weitere Informationen finden Sie unter Systems Manager Parameter Store und Systems Manager Parameter Store Console Walkthrough im Amazon EC2 Systems Manager-Benutzerhandbuch.
> - Zeichenfolgen, die mit angegeben wurdenAWS Secrets Manager. Weitere Informationen finden Sie unter Schlüsselverwaltung.

So zeigen Sie detaillierte Build-Informationen an

1. Während die Seite mit den Build-Details immer noch vom vorherigen Schritt angezeigt wird, werden die letzten 10,000 Zeilen des Build-Protokolls unter Build logs angezeigt. Um das gesamte Build-Protokoll in CloudWatch Logs anzuzeigen, wählen Sie den Link Gesamtes Protokoll anzeigen.
2. Im CloudWatch Protokollstream Logs können Sie die Protokollereignisse durchsuchen. Standardmäßig werden nur die letzten Protokollereignisse angezeigt. Um ältere Protokollereignisse anzuzeigen, blättern Sie zum Anfang der Liste.
3. In diesem Tutorial enthalten die meisten Protokollereignisse wahrscheinlich nicht benötigte detaillierte Informationen über das Herunterladen von Build-Abhängigkeitsdateien durch CodeBuild in die Build-Umgebung und deren Installation. Sie können die angezeigten Informationen über das Feld Filter events reduzieren. Wenn Sie beispielsweise `"[INFO]"`

in das Feld Filter events (Ereignisse filtern) eingeben, werden nur Ereignisse angezeigt, die `[INFO]` enthalten. Weitere Informationen finden Sie unter Filter- und Mustersyntax im Amazon-CloudWatch Benutzerhandbuch.

Nächster Schritt

Schritt 9: Abrufen des Build-Ausgabeartefakts

Schritt 9: Abrufen des Build-Ausgabeartefakts

(Vorheriger Schritt: Schritt 8: Anzeigen der detaillierten Build-Informationen)

In diesem Schritt erhalten Sie die `messageUtil-1.0.jar` Datei, die CodeBuild erstellt und in den Ausgabe-Bucket hochgeladen hat.

Sie können die - CodeBuild Konsole oder die Amazon S3-Konsole verwenden, um diesen Schritt abzuschließen.

So rufen Sie das Build-Ausgabeartefakt ab (AWS CodeBuild-Konsole)

1. Wenn die CodeBuild Konsole noch geöffnet ist und die Seite mit den Build-Details aus dem vorherigen Schritt noch angezeigt wird, wählen Sie die Registerkarte Build-Details und scrollen Sie nach unten zum Abschnitt Artefakte.

   > **Note**
   >
   > Wenn die Build-Detailseite nicht angezeigt wird, wählen Sie in der Navigationsleiste Build-Verlauf und dann den Link Build-Ausführung aus.

2. Der Link zum Amazon S3-Ordner befindet sich unter dem Speicherort des Artifacts-Uploads . Dieser Link öffnet den Ordner in Amazon S3, in dem Sie die `messageUtil-1.0.jar` Build-Ausgabeartefaktdatei finden.

So rufen Sie das Build-Ausgabeartefakt ab (Amazon S3-Konsole)

1. Öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Öffnen Sie `codebuild-`*`region-ID`*`-`*`account-ID`*`-output-bucket`.
3. Öffnen Sie das Verzeichnis `codebuild-demo-project`.

4. Öffnen Sie den Ordner `target`, in dem Sie die Build-Ausgabeartefaktdatei `messageUtil-1.0.jar` finden.

Nächster Schritt

Schritt 10: Löschen der S3-Buckets

Schritt 10: Löschen der S3-Buckets

(Vorheriger Schritt: Schritt 9: Abrufen des Build-Ausgabeartefakts)

Um laufende Gebühren für Ihr AWS Konto zu vermeiden, können Sie die in diesem Tutorial verwendeten Eingabe- und Ausgabe-Buckets löschen. Anweisungen finden Sie unter Löschen oder Leeren eines Buckets im Benutzerhandbuch für Amazon Simple Storage Service.

Wenn Sie den IAM-Benutzer oder einen Administrator-IAM-Benutzer zum Löschen dieser Buckets verwenden, muss der Benutzer über weitere Zugriffsberechtigungen verfügen. Fügen Sie die folgende Anweisung zwischen den Markern (*### BEGIN ADDING STATEMENT HERE ###* und *### END ADDING STATEMENTS HERE ###*) zu einer vorhandenen Zugriffsrichtlinie für den Benutzer hinzu.

Die Ellipsen (...) in dieser Anweisung dienen der Übersichtlichkeit der Darstellung. Entfernen Sie keine Anweisungen aus der vorhandenen Zugriffsrichtlinie. Geben Sie keine Auslassungspunkte in die Richtlinie ein.

```
{
  "Version": "2012-10-17",
  "Id": "...",
  "Statement": [
    ### BEGIN ADDING STATEMENT HERE ###
    {
      "Effect": "Allow",
      "Action": [
        "s3:DeleteBucket",
        "s3:DeleteObject"
      ],
      "Resource": "*"
    }
    ### END ADDING STATEMENT HERE ###
  ]
```

```
}
```

Nächster Schritt

Nachbearbeitung

Nachbearbeitung

In diesem Tutorial haben Sie AWS CodeBuild verwendet, um einen Satz Java-Klassendateien als Build in einer JAR-Datei zu erstellen. Anschließend haben Sie die Build-Ergebnisse angezeigt.

Sie können jetzt versuchen, CodeBuild in Ihren eigenen Szenarien zu verwenden. Folgen Sie den Anweisungen in Planen eines Builds. Wenn Sie dazu noch nicht bereit sind, können Sie einige der Beispiele als Build erstellen. Weitere Informationen finden Sie unter Beispiele.

Erste Schritte mit AWS CodeBuild unter Verwendung der AWS CLI

In diesem Tutorial erstellen Sie mit AWS CodeBuild aus einer Sammlung von Beispielquellcode-Eingabedateien (Build-Eingabeartefakte oder Build-Eingabe) eine bereitstellbare Version des Quellcodes (Build-Ausgabeartefakt oder Build-Ausgabe). Insbesondere weisen Sie CodeBuild an, Apache Maven, ein gängiges Build-Tool, zu verwenden, um eine Reihe von Java-Klassendateien in einer Java Archive (JAR)-Datei zu erstellen. Dieses Tutorial setzt nicht voraus, dass Sie mit Apache Maven oder Java vertraut sind.

Sie können mit CodeBuild über die CodeBuild Konsole, AWS CodePipeline, die AWS CLIoder die AWS SDKs arbeiten. SDKs In diesem Tutorial wird gezeigt, wie Sie CodeBuild mit dem verwendenAWS CLI. Informationen zur Verwendung von finden Sie CodePipelineunter Verwenden von CodePipeline mit CodeBuild.

> **Important**
>
> Die Schritte in diesem Tutorial setzen voraus, dass Sie Ressourcen (z. B. einen S3-Bucket) erstellen, die Kosten für das AWS-Konto verursachen können. Dazu gehören mögliche Gebühren für CodeBuild und für AWS Ressourcen und Aktionen im Zusammenhang mit Amazon S3, AWS KMSund - CloudWatch Protokollen. Weitere Informationen finden Sie unter CodeBuild -Preise, Amazon S3-Preise, AWS Key Management Service -Preise und Amazon-CloudWatch Preise.

Schritte

Schritt 1: Erstellen des Quellcodes

(Teil von: Erste Schritte mit AWS CodeBuild unter Verwendung der AWS CLI)

In diesem Schritt erstellen Sie den Quellcode, den Sie CodeBuild im Ausgabe-Bucket erstellen möchten. Dieser Quellcode besteht aus zwei Java-Klassendateien und einer Apache Maven Project Object Model (POM)-Datei.

1. Erstellen Sie in einem leeren Verzeichnis auf Ihrem lokalen Computer oder der Instance folgende Verzeichnisstruktur.

```
(root directory name)
    `-- src
         |-- main
         |     `-- java
         `-- test
               `-- java
```

2. Erstellen Sie diese Datei mithilfe eines Texteditors, benennen Sie sie `MessageUtil.java` und speichern Sie sie im Verzeichnis `src/main/java`.

```
public class MessageUtil {
  private String message;
```

```
  public MessageUtil(String message) {
    this.message = message;
  }

  public String printMessage() {
    System.out.println(message);
    return message;
  }

  public String salutationMessage() {
    message = "Hi!" + message;
    System.out.println(message);
    return message;
  }
}
```

Diese Klassendatei erstellt die Zeichenfolge, die an sie übergeben wurde, als Ausgabe. Der `MessageUtil`-Konstruktor legt die Zeichenfolge fest. Die `printMessage`-Methode erstellt die Ausgabe. Die `salutationMessage`-Methode gibt `Hi!` gefolgt von der Zeichenfolge aus.

3. Erstellen Sie diese Datei, benennen Sie sie `TestMessageUtil.java` und speichern Sie sie im Verzeichnis `/src/test/java`.

```
import org.junit.Test;
import org.junit.Ignore;
import static org.junit.Assert.assertEquals;

public class TestMessageUtil {

  String message = "Robert";
  MessageUtil messageUtil = new MessageUtil(message);

  @Test
  public void testPrintMessage() {
    System.out.println("Inside testPrintMessage()");
    assertEquals(message,messageUtil.printMessage());
  }

  @Test
  public void testSalutationMessage() {
    System.out.println("Inside testSalutationMessage()");
    message = "Hi!" + "Robert";
```

```
    assertEquals(message,messageUtil.salutationMessage());
  }
}
```

Diese Klassendatei setzt die `message`-Variable in der Klasse `MessageUtil` auf `Robert`. Anschließend führt sie einen Test durch, um zu sehen, ob die `message`-Variable erfolgreich festgelegt wurde, indem sie überprüft, ob die Zeichenfolgen `Robert` und `Hi!Robert` in der Ausgabe vorhanden sind.

4. Erstellen Sie diese Datei, benennen Sie sie `pom.xml` und speichern Sie sie im Stammverzeichnis (oberste Ebene).

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/
maven-v4_0_0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>org.example</groupId>
  <artifactId>messageUtil</artifactId>
  <version>1.0</version>
  <packaging>jar</packaging>
  <name>Message Utility Java Sample App</name>
  <dependencies>
    <dependency>
      <groupId>junit</groupId>
      <artifactId>junit</artifactId>
      <version>4.11</version>
      <scope>test</scope>
    </dependency>
  </dependencies>
  <build>
    <plugins>
      <plugin>
        <groupId>org.apache.maven.plugins</groupId>
        <artifactId>maven-compiler-plugin</artifactId>
        <version>3.8.0</version>
      </plugin>
    </plugins>
  </build>
</project>
```

Apache Maven verwendet die Anweisungen in dieser Datei, um die Dateien `MessageUtil.java` und `TestMessageUtil.java` in eine Datei namens `messageUtil-1.0.jar` zu konvertieren und dann die angegebenen Tests auszuführen.

An diesem Punkt sollte Ihre Dateistruktur wie folgt aussehen:

```
(root directory name)
    |-- pom.xml
    `-- src
         |-- main
         |     `-- java
         |           `-- MessageUtil.java
         `-- test
               `-- java
                     `-- TestMessageUtil.java
```

Nächster Schritt

Schritt 2: Erstellen der buildspec-Datei

Schritt 2: Erstellen der buildspec-Datei

(Vorheriger Schritt: Schritt 1: Erstellen des Quellcodes)

In diesem Schritt erstellen Sie eine Build-Spezifikationsdatei. Eine Build-Spezifikation ist eine Sammlung von Build-Befehlen und zugehörigen Einstellungen im YAML-Format, die CodeBuild verwendet, um einen Build auszuführen. Ohne eine Build-Spezifikation CodeBuild kann Ihre Build-Eingabe nicht erfolgreich in die Build-Ausgabe konvertieren oder das Build-Ausgabeartefakt in der Build-Umgebung finden, das in Ihren Ausgabe-Bucket hochgeladen werden soll.

Erstellen Sie diese Datei, benennen Sie sie `buildspec.yml` und speichern Sie sie im Stammverzeichnis (oberste Ebene).

```
version: 0.2

phases:
  install:
    runtime-versions:
      java: corretto11
  pre_build:
```

```
    commands:
      - echo Nothing to do in the pre_build phase...
  build:
    commands:
      - echo Build started on `date`
      - mvn install
  post_build:
    commands:
      - echo Build completed on `date`
artifacts:
  files:
    - target/messageUtil-1.0.jar
```

Important

Da es sich bei der Build-Spezifikationsdeklaration um eine gültige YAML handelt, ist die Formatierung in einer Build-Spezifikationsdeklaration wichtig. Wenn die Anzahl der Leerzeichen in der Build-Spezifikationsdeklaration nicht mit dieser übereinstimmt, schlägt der Build ggf. sofort fehl. Verwenden Sie eine YAML-Validierung, um zu testen, ob es sich bei der Build-Spezifikationsdeklaration um eine gültige YAML handelt.

Note

Statt eine Build-Spezifikationsdatei in Ihren Quellcode einzuschließen, können Sie Build-Befehle beim Erstellen eines Build-Projekts separat deklarieren. Das ist hilfreich, wenn Sie Ihren Quellcode mit unterschiedlichen Build-Befehlen erstellen möchten, ohne jedes Mal das Quellcode-Repository zu aktualisieren. Weitere Informationen finden Sie unter Syntax der Build-Spezifikation.

In dieser Build-Spezifikationsdeklaration gilt:

- `version` steht für die Version des verwendeten Build-Spezifikationsstandards. Diese Build-Spezifikationsdeklaration verwendet die aktuelle Version, `0.2`.
- `phases` steht für die Build-Phasen, in denen Sie CodeBuild anweisen können, Befehle auszuführen. Diese Build-Phasen sind hier als `install`, `pre_build`, `build` und `post_build` aufgelistet. Sie können die Schreibweise dieser Build-Phasennamen nicht ändern und Sie können keine zusätzlichen Build-Phasennamen erstellen.

In diesem Beispiel CodeBuild führt während der -`build`Phase den `mvn install` Befehl aus. Dieser Befehl weist Apache Maven an, die Java-Klassendateien zu kompilieren, zu testen und die kompilierten Dateien in ein Build-Ausgabeartefakt zu verpacken. Der Vollständigkeit halber sind in diesem Beispiel einige `echo`-Befehle in jeder Build-Phase platziert. Wenn Sie die detaillierten Build-Informationen später in diesem Tutorial anzeigen, können Sie der Ausgabe dieser `echo`-Befehle entnehmen, wie und in welcher Reihenfolge CodeBuild Befehle ausführt. (Auch wenn alle Build-Phasen in diesem Beispiel enthalten sind, müssen Sie nicht unbedingt eine Build-Phase einschließen, wenn Sie nicht vorhaben, während dieser Phase Befehle auszuführen.) Führt für jede Build CodeBuild -Phase jeden angegebenen Befehl nacheinander in der aufgeführten Reihenfolge aus, von Anfang bis Ende.

- `artifacts` stellt den Satz von Build-Ausgabeartefakten dar, CodeBuild die in den Ausgabe-Bucket hochlädt. `files` stellt die Dateien dar, die in die Build-Ausgabe aufgenommen werden sollen. CodeBuild lädt die einzelne `messageUtil-1.0.jar` Datei hoch, die im `target` relativen Verzeichnis in der Build-Umgebung gefunden wird. Der Dateiname `messageUtil-1.0.jar` und der Verzeichnisname `target`, unter denen Apache Maven Build-Ausgabeartefakte erstellt und speichert, gelten nur für dieses Beispiel. In Ihren eigenen Builds lauten diese Dateinamen und Verzeichnisse anders.

Weitere Informationen hierzu finden Sie unter Build-Spezifikationsreferenz.

An diesem Punkt sollte Ihre Dateistruktur wie folgt aussehen:

```
(root directory name)
    |-- pom.xml
    |-- buildspec.yml
    `-- src
         |-- main
         |     `-- java
         |           `-- MessageUtil.java
         `-- test
               `-- java
                     `-- TestMessageUtil.java
```

Nächster Schritt

Schritt 3: Erstellen von zwei S3-Buckets

Schritt 3: Erstellen von zwei S3-Buckets

(Vorheriger Schritt: Schritt 2: Erstellen der buildspec-Datei)

Auch wenn Sie einen einzelnen Bucket für dieses Tutorial verwenden können, ist mit zwei Buckets einfacher zu erkennen, woher die Build-Eingabe stammt und wohin die Build-Ausgabe geschrieben wird.

- Einer dieser Buckets (Eingangs-Bucket) speichert die Build-Eingabe. In diesem Tutorial lautet der Name dieses Eingabe-Buckets `codebuild-region-ID-account-ID-input-bucket`, wobei *`region-ID`* die AWS-Region des Buckets und *`account-ID`* die AWS-Konto-ID angibt.
- Der andere Bucket (Ausgabe-Bucket) nimmt die Build-Ausgabe auf. In diesem Tutorial ist der Name dieses Ausgabe-Buckets `codebuild-region-ID-account-ID-output-bucket`.

Wenn Sie andere Namen für diese Buckets gewählt haben, müssen Sie diese Namen im gesamten Tutorial verwenden.

Die beiden Buckets müssen sich in derselben AWS-Region wie Ihre Builds befinden. Wenn Sie beispielsweise CodeBuild anweisen, einen Build in der Region USA Ost (Ohio) auszuführen, müssen sich diese Buckets auch in der Region USA Ost (Ohio) befinden.

Weitere Informationen erhalten Sie unter Erstellen eines Buckets im Benutzerhandbuch für Amazon Simple Storage Service.

> **Note**
>
> Obwohl CodeBuild auch Build-Eingaben unterstützt GitHub, die in CodeCommit, und Bitbucket-Repositorys gespeichert sind, zeigt Ihnen dieses Tutorial nicht, wie Sie sie verwenden. Weitere Informationen finden Sie unter Planen eines Builds.

Nächster Schritt

Schritt 4: Hochladen des Quellcodes und der Build-Spezifikationsdatei

Schritt 4: Hochladen des Quellcodes und der Build-Spezifikationsdatei

(Vorheriger Schritt: Schritt 3: Erstellen von zwei S3-Buckets)

In diesem Schritt fügen Sie den Quellcode und die Build-Spezifikationsdatei zum Empfangs-Bucket hinzu.

Erstellen Sie mit dem ZIP-Dienstprogramm Ihres Betriebssystems eine Datei namens `MessageUtil.zip`, die `MessageUtil.java`, `TestMessageUtil.java`, `pom.xml` und `buildspec.yml` enthält.

Die Verzeichnisstruktur der Datei `MessageUtil.zip` muss wie folgt aussehen:

```
MessageUtil.zip
    |-- pom.xml
    |-- buildspec.yml
    `-- src
         |-- main
         |     `-- java
         |           `-- MessageUtil.java
         `-- test
               `-- java
                     `-- TestMessageUtil.java
```

> **Important**
>
> Schließen Sie nicht das Verzeichnis *(root directory name)* ein, sondern nur die Verzeichnisse und Dateien, die im Verzeichnis *(root directory name)* enthalten sind.

Laden Sie die Datei `MessageUtil.zip` in den Empfangs-Bucket namens `codebuild-`*region-ID*`-`*account-ID*`-input-bucket`.

> **Important**
>
> Für - CodeCommit GitHub, - und Bitbucket-Repositorys müssen Sie standardmäßig eine Build-Spezifikationsdatei mit dem Namen `buildspec.yml` im Stammverzeichnis (der obersten Ebene) jedes Repositorys speichern oder die Deklaration der Build-Spezifikation als Teil der Build-Projektdefinition einschließen. Erstellen Sie keine ZIP-Datei, die den Quellcode und die Build-Spezifikationsdatei des Repositorys enthält.
> Bei Build-Eingaben, die nur in S3-Buckets gespeichert sind, müssen Sie eine ZIP-Datei erstellen, die den Quellcode enthält, und – gemäß Konvention – eine Build-Spezifikationsdatei namens `buildspec.yml` im Stammverzeichnis (oberste Ebene) speichern oder die Build-Spezifikationsdeklaration in die Build-Projektdefinition einschließen.

Wenn Sie einen anderen Namen für Ihre Build-Spezifikationsdatei verwenden oder auf eine Build-Spezifikation verweisen möchten, die nicht im Stammverzeichnis gespeichert ist, können Sie eine Überschreibung der Build-Spezifikation im Rahmen der Build-Projektdefinition angeben. Weitere Informationen finden Sie unter Dateiname der Build-Spezifikation und Speicherort.

Nächster Schritt

Schritt 5: Erstellen des Build-Projekts

Schritt 5: Erstellen des Build-Projekts

(Vorheriger Schritt: Schritt 4: Hochladen des Quellcodes und der Build-Spezifikationsdatei)

In diesem Schritt erstellen Sie ein Build-Projekt, das AWS CodeBuild zum Ausführen des Builds verwendet. Ein Build-Projekt enthält Informationen zum Ausführen eines Builds, einschließlich des Quellcodes, der zu verwendenden Build-Umgebung, der auszuführenden Build-Befehle und des Speicherorts der Build-Ausgabe. Eine Build-Umgebung stellt eine Kombination aus Betriebssystem, Laufzeit der Programmiersprache und Tools dar, die zum Ausführen eines Builds CodeBuild verwendet. Die Build-Umgebung wird als Docker-Image ausgedrückt. Weitere Informationen finden Sie unter Docker Overview auf der Docker Docs-Website.

Für diese Build-Umgebung weisen Sie an, ein Docker-Image CodeBuild zu verwenden, das eine Version des Java Development Kit (JDK) und Apache Maven enthält.

So erstellen Sie ein Build-Projekt

1. Verwenden Sie die AWS CLI zum Ausführen des create-project-Befehls:

   ```
   aws codebuild create-project --generate-cli-skeleton
   ```

   Daten im JSON-Format werden in der Ausgabe angezeigt. Kopieren Sie die Daten in eine Datei namens `create-project.json` auf dem lokalen Computer oder einer lokalen Instance, auf der die AWS CLI installiert ist. Wenn Sie einen anderen Dateinamen verwenden, muss dieser Name im gesamten Tutorial verwendet werden.

   Ändern Sie die kopierten Daten entsprechend dem nachfolgenden Format und speichern Sie die Ergebnisse:

```
{
  "name": "codebuild-demo-project",
  "source": {
    "type": "S3",
    "location": "codebuild-region-ID-account-ID-input-bucket/MessageUtil.zip"
  },
  "artifacts": {
    "type": "S3",
    "location": "codebuild-region-ID-account-ID-output-bucket"
  },
  "environment": {
    "type": "LINUX_CONTAINER",
    "image": "aws/codebuild/standard:5.0",
    "computeType": "BUILD_GENERAL1_SMALL"
  },
  "serviceRole": "serviceIAMRole"
}
```

Ersetzen Sie *serviceIAMRole* durch den Amazon-Ressourcennamen (ARN) einer CodeBuild Servicerolle (z. B. `arn:aws:iam::`*account-ID*`:role/`*role-name*). Informationen zum Erstellen finden Sie unter Erstellen Sie eine CodeBuild Servicerolle.

In diesen Daten:

- `name` stellt eine erforderliche ID für dieses Build-Projekt dar (in diesem Beispiel `codebuild-demo-project`). Build-Projektnamen müssen in allen Build-Projekten in Ihrem Konto eindeutig sein.
- Für `type` ist ein erforderlicher Wert`source`, der den Repository-Typ des Quellcodes darstellt (in diesem Beispiel `S3` für einen Amazon S3-Bucket).
- Für `source` stellt `location` den Pfad zum Quellcode dar (in diesem Beispiel der Name des Empfangs-Buckets gefolgt vom ZIP-Dateinamen).
- Für `type` ist ein erforderlicher Wert`artifacts`, der den Repository-Typ des Build-Ausgabeartefakts darstellt (in diesem Beispiel `S3` für einen Amazon S3-Bucket).
- Für `artifacts` stellt `location` den Namen des Ausgabe-Buckets dar, den Sie zuvor erstellt oder definiert haben (in diesem Beispiel `codebuild-`*region-ID*`-`*account-ID*`-output-bucket`).
- Für `type` ist ein erforderlicher Wert`environment`, der den Typ der Build-Umgebung darstellt (in diesem Beispiel `LINUX_CONTAINER`).

- Für `image` ist ein erforderlicher Wert`environment`, der die Kombination aus Docker-Image-Name und Tag darstellt, die dieses Build-Projekt verwendet, wie durch den Docker-Image-Repository-Typ angegeben (in diesem Beispiel `aws/codebuild/standard:5.0` für ein Docker-Image im CodeBuild Docker-Image-Repository). `aws/codebuild/standard` ist der Name des Docker-Images. `5.0` ist das Tag des Docker-Images.

  Weitere Docker-Images, die Sie in Ihren Szenarien verwenden können, finden Sie unter Build-Umgebungsreferenz.
- Für `computeType` ist ein erforderlicher Wert`environment`, der die CodeBuild von verwendeten Rechenressourcen darstellt (in diesem Beispiel `BUILD_GENERAL1_SMALL`).

> **Note**
>
> Andere verfügbare Werte in den ursprünglichen JSON-formatierten Daten wie `description`, `buildspec`, `auth` (einschließlich `type` und `resource`), `path`, `namespaceType`, `name` (für `artifacts`), `packaging`, `environmentVariables` (einschließlich `name` und `value`), `timeoutInMinutes`, `encryptionKey` und `tags` (einschließlich `key` und `value`) sind optional. Sie werden in diesem Tutorial nicht verwendet und deshalb hier nicht aufgelistet. Weitere Informationen finden Sie unter Erstellen eines Build-Projekts (AWS CLI).

2. Wechseln Sie in das Verzeichnis, das die soeben gespeicherte Datei enthält, und führen Sie den Befehl create-project erneut aus.

```
aws codebuild create-project --cli-input-json file://create-project.json
```

Ist der Befehl erfolgreich, gibt er als Ausgabe Daten zurück, die wie folgt aussehen sollten.

```
{
  "project": {
    "name": "codebuild-demo-project",
    "serviceRole": "serviceIAMRole",
    "tags": [],
    "artifacts": {
      "packaging": "NONE",
      "type": "S3",
      "location": "codebuild-region-ID-account-ID-output-bucket",
      "name": "message-util.zip"
```

```
    },
    "lastModified": 1472661575.244,
    "timeoutInMinutes": 60,
    "created": 1472661575.244,
    "environment": {
      "computeType": "BUILD_GENERAL1_SMALL",
      "image": "aws/codebuild/standard:5.0",
      "type": "LINUX_CONTAINER",
      "environmentVariables": []
    },
    "source": {
      "type": "S3",
      "location": "codebuild-region-ID-account-ID-input-bucket/MessageUtil.zip"
    },
    "encryptionKey": "arn:aws:kms:region-ID:account-ID:alias/aws/s3",
    "arn": "arn:aws:codebuild:region-ID:account-ID:project/codebuild-demo-project"
  }
}
```

- `project` enthält Informationen zu diesem Build-Projekt.
  - `tags` stellt alle deklarierten Tags dar.
  - `packaging` gibt an, wie die Build-Ausgabeartefakte im Ausgabe-Bucket gespeichert werden. `NONE` bedeutet, dass ein Ordner im Ausgabe-Bucket erstellt wird. Das Build-Ausgabeartefakt wir in diesem Ordner gespeichert.
  - `lastModified` zeigt die Uhrzeit der letzten Änderung des Build-Projekts im Unix-Zeitformat an.
  - `timeoutInMinutes` stellt die Anzahl der Minuten dar, nach denen den Build CodeBuild stoppt, wenn der Build nicht abgeschlossen wurde. (Der Standardwert ist 60 Minuten.)
  - `created` zeigt die Uhrzeit der Erstellung des Build-Projekts im Unix-Zeitformat an.
  - `environmentVariables` repräsentiert alle Umgebungsvariablen, die deklariert wurden und für CodeBuild die Verwendung während des Builds verfügbar sind.
  - `encryptionKey` stellt den ARN des vom Kunden verwalteten Schlüssels dar, mit dem das Build-Ausgabeartefakt verschlüsselt CodeBuild wurde.
  - `arn` zeigt den ARN des Build-Projekts an.

> **Note**
>
> Nachdem Sie den create-project Befehl ausgeführt haben, wird möglicherweise eine Fehlermeldung ähnlich der folgenden ausgegeben: User: ***user-ARN*** is not authorized to perform: codebuild:CreateProject. Dies liegt höchstwahrscheinlich daran, dass Sie die AWS CLI mit den Anmeldeinformationen eines -Benutzers konfiguriert haben, der nicht über ausreichende Berechtigungen zum Erstellen von Build CodeBuild -Projekten verfügt. Um dies zu beheben, konfigurieren Sie mit AWS CLI Anmeldeinformationen, die zu einer der folgenden IAM-Entitäten gehören:
>
> - Ein Administratorbenutzer in Ihrem AWS Konto. Weitere Informationen finden Sie unter Erstellen Ihres ersten AWS-Konto Stammbenutzers und Ihrer ersten Stammgruppe im -Benutzerhandbuch.
> - Ein -Benutzer in Ihrem AWS Konto mit den `IAMFullAccess` verwalteten Richtlinien `AWSCodeBuildAdminAccess`, und `AmazonS3ReadOnlyAccess`, die diesem Benutzer oder einer IAM-Gruppe zugeordnet sind, zu der der Benutzer gehört. Wenn Sie in Ihrem AWS Konto keinen -Benutzer oder keine -Gruppe mit diesen Berechtigungen haben und diese Berechtigungen Ihrem Benutzer oder Ihrer Gruppe nicht hinzufügen können, wenden Sie sich an Ihren AWS Kontoadministrator, um Unterstützung zu erhalten. Weitere Informationen finden Sie unter AWS verwaltete (vordefinierte) Richtlinien für AWS CodeBuild.

Nächster Schritt

Schritt 6: Ausführen des Builds

Schritt 6: Ausführen des Builds

(Vorheriger Schritt: Schritt 5: Erstellen des Build-Projekts)

In diesem Schritt weisen Sie AWS CodeBuild an, den Build mit den Einstellungen des Build-Projekts auszuführen.

So führen Sie den Build aus

1. Verwenden Sie die AWS CLI zum Ausführen des start-build-Befehls:

```
aws codebuild start-build --project-name project-name
```

Ersetzen Sie *project-name* durch den Build-Projektnamen aus dem vorherigen Schritt (z. B. `codebuild-demo-project`).

2. Ist der Befehl erfolgreich, gibt er als Ausgabe Daten zurück, die wie folgt aussehen sollten:

```
{
  "build": {
    "buildComplete": false,
    "initiator": "user-name",
    "artifacts": {
      "location": "arn:aws:s3:::codebuild-region-ID-account-ID-output-bucket/
message-util.zip"
    },
    "projectName": "codebuild-demo-project",
    "timeoutInMinutes": 60,
    "buildStatus": "IN_PROGRESS",
    "environment": {
      "computeType": "BUILD_GENERAL1_SMALL",
      "image": "aws/codebuild/standard:5.0",
      "type": "LINUX_CONTAINER",
      "environmentVariables": []
    },
    "source": {
      "type": "S3",
      "location": "codebuild-region-ID-account-ID-input-bucket/MessageUtil.zip"
    },
    "currentPhase": "SUBMITTED",
    "startTime": 1472848787.882,
    "id": "codebuild-demo-project:0cfbb6ec-3db9-4e8c-992b-1ab28EXAMPLE",
    "arn": "arn:aws:codebuild:region-ID:account-ID:build/codebuild-demo-
project:0cfbb6ec-3db9-4e8c-992b-1ab28EXAMPLE"
  }
}
```

- `build` enthält Informationen zu diesem Build.
  - `buildComplete` gibt an, ob der Build abgeschlossen wurde (`true`). Andernfalls `false`.
  - `initiator` steht für die Entität, die den Build gestartet hat.
  - `artifacts` enthält Informationen über die Build-Ausgabe, einschließlich Speicherort.

- `projectName` zeigt den Namen des Build-Projekts an.
- `buildStatus` stellt den aktuellen Build-Status dar, wenn der start-build-Befehl ausgeführt wurde.
- `currentPhase` stellt die aktuelle Build-Phase dar, wenn der start-build-Befehl ausgeführt wurde.
- `startTime` zeigt die Uhrzeit für den Start des Build-Prozesses im Unix-Zeitformat an.
- `id` enthält die Build-ID.
- `arn` zeigt den ARN des Builds an.

Notieren Sie sich den Wert für die `id`. Sie benötigen ihn im nächsten Schritt.

Nächster Schritt

Schritt 7: Anzeigen der Zusammenfassung der Build-Informationen

Schritt 7: Anzeigen der Zusammenfassung der Build-Informationen

(Vorheriger Schritt: Schritt 6: Ausführen des Builds)

In diesem Schritt zeigen Sie eine Zusammenfassung der Informationen über den Build-Status an.

So zeigen Sie eine Zusammenfassung der Build-Informationen an

Verwenden Sie die AWS CLI zum Ausführen des batch-get-builds-Befehls.

```
aws codebuild batch-get-builds --ids id
```

Ersetzen Sie *id* durch den `id`-Wert, der im vorhergehenden Schritt in der Ausgabe angezeigt wurde.

Ist der Befehl erfolgreich, gibt er als Ausgabe Daten zurück, die wie folgt aussehen sollten.

```
{
  "buildsNotFound": [],
  "builds": [
    {
      "buildComplete": true,
      "phases": [
        {
          "phaseStatus": "SUCCEEDED",
```

```
          "endTime": 1472848788.525,
          "phaseType": "SUBMITTED",
          "durationInSeconds": 0,
          "startTime": 1472848787.882
        },
        ... The full list of build phases has been omitted for brevity ...
        {
          "phaseType": "COMPLETED",
          "startTime": 1472848878.079
        }
      ],
      "logs": {
        "groupName": "/aws/codebuild/codebuild-demo-project",
        "deepLink": "https://console.aws.amazon.com/cloudwatch/home?region=region-
ID#logEvent:group=/aws/codebuild/codebuild-demo-project;stream=38ca1c4a-e9ca-4dbc-bef1-
d52bfEXAMPLE",
        "streamName": "38ca1c4a-e9ca-4dbc-bef1-d52bfEXAMPLE"
      },
      "artifacts": {
        "md5sum": "MD5-hash",
        "location": "arn:aws:s3:::codebuild-region-ID-account-ID-output-bucket/message-
util.zip",
        "sha256sum": "SHA-256-hash"
      },
      "projectName": "codebuild-demo-project",
      "timeoutInMinutes": 60,
      "initiator": "user-name",
      "buildStatus": "SUCCEEDED",
      "environment": {
        "computeType": "BUILD_GENERAL1_SMALL",
        "image": "aws/codebuild/standard:5.0",
        "type": "LINUX_CONTAINER",
        "environmentVariables": []
      },
      "source": {
        "type": "S3",
        "location": "codebuild-region-ID-account-ID-input-bucket/MessageUtil.zip"
      },
      "currentPhase": "COMPLETED",
      "startTime": 1472848787.882,
      "endTime": 1472848878.079,
      "id": "codebuild-demo-project:38ca1c4a-e9ca-4dbc-bef1-d52bfEXAMPLE",
      "arn": "arn:aws:codebuild:region-ID:account-ID:build/codebuild-demo-
project:38ca1c4a-e9ca-4dbc-bef1-d52bfEXAMPLE"
```

```
    }
  ]
}
```

- `buildsNotFound` zeigt die Build-IDs für alle Builds an, für die keine Informationen verfügbar sind. In diesem Beispiel sollte dies leer sein.
- `builds` zeigt die Informationen für alle Builds an, für die Informationen verfügbar sind. In diesem Beispiel sollten nur über einen Build Informationen in der Ausgabe angezeigt werden.
  - `phases` stellt den Satz von Build-Phasen dar, die während des Build-Prozesses CodeBuild ausgeführt werden. Informationen über die einzelnen Build-Phasen werden separat als `startTime`, `endTime` und `durationInSeconds` aufgelistet (wann die Build-Phase gestartet und beendet wurde, im Unix-Zeitformat, und wie lange sie in Sekunden gedauert hat) sowie der `phaseType` (z. B. `SUBMITTED`, `PROVISIONING`, `DOWNLOAD_SOURCE`, `INSTALL`, `PRE_BUILD`, `BUILD`, `POST_BUILD`, `UPLOAD_ARTIFACTS`, `FINALIZING` oder `COMPLETED`) und `phaseStatus` (z. B. `SUCCEEDED`, `FAILED`, `FAULT`, `TIMED_OUT`, `IN_PROGRESS` oder `STOPPED`). Wenn Sie den Befehl batch-get-builds zum ersten Mal ausführen, sind möglicherweise nicht viele (oder gar keine) Phasen vorhanden. Nach mehrmaligem Ausführen des Befehls batch-get-builds mit derselben Build-ID sollten mehr Build-Phasen in der Ausgabe angezeigt werden.
  - `logs` stellt Informationen in Amazon CloudWatch Logs über die Protokolle des Builds dar.
  - `md5sum` und `sha256sum` zeigen die Hash-Werte von MD5 und SHA-256 des Build-Ausgabeartefakts an. Diese werden nur dann in der Ausgabe angezeigt, wenn der `packaging`-Wert des Build-Projekts auf `ZIP` gesetzt ist. (Sie haben diesen Wert nicht in diesem Tutorial festgelegt.) Sie können die Hash-Werte zusammen mit dem Prüfsummen-Tool verwenden, um die Dateiintegrität und die Authentizität zu bestätigen.

    > **Note**
    >
    > Sie können diese Hashes auch über die Amazon S3-Konsole anzeigen. Aktivieren Sie das Kontrollkästchen neben dem Build-Ausgabeartefakt. Wählen Sie dann Actions (Aktionen) und schließlich Properties (Eigenschaften) aus. Erweitern Sie im Bereich Eigenschaften die Option Metadaten und zeigen Sie die Werte für x-amz-meta-codebuild-content-md5 und x-amz-meta-codebuild-content-sha256 an. (In der Amazon S3-Konsole sollte der ETag-Wert des Build-Ausgabeartefakts nicht als MD5- oder SHA-256-Hash interpretiert werden.)

Wenn Sie die AWS SDKs zum Abrufen dieser Hash-Werte verwenden, haben die Werte den Namen `codebuild-content-md5` und `codebuild-content-sha256`.

- `endTime` zeigt die Uhrzeit für das Ende des Build-Prozesses im Unix-Zeitformat an.

> **Note**
>
> Amazon-S3-Metadaten haben einen CodeBuild Header mit dem Namen `x-amz-meta-codebuild-buildarn`, der den `buildArn` des CodeBuild Builds enthält, der Artefakte in Amazon S3 veröffentlicht. Der `buildArn` wird hinzugefügt, um die Quellverfolgung für Benachrichtigungen zu ermöglichen und darauf zu verweisen, von welchem Build das Artefakt generiert wird.

Nächster Schritt

Schritt 8: Anzeigen der detaillierten Build-Informationen

Schritt 8: Anzeigen der detaillierten Build-Informationen

(Vorheriger Schritt: Schritt 7: Anzeigen der Zusammenfassung der Build-Informationen)

In diesem Schritt zeigen Sie detaillierte Informationen zu Ihrem Build in - CloudWatch Protokollen an.

> **Note**
>
> Um vertrauliche Informationen zu schützen, werden die folgenden in CodeBuild Protokollen ausgeblendet:
>
> - AWS-Zugriffsschlüssel-IDs. Weitere Informationen finden Sie unter Verwalten von Zugriffsschlüsseln für IAM-Benutzer im AWS Identity and Access Management - Benutzerhandbuch.
> - Mit dem Parameter Store angegebene Zeichenfolgen. Weitere Informationen finden Sie unter Systems Manager Parameter Store und Systems Manager Parameter Store Console Walkthrough im Amazon EC2 Systems Manager-Benutzerhandbuch.
> - Zeichenfolgen, die mit angegeben wurdenAWS Secrets Manager. Weitere Informationen finden Sie unter Schlüsselverwaltung.

So zeigen Sie detaillierte Build-Informationen an

1. Wechseln Sie in Ihrem Webbrowser zum `deepLink`-Speicherort, der in der Ausgabe des vorherigen Schrittes angezeigt wurde (beispielsweise `https://console.aws.amazon.com/cloudwatch/home?region=region-ID#logEvent:group=/aws/codebuild/codebuild-demo-project;stream=38ca1c4a-e9ca-4dbc-bef1-d52bfEXAMPLE`).
2. Im CloudWatch Protokollstream Logs können Sie die Protokollereignisse durchsuchen. Standardmäßig werden nur die letzten Protokollereignisse angezeigt. Um ältere Protokollereignisse anzuzeigen, blättern Sie zum Anfang der Liste.
3. In diesem Tutorial enthalten die meisten Protokollereignisse wahrscheinlich nicht benötigte detaillierte Informationen über das Herunterladen von Build-Abhängigkeitsdateien durch CodeBuild in die Build-Umgebung und deren Installation. Sie können die angezeigten Informationen über das Feld Filter events reduzieren. Wenn Sie beispielsweise `"[INFO]"` in das Feld Filter events (Ereignisse filtern) eingeben, werden nur Ereignisse angezeigt, die `[INFO]` enthalten. Weitere Informationen finden Sie unter Filter- und Mustersyntax im Amazon-CloudWatch Benutzerhandbuch.

Diese Teile eines CloudWatch Logs-Protokollstreams beziehen sich auf dieses Tutorial.

```
...
[Container] 2016/04/15 17:49:42 Entering phase PRE_BUILD
[Container] 2016/04/15 17:49:42 Running command echo Entering pre_build phase...
[Container] 2016/04/15 17:49:42 Entering pre_build phase...
[Container] 2016/04/15 17:49:42 Phase complete: PRE_BUILD Success: true
[Container] 2016/04/15 17:49:42 Entering phase BUILD
[Container] 2016/04/15 17:49:42 Running command echo Entering build phase...
[Container] 2016/04/15 17:49:42 Entering build phase...
[Container] 2016/04/15 17:49:42 Running command mvn install
[Container] 2016/04/15 17:49:44 [INFO] Scanning for projects...
[Container] 2016/04/15 17:49:44 [INFO]
[Container] 2016/04/15 17:49:44 [INFO]
 ------------------------------------------------------------------------
[Container] 2016/04/15 17:49:44 [INFO] Building Message Utility Java Sample App 1.0
[Container] 2016/04/15 17:49:44 [INFO]
 ------------------------------------------------------------------------
...
[Container] 2016/04/15 17:49:55
 -------------------------------------------------------
[Container] 2016/04/15 17:49:55  T E S T S
```

```
[Container] 2016/04/15 17:49:55
 -------------------------------------------------------
[Container] 2016/04/15 17:49:55 Running TestMessageUtil
[Container] 2016/04/15 17:49:55 Inside testSalutationMessage()
[Container] 2016/04/15 17:49:55 Hi!Robert
[Container] 2016/04/15 17:49:55 Inside testPrintMessage()
[Container] 2016/04/15 17:49:55 Robert
[Container] 2016/04/15 17:49:55 Tests run: 2, Failures: 0, Errors: 0, Skipped: 0, Time
 elapsed: 0.018 sec
[Container] 2016/04/15 17:49:55
[Container] 2016/04/15 17:49:55 Results :
[Container] 2016/04/15 17:49:55
[Container] 2016/04/15 17:49:55 Tests run: 2, Failures: 0, Errors: 0, Skipped: 0
...
[Container] 2016/04/15 17:49:56 [INFO]
 ------------------------------------------------------------------------
[Container] 2016/04/15 17:49:56 [INFO] BUILD SUCCESS
[Container] 2016/04/15 17:49:56 [INFO]
 ------------------------------------------------------------------------
[Container] 2016/04/15 17:49:56 [INFO] Total time: 11.845 s
[Container] 2016/04/15 17:49:56 [INFO] Finished at: 2016-04-15T17:49:56+00:00
[Container] 2016/04/15 17:49:56 [INFO] Final Memory: 18M/216M
[Container] 2016/04/15 17:49:56 [INFO]
 ------------------------------------------------------------------------
[Container] 2016/04/15 17:49:56 Phase complete: BUILD Success: true
[Container] 2016/04/15 17:49:56 Entering phase POST_BUILD
[Container] 2016/04/15 17:49:56 Running command echo Entering post_build phase...
[Container] 2016/04/15 17:49:56 Entering post_build phase...
[Container] 2016/04/15 17:49:56 Phase complete: POST_BUILD Success: true
[Container] 2016/04/15 17:49:57 Preparing to copy artifacts
[Container] 2016/04/15 17:49:57 Assembling file list
[Container] 2016/04/15 17:49:57 Expanding target/messageUtil-1.0.jar
[Container] 2016/04/15 17:49:57 Found target/messageUtil-1.0.jar
[Container] 2016/04/15 17:49:57 Creating zip artifact
```

In diesem Beispiel hat die Build-Phasen vor, nach und nach der Erstellung CodeBuild erfolgreich abgeschlossen. Es hat die Komponententests ausgeführt und die Datei `messageUtil-1.0.jar` erfolgreich erstellt.

Nächster Schritt

Schritt 9: Abrufen des Build-Ausgabeartefakts

Schritt 9: Abrufen des Build-Ausgabeartefakts

(Vorheriger Schritt: Schritt 8: Anzeigen der detaillierten Build-Informationen)

In diesem Schritt erhalten Sie die `messageUtil-1.0.jar` Datei, die CodeBuild erstellt und in den Ausgabe-Bucket hochgeladen hat.

Sie können diesen Schritt mit der CodeBuild -Konsole oder der Amazon S3-Konsole ausführen.

So rufen Sie das Build-Ausgabeartefakt ab (AWS CodeBuild-Konsole)

1. Wenn die CodeBuild Konsole noch geöffnet ist und die Seite mit den Build-Details aus dem vorherigen Schritt noch angezeigt wird, wählen Sie die Registerkarte Build-Details und scrollen Sie nach unten zum Abschnitt Artefakte.

   > **Note**
   >
   > Wenn die Seite mit den Build-Details nicht angezeigt wird, wählen Sie in der Navigationsleiste Build-Verlauf und dann den Link Build-Ausführung aus.

2. Der Link zum Amazon S3-Ordner befindet sich unter dem Speicherort des Artifacts-Uploads. Dieser Link öffnet den Ordner in Amazon S3, in dem Sie die `messageUtil-1.0.jar` Build-Ausgabeartefaktdatei finden.

So rufen Sie das Build-Ausgabeartefakt ab (Amazon S3-Konsole)

1. Öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Öffnen Sie `codebuild-`*`region-ID`*`-`*`account-ID`*`-output-bucket`.
3. Öffnen Sie das Verzeichnis `codebuild-demo-project`.
4. Öffnen Sie den Ordner `target`, in dem Sie die Build-Ausgabeartefaktdatei `messageUtil-1.0.jar` finden.

Nächster Schritt

Schritt 10: Löschen der S3-Buckets

Schritt 10: Löschen der S3-Buckets

(Vorheriger Schritt: Schritt 9: Abrufen des Build-Ausgabeartefakts)

Um laufende Gebühren für Ihr AWS Konto zu vermeiden, können Sie die in diesem Tutorial verwendeten Eingabe- und Ausgabe-Buckets löschen. Anweisungen finden Sie unter Löschen oder Leeren eines Buckets im Benutzerhandbuch für Amazon Simple Storage Service.

Wenn Sie den IAM-Benutzer oder einen Administrator-IAM-Benutzer zum Löschen dieser Buckets verwenden, muss der Benutzer über weitere Zugriffsberechtigungen verfügen. Fügen Sie die folgende Anweisung zwischen den Markern (*### BEGIN ADDING STATEMENT HERE ###* und *### END ADDING STATEMENTS HERE ###*) zu einer vorhandenen Zugriffsrichtlinie für den Benutzer hinzu.

Die Ellipsen (...) in dieser Anweisung dienen der Übersichtlichkeit der Darstellung. Entfernen Sie keine Anweisungen aus der vorhandenen Zugriffsrichtlinie. Geben Sie keine Auslassungspunkte in die Richtlinie ein.

```
{
  "Version": "2012-10-17",
  "Id": "...",
  "Statement": [
    ### BEGIN ADDING STATEMENT HERE ###
    {
      "Effect": "Allow",
      "Action": [
        "s3:DeleteBucket",
        "s3:DeleteObject"
      ],
      "Resource": "*"
    }
    ### END ADDING STATEMENT HERE ###
  ]
}
```

Nächster Schritt

Nachbearbeitung

Nachbearbeitung

In diesem Tutorial haben Sie AWS CodeBuild verwendet, um einen Satz Java-Klassendateien als Build in einer JAR-Datei zu erstellen. Anschließend haben Sie die Build-Ergebnisse angezeigt.

Sie können jetzt versuchen, CodeBuild in Ihren eigenen Szenarien zu verwenden. Folgen Sie den Anweisungen in Planen eines Builds. Wenn Sie dazu noch nicht bereit sind, können Sie einige der Beispiele als Build erstellen. Weitere Informationen finden Sie unter Beispiele.

CodeBuild Beispiele

Diese Gruppen von Proben können verwendet werden, um mit folgenden Objekten zu experimentieren AWS CodeBuild:

Themen

Verwenden Sie fallbasierte Stichproben für CodeBuild

Sie können diese auf Anwendungsfällen basierenden Beispiele verwenden, um mit folgenden Dingen zu experimentieren: AWS CodeBuild

Serviceübergreifende Stichproben

Eine Liste von dienstübergreifenden Beispielen, mit denen Sie experimentieren können. AWS CodeBuild

Build Badges-Beispiel

Zeigt die Einrichtung CodeBuild mit Build-Badges.

Erstellen eines Testberichts mithilfe des AWS CLI -Beispiels

Verwendet den, AWS CLI um einen Testbericht zu erstellen, auszuführen und die Ergebnisse eines Testberichts anzuzeigen.

Docker-Beispiele für CodeBuild

Zeigt, wie Sie benutzerdefinierte Docker-Images verwenden, Docker-Images in einem Repository in Amazon ECR veröffentlichen und Docker-Images in einer privaten Registrierung verwenden.

Hosten der Build-Ausgabe in einem S3-Bucket

Zeigt das Erstellen einer statischen Website in einem S3-Bucket mit unverschlüsselten Build-Artefakten.

Beispiel für mehrere Eingabequellen und Ausgabeartefakte

Demonstriert die Verwendung mehrerer Eingabequellen und mehrerer Ausgabeartefakte in einem Build-Projekt.

Laufzeitversionen im Build-Spezfikationsdateibeispiel

Zeigt, wie Sie Laufzeiten und deren Versionen in der Build-Spezifikationsdatei angeben.

Beispiel für eine Quellversion

Zeigt, wie Sie eine bestimmte Version Ihrer Quelle in einem Build-Projekt verwenden. CodeBuild

Quell-Repository-Beispiele von Drittanbietern für CodeBuild

Zeigt BitBucket, wie Sie GitHub Enterprise Server- und GitHub Pull-Requests mithilfe von CodeBuild Webhooks erstellen.

Beispiel für die Verwendung des semantischen Versionings zum Benennen von Build-Artefakten

Illustriert die Verwendung des semantischen Versionings zum Erstellen eines Artefaktnamens während der Erstellung des Builds.

Serviceübergreifende Beispiele für CodeBuild

Sie können diese serviceübergreifenden Beispiele verwenden, um mit folgenden Dingen zu experimentieren: AWS CodeBuild

Amazon ECR-Beispiel

Verwendet ein Docker-Image in einem Amazon ECR-Repository, um mit Apache Maven eine einzelne JAR-Datei zu erstellen.

Amazon EFS-Beispiel

Zeigt, wie eine Buildspec-Datei so konfiguriert wird, dass ein CodeBuild Projekt auf einem Amazon EFS-Dateisystem bereitgestellt und darauf aufgebaut wird.

AWS CodePipeline Proben

Zeigt, wie man einen Build mit Batch-Builds sowie mehreren Eingabequellen und mehreren Ausgabeartefakten erstellt. AWS CodePipeline

AWS Config Probe

Zeigt, wie man es einrichtet AWS Config. Führt auf, welche CodeBuild Ressourcen nachverfolgt werden, und beschreibt, wie CodeBuild Projekte nachgeschlagen werden können AWS Config.

Build-Benachrichtigungsbeispiel

Verwendet Apache Maven, um eine einzelne JAR-Datei zu erstellen. Sendet eine Build-Benachrichtigung an Abonnenten eines Amazon SNS SNS-Themas.

Amazon ECR-Beispiel für CodeBuild

In diesem Beispiel wird ein Docker-Image in einem Image-Repository von Amazon Elastic Container Registry (Amazon ECR) verwendet, um ein Go-Beispielprojekt zu erstellen.

> **Important**
>
> Die Ausführung dieses Beispiels kann dazu führen, dass Ihr AWS Konto belastet wird. Dazu gehören mögliche Gebühren für AWS CodeBuild und für AWS Ressourcen und Aktionen im Zusammenhang mit Amazon S3 AWS KMS, CloudWatch Logs und Amazon ECR. Weitere Informationen finden Sie unter CodeBuild Preise, Amazon S3 S3-Preise, AWS Key Management Service Preise, CloudWatch Amazon-Preise und Amazon Elastic Container Registry — Preise.

Ausführen des Beispiels

So führen Sie das Beispiel aus

1. Um das Docker-Image zu erstellen und in Ihr Image-Repository in Amazon ECR zu übertragen, führen Sie die Schritte im Abschnitt „Das Beispiel ausführen“ der aus. Docker-Image in einem Amazon ECR-Image-Repository-Beispiel veröffentlichen
2. Erstellen eines Go-Projekts:

   a. Erstellen Sie die Dateien wie in den Go-Projektdateien Abschnitten Go-Projektstruktur und in diesem Thema beschrieben und laden Sie sie dann in einen S3-Eingabe-Bucket oder ein AWS CodeCommit, GitHub, oder Bitbucket-Repository hoch.

   > **Important**
   >
   > Laden Sie nicht *(root directory name)* hoch, sondern nur die Dateien in *(root directory name)*.
   > Wenn Sie einen S3-Empfangs-Bucket verwenden, sollten Sie eine ZIP-Datei erstellen, die die Dateien enthält, und diese dann in den Empfangs-Bucket

hochladen. Fügen Sie *(root directory name)* nicht zur ZIP-Datei hinzu, sondern nur die Dateien in *(root directory name)*.

b. Erstelle ein Build-Projekt, führe den Build aus und sieh dir die zugehörigen Build-Informationen an.

Wenn Sie das AWS CLI Build-Projekt mit erstellen, sieht die Eingabe des `create-project` Befehls im JSON-Format möglicherweise ähnlich aus. (Ersetzen Sie die Platzhalter durch Ihre eigenen Werte.)

```
{
  "name": "sample-go-project",
  "source": {
    "type": "S3",
    "location": "codebuild-region-ID-account-ID-input-bucket/GoSample.zip"
  },
  "artifacts": {
    "type": "S3",
    "location": "codebuild-region-ID-account-ID-output-bucket",
    "packaging": "ZIP",
    "name": "GoOutputArtifact.zip"
  },
  "environment": {
    "type": "LINUX_CONTAINER",
    "image": "aws/codebuild/standard:5.0",
    "computeType": "BUILD_GENERAL1_SMALL"
  },
  "serviceRole": "arn:aws:iam::account-ID:role/role-name",
  "encryptionKey": "arn:aws:kms:region-ID:account-ID:key/key-ID"
}
```

c. Zum Abrufen des Build-Ausgabeartefakts, öffnen Sie Ihren S3-Ausgabe-Bucket.

d. Laden Sie die Datei *GoOutputArtifact*`.zip` auf Ihren lokalen Computer oder Ihre lokale Instance herunter. Extrahieren Sie anschließend den Inhalt der Datei . Rufen Sie im extrahierten Inhalt die Datei `hello` auf.

3. Wenn eine der folgenden Bedingungen zutrifft, müssen Sie Ihrem Image-Repository in Amazon ECR Berechtigungen hinzufügen, damit das zugehörige Docker-Image in die Build-Umgebung abgerufen werden AWS CodeBuild kann.

- Ihr Projekt verwendet CodeBuild Anmeldeinformationen, um Amazon ECR-Bilder abzurufen. Dies wird durch den Wert von `CODEBUILD` im Attribut `imagePullCredentialsType` Ihrer `ProjectEnvironment` angezeigt.
- Ihr Projekt verwendet ein kontoübergreifendes Amazon ECR-Image. In diesem Fall muss Ihr Projekt seine Servicerolle verwenden, um Amazon ECR-Images abzurufen. Um dieses Verhalten zu aktivieren, setzen Sie das Attribut `imagePullCredentialsType` Ihrer `ProjectEnvironment` auf `SERVICE_ROLE`.

1. Öffnen Sie die Amazon ECR-Konsole unter https://console.aws.amazon.com/ecr/.
2. Klicken Sie in der Liste der Repository-Namen den Namen des Repositories aus, das Sie erstellt oder ausgewählt haben.
3. Wählen Sie im Navigationsbereich Permissions (Berechtigungen) und dann Edit (Bearbeiten) aus. Klicken Sie anschließend auf Add statement (Anweisung hinzufügen).
4. Geben Sie in Statement name einen Bezeichner (z. B. **`CodeBuildAccess`**) ein.
5. Für Effect (Effekt) lassen Sie Allow (Zulassen) ausgewählt. Dies zeigt an, dass Sie den Zugriff auf ein anderes AWS -Konto zulassen möchten.
6. Wählen Sie für Principal (Prinzipal) eine der folgenden Vorgehensweisen:
   - Wenn Ihr Projekt CodeBuild Anmeldeinformationen verwendet, um ein Amazon ECR-Image abzurufen, geben **`codebuild.amazonaws.com`** Sie im Feld Service Principal den folgenden Wert ein.
   - Wenn Ihr Projekt ein kontoübergreifendes Amazon ECR-Image verwendet, geben Sie für AWS Konto-IDs die IDs der AWS Konten ein, denen Sie Zugriff gewähren möchten.
7. Überspringen Sie die Liste All IAM entities.
8. Wählen Sie unter Aktion die Aktionen, die nur abgerufen werden können: ecr:GetDownloadUrlForLayer, ecr: und ecr: aus. BatchGetImage BatchCheckLayerAvailability
9. Fügen Sie unter Bedingungen Folgendes hinzu:

```
{
   "StringEquals":{
      "aws:SourceAccount":"<AWS-account-ID>",
      "aws:SourceArn":"arn:aws:codebuild:<region>:<AWS-account-
ID>:project/<project-name>"
   }
}
```

10. Wählen Sie Speichern.

Diese Richtlinie wird in Permissions (Berechtigungen) angezeigt. Der Prinzipal entspricht dem, was Sie in Schritt 3 dieses Verfahrens für Principal (Prinzipal) eingegeben haben:

- Wenn Ihr Projekt CodeBuild Anmeldeinformationen verwendet, um ein Amazon ECR-Image abzurufen, `"codebuild.amazonaws.com"` wird es unter Service Principals angezeigt.
- Wenn Ihr Projekt ein kontoübergreifendes Amazon ECR-Image verwendet, wird die ID des AWS Kontos, dem Sie Zugriff gewähren möchten, unter AWS Konto-IDs angezeigt.

  Die folgende Beispielrichtlinie verwendet sowohl CodeBuild Anmeldeinformationen als auch ein kontoübergreifendes Amazon ECR-Image.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"CodeBuildAccessPrincipal",
         "Effect":"Allow",
         "Principal":{
            "Service":"codebuild.amazonaws.com"
         },
         "Action":[
            "ecr:GetDownloadUrlForLayer",
            "ecr:BatchGetImage",
            "ecr:BatchCheckLayerAvailability"
         ],
         "Condition":{
            "StringEquals":{
               "aws:SourceArn":"arn:aws:codebuild:<region>:<aws-account-
id>:project/<project-name>",
               "aws:SourceAccount":"<aws-account-id>"
            }
         }
      },
      {
         "Sid":"CodeBuildAccessCrossAccount",
         "Effect":"Allow",
         "Principal":{
            "AWS":"arn:aws:iam::<AWS-account-ID>:root"
         },
         "Action":[
            "ecr:GetDownloadUrlForLayer",
```

```
            "ecr:BatchGetImage",
            "ecr:BatchCheckLayerAvailability"
         ]
      }
   ]
}
```

- Wenn Ihre Projekte CodeBuild Anmeldeinformationen verwenden und Sie möchten, dass Ihre CodeBuild Projekte offenen Zugriff auf das Amazon ECR-Repository haben, können Sie die `Condition` Schlüssel weglassen und die folgende Beispielrichtlinie hinzufügen.

```
{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Sid":"CodeBuildAccessPrincipal",
      "Effect":"Allow",
      "Principal":{
        "Service":"codebuild.amazonaws.com"
      },
      "Action":[
        "ecr:GetDownloadUrlForLayer",
        "ecr:BatchGetImage",
        "ecr:BatchCheckLayerAvailability"
      ]
    },
    {
      "Sid":"CodeBuildAccessCrossAccount",
      "Effect":"Allow",
      "Principal":{
        "AWS":"arn:aws:iam::<AWS-account-ID>:root"
      },
      "Action":[
        "ecr:GetDownloadUrlForLayer",
        "ecr:BatchGetImage",
        "ecr:BatchCheckLayerAvailability"
      ]
    }
  ]
}
```

4. Erstellen Sie ein Build-Projekt, führen Sie den Build aus und sehen Sie sich die Build-Informationen an.

Wenn Sie das AWS CLI Build-Projekt mit erstellen, sieht die Eingabe des `create-project` Befehls im JSON-Format möglicherweise ähnlich aus. (Ersetzen Sie die Platzhalter durch Ihre eigenen Werte.)

```
{
  "name": "amazon-ecr-sample-project",
  "source": {
    "type": "S3",
    "location": "codebuild-region-ID-account-ID-input-bucket/GoSample.zip"
  },
  "artifacts": {
    "type": "S3",
    "location": "codebuild-region-ID-account-ID-output-bucket",
    "packaging": "ZIP",
    "name": "GoOutputArtifact.zip"
  },
  "environment": {
    "type": "LINUX_CONTAINER",
    "image": "account-ID.dkr.ecr.region-ID.amazonaws.com/your-Amazon-ECR-repo-
name:tag",
    "computeType": "BUILD_GENERAL1_SMALL"
  },
  "serviceRole": "arn:aws:iam::account-ID:role/role-name",
  "encryptionKey": "arn:aws:kms:region-ID:account-ID:key/key-ID"
}
```

5. Zum Abrufen des Build-Ausgabeartefakts, öffnen Sie Ihren S3-Ausgabe-Bucket.
6. Laden Sie die Datei *GoOutputArtifact*`.zip` auf Ihren lokalen Computer oder Ihre lokale Instance herunter. Extrahieren Sie anschließend den Inhalt der Datei *GoOutputArtifact*`.zip`. Rufen Sie im extrahierten Inhalt die Datei `hello` auf.

Go-Projektstruktur

In diesem Beispiel wird von dieser Verzeichnisstruktur ausgegangen.

```
(root directory name)
### buildspec.yml
### hello.go
```

Go-Projektdateien

In diesem Beispiel werden diese Dateien verwendet.

`buildspec.yml` (in *(root directory name)*)

```
version: 0.2

phases:
  install:
   runtime-versions:
     golang: 1.13
  build:
    commands:
      - echo Build started on `date`
      - echo Compiling the Go code
      - go build hello.go
  post_build:
    commands:
      - echo Build completed on `date`
artifacts:
  files:
    - hello
```

`hello.go` (in *(root directory name)*)

```
package main
import "fmt"

func main() {
  fmt.Println("hello world")
  fmt.Println("1+1 =", 1+1)
  fmt.Println("7.0/3.0 =", 7.0/3.0)
  fmt.Println(true && false)
  fmt.Println(true || false)
  fmt.Println(!true)
}
```

Zugehörige Ressourcen

- Hinweise zu den ersten Schritten mit AWS CodeBuild finden Sie unter. Erste Schritte mit AWS CodeBuild unter Verwendung der Konsole

- Informationen zur Behebung von Problemen finden Sie unterFehlerbehebung AWS CodeBuild. CodeBuild
- Informationen zu Kontingenten in CodeBuild finden Sie unterKontingente für AWS CodeBuild.

Amazon Elastic File System-Beispiel für AWS CodeBuild

Möglicherweise möchten Sie Ihre AWS CodeBuild Builds auf Amazon Elastic File System erstellen, einem skalierbaren, gemeinsam genutzten Dateidienst für Amazon EC2 EC2-Instances. Die Speicherkapazität von Amazon EFS ist elastisch, sodass sie mit dem Hinzufügen und Entfernen von Dateien wächst oder schrumpft. Über eine einfache Web-Service-Schnittstelle können Sie Dateisysteme erstellen und konfigurieren. Die gesamte Dateispeicher-Infrastruktur wird für Sie verwaltet. Sie müssen sich damit nicht um Bereitstellung, Patching oder Wartung von Dateisystem-Konfigurationen kümmern. Weitere Informationen finden Sie unter Was ist Amazon Elastic File System? im Amazon Elastic File System-Benutzerhandbuch.

Dieses Beispiel zeigt Ihnen, wie Sie ein CodeBuild Projekt so konfigurieren, dass es eine Java-Anwendung in einem Amazon EFS-Dateisystem mountet und dann erstellt. Bevor Sie beginnen, müssen Sie eine Java-Anwendung zur Erstellung bereit haben, die in einen S3-Eingabe-Bucket oder ein AWS CodeCommit, GitHub, GitHub Enterprise Server- oder Bitbucket-Repository hochgeladen wird.

Daten, die für Ihr Dateisystem übertragen werden, werden verschlüsselt. Informationen zum Verschlüsseln von Daten während der Übertragung mit einem anderen Bild finden Sie unter Daten im Transit verschlüsseln.

Allgemeine Schritte

Dieses Beispiel behandelt die drei grundlegenden Schritte, die für die Verwendung von Amazon EFS erforderlich sind mit AWS CodeBuild:

1. Erstellen Sie eine virtuelle private Cloud (VPC) in Ihrem AWS Konto.
2. Erstellen Sie ein Dateisystem, das diese VPC verwendet.
3. Erstellen und erstellen Sie ein CodeBuild Projekt, das die VPC verwendet. Das CodeBuild Projekt verwendet Folgendes, um das Dateisystem zu identifizieren:
   - Eine eindeutige Dateisystemkennung. Sie wählen die Kennung, wenn Sie das Dateisystem in Ihrem Build-Projekt angeben.

- Die Dateisystem-ID. Die ID wird angezeigt, wenn Sie Ihr Dateisystem in der Amazon EFS-Konsole aufrufen.
- Ein Mountingpunkt. Dies ist ein Verzeichnis in Ihrem Docker-Container, das das Dateisystem mountet.
- Mountingoptionen. Dazu gehören Details zum Mounten des Dateisystems.

> **Note**
>
> Ein in Amazon EFS erstelltes Dateisystem wird nur auf Linux-Plattformen unterstützt.

Erstellen Sie eine VPC mit AWS CloudFormation

Erstellen Sie Ihre VPC mit einer AWS CloudFormation Vorlage.

1. Folgen Sie den Anweisungen unter AWS CloudFormation VPC-Vorlage AWS CloudFormation So erstellen Sie eine VPC.

   > **Note**
   >
   > Die mit dieser AWS CloudFormation Vorlage erstellte VPC hat zwei private Subnetze und zwei öffentliche Subnetze. Sie dürfen private Subnetze nur verwenden, wenn Sie das Dateisystem mounten AWS CodeBuild , das Sie in Amazon EFS erstellt haben. Bei Verwendung eines der öffentlichen Subnetze schlägt der Build fehl.

2. Melden Sie sich bei der Amazon VPC-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/vpc/.
3. Wählen Sie die VPC aus, mit AWS CloudFormation der Sie erstellt haben.
4. Notieren Sie sich den Namen der VPC und ihre ID, die auf der Registerkarte Description (Beschreibung) angezeigt werden. Beide sind erforderlich, wenn Sie Ihr AWS CodeBuild Projekt später in diesem Beispiel erstellen.

Erstellen Sie mit Ihrer VPC ein Amazon Elastic File System-Dateisystem

Erstellen Sie ein einfaches Amazon EFS-Dateisystem für dieses Beispiel mit der VPC, die Sie zuvor erstellt haben.

1. Melden Sie sich bei der Amazon EFS-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/efs/.
2. Wählen Sie Create file system (Dateisystem erstellen) aus.
3. Wählen Sie unter VPC den zuvor in diesem Beispiel notierten VPC-Namen aus.
4. Lassen Sie die Availability Zones für Ihre Subnetze ausgewählt.
5. Wählen Sie Next Step (Weiter) aus.
6. Geben Sie unter Tags hinzufügen für den Standardschlüssel Name im Feld Wert den Namen Ihres Amazon EFS-Dateisystems ein.
7. Lassen Sie Bursting und General Purpose (Universal) als Standardleistungs- und -durchsatzmodus ausgewählt. Wählen Sie dann Next Step (Nächster Schritt) aus.
8. Wählen Sie unter Configure client access (Client-Zugriff konfigurieren) die Option Next Step (Nächster Schritt).
9. Klicken Sie auf Create File System (Dateisystem erstellen).
10. (Optional) Wir empfehlen, Ihrem Amazon EFS-Dateisystem eine Richtlinie hinzuzufügen, die die Verschlüsselung von Daten bei der Übertragung erzwingt. Wählen Sie in der Amazon EFS-Konsole Dateisystemrichtlinie, klicken Sie auf Bearbeiten, aktivieren Sie das Kästchen Verschlüsselung während der Übertragung für alle Clients erzwingen und wählen Sie dann Speichern aus.

Erstellen Sie ein CodeBuild Projekt zur Verwendung mit Amazon EFS

Erstellen Sie ein AWS CodeBuild Projekt, das die VPC verwendet, die Sie zuvor in diesem Beispiel erstellt haben. Wenn der Build ausgeführt wird, hängt er das zuvor erstellte Amazon EFS-Dateisystem ein. Als Nächstes speichert es die von Ihrer Java-Anwendung erstellte .jar-Datei im Mountingpunkt-Verzeichnis Ihres Dateisystems.

1. Öffnen Sie die AWS CodeBuild Konsole unter https://console.aws.amazon.com/codesuite/codebuild/home.
2. Wählen Sie im Navigationsbereich Build projects (Build-Projekte) gefolgt von Create build project (Build-Projekt erstellen) aus.
3. Geben Sie im Feld Projektname einen Namen für Ihr Projekt an.
4. Wählen Sie unter Source provider (Quellanbieter) das Repository aus, in dem sich die zu erstellende Java-Anwendung befindet.

5. Geben Sie Informationen ein, z. B. eine Repository-URL, CodeBuild anhand derer Ihre Anwendung gefunden wird. Die Optionen unterscheiden sich je nach Quellanbieter. Weitere Informationen finden Sie unter Choose source provider.
6. Wählen Sie unter Environment image (Umgebungs-Image) die Option Managed image (Verwaltetes Image) aus.
7. Wählen Sie unter Operating system (Betriebssystem) die Option Amazon Linux 2 aus.
8. Wählen Sie unter Runtime(s) (Laufzeit(en)) die Option Standard aus.
9. Wählen Sie unter Image die Option aws/codebuild/amazonlinux2-x86_64-standard:4.0 aus.
10. Wählen Sie für Environment type (Umgebungstyp) die Option Linux aus.
11. Wählen Sie unter Service role (Servicerolle) die Option New service role (Neue Servicerolle) aus. Geben Sie im Feld Rollenname einen Namen für die Rolle ein, die für Sie erstellt wurde. CodeBuild
12. Erweitern Sie Additional configuration (Zusätzliche Konfiguration).
13. Wählen Sie Enable this flag if you want to build Docker images or want your builds to get elevated privileges (Dieses Flag aktivieren, wenn Docker-Abbilder erstellt oder die Builds erweiterte Berechtigungen erhalten sollen) aus.

    > **Note**
    >
    > Standardmäßig ist der Docker-Daemon für Nicht-VPC-Builds aktiviert. Wenn Sie Docker-Container für VPC-Builds verwenden möchten, lesen Sie auf der Docker Docs-Website unter Runtime Privilege and Linux Capabilities nach und aktivieren Sie den privilegierten Modus. Außerdem unterstützt Windows den privilegierten Modus nicht.

14. Wählen Sie unter VPC die VPC-ID aus.
15. Wählen Sie unter Subnets (Subnetze) mindestens eines der privaten Subnetze aus, die Ihrer VPC zugeordnet sind. Sie müssen private Subnetze in einem Build verwenden, der ein Amazon EFS-Dateisystem mountet. Bei Verwendung eines öffentlichen Subnetzes schlägt der Build fehl.
16. Wählen Sie unter Security groups (Sicherheitsgruppen) die Standardsicherheitsgruppe aus.
17. Geben Sie unter File systems (Dateisysteme) die folgenden Informationen ein:

    - Geben Sie unter Identifier (Bezeichner), einen eindeutigen Dateisystembezeichner ein. Es darf weniger als 129 Zeichen lang sein und darf nur alphanumerische Zeichen und Unterstriche enthalten. CodeBuild verwendet diesen Bezeichner, um eine Umgebungsvariable zu erstellen, die das elastische Dateisystem identifiziert. Das Format der Umgebungsvariablen ist

`CODEBUILD_<file_system_identifier>` in Großbuchstaben. Wenn Sie beispielsweise `my_efs` eingeben, lautet die Umgebungsvariable `CODEBUILD_MY_EFS`.

- Wählen Sie für ID die Dateisystem-ID aus.
- (Optional) Geben Sie ein Verzeichnis im Dateisystem ein. CodeBuild hängt dieses Verzeichnis ein. Wenn Sie den Verzeichnispfad leer lassen, wird CodeBuild das gesamte Dateisystem eingehängt. Der Dateipfad ist relativ zum Stamm des Dateisystems.
- Geben Sie als Einhängepunkt den absoluten Pfad des Verzeichnisses in Ihrem Build-Container ein, in dem das Dateisystem eingehängt ist. Wenn dieses Verzeichnis nicht existiert, CodeBuild wird es während des Builds erstellt.
- (Optional) Geben Sie Mountingoptionen ein. Wenn Sie das Feld Einhängeoptionen leer lassen, CodeBuild werden die Standard-Einhängeoptionen verwendet:

```
nfsvers=4.1
rsize=1048576
wsize=1048576
hard
timeo=600
retrans=2
```

Weitere Informationen finden Sie unter Empfohlene NFS-Mount-Optionen im Amazon Elastic File System-Benutzerhandbuch.

18. Wählen Sie unter Build specification (Build-Spezifikation) die Option Insert build commands (Build-Befehle einfügen) und anschließend Switch to editor (Zum Editor wechseln) aus.
19. Geben Sie die folgenden Build-Spezifikationsbefehle in den Editor ein. Ersetzen Sie *<file_system_identifier>* durch den Bezeichner, den Sie in Schritt 17 eingegeben haben. Verwenden Sie Großbuchstaben (z. B. `CODEBUILD_MY_EFS`).

```
version: 0.2
phases:
  install:
    runtime-versions:
      java: corretto11
  build:
    commands:
      - mvn compile -Dgpg.skip=true -Dmaven.repo.local=
$CODEBUILD_<file_system_identifier>
```

20. Verwenden Sie für alle anderen Einstellungen die Standardwerte und wählen Sie Create build project (Build-Projekt erstellen) aus. Wenn Ihr Build abgeschlossen ist, wird die Konsolenseite für Ihr Projekt angezeigt.
21. Wählen Sie Start build (Build starten).

CodeBuild und Zusammenfassung der Amazon EFS-Beispiele

Nachdem Ihr AWS CodeBuild Projekt erstellt wurde:

- Sie haben eine von Ihrer Java-Anwendung erstellte JAR-Datei, die für Ihr Amazon EFS-Dateisystem in Ihrem Mount-Point-Verzeichnis erstellt wurde.
- Eine Umgebungsvariable, die Ihr Dateisystem identifiziert, wird mithilfe der beim Erstellen des Projekts eingegebenen Dateisystembezeichner erstellt.

Weitere Informationen finden Sie unter Mounten von Dateisystemen im Amazon Elastic File System-Benutzerhandbuch.

Fehlerbehebung

Die folgenden Fehler können bei der Einrichtung von Amazon EFS auftreten CodeBuild.

Themen

- CLIENT_ERROR: Das Mounten von '127.0.0.1: /' ist fehlgeschlagen. Zugriff verweigert
- CLIENT_ERROR: Das Mounten von '127.0.0.1: /' ist fehlgeschlagen. Verbindung wurde durch Peer zurückgesetzt
- VPC_CLIENT_ERROR: Unerwarteter EC2-Fehler: UnauthorizedOperation

CLIENT_ERROR: Das Mounten von '127.0.0.1: /' ist fehlgeschlagen. Zugriff verweigert

Die IAM-Autorisierung wird für das Mounten von Amazon EFS mit CodeBuild nicht unterstützt. Wenn Sie eine benutzerdefinierte Amazon EFS-Dateisystemrichtlinie verwenden, müssen Sie allen IAM-Prinzipalen Lese- und Schreibzugriff gewähren. Beispielsweise:

```
"Principal": {
  "AWS": "*"
}
```

CLIENT_ERROR: Das Mounten von '127.0.0.1: /' ist fehlgeschlagen. Verbindung wurde durch Peer zurückgesetzt

Es gibt zwei mögliche Ursachen für diesen Fehler:

- Das CodeBuild VPC-Subnetz befindet sich in einer anderen Availability Zone als das Amazon EFS-Mount-Ziel. Sie können dieses Problem lösen, indem Sie ein VPC-Subnetz in derselben Availability Zone wie das Amazon EFS-Mount-Ziel hinzufügen.
- Die Sicherheitsgruppe ist nicht berechtigt, mit Amazon EFS zu kommunizieren. Sie können dieses Problem lösen, indem Sie eine Regel für eingehenden Datenverkehr hinzufügen, die den gesamten Datenverkehr entweder von der VPC (fügen Sie den primären CIDR-Block für Ihre VPC hinzu) oder von der Sicherheitsgruppe selbst zulässt.

VPC_CLIENT_ERROR: Unerwarteter EC2-Fehler: UnauthorizedOperation

Dieser Fehler tritt auf, wenn alle Subnetze in Ihrer VPC-Konfiguration für das CodeBuild Projekt öffentliche Subnetze sind. Sie müssen mindestens ein privates Subnetz in der VPC haben, um die Netzwerkkonnektivität sicherzustellen.

CodePipeline Beispiele für CodeBuild

Themen

- AWS CodePipeline Integration mit CodeBuild und Batch-Builds
- AWS CodePipeline Beispiel für Integration mit CodeBuild und mehrere Eingabequellen und Ausgabeartefakte

AWS CodePipeline Integration mit CodeBuild und Batch-Builds

AWS CodeBuild unterstützt jetzt Batch-Builds. Dieses Beispiel zeigt, wie Sie AWS CodePipeline ein Build-Projekt erstellen, das Batch-Builds verwendet.

Sie können eine Datei im JSON-Format verwenden, die die Struktur Ihrer Pipeline definiert, und sie dann zusammen mit der verwenden, AWS CLI um die Pipeline zu erstellen. Weitere Informationen finden Sie unter AWS CodePipeline Pipeline-Strukturreferenz im AWS CodePipeline Benutzerhandbuch.

Batch-Erstellung mit einzelnen Artefakten

Verwenden Sie die folgende JSON-Datei als Beispiel für eine Pipeline-Struktur, die einen Batch-Build mit separaten Artefakten erstellt. Um Batch-Builds zu aktivieren CodePipeline, setzen Sie den `BatchEnabled` Parameter des `configuration` Objekts auf`true`.

```
{
  "pipeline": {
    "roleArn": "arn:aws:iam::account-id:role/my-AWS-CodePipeline-service-role-name",
    "stages": [
      {
        "name": "Source",
        "actions": [
          {
            "inputArtifacts": [],
            "name": "Source1",
            "actionTypeId": {
              "category": "Source",
              "owner": "AWS",
              "version": "1",
              "provider": "S3"
            },
            "outputArtifacts": [
              {
                "name": "source1"
              }
            ],
            "configuration": {
              "S3Bucket": "<my-input-bucket-name>",
              "S3ObjectKey": "my-source-code-file-name.zip"
            },
            "runOrder": 1
          },
          {
            "inputArtifacts": [],
            "name": "Source2",
            "actionTypeId": {
              "category": "Source",
              "owner": "AWS",
              "version": "1",
              "provider": "S3"
            },
            "outputArtifacts": [
```

```
            {
              "name": "source2"
            }
          ],
          "configuration": {
            "S3Bucket": "<my-other-input-bucket-name>",
            "S3ObjectKey": "my-other-source-code-file-name.zip"
          },
          "runOrder": 1
        }
      ]
    },
    {
      "name": "Build",
      "actions": [
        {
          "inputArtifacts": [
            {
              "name": "source1"
            },
            {
              "name": "source2"
            }
          ],
          "name": "Build",
          "actionTypeId": {
            "category": "Build",
            "owner": "AWS",
            "version": "1",
            "provider": "CodeBuild"
          },
          "outputArtifacts": [
            {
              "name": "build1"
            },
            {
              "name": "build1_artifact1"
            },
            {
              "name": "build1_artifact2"
            },
            {
              "name": "build2_artifact1"
            },
```

```
              {
                "name": "build2_artifact2"
              }
            ],
            "configuration": {
              "ProjectName": "my-build-project-name",
              "PrimarySource": "source1",
              "BatchEnabled": "true"
            },
            "runOrder": 1
          }
        ]
      }
    ],
    "artifactStore": {
      "type": "S3",
      "location": "<AWS-CodePipeline-internal-bucket-name>"
    },
    "name": "my-pipeline-name",
    "version": 1
  }
}
```

Im Folgenden finden Sie ein Beispiel für eine CodeBuild Buildspec-Datei, die mit dieser Pipeline-Konfiguration funktioniert.

```
version: 0.2
batch:
  build-list:
    - identifier: build1
      env:
        compute-type: BUILD_GENERAL1_SMALL
    - identifier: build2
      env:
        compute-type: BUILD_GENERAL1_MEDIUM

phases:
  build:
    commands:
      - echo 'file' > output_file

artifacts:
  files:
```

```
    - output_file
  secondary-artifacts:
    artifact1:
      files:
        - output_file
    artifact2:
      files:
        - output_file
```

Die Namen der in der JSON-Datei der Pipeline angegebenen Ausgabeartefakte müssen mit der Kennung der Builds und Artefakte übereinstimmen, die in Ihrer Buildspec-Datei definiert sind. *Die Syntax lautet buildIdentifier für die primären Artefakte und buildIdentifier _ artifactIdentifier für die sekundären Artefakte.*

Wird beispielsweise für den Namen des Ausgabeartefakts das primäre Artefakt von an den Speicherort von `build1` hochgeladen CodeBuild . `build1` `build1` Als Ausgabename `build1_artifact1` CodeBuild wird das sekundäre Artefakt `artifact1` von `build1` an den Speicherort von `build1_artifact1` hochgeladen usw. Wenn nur ein Ausgabespeicherort angegeben ist, sollte der Name nur *buildIdentifier lauten.*

Nachdem Sie die JSON-Datei erstellt haben, können Sie die Pipeline erstellen. Verwenden Sie den AWS CLI , um den Befehl create-pipeline auszuführen und die Datei an den Parameter zu übergeben. `--cli-input-json` Weitere Informationen finden Sie unter Erstellen einer Pipeline (CLI) im AWS CodePipeline Benutzerhandbuch.

Batch-Build mit kombinierten Artefakten

Verwenden Sie die folgende JSON-Datei als Beispiel für eine Pipeline-Struktur, die einen Batch-Build mit kombinierten Artefakten erstellt. Um Batch-Builds zu aktivieren CodePipeline, setzen Sie den `BatchEnabled` Parameter des `configuration` Objekts auf`true`. Um die Build-Artefakte an derselben Stelle zu kombinieren, setzen Sie den `CombineArtifacts` Parameter des `configuration` Objekts auf`true`.

```
{
 "pipeline": {
  "roleArn": "arn:aws:iam::account-id:role/my-AWS-CodePipeline-service-role-name",
  "stages": [
    {
      "name": "Source",
      "actions": [
        {
```

```
          "inputArtifacts": [],
          "name": "Source1",
          "actionTypeId": {
            "category": "Source",
            "owner": "AWS",
            "version": "1",
            "provider": "S3"
          },
          "outputArtifacts": [
            {
              "name": "source1"
            }
          ],
          "configuration": {
            "S3Bucket": "<my-input-bucket-name>",
            "S3ObjectKey": "my-source-code-file-name.zip"
          },
          "runOrder": 1
        },
        {
          "inputArtifacts": [],
          "name": "Source2",
          "actionTypeId": {
            "category": "Source",
            "owner": "AWS",
            "version": "1",
            "provider": "S3"
          },
          "outputArtifacts": [
            {
              "name": "source2"
            }
          ],
          "configuration": {
            "S3Bucket": "<my-other-input-bucket-name>",
            "S3ObjectKey": "my-other-source-code-file-name.zip"
          },
          "runOrder": 1
        }
      ]
    },
    {
      "name": "Build",
      "actions": [
```

```
        {
          "inputArtifacts": [
            {
              "name": "source1"
            },
            {
              "name": "source2"
            }
          ],
          "name": "Build",
          "actionTypeId": {
            "category": "Build",
            "owner": "AWS",
            "version": "1",
            "provider": "CodeBuild"
          },
          "outputArtifacts": [
            {
              "name": "output1 "
            }
          ],
          "configuration": {
            "ProjectName": "my-build-project-name",
            "PrimarySource": "source1",
             "BatchEnabled": "true",
             "CombineArtifacts": "true"
          },
          "runOrder": 1
        }
      ]
    }
  ],
  "artifactStore": {
    "type": "S3",
    "location": "<AWS-CodePipeline-internal-bucket-name>"
  },
  "name": "my-pipeline-name",
  "version": 1
 }
}
```

Im Folgenden finden Sie ein Beispiel für eine CodeBuild Buildspec-Datei, die mit dieser Pipeline-Konfiguration funktioniert.

```
version: 0.2
batch:
  build-list:
    - identifier: build1
      env:
        compute-type: BUILD_GENERAL1_SMALL
    - identifier: build2
      env:
        compute-type: BUILD_GENERAL1_MEDIUM

phases:
  build:
    commands:
      - echo 'file' > output_file

artifacts:
  files:
    - output_file
```

Wenn kombinierte Artefakte für den Batch-Build aktiviert sind, ist nur eine Ausgabe zulässig. CodeBuild kombiniert die primären Artefakte aller Builds in einer einzigen ZIP-Datei.

Nachdem Sie die JSON-Datei erstellt haben, können Sie die Pipeline erstellen. Verwenden Sie den AWS CLI , um den Befehl create-pipeline auszuführen und die Datei an den Parameter zu übergeben. `--cli-input-json` Weitere Informationen finden Sie unter Erstellen einer Pipeline (CLI) im AWS CodePipeline Benutzerhandbuch.

AWS CodePipeline Beispiel für Integration mit CodeBuild und mehrere Eingabequellen und Ausgabeartefakte

Ein AWS CodeBuild Projekt kann mehr als eine Eingabequelle verwenden. Es kann zudem mehr als ein Ausgabeartefakt erstellen. Dieses Beispiel zeigt, wie Sie AWS CodePipeline ein Build-Projekt erstellen, das mehrere Eingabequellen verwendet, um mehrere Ausgabeartefakte zu erstellen. Weitere Informationen finden Sie unter Beispiel für mehrere Eingabequellen und Ausgabeartefakte.

Sie können eine Datei im JSON-Format verwenden, die die Struktur Ihrer Pipeline definiert, und sie dann zusammen mit der verwenden, AWS CLI um die Pipeline zu erstellen. Verwenden Sie die folgende JSON-Datei als Beispiel für eine Pipeline-Struktur, die einen Build mit mehr als einer Eingabequelle und mehr als einem Ausgabeartefakt erstellt. Unten in diesem Beispiel werden Sie sehen, wie diese Datei mehrere Eingaben und Ausgaben angibt. Weitere Informationen finden Sie in der Referenz zur CodePipeline Pipeline-Struktur im AWS CodePipeline Benutzerhandbuch.

```
{
 "pipeline": {
  "roleArn": "arn:aws:iam::account-id:role/my-AWS-CodePipeline-service-role-name",
  "stages": [
    {
      "name": "Source",
      "actions": [
        {
          "inputArtifacts": [],
          "name": "Source1",
          "actionTypeId": {
            "category": "Source",
            "owner": "AWS",
            "version": "1",
            "provider": "S3"
          },
          "outputArtifacts": [
            {
              "name": "source1"
            }
          ],
          "configuration": {
            "S3Bucket": "my-input-bucket-name",
            "S3ObjectKey": "my-source-code-file-name.zip"
          },
          "runOrder": 1
        },
        {
          "inputArtifacts": [],
          "name": "Source2",
          "actionTypeId": {
            "category": "Source",
            "owner": "AWS",
            "version": "1",
            "provider": "S3"
          },
          "outputArtifacts": [
            {
              "name": "source2"
            }
          ],
          "configuration": {
            "S3Bucket": "my-other-input-bucket-name",
```

```
            "S3ObjectKey": "my-other-source-code-file-name.zip"
          },
          "runOrder": 1
        }
      ]
    },
    {
      "name": "Build",
      "actions": [
        {
          "inputArtifacts": [
            {
              "name": "source1"
            },
            {
              "name": "source2"
            }
          ],
          "name": "Build",
          "actionTypeId": {
            "category": "Build",
            "owner": "AWS",
            "version": "1",
            "provider": "AWS CodeBuild"
          },
          "outputArtifacts": [
            {
              "name": "artifact1"
            },
            {
              "name": "artifact2"
            }
          ],
          "configuration": {
            "ProjectName": "my-build-project-name",
            "PrimarySource": "source1"
          },
          "runOrder": 1
        }
      ]
    }
  ],
  "artifactStore": {
    "type": "S3",
```

```
    "location": "AWS-CodePipeline-internal-bucket-name"
  },
  "name": "my-pipeline-name",
  "version": 1
 }
}
```

Für diese JSON-Datei gilt:

- Eine der Eingabequellen muss als `PrimarySource` fungieren. Diese Quelle ist das Verzeichnis, in dem CodeBuild nach Ihrer Buildspec-Datei gesucht und diese ausgeführt wird. Das Schlüsselwort `PrimarySource` wird verwendet, um die Primärquelle im `configuration` Abschnitt der CodeBuild Phase in der JSON-Datei anzugeben.
- Jede Eingabequelle ist in einem eigenen Verzeichnis installiert. Dieses Verzeichnis ist in der integrierten Umgebungsvariable `$CODEBUILD_SRC_DIR` für die primäre Quelle und `$CODEBUILD_SRC_DIR_yourInputArtifactName` für alle anderen Quellen gespeichert. Für die Pipeline in diesem Beispiel lauten die beiden Eingabequellverzeichnisse `$CODEBUILD_SRC_DIR` und `$CODEBUILD_SRC_DIR_source2`. Weitere Informationen finden Sie unter Umgebungsvariablen in Build-Umgebungen.
- Die in der JSON-Datei der Pipeline angegebenen Namen der Ausgabeartefakte müssen den Namen der sekundären Artefakte entsprechen, die in der buildspec-Datei definiert sind. Diese Pipeline verwendet die folgende buildspec-Datei. Weitere Informationen finden Sie unter Syntax der Build-Spezifikation.

```
version: 0.2

phases:
  build:
    commands:
      - touch source1_file
      - cd $CODEBUILD_SRC_DIR_source2
      - touch source2_file

artifacts:
  files:
    - '**/*'
  secondary-artifacts:
    artifact1:
      base-directory: $CODEBUILD_SRC_DIR
```

```
    files:
      - source1_file
  artifact2:
    base-directory: $CODEBUILD_SRC_DIR_source2
    files:
      - source2_file
```

Nachdem Sie die JSON-Datei erstellt haben, können Sie die Pipeline erstellen. Verwenden Sie den AWS CLI , um den Befehl create-pipeline auszuführen und die Datei an den Parameter zu übergeben. `--cli-input-json` Weitere Informationen finden Sie unter Erstellen einer Pipeline (CLI) im AWS CodePipeline Benutzerhandbuch.

AWS Config Mit CodeBuild Beispiel verwenden

AWS Config bietet eine Bestandsaufnahme Ihrer AWS Ressourcen und einen Verlauf der Konfigurationsänderungen an diesen Ressourcen. AWS Config wird jetzt AWS CodeBuild als AWS Ressource unterstützt, was bedeutet, dass der Dienst Ihre CodeBuild Projekte verfolgen kann. Weitere Informationen zu AWS Config finden Sie unter Was ist AWS Config? im AWS Config Entwicklerhandbuch.

Auf der Seite „Ressourceninventar“ in der AWS Config Konsole finden Sie die folgenden Informationen zu CodeBuild Ressourcen:

- Eine Zeitleiste Ihrer CodeBuild Konfigurationsänderungen.
- Konfigurationsdetails für jedes CodeBuild Projekt.
- Beziehungen zu anderen AWS Ressourcen.
- Eine Liste der Änderungen an Ihren CodeBuild Projekten.

Die Verfahren in diesem Thema zeigen Ihnen, wie Sie CodeBuild Projekte einrichten, nachschlagen AWS Config und anzeigen.

Themen

- Voraussetzungen
- Richten Sie ein AWS Config
- Suchen Sie nach AWS CodeBuild Projekten
- AWS CodeBuild Konfigurationsdetails in der AWS Config Konsole anzeigen

Voraussetzungen

Erstellen Sie Ihr AWS CodeBuild Projekt. Anweisungen finden Sie unter Erstellen eines Build-Projekts.

Richten Sie ein AWS Config

- Einrichtung von AWS Config (Konsole)
- Einrichten AWS Config (AWS CLI)

> **Note**
>
> Nach Abschluss der Einrichtung kann es bis zu 10 Minuten dauern, bis AWS CodeBuild Projekte in der AWS Config Konsole angezeigt werden.

Suchen Sie nach AWS CodeBuild Projekten

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die AWS Config Konsole unter https://console.aws.amazon.com/config.
2. Wählen Sie auf der Seite Ressourceninventar unter Ressourcentyp die Option AWS CodeBuild Projekt aus. Scrollen Sie nach unten und aktivieren Sie das CodeBuild Projekt-Kontrollkästchen.
3. Wählen Sie Look up.
4. Nachdem die CodeBuild Projektliste hinzugefügt wurde, wählen Sie in der Spalte Config Timeline den Link mit dem CodeBuild Projektnamen aus.

AWS CodeBuild Konfigurationsdetails in der AWS Config Konsole anzeigen

Wenn Sie auf der Seite „Ressourcenbestand“ nach Ressourcen suchen, können Sie den AWS Config Zeitplan auswählen, um Details zu Ihrem CodeBuild Projekt anzuzeigen. Die Detailseite für eine Ressource bietet Informationen über die Konfiguration, die Beziehungen und die Anzahl der vorgenommenen Änderungen dieser Ressource.

Die Blöcke oben auf der Seite werden zusammengefasst als Timeline bezeichnet. Die Timeline zeigt das Datum und die Uhrzeit der Aufzeichnung.

Weitere Informationen finden Sie im AWS Config Entwicklerhandbuch unter Konfigurationsdetails in der AWS Config Konsole anzeigen.

Beispiel für Benachrichtigungen erstellen für CodeBuild

Amazon CloudWatch Events bietet integrierte Unterstützung für AWS CodeBuild. CloudWatch Events ist ein Stream von Systemereignissen, die Änderungen an Ihren AWS Ressourcen beschreiben. Mit CloudWatch Ereignissen schreiben Sie deklarative Regeln, um interessante Ereignisse mit zu ergreifenden automatisierten Aktionen zu verknüpfen. In diesem Beispiel werden Amazon CloudWatch Events und Amazon Simple Notification Service (Amazon SNS) verwendet, um Build-Benachrichtigungen an Abonnenten zu senden, wenn Builds erfolgreich sind, fehlschlagen, von einer Build-Phase zur nächsten wechseln oder wenn eine Kombination dieser Ereignisse vorliegt.

> **Important**
>
> Die Ausführung dieses Beispiels kann dazu führen, dass Ihr AWS Konto belastet wird. Dazu gehören mögliche Gebühren für CodeBuild und für AWS Ressourcen und Aktionen im Zusammenhang mit Amazon CloudWatch und Amazon SNS. Weitere Informationen finden Sie unter CodeBuild Preise, CloudWatchAmazon-Preise und Amazon SNS-Preise.

Ausführen des Beispiels

So führen Sie das Beispiel aus

1. Wenn Sie in Amazon SNS bereits ein Thema eingerichtet und abonniert haben, das Sie für dieses Beispiel verwenden möchten, fahren Sie mit Schritt 4 fort. Andernfalls, wenn Sie einen IAM-Benutzer anstelle eines AWS Root-Kontos oder eines Administratorbenutzers für die Arbeit mit Amazon SNS verwenden, fügen Sie dem Benutzer (oder der IAM-Gruppe, der der Benutzer zugeordnet ist) die folgende *Anweisung (zwischen ### BEGIN ADDING STATEMENT HERE ### und ### END ADDING STATEMENT HERE ###) hinzu*. Die Verwendung eines Root-Kontos wird nicht empfohlen. AWS Diese Erklärung ermöglicht das Anzeigen, Erstellen, Abonnieren und Testen des Versands von Benachrichtigungen zu Themen in Amazon SNS. Auslassungspunkte (`...`) werden zur Abkürzung verwendet und weisen auf die Stellen hin, an denen die Anweisung hinzugefügt wird. Entfernen Sie keine Anweisungen und geben Sie die Auslassungspunkte nicht in die vorhandene Richtlinie ein.

```
{
  "Statement": [
    ### BEGIN ADDING STATEMENT HERE ###
    {
      "Action": [
```

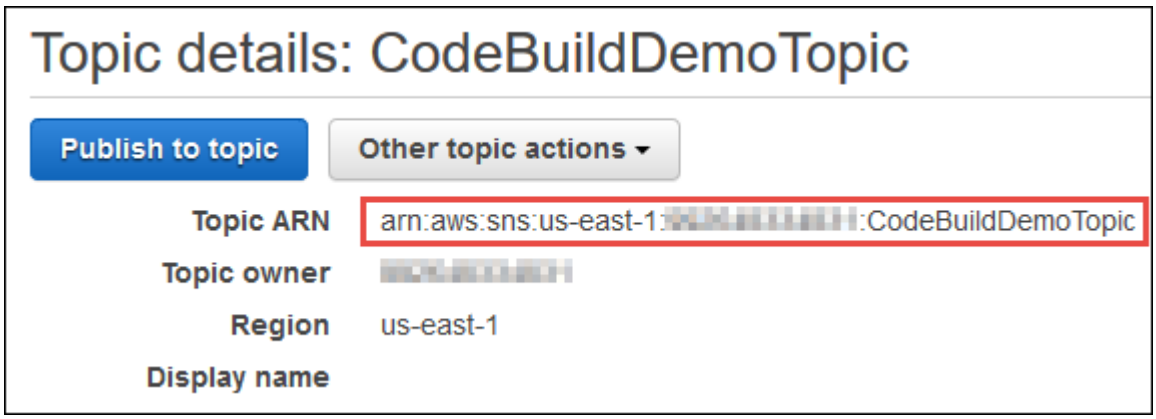
```
        "sns:CreateTopic",
        "sns:GetTopicAttributes",
        "sns:List*",
        "sns:Publish",
        "sns:SetTopicAttributes",
        "sns:Subscribe"
      ],
      "Resource": "*",
      "Effect": "Allow"
    },
    ### END ADDING STATEMENT HERE ###
    ...
  ],
  "Version": "2012-10-17"
}
```

> **Note**
>
> Die IAM-Entität, die diese Richtlinie ändert, muss in IAM über die Erlaubnis verfügen, Richtlinien zu ändern.
> Weitere Informationen finden Sie unter Vom Kunden verwaltete Richtlinien bearbeiten oder im Abschnitt „So bearbeiten oder löschen Sie eine Inline-Richtlinie für eine Gruppe, einen Benutzer oder eine Rolle“ unter Arbeiten mit Inline-Richtlinien (Konsole) im IAM-Benutzerhandbuch.

2. Erstellen oder identifizieren Sie ein Thema in Amazon SNS. AWS CodeBuild verwendet CloudWatch Events, um Build-Benachrichtigungen zu diesem Thema über Amazon SNS zu senden.

   Erstellen Sie ein Thema wie folgt:

   1. Öffnen Sie die Amazon SNS SNS-Konsole unter https://console.aws.amazon.com/sns.
   2. Wählen Sie Thema erstellen aus.
   3. Geben Sie unter Create new topic (Neues Thema erstellen) für Topic name (Themenname) einen Namen für das Thema ein (z. B. **`CodeBuildDemoTopic`**). (Wenn Sie einen anderen Namen verwenden, muss dieser im gesamten Beispiel verwendet werden.)
   4. Wählen Sie Thema erstellen aus.
   5. Kopieren Sie auf der CodeBuildDemoTopic Seite Themendetails: den ARN-Wert für das Thema. Sie benötigen diesen Wert im nächsten Schritt.

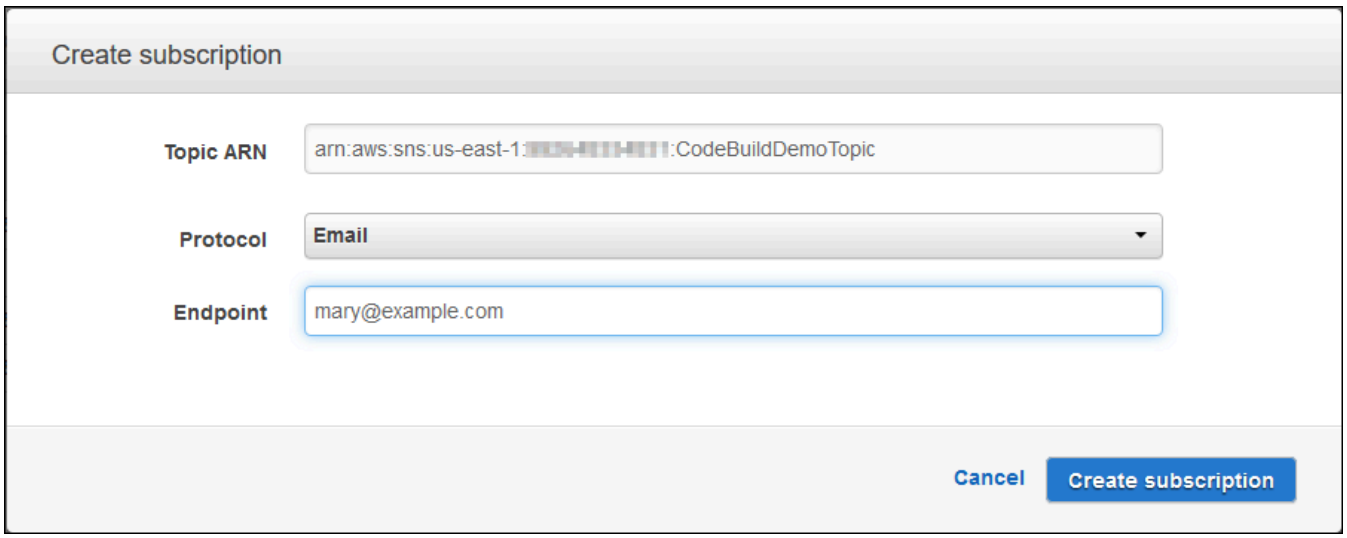


Weitere Informationen finden Sie unter Thema erstellen im Amazon SNS SNS-Entwicklerhandbuch.

3. Abonnieren Sie das Thema für einen oder mehrere Empfänger, um E-Mail-Benachrichtigungen zu empfangen.

   So abonnieren Sie ein Thema für einen Empfänger:

   1. Wählen Sie bei geöffneter Amazon SNS SNS-Konsole aus dem vorherigen Schritt im Navigationsbereich Abonnements und anschließend Abonnement erstellen aus.
   2. Fügen Sie unter Create subscription (Abonnement erstellen), für Topic ARN (Thema-ARN), den Thema-ARN ein, den Sie im vorherigen Schritt kopiert haben.
   3. Wählen Sie unter Protocol (Protokoll) die Option Email (E-Mail) aus.
   4. Geben Sie für Endpoint (Endpunkt) die vollständige E-Mail-Adresse des Empfängers ein.



5. Klicken Sie auf Create subscription (Abonnement erstellen).
6. Amazon SNS sendet eine Bestätigungs-E-Mail für das Abonnement an den Empfänger. Um Benachrichtigungen zu erhalten, muss der Empfänger den Link Confirm subscription in der Bestätigungs-E-Mail wählen. Nachdem der Empfänger auf den Link geklickt hat, zeigt Amazon SNS bei erfolgreichem Abonnement eine Bestätigungsnachricht im Webbrowser des Empfängers an.

Weitere Informationen finden Sie unter Abonnieren eines Themas im Amazon SNS Developer Guide.

4. Wenn Sie einen Benutzer anstelle eines AWS Root-Kontos oder eines Administrator-Benutzers für die Arbeit mit CloudWatch Events verwenden, fügen Sie dem Benutzer (oder der IAM-Gruppe, der der Benutzer zugeordnet ist) die folgende *Anweisung hinzu (zwischen ### BEGIN* ADDING *STATEMENT HERE ### und ### END ADDING STATEMENT HERE ###).* Die Verwendung eines AWS Root-Kontos wird nicht empfohlen. Diese Anweisung wird verwendet, um dem Benutzer die Arbeit mit CloudWatch Ereignissen zu ermöglichen. Auslassungspunkte (`...`) werden zur Abkürzung verwendet und weisen auf die Stellen hin, an denen die Anweisung hinzugefügt wird. Entfernen Sie keine Anweisungen und geben Sie die Auslassungspunkte nicht in die vorhandene Richtlinie ein.

```
{
  "Statement": [
    ### BEGIN ADDING STATEMENT HERE ###
    {
      "Action": [
```

```
        "events:*",
        "iam:PassRole"
      ],
      "Resource": "*",
      "Effect": "Allow"
    },
    ### END ADDING STATEMENT HERE ###
    ...
  ],
  "Version": "2012-10-17"
}
```

> **Note**
>
> Die IAM-Entität, die diese Richtlinie ändert, muss in IAM über die Berechtigung zum Ändern von Richtlinien verfügen.
> Weitere Informationen finden Sie unter Vom Kunden verwaltete Richtlinien bearbeiten oder im Abschnitt „So bearbeiten oder löschen Sie eine Inline-Richtlinie für eine Gruppe, einen Benutzer oder eine Rolle“ unter Arbeiten mit Inline-Richtlinien (Konsole) im IAM-Benutzerhandbuch.

5. Erstellen Sie eine Regel unter CloudWatch Ereignisse. Öffnen Sie dazu die CloudWatch Konsole unter https://console.aws.amazon.com/cloudwatch.
6. Wählen Sie im Navigationsbereich unter Events Rules aus und anschließend Create rule.
7. Auf der Seite Step 1: Create rule page (Schritt 1: Regel erstellen), sollten bereits die Optionen Event Pattern (Ereignismuster) und Build event pattern to match events by service (Ereignismuster erstellen, um Ereignisse nach Service abzugleichen) ausgewählt sein.
8. Wählen Sie für Servicename CodeBuild aus. Bei Event Type (Ereignistyp) sollte bereits All Events (Alle Ereignisse) ausgewählt sein.
9. Der folgende Code sollte in Event Pattern Preview (Ereignismustervorschau) angezeigt werden:

```
{
  "source": [
    "aws.codebuild"
  ]
}
```

10. Wählen Sie die Option Edit (Bearbeiten) und ersetzen Sie den Code in Event Pattern Preview (Ereignismustervorschau) durch eines der beiden folgenden Regelmuster.

Das erste Regelmuster löst für die in AWS CodeBuild angegebenen Build-Projekte ein Ereignis aus, sobald ein Build gestartet oder abgeschlossen wird.

```
{
  "source": [
    "aws.codebuild"
  ],
  "detail-type": [
    "CodeBuild Build State Change"
  ],
  "detail": {
    "build-status": [
      "IN_PROGRESS",
      "SUCCEEDED",
      "FAILED",
      "STOPPED"
    ],
    "project-name": [
      "my-demo-project-1",
      "my-demo-project-2"
    ]
  }
}
```

Nehmen Sie in der vorhergehenden Regel die folgenden Code-Änderungen nach Bedarf vor.

- Um ein Ereignis auszulösen, sobald ein Build beginnt oder abgeschlossen ist, lassen Sie entweder alle Werte wie im `build-status` Array gezeigt stehen oder entfernen Sie `build-status` das Array vollständig.
- Um ein Ereignis nur dann auszulösen, wenn ein Build abgeschlossen ist, entfernen Sie `IN_PROGRESS` aus dem `build-status` Array.
- Um ein Ereignis nur dann auszulösen, wenn ein Build startet, entfernen Sie alle Werte außer `IN_PROGRESS` aus dem `build-status` Array.
- Um Ereignisse für alle Build-Projekte auszulösen, entfernen Sie das `project-name` Array vollständig.
- Um Ereignisse nur für einzelne Build-Projekte auszulösen, geben Sie den Namen des jeweiligen Build-Projekts im Array `project-name` an.

Dieses zweite Regelmuster löst ein Ereignis aus, sobald bei den in AWS CodeBuild angegebenen Build-Projekten ein Build von einer Build-Phase in eine andere wechselt.

```
{
  "source": [
    "aws.codebuild"
  ],
  "detail-type": [
    "CodeBuild Build Phase Change"
  ],
  "detail": {
    "completed-phase": [
      "SUBMITTED",
      "PROVISIONING",
      "DOWNLOAD_SOURCE",
      "INSTALL",
      "PRE_BUILD",
      "BUILD",
      "POST_BUILD",
      "UPLOAD_ARTIFACTS",
      "FINALIZING"
    ],
    "completed-phase-status": [
      "TIMED_OUT",
      "STOPPED",
      "FAILED",
      "SUCCEEDED",
      "FAULT",
      "CLIENT_ERROR"
    ],
    "project-name": [
      "my-demo-project-1",
      "my-demo-project-2"
    ]
  }
}
```

Nehmen Sie in der vorhergehenden Regel die folgenden Code-Änderungen nach Bedarf vor.

- Um ein Ereignis für jeden Build-Phasenwechsel auszulösen (wodurch bis zu neun Benachrichtigungen für jeden Build gesendet werden können), lassen Sie entweder alle

Werte, wie im `completed-phase`-Array gezeigt, stehen oder entfernen Sie das `completed-phase`-Array vollständig.

- Um Ereignisse nur für einzelne Build-Phasenänderungen auszulösen, entfernen Sie den Namen jeder Build-Phase in dem `completed-phase`-Array, für das Sie kein Ereignis auslösen möchten.
- Um ein Ereignis für jede Statusänderung einer Build-Phase auszulösen, lassen Sie entweder alle Werte, wie im `completed-phase-status`-Array gezeigt, stehen oder entfernen Sie das `completed-phase-status`-Array vollständig.
- Um Ereignisse nur für einzelne Statusänderungen von Build-Phasen auszulösen, entfernen Sie den Namen des jeweiligen Status einer Build-Phase in dem `completed-phase-status`-Array, für das Sie kein Ereignis auslösen möchten.
- Um Ereignisse für alle Build-Projekte auszulösen, entfernen Sie das `project-name`-Array.
- Um Ereignisse für einzelne Build-Projekte auszulösen, geben Sie den Namen des jeweiligen Build-Projekts im `project-name`-Array an.

Weitere Informationen zu Ereignismustern finden Sie unter Ereignismuster im EventBridge Amazon-Benutzerhandbuch.

Weitere Informationen zum Filtern mit Ereignismustern finden Sie unter Inhaltsbasiertes Filtern mit Ereignismustern im EventBridge Amazon-Benutzerhandbuch.

> **Note**
>
> Wenn Sie Ereignisse sowohl für Build-Statusänderungen als auch für Build-Phasenänderungen auslösen möchten, müssen Sie zwei separate Regeln erstellen: eine für Build-Statusänderungen und eine für Build-Phasenänderungen. Wenn Sie versuchen, beide Regeln zu einer einzigen Regel zusammenzufassen, führt die kombinierte Regel ggf. zu unerwarteten Ergebnissen oder funktioniert ggf. gar nicht mehr.

Wenn Sie mit dem Ersetzen des Codes fertig sind, wählen Sie Save (Speichern) aus.

11. Wählen Sie für Targets die Option Add target aus.
12. Wählen Sie in der Liste der Ziele SNS topic aus.
13. Als Topic wählen Sie das Thema, das Sie zuvor identifiziert oder erstellt haben.
14. Erweitern Sie Configure input und wählen Sie dann Input Transformer aus.

15. Geben Sie im Feld Input Path (Eingabepfad) einen der folgenden Eingabepfade ein.

    Geben Sie für eine Regel mit einem `detail-type`-Wert von `CodeBuild Build State Change` Folgendes ein.

    ```
    {"build-id":"$.detail.build-id","project-name":"$.detail.project-name","build-status":"$.detail.build-status"}
    ```

    Geben Sie für eine Regel mit einem `detail-type`-Wert von `CodeBuild Build Phase Change` Folgendes ein.

    ```
    {"build-id":"$.detail.build-id","project-name":"$.detail.project-name","completed-phase":"$.detail.completed-phase","completed-phase-status":"$.detail.completed-phase-status"}
    ```

    Weitere Informationen finden Sie unter Eingabeformat-Referenz für Build-Benachrichtigungen.

16. Geben Sie im Feld Input Template (Eingabevorlage) eine der folgenden Eingabevorlagen ein.

    Geben Sie für eine Regel mit einem `detail-type`-Wert von `CodeBuild Build State Change` Folgendes ein.

    ```
    "Build '<build-id>' for build project '<project-name>' has reached the build status of '<build-status>'."
    ```

    Geben Sie für eine Regel mit einem `detail-type`-Wert von `CodeBuild Build Phase Change` Folgendes ein.

    ```
    "Build '<build-id>' for build project '<project-name>' has completed the build phase of '<completed-phase>' with a status of '<completed-phase-status>'."
    ```

17. Wählen Sie Details konfigurieren.
18. Geben Sie auf der Seite Step 2: Configure rule details (Schritt 2: Konfigurieren von Regeldetails) einen Namen ein und eine optionale Beschreibung ein. Lassen Sie unter Status die Option Enabled (Aktiviert) ausgewählt.
19. Wählen Sie Regel erstellen aus.
20. Erstellen Sie Build-Projekte, führen Sie die Builds aus und zeigen Sie Build-Informationen an.

21. Bestätigen Sie, CodeBuild dass Build-Benachrichtigungen jetzt erfolgreich gesendet werden. Überprüfen Sie beispielsweise, ob sich die Build-Benachrichtigungs-E-Mails jetzt in Ihrem Posteingang befinden.

Um das Verhalten einer Regel zu ändern, wählen Sie in der CloudWatch Konsole die Regel aus, die Sie ändern möchten, klicken Sie auf Aktionen und dann auf Bearbeiten. Nehmen Sie Änderungen an der Regel vor, wählen Sie Configure details (Details konfigurieren), und klicken Sie dann auf Update rule (Regel aktualisieren).

Um eine Regel nicht mehr zum Senden von Build-Benachrichtigungen zu verwenden, wählen Sie in der CloudWatch Konsole die Regel aus, die Sie nicht mehr verwenden möchten, wählen Sie Aktionen und dann Deaktivieren aus.

Um eine Regel vollständig zu löschen, wählen Sie in der CloudWatch Konsole die Regel aus, die Sie löschen möchten, wählen Sie Aktionen und dann Löschen aus.

Zugehörige Ressourcen

- Informationen zu den ersten Schritten finden Sie unterErste Schritte mit AWS CodeBuild unter Verwendung der Konsole. AWS CodeBuild
- Informationen zur Behebung von Problemen finden Sie unterFehlerbehebung AWS CodeBuild. CodeBuild
- Informationen zu Kontingenten in CodeBuild finden Sie unterKontingente für AWS CodeBuild.

Eingabeformat-Referenz für Build-Benachrichtigungen

CloudWatch übermittelt Benachrichtigungen im JSON-Format.

Benachrichtigungen zu Statusänderungen von Builds verwenden das folgende Format:

```
{
  "version": "0",
  "id": "c030038d-8c4d-6141-9545-00ff7b7153EX",
  "detail-type": "CodeBuild Build State Change",
  "source": "aws.codebuild",
  "account": "123456789012",
  "time": "2017-09-01T16:14:28Z",
  "region": "us-west-2",
  "resources":[
```

```
    "arn:aws:codebuild:us-west-2:123456789012:build/my-sample-project:8745a7a9-
c340-456a-9166-edf953571bEX"
  ],
  "detail":{
    "build-status": "SUCCEEDED",
    "project-name": "my-sample-project",
    "build-id": "arn:aws:codebuild:us-west-2:123456789012:build/my-sample-
project:8745a7a9-c340-456a-9166-edf953571bEX",
    "additional-information": {
      "artifact": {
        "md5sum": "da9c44c8a9a3cd4b443126e823168fEX",
        "sha256sum":
 "6ccc2ae1df9d155ba83c597051611c42d60e09c6329dcb14a312cecc0a8e39EX",
        "location": "arn:aws:s3:::codebuild-123456789012-output-bucket/my-output-
artifact.zip"
      },
      "environment": {
        "image": "aws/codebuild/standard:5.0",
        "privileged-mode": false,
        "compute-type": "BUILD_GENERAL1_SMALL",
        "type": "LINUX_CONTAINER",
        "environment-variables": []
      },
      "timeout-in-minutes": 60,
      "build-complete": true,
      "initiator": "MyCodeBuildDemoUser",
      "build-start-time": "Sep 1, 2017 4:12:29 PM",
      "source": {
        "location": "codebuild-123456789012-input-bucket/my-input-artifact.zip",
        "type": "S3"
      },
      "logs": {
        "group-name": "/aws/codebuild/my-sample-project",
        "stream-name": "8745a7a9-c340-456a-9166-edf953571bEX",
        "deep-link": "https://console.aws.amazon.com/cloudwatch/home?region=us-
west-2#logEvent:group=/aws/codebuild/my-sample-project;stream=8745a7a9-c340-456a-9166-
edf953571bEX"
      },
      "phases": [
        {
          "phase-context": [],
          "start-time": "Sep 1, 2017 4:12:29 PM",
          "end-time": "Sep 1, 2017 4:12:29 PM",
          "duration-in-seconds": 0,
```

```
      "phase-type": "SUBMITTED",
      "phase-status": "SUCCEEDED"
    },
    {
      "phase-context": [],
      "start-time": "Sep 1, 2017 4:12:29 PM",
      "end-time": "Sep 1, 2017 4:13:05 PM",
      "duration-in-seconds": 36,
      "phase-type": "PROVISIONING",
      "phase-status": "SUCCEEDED"
    },
    {
      "phase-context": [],
      "start-time": "Sep 1, 2017 4:13:05 PM",
      "end-time": "Sep 1, 2017 4:13:10 PM",
      "duration-in-seconds": 4,
      "phase-type": "DOWNLOAD_SOURCE",
      "phase-status": "SUCCEEDED"
    },
    {
      "phase-context": [],
      "start-time": "Sep 1, 2017 4:13:10 PM",
      "end-time": "Sep 1, 2017 4:13:10 PM",
      "duration-in-seconds": 0,
      "phase-type": "INSTALL",
      "phase-status": "SUCCEEDED"
    },
    {
      "phase-context": [],
      "start-time": "Sep 1, 2017 4:13:10 PM",
      "end-time": "Sep 1, 2017 4:13:10 PM",
      "duration-in-seconds": 0,
      "phase-type": "PRE_BUILD",
      "phase-status": "SUCCEEDED"
    },
    {
      "phase-context": [],
      "start-time": "Sep 1, 2017 4:13:10 PM",
      "end-time": "Sep 1, 2017 4:14:21 PM",
      "duration-in-seconds": 70,
      "phase-type": "BUILD",
      "phase-status": "SUCCEEDED"
    },
    {
```

```
          "phase-context": [],
          "start-time": "Sep 1, 2017 4:14:21 PM",
          "end-time": "Sep 1, 2017 4:14:21 PM",
          "duration-in-seconds": 0,
          "phase-type": "POST_BUILD",
          "phase-status": "SUCCEEDED"
        },
        {
          "phase-context": [],
          "start-time": "Sep 1, 2017 4:14:21 PM",
          "end-time": "Sep 1, 2017 4:14:21 PM",
          "duration-in-seconds": 0,
          "phase-type": "UPLOAD_ARTIFACTS",
          "phase-status": "SUCCEEDED"
        },
         {
          "phase-context": [],
          "start-time": "Sep 1, 2017 4:14:21 PM",
          "end-time": "Sep 1, 2017 4:14:26 PM",
          "duration-in-seconds": 4,
          "phase-type": "FINALIZING",
          "phase-status": "SUCCEEDED"
        },
        {
          "start-time": "Sep 1, 2017 4:14:26 PM",
          "phase-type": "COMPLETED"
        }
      ]
    },
    "current-phase": "COMPLETED",
    "current-phase-context": "[]",
    "version": "1"
  }
}
```

Benachrichtigungen zu Phasenänderungen von Builds verwenden das folgende Format:

```
{
  "version": "0",
  "id": "43ddc2bd-af76-9ca5-2dc7-b695e15adeEX",
  "detail-type": "CodeBuild Build Phase Change",
  "source": "aws.codebuild",
  "account": "123456789012",
```

```
  "time": "2017-09-01T16:14:21Z",
  "region": "us-west-2",
  "resources":[
    "arn:aws:codebuild:us-west-2:123456789012:build/my-sample-project:8745a7a9-
c340-456a-9166-edf953571bEX"
  ],
  "detail":{
    "completed-phase": "COMPLETED",
    "project-name": "my-sample-project",
    "build-id": "arn:aws:codebuild:us-west-2:123456789012:build/my-sample-
project:8745a7a9-c340-456a-9166-edf953571bEX",
    "completed-phase-context": "[]",
    "additional-information": {
      "artifact": {
        "md5sum": "da9c44c8a9a3cd4b443126e823168fEX",
        "sha256sum":
 "6ccc2ae1df9d155ba83c597051611c42d60e09c6329dcb14a312cecc0a8e39EX",
        "location": "arn:aws:s3:::codebuild-123456789012-output-bucket/my-output-
artifact.zip"
      },
      "environment": {
        "image": "aws/codebuild/standard:5.0",
        "privileged-mode": false,
        "compute-type": "BUILD_GENERAL1_SMALL",
        "type": "LINUX_CONTAINER",
        "environment-variables": []
      },
      "timeout-in-minutes": 60,
      "build-complete": true,
      "initiator": "MyCodeBuildDemoUser",
      "build-start-time": "Sep 1, 2017 4:12:29 PM",
      "source": {
        "location": "codebuild-123456789012-input-bucket/my-input-artifact.zip",
        "type": "S3"
      },
      "logs": {
        "group-name": "/aws/codebuild/my-sample-project",
        "stream-name": "8745a7a9-c340-456a-9166-edf953571bEX",
        "deep-link": "https://console.aws.amazon.com/cloudwatch/home?region=us-
west-2#logEvent:group=/aws/codebuild/my-sample-project;stream=8745a7a9-c340-456a-9166-
edf953571bEX"
      },
      "phases": [
        {
```

```
      "phase-context": [],
      "start-time": "Sep 1, 2017 4:12:29 PM",
      "end-time": "Sep 1, 2017 4:12:29 PM",
      "duration-in-seconds": 0,
      "phase-type": "SUBMITTED",
      "phase-status": "SUCCEEDED"
    },
    {
      "phase-context": [],
      "start-time": "Sep 1, 2017 4:12:29 PM",
      "end-time": "Sep 1, 2017 4:13:05 PM",
      "duration-in-seconds": 36,
      "phase-type": "PROVISIONING",
      "phase-status": "SUCCEEDED"
    },
    {
      "phase-context": [],
      "start-time": "Sep 1, 2017 4:13:05 PM",
      "end-time": "Sep 1, 2017 4:13:10 PM",
      "duration-in-seconds": 4,
      "phase-type": "DOWNLOAD_SOURCE",
      "phase-status": "SUCCEEDED"
    },
    {
      "phase-context": [],
      "start-time": "Sep 1, 2017 4:13:10 PM",
      "end-time": "Sep 1, 2017 4:13:10 PM",
      "duration-in-seconds": 0,
      "phase-type": "INSTALL",
      "phase-status": "SUCCEEDED"
    },
    {
      "phase-context": [],
      "start-time": "Sep 1, 2017 4:13:10 PM",
      "end-time": "Sep 1, 2017 4:13:10 PM",
      "duration-in-seconds": 0,
      "phase-type": "PRE_BUILD",
      "phase-status": "SUCCEEDED"
    },
    {
      "phase-context": [],
      "start-time": "Sep 1, 2017 4:13:10 PM",
      "end-time": "Sep 1, 2017 4:14:21 PM",
      "duration-in-seconds": 70,
```

```
          "phase-type": "BUILD",
          "phase-status": "SUCCEEDED"
        },
        {
          "phase-context": [],
          "start-time": "Sep 1, 2017 4:14:21 PM",
          "end-time": "Sep 1, 2017 4:14:21 PM",
          "duration-in-seconds": 0,
          "phase-type": "POST_BUILD",
          "phase-status": "SUCCEEDED"
        },
        {
          "phase-context": [],
          "start-time": "Sep 1, 2017 4:14:21 PM",
          "end-time": "Sep 1, 2017 4:14:21 PM",
          "duration-in-seconds": 0,
          "phase-type": "UPLOAD_ARTIFACTS",
          "phase-status": "SUCCEEDED"
        },
        {
          "phase-context": [],
          "start-time": "Sep 1, 2017 4:14:21 PM",
          "end-time": "Sep 1, 2017 4:14:26 PM",
          "duration-in-seconds": 4,
          "phase-type": "FINALIZING",
          "phase-status": "SUCCEEDED"
        },
        {
          "start-time": "Sep 1, 2017 4:14:26 PM",
          "phase-type": "COMPLETED"
        }
      ]
    },
    "completed-phase-status": "SUCCEEDED",
    "completed-phase-duration-seconds": 4,
    "version": "1",
    "completed-phase-start": "Sep 1, 2017 4:14:21 PM",
    "completed-phase-end": "Sep 1, 2017 4:14:26 PM"
  }
}
```

Beispiel für Badges erstellen mit CodeBuild

AWS CodeBuild unterstützt jetzt die Verwendung von Build-Badges, die ein integrierbares, dynamisch generiertes Bild (Badge) bereitstellen, das den Status des letzten Builds für ein Projekt anzeigt. Auf dieses Bild kann über eine öffentlich verfügbare URL zugegriffen werden, die für Ihr Projekt generiert wurde. CodeBuild Auf diese Weise kann jeder den Status eines CodeBuild Projekts einsehen. Build Badges enthalten keine Sicherheitsinformationen, sodass keine Authentifizierung erforderlich ist.

Erstellen eines Build-Projekts mit aktivierten Build Badges (Konsole)

1. Öffnen Sie die AWS CodeBuild Konsole unter https://console.aws.amazon.com/codesuite/codebuild/home.
2. Wenn eine CodeBuild Informationsseite angezeigt wird, wählen Sie Create build project aus. Erweitern Sie andernfalls im Navigationsbereich Build, wählen Sie Build projects und dann Create build project aus.
3. Geben Sie unter Project name (Projektname) einen Namen für dieses Build-Projekt ein. Die Namen von Build-Projekten müssen für jedes AWS Konto eindeutig sein. Sie können auch eine optionale Beschreibung des Build-Projekts hinzufügen, damit andere Benutzer verstehen, wofür dieses Projekt verwendet wird.
4. Wählen Sie unter Source (Quelle) für Source provider (Quellanbieter) den Quellcode-Anbietertyp aus und führen Sie dann einen der folgenden Schritte aus:

   > **Note**
   >
   > CodeBuild unterstützt keine Build-Badges mit dem Amazon S3 S3-Quellanbieter. Da Amazon S3 für Artefaktübertragungen AWS CodePipeline verwendet wird, werden Build-Badges nicht für Build-Projekte unterstützt, die Teil einer Pipeline sind, die in erstellt wurde. CodePipeline

   - Wenn Sie wählen CodeCommit, wählen Sie für Repository den Namen des Repositorys. Wählen Sie Enable build badge (Build Badge aktivieren) aus, damit der Build-Status Ihres Projekts sichtbar und integrierbar ist.
   - Wenn Sie möchten GitHub, folgen Sie den Anweisungen, um eine Verbindung herzustellen (oder die Verbindung erneut herzustellen). GitHub Wählen Sie auf der Seite Anwendung GitHub autorisieren für Organisationszugriff neben jedem Repository, auf das Sie zugreifen

können möchten, AWS CodeBuild die Option Zugriff anfordern aus. Nach der Auswahl von Authorize application (Anwendung autorisieren) wählen Sie in der AWS CodeBuild -Konsole für Repository den Namen des Repositorys aus, der den Quellcode enthält. Wählen Sie Enable build badge (Build Badge aktivieren) aus, damit der Build-Status Ihres Projekts sichtbar und integrierbar ist.

- Wenn Sie Bitbucket ausgewählt haben, folgen Sie den Anweisungen zur Verbindung (oder erneuten Verbindung) mit Bitbucket. Wählen Sie auf der Bitbucket-Seite Confirm access to your account für Organization access die Option Grant access aus. Nachdem Sie Zugriff gewähren ausgewählt haben, wählen Sie in der AWS CodeBuild Konsole für Repository den Namen des Repositorys aus, das den Quellcode enthält. Wählen Sie Enable build badge (Build Badge aktivieren) aus, damit der Build-Status Ihres Projekts sichtbar und integrierbar ist.

> **Important**
>
> Wenn Sie Ihre Projektquelle aktualisieren, kann das Auswirkungen auf die Richtigkeit der Build Badges Ihres Projekts haben.

5. In Environment (Umgebung):

   Führen Sie für Environment image (Umgebungs-Image) einen der folgenden Schritte aus:

   - Um ein Docker-Image zu verwenden, das von verwaltet wird AWS CodeBuild, wählen Sie Verwaltetes Image aus und treffen Sie dann eine Auswahl unter Betriebssystem, Runtime (s), Image und Image-Version. Treffen Sie eine Auswahl unter Environment type (Umgebungstyp), sofern verfügbar.
   - Wenn Sie ein anderes Docker-Image verwenden möchten, wählen Sie Custom image (Benutzerdefiniertes Image) aus. Wählen Sie als Umgebungstyp ARM, Linux, Linux GPU oder Windows aus. Wenn Sie Andere Registrierung wählen, geben Sie für Externe Registrierungs-URL den Namen und das Tag des Docker-Images in Docker Hub ein. Verwenden Sie dabei das Format. *`docker repository`*/*`docker image name`* Wenn Sie sich für Amazon ECR entscheiden, verwenden Sie das Amazon ECR-Repository und das Amazon ECR-Image, um das Docker-Image in Ihrem Konto auszuwählen. AWS
   - Um ein privates Docker-Image zu verwenden, wählen Sie Benutzerdefiniertes Image. Wählen Sie als Umgebungstyp ARM, Linux, Linux GPU oder Windows aus. Wählen Sie unter Image registry (Abbildregistrierung) die Option Other registry (Andere Registrierung) aus und geben Sie dann den ARN der Anmeldeinformationen für Ihr privates Docker-Image ein. Die Anmeldeinformationen müssen von Secrets Manager erstellt werden. Weitere

Informationen finden Sie unter Was ist AWS Secrets Manager? im AWS Secrets Manager - Benutzerhandbuch.

6. Führen Sie unter Service role (Service-Rolle) einen der folgenden Schritte aus:

   - Wenn Sie keine CodeBuild Servicerolle haben, wählen Sie Neue Servicerolle. Geben Sie im Feld Rollenname einen Namen für die neue Rolle ein.
   - Wenn Sie eine CodeBuild Servicerolle haben, wählen Sie Bestehende Servicerolle aus. Wählen Sie unter Role ARN die Servicerolle aus.

   > **Note**
   >
   > Wenn Sie die Konsole verwenden, um ein Build-Projekt zu erstellen oder zu aktualisieren, können Sie gleichzeitig eine CodeBuild Servicerolle erstellen. In der Standardeinstellung funktioniert diese Rolle ausschließlich mit diesem Projekt. Wenn Sie die Konsole verwenden, um die Servicerolle mit einem anderen Build-Projekt zu verknüpfen, wird die Rolle so aktualisiert, dass sie mit dem anderen Build-Projekt funktioniert. Eine Servicerolle kann in bis zu zehn Build-Projekten verwendet werden.

7. Führen Sie in Buildspec einen der folgenden Schritte aus:

   - Wählen Sie Buildspec-Datei verwenden, um die Datei buildspec.yml im Quellcode-Stammverzeichnis zu verwenden.
   - Wählen Sie Build-Befehle einfügen, um die Konsole zum Einfügen von Build-Befehlen zu verwenden.

   Weitere Informationen hierzu finden Sie unter Build-Spezifikationsreferenz.

8. Führen Sie unter Artifacts (Artefakte) für Type (Typ) einen der folgenden Schritte aus:

   - Wenn keine Build-Ausgabeartefakte erstellt werden sollen, klicken Sie auf die Option No artifacts (Keine Artefakte).
   - Um die Build-Ausgabe in einem S3-Bucket zu speichern, wählen Sie Amazon S3 und gehen Sie dann wie folgt vor:
     - Lassen Sie Name leer, wenn Sie den Projektnamen für die ZIP-Datei mit der Build-Ausgabe verwenden möchten. Geben Sie andernfalls den Namen ein. Standardmäßig ist der Artefaktname der Projektname. Wenn Sie einen anderen Namen verwenden möchten,

geben Sie diesen in das Feld für den Artefaktnamen ein. Wenn Sie eine ZIP-Datei ausgeben möchten, schließen Sie die ZIP-Erweiterung mit ein.

- Wählen Sie für Bucket name den Namen des Ausgabe-Buckets aus.
- Wenn Sie in diesem Vorgang zuvor die Option Insert build commands (Build-Befehle einfügen) verwendet haben, geben Sie für Output files (Ausgabedateien) die Speicherorte der Build-Dateien ein, die in der ZIP-Datei oder in dem Ordner für die Build-Ausgabe enthalten sein sollen. Bei mehreren Speicherorten trennen Sie die einzelnen Speicherorte durch ein Komma, (wie z. B. `appspec.yml, target/my-app.jar`). Weitere Informationen finden Sie in der Beschreibung von `files` in Syntax der Build-Spezifikation.

9. Erweitern Sie Additional configuration (Zusätzliche Einstellungen) und wählen Sie die entsprechenden Optionen.
10. Wählen Sie Create build project (Build-Projekt erstellen) aus. Klicken Sie auf der Seite Review (Überprüfen) auf Start build (Build starten), um den Build auszuführen.

Erstellen eines Build-Projekts mit aktivierten Build Badges (CLI)

Informationen über das Erstellen eines Build-Projekts finden Sie unter Erstellen eines Build-Projekts (AWS CLI). Wenn Sie Build Badges in Ihr AWS CodeBuild -Projekt einschließen möchten, müssen Sie für *`badgeEnabled`* den Wert `true` angeben.

Greifen Sie auf Ihre AWS CodeBuild Build-Badges zu

Sie können die AWS CodeBuild Konsole oder die verwenden AWS CLI , um auf Build-Badges zuzugreifen.

- Wählen Sie in der CodeBuild Konsole in der Liste der Build-Projekte in der Spalte Name den Link aus, der dem Build-Projekt entspricht. Wählen Sie auf der Seite Build project: (Build-Projekt:) ***`Projektname`*** unter Configuration (Konfiguration) die Option Copy badge URL (Badge-URL kopieren) aus. Weitere Informationen finden Sie unter Anzeigen der Details eines Build-Projekts (Konsole).
- Führen Sie in der AWS CLI den `batch-get-projects` Befehl aus. Die Build Badge-URL ist im Abschnitt zu den Projektumgebungsdetails der Ausgabe enthalten. Weitere Informationen finden Sie unter Anzeigen der Details eines Build-Projekts (AWS CLI).

Die URL für die Build-Badge-Anfrage wird mit einem gemeinsamen Standard-Branch generiert. Sie können jedoch jeden Branch in Ihrem Quell-Repository angeben, den Sie zum Ausführen eines Builds verwendet haben. Beispielsweise:

```
https://codebuild.us-east-1.amazon.com/badges?uuid=...&branch=<branch>
```

Sie können auch ein Tag aus Ihrem Quell-Repository angeben, indem Sie den Parameter durch den `branch` `tag` Parameter in der Badge-URL ersetzen. Beispielsweise:

```
https://codebuild.us-east-1.amazon.com/badges?uuid=...&tag=<tag>
```

Veröffentlichen Sie Ihre Build-Badges CodeBuild

Sie können den Status des neuesten Builds in einer Markdown-Datei anzeigen, indem Sie Ihre Build-Badge-URL in einem Markdown-Bild verwenden. Dies ist nützlich, um den Status des neuesten Builds in der Datei readme.md in Ihrem Quell-Repository anzuzeigen (zum Beispiel oder). GitHub CodeCommit Beispielsweise:

```
![](<build badge URL>)
```

CodeBuild Status der Badges

- PASSING Der neueste Build der angegebenen Verzweigung wurde übergeben.
- FAILING Der neueste Build der angegebenen Verzweigung hat das Zeitlimit überschritten, ist fehlgeschlagen, fehlerhaft oder wurde gestoppt.
- IN_PROGRESS Der neueste Build für die angegebene Verzweigung wird verarbeitet.
- UNKNOWN Das Projekt hat für die angegebene Verzweigung oder generell noch keinen Build ausgeführt. Darüber hinaus ist die Funktion zum Erstellen von Badges möglicherweise deaktiviert.

Erstellen Sie CodeBuild anhand des Beispiels einen Testbericht AWS CLI

Tests, die Sie in der buildspec-Datei angeben, werden während des Builds ausgeführt. Dieses Beispiel zeigt Ihnen, wie Sie mit AWS CLI dem Tests in Builds integrieren können CodeBuild. Sie können JUnit verwenden, um Komponententests zu erstellen, oder ein anderes Werkzeug, um Konfigurationstests zu erstellen. Anschließend können Sie die Testergebnisse auswerten, um Probleme zu beheben oder Ihre Anwendung zu optimieren.

Sie können die CodeBuild API oder die AWS CodeBuild Konsole verwenden, um auf die Testergebnisse zuzugreifen. In diesem Beispiel wird gezeigt, wie Sie den Bericht so konfigurieren, dass die Testergebnisse in einen S3-Bucket exportiert werden.

Themen

Voraussetzungen

- Erstellen Sie Ihre Testfälle. Dieses Beispiel geht von der Annahme aus, dass Sie Testfälle in Ihren Testbericht aufnehmen müssen. Sie geben den Speicherort Ihrer Testdateien in der buildspec-Datei an.

  Die folgenden Dateiformate für Testberichte werden unterstützt:

  - Gurke JSON (.json)
  - Einheits-XML (.xml)
  - Einheit XML (.xml)
  - NUnit3-XML (.xml)
  - TestNG XML (.xml)
  - Visual Studio TRX (.trx)
  - Visual Studio TRX XML (.xml)

  Erstellen Sie Ihre Testfälle mit einem beliebigen Test-Framework, das Testberichtdateien in einem dieser Formate erstellen kann (zum Beispiel Surefire JUnit-Plugin, TestNG oder Cucumber).
- Erstellen Sie einen S3-Bucket und notieren Sie sich dessen Namen. Weitere Informationen finden Sie unter Wie erstelle ich einen S3-Bucket? im Amazon S3 S3-Benutzerhandbuch.
- Erstellen Sie eine IAM-Rolle und notieren Sie sich ihren ARN. Sie benötigen den ARN, wenn Sie Ihr Build-Projekt erstellen.
- Wenn Ihre Rolle nicht über die folgenden Berechtigungen verfügt, fügen Sie sie hinzu.

```
{
    "Effect": "Allow",
```

```
    "Resource": [
        "*"
    ],
    "Action": [
        "codebuild:CreateReportGroup",
        "codebuild:CreateReport",
        "codebuild:UpdateReport",
        "codebuild:BatchPutTestCases"
    ]
}
```

Weitere Informationen finden Sie unter Berechtigungen für Testberichtoperationen.

Erstellen einer Berichtsgruppe

1. Erstellen Sie eine Datei namens `CreateReportGroupInput.json`.
2. Erstellen Sie einen Ordner in Ihrem S3-Bucket, in den die Testergebnisse exportiert werden.
3. Kopieren Sie Folgendes in `CreateReportGroupInput.json`. Verwenden Sie für *<bucket-name>* den Namen des S3-Buckets. Geben Sie für *<path-to-folder>* den Pfad zu dem Ordner in Ihrem S3-Bucket ein.

```
{
  "name": "<report-name>",
  "type": "TEST",
  "exportConfig": {
    "exportConfigType": "S3",
    "s3Destination": {
      "bucket": "<bucket-name>",
      "path": "<path-to-folder>",
      "packaging": "NONE"
    }
  }
}
```

4. Führen Sie im Verzeichnis, das `CreateReportGroupInput.json` enthält, den folgenden Befehl aus.

```
aws codebuild create-report-group --cli-input-json file://
CreateReportGroupInput.json
```

Die Ausgabe sieht wie folgt aus. Notieren Sie sich den ARN für die `reportGroup`. Sie verwenden diesen, wenn Sie ein Projekt erstellen, das diese Berichtsgruppe verwendet.

```
{
  "reportGroup": {
    "arn": "arn:aws:codebuild:us-west-2:123456789012:report-group/<report-name>",
    "name": "<report-name>",
    "type": "TEST",
    "exportConfig": {
      "exportConfigType": "S3",
      "s3Destination": {
        "bucket": "<s3-bucket-name>",
        "path": "<folder-path>",
        "packaging": "NONE",
        "encryptionKey": "arn:aws:kms:us-west-2:123456789012:alias/aws/s3"
      }
    },
    "created": 1570837165.885,
    "lastModified": 1570837165.885
  }
}
```

Konfigurieren eines Projekts mit einer Berichtsgruppe

Um einen Bericht auszuführen, erstellen Sie zunächst ein CodeBuild Build-Projekt, das mit Ihrer Berichtsgruppe konfiguriert ist. Testfälle, die für Ihre Berichtsgruppe angegeben wurden, werden ausgeführt, wenn Sie einen Build ausführen.

1. Erstellen Sie eine buildspec-Datei mit dem Namen `buildspec.yml`.
2. Verwenden Sie die folgende YAML als Vorlage für Ihre `buildspec.yml`-Datei. Stellen Sie sicher, dass Sie die Befehle einschließen, die Ihre Tests ausführen. Geben Sie im `reports`-Abschnitt die Dateien an, die die Ergebnisse Ihrer Testfälle enthalten. In diesen Dateien werden die Testergebnisse gespeichert, auf die Sie zugreifen können CodeBuild. Sie verfallen 30 Tage nach ihrer Erstellung. Diese Dateien unterscheiden sich von den Roh-Testfall-Ergebnisdateien, die Sie in einen S3-Bucket exportieren.

```
version: 0.2
    phases:
    install:
```

```
        runtime-versions:
            java: openjdk8
    build:
      commands:
        - echo Running tests
        - <enter commands to run your tests>

    reports:
      <report-name-or-arn>: #test file information
      files:
        - '<test-result-files>'
      base-directory: '<optional-base-directory>'
      discard-paths: false #do not remove file paths from test result files
```

> **Note**
>
> Anstelle des ARN einer vorhandenen Berichtsgruppe können Sie auch einen Namen für eine nicht erstellte Berichtsgruppe angeben. Wenn Sie einen Namen anstelle eines ARN angeben, CodeBuild wird bei der Ausführung eines Builds eine Berichtsgruppe erstellt. Der Name enthält Ihren Projektnamen und den Namen, den Sie in der buildspec-Datei in folgendem Format angeben: `project-name-report-group-name`. Weitere Informationen finden Sie unter Erstellen eines Testberichts und Benennung von Berichtsgruppen.

3. Erstellen Sie eine Datei namens `project.json`. Diese Datei enthält die Eingabe für den create-project-Befehl.

4. Kopieren Sie den folgenden JSON in `project.json`. Geben Sie für `source` den Typ und den Speicherort des Repositorys ein, das die Quelldateien enthält. Geben Sie für `serviceRole` den ARN der Rolle an, die Sie verwenden.

```
{
  "name": "test-report-project",
  "description": "sample-test-report-project",
  "source": {
    "type": "CODECOMMIT|CODEPIPELINE|GITHUB|S3|BITBUCKET|GITHUB_ENTERPRISE|
NO_SOURCE",
    "location": "<your-source-url>"
  },
  "artifacts": {
    "type": "NO_ARTIFACTS"
```

```
  },
  "cache": {
    "type": "NO_CACHE"
  },
  "environment": {
    "type": "LINUX_CONTAINER",
    "image": "aws/codebuild/standard:5.0",
    "computeType": "small"
  },
  "serviceRole": "arn:aws:iam::<your-aws-account-id>:role/service-role/<your-role-name>"
}
```

5. Führen Sie im Verzeichnis, das `project.json` enthält, den folgenden Befehl aus. Dadurch wird ein Projekt mit dem Namen `test-project` erstellt.

```
aws codebuild create-project --cli-input-json file://project.json
```

Ausführen und Anzeigen der Ergebnisse eines Berichts

In diesem Abschnitt führen Sie einen Build des zuvor erstellten Projekts aus. CodeBuild Erstellt während des Erstellungsprozesses einen Bericht mit den Ergebnissen der Testfälle. Der Bericht ist in der von Ihnen angegebenen Berichtsgruppe enthalten.

1. Führen Sie den folgenden Befehl aus, um einen Build zu starten. `test-report-project`ist der Name des oben erstellten Build-Projekts. Notieren Sie sich die Build-ID, die in der Ausgabe angezeigt wird.

```
aws codebuild start-build --project-name test-report-project
```

2. Führen Sie den folgenden Befehl aus, um Informationen zu Ihrem Build abzurufen, einschließlich des ARN Ihres Berichts. Geben Sie für *<build-id>* Ihre Build-ID an. Notieren Sie sich den Berichts-ARN in der `reportArns` Eigenschaft der Ausgabe.

```
aws codebuild batch-get-builds --ids <build-id>
```

3. Führen Sie den folgenden Befehl aus, um Details zu Ihrem Bericht abzurufen. Geben Sie für *<report-arn>* den Berichts-ARN an.

```
aws codebuild batch-get-reports --report-arns <report-arn>
```

Die Ausgabe sieht wie folgt aus. Diese Beispielausgabe zeigt, wie viele der Tests erfolgreich waren, fehlschlugen, übersprungen wurden, zu einem Fehler geführt haben oder einen unbekannten Status zurückgaben.

```
{
  "reports": [
    {
      "status": "FAILED",
      "reportGroupArn": "<report-group-arn>",
      "name": "<report-group-name>",
      "created": 1573324770.154,
      "exportConfig": {
        "exportConfigType": "S3",
        "s3Destination": {
          "bucket": "<your-S3-bucket>",
          "path": "<path-to-your-report-results>",
          "packaging": "NONE",
          "encryptionKey": "<encryption-key>"
        }
      },
      "expired": 1575916770.0,
      "truncated": false,
      "executionId": "arn:aws:codebuild:us-west-2:123456789012:build/<name-of-
build-project>:2c254862-ddf6-4831-a53f-6839a73829c1",
      "type": "TEST",
      "arn": "<report-arn>",
      "testSummary": {
        "durationInNanoSeconds": 6657770,
        "total": 11,
        "statusCounts": {
          "FAILED": 3,
          "SKIPPED": 7,
          "ERROR": 0,
          "SUCCEEDED": 1,
          "UNKNOWN": 0
        }
      }
    }
  ],
```

```
  "reportsNotFound": []
}
```

4. Führen Sie den folgenden Befehl aus, um Informationen zu Testfällen für Ihren Bericht aufzulisten. Geben Sie für *<report-arn>* den ARN Ihres Berichts an. Für den optionalen `--filter`-Parameter können Sie ein Statusergebnis (`SUCCEEDED`, `FAILED`, `SKIPPED`, `ERROR` oder `UNKNOWN`) angeben.

```
aws codebuild describe-test-cases \
    --report-arn <report-arn> \
    --filter status=SUCCEEDED|FAILED|SKIPPED|ERROR|UNKNOWN
```

Die Ausgabe sieht wie folgt aus.

```
{
  "testCases": [
    {
      "status": "FAILED",
      "name": "Test case 1",
      "expired": 1575916770.0,
      "reportArn": "<report-arn>",
      "prefix": "Cucumber tests for agent",
      "message": "A test message",
      "durationInNanoSeconds": 1540540,
      "testRawDataPath": "<path-to-output-report-files>"
    },
    {
      "status": "SUCCEEDED",
      "name": "Test case 2",
      "expired": 1575916770.0,
      "reportArn": "<report-arn>",
      "prefix": "Cucumber tests for agent",
      "message": "A test message",
      "durationInNanoSeconds": 1540540,
      "testRawDataPath": "<path-to-output-report-files>"
    }
  ]
}
```

Docker-Beispiele für CodeBuild

Themen

Docker im Beispiel eines benutzerdefinierten Images für CodeBuild

In diesem Beispiel wird ein Docker-Image mithilfe eines benutzerdefinierten Docker-Build-Images (`docker:dind`in Docker Hub) erstellt AWS CodeBuild und ausgeführt.

Informationen dazu, wie Sie ein Docker-Image erstellen, indem Sie stattdessen ein Build-Image verwenden, das von der Docker-Unterstützung bereitgestellt wird CodeBuild , finden Sie in unserer. Docker-Image in einem Amazon ECR-Image-Repository-Beispiel veröffentlichen

> **Important**
>
> Die Ausführung dieses Beispiels kann zu Gebühren für Ihr AWS Konto führen. Dazu gehören mögliche Gebühren für CodeBuild und für AWS Ressourcen und Aktionen im Zusammenhang mit Amazon S3 und CloudWatch Logs. AWS KMS Weitere Informationen finden Sie unter CodeBuild Preise, Amazon S3 S3-Preise, AWS Key Management Service Preise und CloudWatchAmazon-Preise.

Themen

Ausführen des Beispiels

So führen Sie das Beispiel aus

1. Erstellen Sie die Dateien wie in den Abschnitten „Verzeichnisstruktur“ und „Dateien“ dieses Themas beschrieben und laden Sie sie dann in einen S3-Eingabe-Bucket oder ein AWS CodeCommit, GitHub, oder Bitbucket-Repository hoch.

   > **Important**
   >
   > Laden Sie nicht *(root directory name)* hoch, sondern nur die Dateien in *(root directory name)*.
   > Wenn Sie einen S3-Empfangs-Bucket verwenden, sollten Sie eine ZIP-Datei erstellen, die die Dateien enthält, und diese dann in den Empfangs-Bucket hochladen. Fügen Sie *(root directory name)* nicht zur ZIP-Datei hinzu, sondern nur die Dateien in *(root directory name)*.

2. Erstelle ein Build-Projekt, führe den Build aus und sieh dir die zugehörigen Build-Informationen an.

   Wenn Sie das AWS CLI Build-Projekt mit erstellen, sieht die Eingabe des `create-project` Befehls im JSON-Format möglicherweise ähnlich aus. (Ersetzen Sie die Platzhalter durch Ihre eigenen Werte.)

```
{
  "name": "sample-docker-custom-image-project",
  "source": {
    "type": "S3",
    "location": "codebuild-region-ID-account-ID-input-
bucket/DockerCustomImageSample.zip"
  },
  "artifacts": {
    "type": "NO_ARTIFACTS"
  },
  "environment": {
    "type": "LINUX_CONTAINER",
    "image": "docker:dind",
    "computeType": "BUILD_GENERAL1_SMALL",
    "privilegedMode": false
  },
  "serviceRole": "arn:aws:iam::account-ID:role/role-name",
```

```
  "encryptionKey": "arn:aws:kms:region-ID:account-ID:key/key-ID"
}
```

> **Note**
>
> Standardmäßig ist der Docker-Daemon für Nicht-VPC-Builds aktiviert. Wenn Sie Docker-Container für VPC-Builds verwenden möchten, lesen Sie auf der Docker Docs-Website unter Runtime Privilege and Linux Capabilities nach und aktivieren Sie den privilegierten Modus. Außerdem unterstützt Windows den privilegierten Modus nicht.

3. Zum Anzeigen der Build-Ergebnisse sehen Sie sich die Zeichenfolge `Hello, World!` im Build-Protokoll an. Weitere Informationen finden Sie unter Anzeigen von Build-Details.

Verzeichnisstruktur

In diesem Beispiel wird von dieser Verzeichnisstruktur ausgegangen.

```
(root directory name)
### buildspec.yml
### Dockerfile
```

Dateien

Die Basis-Image des Betriebssystems in diesem Beispiel ist Ubuntu. Das Beispiel verwendet diese Dateien.

`buildspec.yml` (in *`(root directory name)`*)

```
version: 0.2

phases:
  pre_build:
    commands:
      - docker build -t helloworld .
  build:
    commands:
      - docker images
      - docker run helloworld echo "Hello, World!"
```

`Dockerfile` (in *`(root directory name)`*)

```
FROM maven:3.3.9-jdk-8

RUN echo "Hello World"
```

Zugehörige Ressourcen

- Informationen zu den ersten Schritten mit AWS CodeBuild finden Sie unterErste Schritte mit AWS CodeBuild unter Verwendung der Konsole.
- Informationen zur Behebung von Problemen finden Sie unterFehlerbehebung AWS CodeBuild. CodeBuild
- Informationen zu Kontingenten in CodeBuild finden Sie unterKontingente für AWS CodeBuild.

Docker-Image in einem Amazon Elastic Container Registry-Image-Repository veröffentlichen — Beispiel für CodeBuild

Dieses Beispiel erzeugt als Build-Ausgabe ein Docker-Image und überträgt das Docker-Image dann in ein Amazon Elastic Container Registry (Amazon ECR) -Image-Repository. Sie können dieses Beispiel so anpassen, dass das Docker-Image im Docker Hub bereitgestellt wird. Weitere Informationen finden Sie unter Anpassen des Beispiels zur Übertragung des Images zum Docker Hub.

Informationen zum Erstellen eines Docker-Image mit einem benutzerdefinierten Docker Build-Image (`docker:dind` im Docker Hub) finden Sie unter Docker im benutzerdefinierten Image – Beispiel.

Dieses Beispiel wurde mit einem Verweis auf `golang:1.12` getestet.

In diesem Beispiel wird die mehrstufige Builds-Funktion von Docker verwendet, durch die ein Docker-Image als Build-Ausgabe produziert wird. Anschließend wird das Docker-Image in ein Amazon ECR-Image-Repository übertragen. Mehrstufige Docker-Image-Builds reduzieren die Größe des endgültigen Docker-Image. Weitere Informationen finden Sie unter Use multi-stage builds with Docker.

> **Important**
>
> Die Ausführung dieses Beispiels kann dazu führen, dass Ihr Konto belastet wird. AWS Dazu gehören mögliche Gebühren für AWS CodeBuild und für AWS Ressourcen und Aktionen im Zusammenhang mit Amazon S3 AWS KMS, CloudWatch Logs und Amazon ECR. Weitere Informationen finden Sie unter CodeBuild Preise, Amazon S3 S3-Preise, AWS Key

Management Service Preise, CloudWatch Amazon-Preise und Amazon Elastic Container Registry — Preise.

Themen

- Ausführen des Beispiels
- Verzeichnisstruktur
- Dateien
- Anpassen des Beispiels zur Übertragung des Images zum Docker Hub
- Zugehörige Ressourcen

Ausführen des Beispiels

So führen Sie das Beispiel aus

1. Wenn Sie bereits über ein Bild-Repository in Amazon ECR verfügen, das Sie verwenden möchten, fahren Sie mit Schritt 3 fort. Andernfalls, wenn Sie einen Benutzer anstelle eines AWS Root-Kontos oder eines Administrator-Benutzers für die Arbeit mit Amazon ECR verwenden, fügen Sie diese Anweisung (zwischen *### BEGIN ADDING STATEMENT HERE ### und ### END ADDING STATEMENT HERE ###) dem Benutzer (oder der IAM-Gruppe, der der Benutzer zugeordnet ist) hinzu*. Die Verwendung eines AWS Root-Kontos wird nicht empfohlen. Diese Anweisung ermöglicht die Erstellung von Amazon ECR-Repositorys zum Speichern von Docker-Images. Auslassungspunkte (`...`) werden zur Abkürzung verwendet und weisen auf die Stellen hin, an denen die Anweisung hinzugefügt wird. Entfernen Sie keine Anweisungen und geben Sie die Auslassungspunkte nicht in die Richtlinie ein. Weitere Informationen finden Sie unter Arbeiten mit Inline-Richtlinien mithilfe von im Benutzerhandbuch. AWS Management Console

```
{
  "Statement": [
    ### BEGIN ADDING STATEMENT HERE ###
    {
      "Action": [
        "ecr:CreateRepository"
      ],
      "Resource": "*",
      "Effect": "Allow"
```

```
    },
    ### END ADDING STATEMENT HERE ###
    ...
  ],
  "Version": "2012-10-17"
}
```

> **Note**
>
> Die IAM-Entität, die diese Richtlinie ändert, muss in IAM über die Berechtigung zum Ändern von Richtlinien verfügen.

2. Erstellen Sie ein Bild-Repository in Amazon ECR. Stellen Sie sicher, dass Sie das Repository in derselben AWS Region erstellen, in der Sie Ihre Build-Umgebung erstellen und Ihren Build ausführen. Weitere Informationen finden Sie unter Erstellen eines Repositorys im Amazon ECR-Benutzerhandbuch. Der Name dieses Repositorys muss mit dem Namen des Repositorys übereinstimmen, den Sie zu einem späteren Zeitpunkt in diesem Verfahren festlegen und der durch die `IMAGE_REPO_NAME`-Umgebungsvariable dargestellt wird. Stellen Sie sicher, dass die Amazon ECR-Repository-Richtlinie Image-Push-Zugriff für Ihre CodeBuild Service-IAM-Rolle gewährt.
3. Fügen Sie diese Erklärung (zwischen *### BEGIN ADDING STATEMENT HERE ### und ### END ADDING STATEMENT HERE ###) der Richtlinie hinzu*, die Sie Ihrer Servicerolle zugeordnet haben. AWS CodeBuild Diese Anweisung ermöglicht das Hochladen CodeBuild von Docker-Images in Amazon ECR-Repositorys. Auslassungspunkte (`...`) werden zur Abkürzung verwendet und weisen auf die Stellen hin, an denen die Anweisung hinzugefügt wird. Entfernen Sie keine Anweisungen und geben Sie die Auslassungspunkte nicht in die Richtlinie ein.

```
{
  "Statement": [
    ### BEGIN ADDING STATEMENT HERE ###
    {
      "Action": [
        "ecr:BatchCheckLayerAvailability",
        "ecr:CompleteLayerUpload",
        "ecr:GetAuthorizationToken",
        "ecr:InitiateLayerUpload",
        "ecr:PutImage",
        "ecr:UploadLayerPart"
      ],
```

```
      "Resource": "*",
      "Effect": "Allow"
    },
    ### END ADDING STATEMENT HERE ###
    ...
  ],
  "Version": "2012-10-17"
}
```

> **Note**
>
> Die IAM-Entität, die diese Richtlinie ändert, muss in IAM über die Erlaubnis verfügen, Richtlinien zu ändern.

4. Erstellen Sie die Dateien wie in den Abschnitten „Verzeichnisstruktur“ und „Dateien“ dieses Themas beschrieben und laden Sie sie dann in einen S3-Eingabe-Bucket oder ein AWS CodeCommit, GitHub, oder Bitbucket-Repository hoch. Weitere Informationen findest du unter Referenz zur Datei mit Bilddefinitionen im AWS CodePipeline Benutzerhandbuch.

   > **Important**
   >
   > Laden Sie nicht *(root directory name)* hoch, sondern nur die Dateien in *(root directory name)*.
   > Wenn Sie einen S3-Empfangs-Bucket verwenden, sollten Sie eine ZIP-Datei erstellen, die die Dateien enthält, und diese dann in den Empfangs-Bucket hochladen. Fügen Sie *(root directory name)* nicht zur ZIP-Datei hinzu, sondern nur die Dateien in *(root directory name)*.

5. Erstellen Sie ein Build-Projekt, führen Sie den Build aus und zeigen Sie die Build-Informationen an.

   Wenn Sie die Konsole verwenden, um Ihr Projekt zu erstellen:

   a. Wählen Sie für Operating system (Betriebssystem) die Option Ubuntu aus.
   b. Wählen Sie unter Runtime (Laufzeit) die Option Standard aus.
   c. Wählen Sie für Image (Abbild) die Option aws/codebuild/standard:5.0 aus.
   d. Fügen Sie die folgenden Umgebungsvariablen hinzu:

      - AWS_DEFAULT_REGION mit einem Wert für *region-ID*

- AWS_ACCOUNT_ID mit einem Wert für *account-ID*
- IMAGE_TAG mit dem Wert „Latest (Aktuell)“
- IMAGE_REPO_NAME mit einem Wert für *Amazon-ECR-repo-name*

Wenn Sie das AWS CLI Build-Projekt mit erstellen, sieht die Eingabe des `create-project` Befehls im JSON-Format möglicherweise ähnlich aus. (Ersetzen Sie die Platzhalter durch Ihre eigenen Werte.)

```
{
  "name": "sample-docker-project",
  "source": {
    "type": "S3",
    "location": "codebuild-region-ID-account-ID-input-bucket/DockerSample.zip"
  },
  "artifacts": {
    "type": "NO_ARTIFACTS"
  },
  "environment": {
    "type": "LINUX_CONTAINER",
    "image": "aws/codebuild/standard:5.0",
    "computeType": "BUILD_GENERAL1_SMALL",
    "environmentVariables": [
      {
        "name": "AWS_DEFAULT_REGION",
        "value": "region-ID"
      },
      {
        "name": "AWS_ACCOUNT_ID",
        "value": "account-ID"
      },
      {
        "name": "IMAGE_REPO_NAME",
        "value": "Amazon-ECR-repo-name"
      },
      {
        "name": "IMAGE_TAG",
        "value": "latest"
      }
    ],
  },
  "serviceRole": "arn:aws:iam::account-ID:role/role-name",
```

```
  "encryptionKey": "arn:aws:kms:region-ID:account-ID:key/key-ID"
}
```

6. Bestätigen Sie, dass das Docker-Image CodeBuild erfolgreich in das Repository übertragen wurde:

   1. Öffnen Sie die Amazon ECR-Konsole unter https://console.aws.amazon.com/ecr/.
   2. Wählen Sie den Namen des Repositorys aus. Das Bild muss in der Spalte Image Tag (Image-Tag) aufgelistet werden.

Verzeichnisstruktur

In diesem Beispiel wird von dieser Verzeichnisstruktur ausgegangen.

```
(root directory name)
### buildspec.yml
### Dockerfile
```

Dateien

In diesem Beispiel werden diese Dateien verwendet.

`buildspec.yml` (in *(root directory name)*)

```
version: 0.2

phases:
  pre_build:
    commands:
      - echo Logging in to Amazon ECR...
      - aws ecr get-login-password --region $AWS_DEFAULT_REGION | docker login --
username AWS --password-stdin $AWS_ACCOUNT_ID.dkr.ecr.$AWS_DEFAULT_REGION.amazonaws.com
  build:
    commands:
      - echo Build started on `date`
      - echo Building the Docker image...
      - docker build -t $IMAGE_REPO_NAME:$IMAGE_TAG .
      - docker tag $IMAGE_REPO_NAME:$IMAGE_TAG $AWS_ACCOUNT_ID.dkr.ecr.
$AWS_DEFAULT_REGION.amazonaws.com/$IMAGE_REPO_NAME:$IMAGE_TAG
  post_build:
    commands:
```

```
      - echo Build completed on `date`
      - echo Pushing the Docker image...
      - docker push $AWS_ACCOUNT_ID.dkr.ecr.$AWS_DEFAULT_REGION.amazonaws.com/
$IMAGE_REPO_NAME:$IMAGE_TAG
```

`Dockerfile` (in *`(root directory name)`*)

```
FROM golang:1.12-alpine AS build
#Install git
RUN apk add --no-cache git
#Get the hello world package from a GitHub repository
RUN go get github.com/golang/example/hello
WORKDIR /go/src/github.com/golang/example/hello
# Build the project and send the output to /bin/HelloWorld
RUN go build -o /bin/HelloWorld

FROM golang:1.12-alpine
#Copy the build's output binary from the previous build container
COPY --from=build /bin/HelloWorld /bin/HelloWorld
ENTRYPOINT ["/bin/HelloWorld"]
```

> **Note**
>
> CodeBuild überschreibt das `ENTRYPOINT` für benutzerdefinierte Docker-Images.

Anpassen des Beispiels zur Übertragung des Images zum Docker Hub

Um das Docker-Image auf Docker Hub statt auf Amazon ECR zu übertragen, bearbeiten Sie den Code dieses Beispiels.

> **Note**
>
> Wenn Sie eine Version von Docker vor 17.06 verwenden, entfernen Sie die `--no-include-email`-Option.

1. Ersetzen Sie diese Amazon ECR-spezifischen Codezeilen in der `buildspec.yml` Datei:

   ```
   ...
   ```

```
  pre_build:
    commands:
      - echo Logging in to Amazon ECR...
      - aws ecr get-login-password --region $AWS_DEFAULT_REGION |
 docker login --username AWS --password-stdin $AWS_ACCOUNT_ID.dkr.ecr.
$AWS_DEFAULT_REGION.amazonaws.com
  build:
    commands:
      - echo Build started on `date`
      - echo Building the Docker image...
      - docker build -t $IMAGE_REPO_NAME:$IMAGE_TAG .
      - docker tag $IMAGE_REPO_NAME:$IMAGE_TAG $AWS_ACCOUNT_ID.dkr.ecr.
$AWS_DEFAULT_REGION.amazonaws.com/$IMAGE_REPO_NAME:$IMAGE_TAG
  post_build:
    commands:
      - echo Build completed on `date`
      - echo Pushing the Docker image...
      - docker push $AWS_ACCOUNT_ID.dkr.ecr.$AWS_DEFAULT_REGION.amazonaws.com/
$IMAGE_REPO_NAME:$IMAGE_TAG
...
```

Durch die folgenden Docker Hub-spezifischen Codezeilen:

```
...
  pre_build:
    commands:
      - echo Logging in to Docker Hub...
      # Type the command to log in to your Docker Hub account here.
  build:
    commands:
      - echo Build started on `date`
      - echo Building the Docker image...
      - docker build -t $IMAGE_REPO_NAME:$IMAGE_TAG .
      - docker tag $IMAGE_REPO_NAME:$IMAGE_TAG $IMAGE_REPO_NAME:$IMAGE_TAG
  post_build:
    commands:
      - echo Build completed on `date`
      - echo Pushing the Docker image...
      - docker push $IMAGE_REPO_NAME:$IMAGE_TAG
...
```

2. Laden Sie den bearbeiteten Code in einen S3-Eingabe-Bucket oder ein AWS CodeCommit, GitHub, oder Bitbucket-Repository hoch.

> **Important**
>
> Laden Sie nicht *(root directory name)* hoch, sondern nur die Dateien in *(root directory name)*.
> Wenn Sie einen S3-Empfangs-Bucket verwenden, sollten Sie eine ZIP-Datei erstellen, die die Dateien enthält, und diese dann in den Empfangs-Bucket hochladen. Fügen Sie *(root directory name)* nicht zur ZIP-Datei hinzu, sondern nur die Dateien in *(root directory name)*.

3. Ersetzen Sie diese Codezeilen aus der JSON-formatierten Eingabe des `create-project`-Befehls:

```
...
    "environmentVariables": [
      {
        "name": "AWS_DEFAULT_REGION",
        "value": "region-ID"
      },
      {
        "name": "AWS_ACCOUNT_ID",
        "value": "account-ID"
      },
      {
        "name": "IMAGE_REPO_NAME",
        "value": "Amazon-ECR-repo-name"
      },
      {
        "name": "IMAGE_TAG",
        "value": "latest"
      }
    ]
...
```

Durch diese Codezeilen:

```
...
    "environmentVariables": [
      {
        "name": "IMAGE_REPO_NAME",
        "value": "your-Docker-Hub-repo-name"
```

```
      },
      {
        "name": "IMAGE_TAG",
        "value": "latest"
      }
    ]
...
```

4. Erstelle eine Build-Umgebung, führe den Build aus und sieh dir die zugehörigen Build-Informationen an.
5. Bestätigen Sie, dass das Docker-Image AWS CodeBuild erfolgreich in das Repository übertragen wurde. Melden Sie sich beim Docker Hub an, wechseln Sie zum Repository wählen Sie die Registerkarte Tags aus Das `latest`-Tag sollte einen aktuellen Last Updated-Wert enthalten.

Zugehörige Ressourcen

- Informationen zu den ersten Schritten mit finden Sie AWS CodeBuild unterErste Schritte mit AWS CodeBuild unter Verwendung der Konsole.
- Informationen zur Behebung von Problemen finden Sie unterFehlerbehebung AWS CodeBuild. CodeBuild
- Informationen zu Kontingenten in CodeBuild finden Sie unterKontingente für AWS CodeBuild.

Private Registrierung mit AWS Secrets Manager Beispiel für CodeBuild

Dieses Beispiel zeigt Ihnen, wie Sie ein Docker-Image, das in einer privaten Registry gespeichert ist, als Ihre AWS CodeBuild Laufzeitumgebung verwenden. Die Anmeldeinformationen für die private Registrierung sind in AWS Secrets Manager gespeichert. Jede private Registrierung funktioniert mit CodeBuild. In diesem Beispiel wird Docker Hub verwendet.

> **Note**
>
> Geheimnisse sind für Aktionen sichtbar und werden nicht maskiert, wenn sie in eine Datei geschrieben werden.

Anforderungen für ein Beispiel einer privaten Registrierung

Um eine private Registrierung mit verwenden zu können AWS CodeBuild, benötigen Sie Folgendes:

- Ein Secrets Manager Manager-Geheimnis, das Ihre Docker Hub-Anmeldeinformationen speichert. Die Anmeldeinformationen werden für den Zugriff auf Ihr privates Repository verwendet.

> **Note**
>
> Die von Ihnen erstellten Geheimnisse werden Ihnen in Rechnung gestellt.

- Ein privates Repository oder Konto.
- Eine IAM-Richtlinie für eine CodeBuild Servicerolle, die Zugriff auf Ihr Secrets Manager Manager-Geheimnis gewährt.

Gehen Sie wie folgt vor, um diese Ressourcen zu erstellen, und erstellen Sie dann ein CodeBuild Build-Projekt mit den Docker-Images, die in Ihrer privaten Registrierung gespeichert sind.

Erstellen Sie ein CodeBuild Projekt mit einer privaten Registrierung

1. Weitere Informationen dazu, wie Sie ein kostenloses privates Repository erstellen, finden Sie unter Repositories on Docker Hub Sie können auch die folgenden Befehle in einem Terminal ausführen, um eine Pull-Übertragung eines Images durchzuführen, seine ID abzurufen und es in ein neues Repository zu verschieben.

```
docker pull amazonlinux
docker images amazonlinux --format {{.ID}}
docker tag image-id your-username/repository-name:tag
docker login
docker push your-username/repository-name
```

2. Folgen Sie den Schritten unter Ein AWS Secrets Manager Geheimnis erstellen im AWS Secrets Manager Benutzerhandbuch.

   a. Wählen Sie in Schritt 3 unter Geheimtyp auswählen die Option Anderer Geheimtyp aus.

   b. Erstellen Sie unter Schlüssel/Wert-Paare ein Schlüssel-Wert-Paar für Ihren Docker Hub-Benutzernamen und ein Schlüssel-Wert-Paar für Ihr Docker Hub-Passwort.

   c. Folgen Sie weiterhin den Schritten unter Create an Secret. AWS Secrets Manager

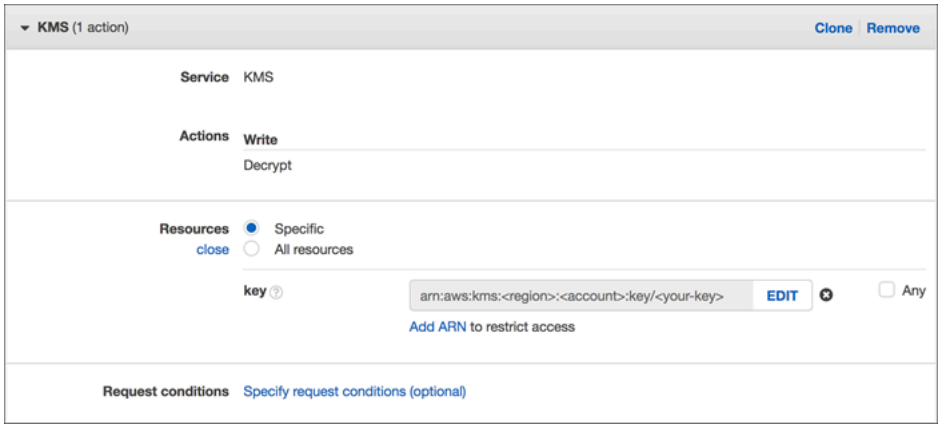
d. Deaktivieren Sie sie in Schritt 5 auf der Seite Automatische Rotation konfigurieren, da die Schlüssel Ihren Docker Hub-Anmeldeinformationen entsprechen.

e. Folgen Sie abschließend den Schritten unter Create an AWS Secrets Manager Secret.

Weitere Informationen finden Sie unter Was ist AWS Secrets Manager?

3. Wenn Sie ein AWS CodeBuild Projekt in der Konsole erstellen, CodeBuild hängt es die für Sie erforderlichen Berechtigungen an. Wenn Sie einen anderen AWS KMS Schlüssel als verwenden`DefaultEncryptionKey`, müssen Sie ihn der Servicerolle hinzufügen. Weitere Informationen finden Sie unter Ändern einer Rolle (Konsole) im IAM-Benutzerhandbuch.

   Damit Ihre Servicerolle mit Secrets Manager funktioniert, muss sie mindestens über die `secretsmanager:GetSecretValue` entsprechende Berechtigung verfügen.



4. Um die Konsole für die Erstellung eines Projekts zu verwenden, dessen Umgebung in einer privaten Registrierung gespeichert ist, gehen Sie während der Projekterstellung wie folgt vor: Weitere Informationen finden Sie unter Erstellen Sie ein Build-Projekt (Konsole).

> **Note**
>
> Wenn sich Ihre private Registrierung in Ihrer VPC befindet, muss sie über einen öffentlichen Internetzugang verfügen. CodeBuild kann kein Image von einer privaten IP-Adresse in einer VPC abrufen.

a. Wählen Sie unter Umgebungsbild die Option Benutzerdefiniertes Bild aus.

b. Wählen Sie für Environment type (Umgebungsart) die Option Linux oder Windows aus.

c. Wählen Sie für Image-Registrierung die Option Andere Registrierung aus.

d. Geben Sie unter Externe Registrierungs-URL den Speicherort des Images und in Registrierungsanmeldedaten — optional den ARN oder den Namen Ihrer Secrets Manager Manager-Anmeldeinformationen ein.

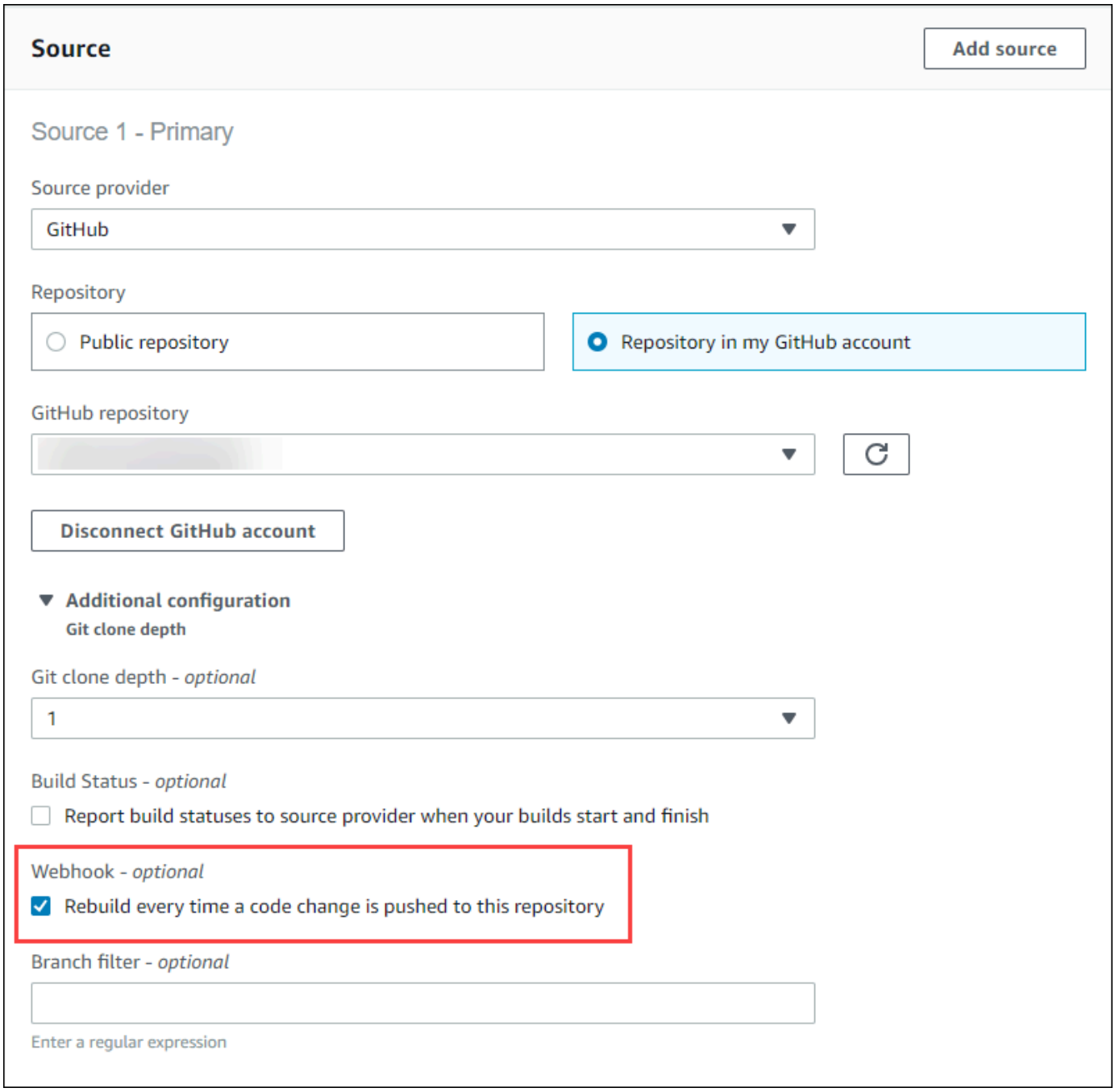
> **Note**
>
> Wenn Ihre Anmeldeinformationen nicht in Ihrer aktuellen Region vorhanden sind, müssen Sie den ARN verwenden. Sie können den Namen der Anmeldeinformationen nicht verwenden, wenn die Anmeldeinformationen in einer anderen Region existieren.

Erstellen einer statischen Website mit in einem S3-Bucket gehosteter Build-Ausgabe

Sie können die Verschlüsselung von Artefakten in einem Build deaktivieren. Dies ist sinnvoll, damit Sie Artefakte an einem Ort veröffentlichen können, der zum Hosten einer Website konfiguriert ist. (Sie können verschlüsselte Artefakte nicht veröffentlichen.) Dieses Beispiel zeigt, wie Sie Webhooks verwenden können, um eine Build-Erstellung auszulösen und die Build-Artefakte in einem S3-Bucket zu veröffentlichen, der als Website konfiguriert wurde.

1. Befolgen Sie die Anleitung unter Einrichten einer statischen Website, um einen S3-Bucket als Website zu konfigurieren.
2. Öffnen Sie die AWS CodeBuild Konsole unter https://console.aws.amazon.com/codesuite/codebuild/home.
3. Wenn eine CodeBuild Informationsseite angezeigt wird, wählen Sie Create build project aus. Erweitern Sie andernfalls im Navigationsbereich Build, wählen Sie Build projects und dann Create build project aus.
4. Geben Sie unter Project name (Projektname) einen Namen für dieses Build-Projekt ein. Die Namen von Build-Projekten müssen für jedes AWS Konto eindeutig sein. Sie können auch eine optionale Beschreibung des Build-Projekts hinzufügen, damit andere Benutzer verstehen, wofür dieses Projekt verwendet wird.
5. Wählen Sie unter Quelle für Quellanbieter die Option GitHub. Folgen Sie den Anweisungen, um eine Verbindung herzustellen (oder erneut herzustellen) GitHub, und wählen Sie dann Autorisieren aus.

Wählen Sie für Webhook die Option Rebuild every time a code change is pushed to this repository (Erneut erstellen, wenn eine Codeänderung an diesen Repository übergeben wird) aus. Sie können dieses Kontrollkästchen nur aktivieren, wenn Sie Use a repository in my account (Ein Repository in meinem Konto verwenden) ausgewählt haben.



6. In Environment (Umgebung):

   Führen Sie für Environment image (Umgebungs-Image) einen der folgenden Schritte aus:

- Um ein Docker-Image zu verwenden, das von verwaltet wird AWS CodeBuild, wählen Sie Verwaltetes Image aus und treffen Sie dann eine Auswahl unter Betriebssystem, Runtime (s), Image und Image-Version. Treffen Sie eine Auswahl unter Environment type (Umgebungstyp), sofern verfügbar.
- Wenn Sie ein anderes Docker-Image verwenden möchten, wählen Sie Custom image (Benutzerdefiniertes Image) aus. Wählen Sie als Umgebungstyp ARM, Linux, Linux GPU oder Windows aus. Wenn Sie Andere Registrierung wählen, geben Sie für Externe Registrierungs-URL den Namen und das Tag des Docker-Images in Docker Hub ein. Verwenden Sie dabei das Format. *docker repository*/*docker image name* Wenn Sie sich für Amazon ECR entscheiden, verwenden Sie das Amazon ECR-Repository und das Amazon ECR-Image, um das Docker-Image in Ihrem Konto auszuwählen. AWS
- Um ein privates Docker-Image zu verwenden, wählen Sie Benutzerdefiniertes Image. Wählen Sie als Umgebungstyp ARM, Linux, Linux GPU oder Windows aus. Wählen Sie unter Image registry (Abbildregistrierung) die Option Other registry (Andere Registrierung) aus und geben Sie dann den ARN der Anmeldeinformationen für Ihr privates Docker-Image ein. Die Anmeldeinformationen müssen von Secrets Manager erstellt werden. Weitere Informationen finden Sie unter Was ist AWS Secrets Manager? im AWS Secrets Manager - Benutzerhandbuch.

7. Führen Sie unter Service role (Service-Rolle) einen der folgenden Schritte aus:

   - Wenn Sie keine CodeBuild Servicerolle haben, wählen Sie Neue Servicerolle. Geben Sie im Feld Rollenname einen Namen für die neue Rolle ein.
   - Wenn Sie eine CodeBuild Servicerolle haben, wählen Sie Bestehende Servicerolle aus. Wählen Sie unter Role ARN die Servicerolle aus.

   > **Note**
   >
   > Wenn Sie die Konsole verwenden, um ein Build-Projekt zu erstellen oder zu aktualisieren, können Sie gleichzeitig eine CodeBuild Servicerolle erstellen. In der Standardeinstellung funktioniert diese Rolle ausschließlich mit diesem Projekt. Wenn Sie die Konsole verwenden, um die Servicerolle mit einem anderen Build-Projekt zu verknüpfen, wird die Rolle so aktualisiert, dass sie mit dem anderen Build-Projekt funktioniert. Eine Servicerolle kann in bis zu zehn Build-Projekten verwendet werden.

8. Führen Sie in Buildspec einen der folgenden Schritte aus:

- Wählen Sie Buildspec-Datei verwenden, um die Datei buildspec.yml im Quellcode-Stammverzeichnis zu verwenden.
- Wählen Sie Build-Befehle einfügen, um die Konsole zum Einfügen von Build-Befehlen zu verwenden.

Weitere Informationen hierzu finden Sie unter Build-Spezifikationsreferenz.

9. Wählen Sie unter Artifacts für Type Amazon S3 aus, um die Build-Ausgabe in einem S3-Bucket zu speichern.
10. Wählen Sie für Bucket name (Bucket-Name) den Namen des S3-Buckets, den Sie in Schritt 1 als Website konfiguriert haben.
11. Wenn Sie Build-Befehle in Umgebung einfügen ausgewählt haben, geben Sie für Ausgabedateien die Speicherorte der Dateien aus dem Build ein, die Sie in den Ausgabe-Bucket einfügen möchten. Wenn Sie mehr als einen Speicherort haben, verwenden Sie ein Komma, um jede Position zu trennen (z. B. **`appspec.yml, target/my-app.jar`**). Weitere Informationen finden Sie unter Artifacts reference-key in the buildspec file.
12. Wählen Sie Disable artifacts encryption (Artefaktverschlüsselung deaktivieren) aus.
13. Erweitern Sie Additional configuration (Zusätzliche Einstellungen) und wählen Sie die entsprechenden Optionen.
14. Wählen Sie Create build project (Build-Projekt erstellen) aus. Wählen Sie auf der Seite für das Erstellen des Projekts in Build history (Build-Verlauf) die Option Start build (Build starten) aus, um den Build auszuführen.
15. (Optional) Folgen Sie den Anweisungen unter Beispiel: Beschleunigen Sie Ihre Website mit Amazon CloudFront im Amazon S3 S3-Entwicklerhandbuch.

Beispiel für mehrere Eingabequellen und Ausgabeartefakte

Sie können ein AWS CodeBuild Build-Projekt mit mehr als einer Eingabequelle und mehr als einem Satz von Ausgabeartefakten erstellen. Dieses Beispiel zeigt, wie Sie ein Build-Projekt mit folgenden Eigenschaften einrichten:

- Mehrere Quellen und Repositorys unterschiedlicher Typen
- Veröffentlichung von Build-Artefakten für mehrere S3-Buckets in einem Build

In diesem Beispiel erstellen Sie ein Build-Projekt und verwenden es zum Ausführen eines Builds. Das Beispiel verwendet die buildspec-Datei des Build-Projekts, um zu demonstrieren, wie mehr als eine Quelle eingeschlossen und mehr als ein Satz Artefakte erstellt wird.

1. Lade deine Quellen in ein oder mehrere S3-Buckets, CodeCommit GitHub, GitHub Enterprise Server- oder Bitbucket-Repositorys hoch.
2. Wählen Sie die als primäre Quelle dienende Quelle. Dies ist die Quelle, in der nach Ihrer CodeBuild Buildspec-Datei gesucht und diese ausgeführt wird.
3. Erstellen Sie ein Build-Projekt. Weitere Informationen finden Sie unter Erstellen eines Build-Projekts in AWS CodeBuild.
4. Erstellen Sie Ihr Build-Projekt, führen Sie den Build aus und rufen Sie Informationen über den Build ab.
5. Wenn Sie das AWS CLI Build-Projekt mit erstellen, sieht die Eingabe des `create-project` Befehls im JSON-Format möglicherweise wie folgt aus:

```
{
  "name": "sample-project",
  "source": {
    "type": "S3",
    "location": "<bucket/sample.zip>"
  },
  "secondarySources": [
    {
      "type": "CODECOMMIT",
      "location": "https://git-codecommit.us-west-2.amazonaws.com/v1/repos/repo",
      "sourceIdentifier": "source1"
    },
    {
      "type": "GITHUB",
      "location": "https://github.com/awslabs/aws-codebuild-jenkins-plugin",
      "sourceIdentifier": "source2"
    }
  ],
  "secondaryArtifacts": [ss
    {
      "type": "S3",
      "location": "<output-bucket>",
      "artifactIdentifier": "artifact1"
    },
    {
```

```
      "type": "S3",
      "location": "<other-output-bucket>",
      "artifactIdentifier": "artifact2"
    }
  ],
  "environment": {
    "type": "LINUX_CONTAINER",
    "image": "aws/codebuild/standard:5.0",
    "computeType": "BUILD_GENERAL1_SMALL"
  },
  "serviceRole": "arn:aws:iam::account-ID:role/role-name",
  "encryptionKey": "arn:aws:kms:region-ID:account-ID:key/key-ID"
}
```

Die primäre Quelle ist im Attribut `source` definiert. Alle weiteren Quellen sind sekundäre Quellen und werden unter `secondarySources` aufgeführt. Alle sekundären Quellen werden in einem eigenen Verzeichnis installiert. Dieses Verzeichnis wird in der integrierten Umgebungsvariable `CODEBUILD_SRC_DIR_`*`sourceIdentifer`* gespeichert. Weitere Informationen finden Sie unter Umgebungsvariablen in Build-Umgebungen.

Das Attribut `secondaryArtifacts` enthält eine Liste der Artefaktdefinitionen. Diese Artefakte verwenden den Block `secondary-artifacts` in der buildspec-Datei, der im Block `artifacts` verschachtelt ist.

Sekundäre Artefakte in der buildspec-Datei haben die gleiche Struktur wie Artefakte und werden durch die jeweiligen Artefaktbezeichner voneinander getrennt.

> **Note**
>
> In der CodeBuild API ist das `artifactIdentifier` auf einem sekundären Artefakt ein erforderliches Attribut in und. `CreateProject` `UpdateProject` Es muss verwendet werden, um auf ein sekundäres Artefakt zu verweisen.

Mit der oben gezeigten Eingabe im JSON-Format ergibt sich eine buildspec-Datei wie die folgende:

```
version: 0.2

phases:
  install:
```

```
    runtime-versions:
      java: openjdk11
  build:
    commands:
      - cd $CODEBUILD_SRC_DIR_source1
      - touch file1
      - cd $CODEBUILD_SRC_DIR_source2
      - touch file2

artifacts:
  files:
    - '**.*'
  secondary-artifacts:
    artifact1:
      base-directory: $CODEBUILD_SRC_DIR_source1
      files:
        - file1
    artifact2:
      base-directory: $CODEBUILD_SRC_DIR_source2
      files:
        - file2
```

Sie können die Version der primären Quelle unter Verwendung der API mit dem Attribut `sourceVersion` in `StartBuild` überschreiben. Mit dem Attribut `secondarySourceVersionOverride` können Sie einzelne oder mehrere Versionen sekundärer Quellen überschreiben.

Die Eingabe im JSON-Format für den `start-build` Befehl in könnte wie folgt aussehen: AWS CLI

```
{
   "projectName": "sample-project",
   "secondarySourcesVersionOverride": [
      {
        "sourceIdentifier": "source1",
        "sourceVersion": "codecommit-branch"
      },
      {
        "sourceIdentifier": "source2",
        "sourceVersion": "github-branch"
      },
   ]
}
```

Beispiel für ein Projekt ohne Quelle

Sie können ein CodeBuild Projekt konfigurieren, indem Sie bei der Konfiguration **`NO_SOURCE`** Ihrer Quelle den Quelltyp auswählen. Wenn Ihr Quelltyp **`NO_SOURCE`** ist, können Sie keine buildspec-Datei angeben, da Ihr Projekt keine Quelle hat. Stattdessen müssen Sie eine YAML-formatierte buildspec-Zeichenfolge im `buildspec`-Attribut der JSON-formatierten Eingabe für den `create-project`-CLI-Befehl angeben. Diese könnte wie folgt aussehen:

```
{
  "name": "project-name",
  "source": {
    "type": "NO_SOURCE",
    "buildspec": "version: 0.2\n\nphases:\n  build:\n    commands:\n      - command"
   },
  "environment": {
    "type": "LINUX_CONTAINER",
    "image": "aws/codebuild/standard:5.0",
    "computeType": "BUILD_GENERAL1_SMALL",
  },
  "serviceRole": "arn:aws:iam::account-ID:role/role-name",
  "encryptionKey": "arn:aws:kms:region-ID:account-ID:key/key-ID"
}
```

Weitere Informationen finden Sie unter Erstellen eines Build-Projekts (AWS CLI).

Informationen zum Erstellen einer Pipeline, die mehrere Quelleingaben verwendet, um mehrere Ausgabeartefakte CodeBuild zu erzeugen, finden Sie unterAWS CodePipeline Beispiel für Integration mit CodeBuild und mehrere Eingabequellen und Ausgabeartefakte .

Beispiel für Laufzeitversionen in der Buildspec-Datei CodeBuild

Wenn Sie das Amazon Linux 2 (AL2) -Standard-Image Version 1.0 oder höher oder das Ubuntu-Standard-Image Version 2.0 oder höher verwenden, können Sie im `runtime-versions` Abschnitt Ihrer Buildspec-Datei eine oder mehrere Laufzeiten angeben. Dieses Beispiel zeigt Ihnen, wie Sie Ihre Projektlaufzeit ändern, mehr als eine Laufzeit angeben und eine Laufzeit angeben können, die von einer anderen Laufzeit abhängt. Hinweise zu unterstützten Laufzeiten finden Sie unter Docker-Images bereitgestellt von CodeBuild.

> **Note**
>
> Wenn Sie Docker in Ihrem Build-Container verwenden, muss Ihr Build im privilegierten Modus ausgeführt werden. Weitere Informationen finden Sie unter Ausführen eines Build in AWS CodeBuild und Erstellen eines Build-Projekts in AWS CodeBuild.

Aktualisieren Ihrer Laufzeitversion

Sie können die von Ihrem Projekt verwendete Laufzeit auf eine neue Version umstellen, indem Sie den `runtime-versions` Abschnitt Ihrer Buildspec-Datei aktualisieren. Die folgenden Beispiele zeigen, wie Sie Java-Versionen 8 und 11 angeben.

- Ein `runtime-versions`-Abschnitt, der Version 8 von Java angibt:

```
phases:
  install:
    runtime-versions:
      java: corretto8
```

- Ein `runtime-versions`-Abschnitt, der Version 11 von Java angibt:

```
phases:
  install:
    runtime-versions:
      java: corretto11
```

Die folgenden Beispiele zeigen, wie verschiedene Versionen von Python mit dem Ubuntu-Standard-Image 5.0 oder dem Amazon Linux 2-Standard-Image 3.0 angegeben werden:

- Ein `runtime-versions` Abschnitt, der Python-Version 3.7 spezifiziert:

```
phases:
  install:
    runtime-versions:
      python: 3.7
```

- Ein `runtime-versions` Abschnitt, der Python-Version 3.8 spezifiziert:

```
phases:
```

```
  install:
    runtime-versions:
      python: 3.8
```

Dieses Beispiel zeigt ein Projekt, das mit der Java-Laufzeitversion 8 beginnt und anschließend auf die Java-Laufzeitversion 10 aktualisiert wird.

1. Laden Sie Maven herunter und installieren Sie es. Weitere Informationen finden Sie unter Downloading Apache Maven und Installing Apache Maven auf der Apache Maven-Website.
2. Wechseln Sie in ein leeres Verzeichnis auf Ihrem lokalen Computer oder der Instance und führen anschließend diesen Maven-Befehl aus.

   ```
   mvn archetype:generate "-DgroupId=com.mycompany.app" "-DartifactId=ROOT" "-
   DarchetypeArtifactId=maven-archetype-webapp" "-DinteractiveMode=false"
   ```

   Nach erfolgreicher Ausführung werden diese Verzeichnisstruktur und die Dateien erstellt.

   ```
   .
   ### ROOT
       ### pom.xml
       ### src
           ### main
               ### resources
               ### webapp
                   ### WEB-INF
                   #   ### web.xml
                   ### index.jsp
   ```
3. Erstellen Sie eine Datei mit dem Namen `buildspec.yml` und dem folgenden Inhalt. Speichern Sie die Datei im Verzeichnis *(root directory name)*`/my-web-app`.

   ```
   version: 0.2

   phases:
     install:
       runtime-versions:
         java: corretto8
     build:
       commands:
         - java -version
   ```

```
      - mvn package
artifacts:
  files:
    - '**/*'
  base-directory: 'target/my-web-app'
```

In der Build-Spezifikationsdatei:

- Der Abschnitt `runtime-versions` gibt an, dass das Projekt die Java-Laufzeitversion 8 verwendet.
- Der Befehl `- java -version` zeigt die Version von Java an, die bei der Ausführung des Projekts verwendet wird .

Ihre Dateistruktur sollte nun wie folgt aussehen:

```
(root directory name)
### my-web-app
    ### src
    #   ### main
    #   ### resources
    #   ### webapp
    #       ### WEB-INF
    #           ### web.xml
    #               ### index.jsp
    ### buildspec.yml
    ### pom.xml
```

4. Laden Sie den Inhalt des `my-web-app` Verzeichnisses in einen S3-Eingabe-Bucket oder ein CodeCommit, GitHub, oder Bitbucket-Repository hoch.

> **Important**
>
> Laden Sie nicht *(root directory name)* oder *(root directory name)*`/my-web-app` hoch, sondern nur die Verzeichnisse und Dateien in *(root directory name)*`/my-web-app`.
> Wenn Sie einen S3-Empfangs-Bucket verwenden, sollten Sie eine ZIP-Datei erstellen, die die Verzeichnisstruktur und die Dateien enthält, und diese dann in den Empfangs-Bucket hochladen. Fügen Sie nicht *(root directory name)* oder *(root*

directory name)/my-web-app zur ZIP-Datei hinzu, sondern nur die Verzeichnisse und Dateien in *(root directory name)*/my-web-app.

5. Öffnen Sie die AWS CodeBuild Konsole unter https://console.aws.amazon.com/codesuite/codebuild/home.
6. Erstellen Sie ein Build-Projekt. Weitere Informationen finden Sie unter Erstellen Sie ein Build-Projekt (Konsole) und Ausführen eines Build (Konsole). Übernehmen Sie für alle Einstellungen die Standardwerte, außer für folgende Einstellungen:
   - Für Environment (Umgebung):
     - Wählen Sie für Environment image (Umgebungs-Abbild) die Option Managed image (Verwaltetes Abbild) aus.
     - Wählen Sie für Operating system (Betriebssystem) die Option Amazon Linux 2 aus.
     - Wählen Sie unter Runtime (Laufzeit) die Option Standard aus.
     - Wählen Sie für Image aws/codebuild/amazonlinux2-x86_64-standard:4.0.
7. Wählen Sie Start build (Build starten).
8. Übernehmen Sie in der Build configuration (Build-Konfiguration) die Standardeinstellungen und wählen Sie dann Start build (Build starten).
9. Überprüfen Sie die Build-Ausgabe unter der Registerkarte Build logs (Build-Protokolle), wenn der Build abgeschlossen ist. Die Ausgabe sollte folgendermaßen oder ähnlich aussehen:

```
[Container] Date Time Phase is DOWNLOAD_SOURCE
[Container] Date Time CODEBUILD_SRC_DIR=/codebuild/output/src460614277/src
[Container] Date Time YAML location is /codebuild/output/src460614277/src/
buildspec.yml
[Container] Date Time Processing environment variables
[Container] Date Time Selecting 'java' runtime version 'corretto8' based on manual
 selections...
[Container] Date Time Running command echo "Installing Java version 8 ..."
Installing Java version 8 ...

[Container] Date Time Running command export JAVA_HOME="$JAVA_8_HOME"

[Container] Date Time Running command export JRE_HOME="$JRE_8_HOME"

[Container] Date Time Running command export JDK_HOME="$JDK_8_HOME"
```

```
[Container] Date Time Running command for tool_path in "$JAVA_8_HOME"/bin/*
 "$JRE_8_HOME"/bin/*;
```

10. Aktualisieren Sie den Abschnitt `runtime-versions` mit Java-Version 11:

```
install:
    runtime-versions:
      java: corretto11
```

11. Nachdem Sie die Änderung gespeichert haben, führen Sie Ihren Build erneut aus und zeigen Sie die Build-Ausgabe an. Es sollte angezeigt werden, dass Java Version 11 installiert ist. Die Ausgabe sollte folgendermaßen oder ähnlich aussehen:

```
[Container] Date Time Phase is DOWNLOAD_SOURCE
[Container] Date Time CODEBUILD_SRC_DIR=/codebuild/output/src460614277/src
[Container] Date Time YAML location is /codebuild/output/src460614277/src/
buildspec.yml
[Container] Date Time Processing environment variables
[Container] Date Time Selecting 'java' runtime version 'corretto11' based on manual
 selections...
Installing Java version 11 ...

[Container] Date Time Running command export JAVA_HOME="$JAVA_11_HOME"

[Container] Date Time Running command export JRE_HOME="$JRE_11_HOME"

[Container] Date Time Running command export JDK_HOME="$JDK_11_HOME"

[Container] Date Time Running command for tool_path in "$JAVA_11_HOME"/bin/*
 "$JRE_11_HOME"/bin/*;
```

Zwei Laufzeiten angeben

Sie können mehr als eine Runtime im selben Build-Projekt angeben. CodeBuild Dieses Beispielprojekt verwendet zwei Quelldateien: eine, die die Go-Laufzeit verwendet und eine, die die Node.js-Laufzeit verwendet.

1. Erstellen Sie ein Verzeichnis mit dem Namen `my-source`.
2. Erstellen Sie im Verzeichnis `my-source` ein Verzeichnis mit dem Namen `golang-app`.

3. Erstellen Sie eine Datei mit dem Namen `hello.go` und dem folgenden Inhalt. Speichern Sie die Datei im Verzeichnis `golang-app`.

```
package main
import "fmt"

func main() {
  fmt.Println("hello world from golang")
  fmt.Println("1+1 =", 1+1)
  fmt.Println("7.0/3.0 =", 7.0/3.0)
  fmt.Println(true && false)
  fmt.Println(true || false)
  fmt.Println(!true)
  fmt.Println("good bye from golang")
}
```

4. Erstellen Sie im Verzeichnis `my-source` ein Verzeichnis mit dem Namen `nodejs-app`. Es sollte sich auf derselben Ebene wie das Verzeichnis `golang-app` befinden.
5. Erstellen Sie eine Datei mit dem Namen `index.js` und dem folgenden Inhalt. Speichern Sie die Datei im Verzeichnis `nodejs-app`.

```
console.log("hello world from nodejs");
console.log("1+1 =" + (1+1));
console.log("7.0/3.0 =" + 7.0/3.0);
console.log(true && false);
console.log(true || false);
console.log(!true);
console.log("good bye from nodejs");
```

6. Erstellen Sie eine Datei mit dem Namen `package.json` und dem folgenden Inhalt. Speichern Sie die Datei im Verzeichnis `nodejs-app`.

```
{
  "name": "mycompany-app",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"run some tests here\""
  },
  "author": "",
  "license": "ISC"
```

```
}
```

7. Erstellen Sie eine Datei mit dem Namen `buildspec.yml` und dem folgenden Inhalt. Speichern Sie die Datei im Verzeichnis `my-source` auf der gleichen Ebene wie die Verzeichnisse `nodejs-app` und `golang-app`. `runtime-versions`In diesem Abschnitt werden die Laufzeiten Node.js Version 12 und Go Version 1.13 angegeben.

```
version: 0.2

phases:
  install:
    runtime-versions:
      golang: 1.13
      nodejs: 12
  build:
    commands:
      - echo Building the Go code...
      - cd $CODEBUILD_SRC_DIR/golang-app
      - go build hello.go
      - echo Building the Node code...
      - cd $CODEBUILD_SRC_DIR/nodejs-app
      - npm run test
artifacts:
  secondary-artifacts:
    golang_artifacts:
      base-directory: golang-app
      files:
        - hello
    nodejs_artifacts:
      base-directory: nodejs-app
      files:
        - index.js
        - package.json
```

8. Ihre Dateistruktur sollte nun wie folgt aussehen:

```
my-source
### golang-app
#   ### hello.go
### nodejs.app
#   ### index.js
#   ### package.json
```

```
### buildspec.yml
```

9. Laden Sie den Inhalt des `my-source` Verzeichnisses in einen S3-Eingabe-Bucket oder ein CodeCommit, GitHub, oder Bitbucket-Repository hoch.

> **Important**
>
> Wenn Sie einen S3-Empfangs-Bucket verwenden, sollten Sie eine ZIP-Datei erstellen, die die Verzeichnisstruktur und die Dateien enthält, und diese dann in den Empfangs-Bucket hochladen. Fügen Sie `my-source` nicht zur ZIP-Datei hinzu, sondern nur die Verzeichnisse und Dateien in `my-source`.

10. Öffnen Sie die AWS CodeBuild Konsole unter https://console.aws.amazon.com/codesuite/codebuild/home.
11. Erstellen Sie ein Build-Projekt. Weitere Informationen finden Sie unter Erstellen Sie ein Build-Projekt (Konsole) und Ausführen eines Build (Konsole). Übernehmen Sie für alle Einstellungen die Standardwerte, außer für folgende Einstellungen:
    - Für Environment (Umgebung):
      - Wählen Sie für Environment image (Umgebungs-Abbild) die Option Managed image (Verwaltetes Abbild) aus.
      - Wählen Sie für Operating system (Betriebssystem) die Option Amazon Linux 2 aus.
      - Wählen Sie unter Runtime (Laufzeit) die Option Standard aus.
      - Wählen Sie für Image aws/codebuild/amazonlinux2-x86_64-standard:4.0.
12. Wählen Sie Create build project (Build-Projekt erstellen) aus.
13. Wählen Sie Start build (Build starten).
14. Übernehmen Sie in der Build configuration (Build-Konfiguration) die Standardeinstellungen und wählen Sie dann Start build (Build starten).
15. Überprüfen Sie die Build-Ausgabe unter der Registerkarte Build logs (Build-Protokolle), wenn der Build abgeschlossen ist. Die Ausgabe sollte in etwa wie folgt aussehen: Es wird die Ausgabe der Go- und Node.js-Laufzeiten angezeigt. Es wird auch die Ausgabe der Go- und Node.js-Anwendungen angezeigt.

```
[Container] Date Time Processing environment variables
[Container] Date Time Selecting 'golang' runtime version '1.13' based on manual
 selections...
```

```
[Container] Date Time Selecting 'nodejs' runtime version '12' based on manual
 selections...
[Container] Date Time Running command echo "Installing Go version 1.13 ..."
Installing Go version 1.13 ...

[Container] Date Time Running command echo "Installing Node.js version 12 ..."
Installing Node.js version 12 ...

[Container] Date Time Running command n $NODE_12_VERSION
   installed : v12.20.1 (with npm 6.14.10)

[Container] Date Time Moving to directory /codebuild/output/src819694850/src
[Container] Date Time Registering with agent
[Container] Date Time Phases found in YAML: 2
[Container] Date Time  INSTALL: 0 commands
[Container] Date Time  BUILD: 1 commands
[Container] Date Time Phase complete: DOWNLOAD_SOURCE State: SUCCEEDED
[Container] Date Time Phase context status code:  Message:
[Container] Date Time Entering phase INSTALL
[Container] Date Time Phase complete: INSTALL State: SUCCEEDED
[Container] Date Time Phase context status code:  Message:
[Container] Date Time Entering phase PRE_BUILD
[Container] Date Time Phase complete: PRE_BUILD State: SUCCEEDED
[Container] Date Time Phase context status code:  Message:
[Container] Date Time Entering phase BUILD
[Container] Date Time Running command echo Building the Go code...
Building the Go code...

[Container] Date Time Running command cd $CODEBUILD_SRC_DIR/golang-app

[Container] Date Time Running command go build hello.go

[Container] Date Time Running command echo Building the Node code...
Building the Node code...

[Container] Date Time Running command cd $CODEBUILD_SRC_DIR/nodejs-app

[Container] Date Time Running command npm run test

> mycompany-app@1.0.0 test /codebuild/output/src924084119/src/nodejs-app
> echo "run some tests here"

run some tests here
```

Beispiel für eine Quellversion mit AWS CodeBuild

In diesem Beispiel wird gezeigt, wie Sie eine Version Ihrer Quelle mithilfe eines anderen Formats als einer Commit-ID (auch bekannt als Commit-SHA) angeben können. Sie haben folgende Möglichkeiten, die Version Ihrer Quelle anzugeben:

- Verwenden Sie für einen Amazon S3 S3-Quellanbieter die Versions-ID des Objekts, das die Build-Eingabe-ZIP-Datei darstellt.
- Verwenden Sie für CodeCommit Bitbucket und GitHub Enterprise Server eine der folgenden Optionen: GitHub
  - Pull-Anforderung als Pull-Anforderungsreferenz (z. B. `refs/pull/1/head`).
  - Branch als Branch-Name.
  - Commit-ID.
  - Tag.
  - Referenz und eine Commit-ID. Die Referenz kann Folgendes sein:
    - Ein Tag (z. B. `refs/tags/mytagv1.0^{full-commit-SHA}`).
    - Ein Branch (z. B. `refs/heads/mydevbranch^{full-commit-SHA}`).
    - Eine Pull-Anforderung (z. B. `refs/pull/1/head^{full-commit-SHA}`).
- Verwende für GitLab und GitLab Self Managed eine der folgenden Optionen:
  - Branch als Branch-Name.
  - Commit-ID.
  - Tag.

> **Note**
>
> Sie können die Version einer Pull-Request-Quelle nur angeben, wenn es sich bei Ihrem Repository um einen GitHub Enterprise Server handelt GitHub .

Wenn Sie eine Referenz und eine Commit-ID zum Angeben einer Version verwenden, ist die `DOWNLOAD_SOURCE`-Phase Ihres Builds schneller, als wenn Sie nur die Version angeben. Das liegt daran, dass beim Hinzufügen einer Referenz CodeBuild nicht das gesamte Repository heruntergeladen werden muss, um den Commit zu finden.

- Sie können eine Quellversion mit nur einer Commit-ID angeben, wie z. B. `1234567890123456789012346789012345678 9`. Wenn Sie dies tun, CodeBuild müssen Sie das gesamte Repository herunterladen, um die Version zu finden.
- Sie können eine Quellversion mit einer Referenz und einer Commit-ID im folgenden Format angeben: *`refs/heads/branchname`*`^{`*`full-commit-SHA`*`}` (z. B. `refs/heads/main^{1234567890123456789012346789012345678 9}`). Wenn Sie dies tun, wird nur der angegebene Zweig CodeBuild heruntergeladen, um die Version zu finden.

> **Note**
>
> Um die `DOWNLOAD_SOURCE` Phase Ihres Builds zu beschleunigen, können Sie die Git-Klontiefe auch auf einen niedrigen Wert setzen. CodeBuild lädt weniger Versionen deines Repositorys herunter.

Um eine GitHub Repository-Version mit einer Commit-ID anzugeben

1. Öffnen Sie die AWS CodeBuild Konsole unter https://console.aws.amazon.com/codesuite/codebuild/home.
2. Erstellen Sie ein Build-Projekt. Weitere Informationen finden Sie unter Erstellen Sie ein Build-Projekt (Konsole) und Ausführen eines Build (Konsole). Übernehmen Sie für alle Einstellungen die Standardwerte, außer für folgende Einstellungen:
   - In Source (Quelle):
     - Wählen Sie als Quellanbieter die Option GitHub. Wenn Sie nicht verbunden sind GitHub, folgen Sie den Anweisungen, um eine Verbindung herzustellen.
     - Wählen Sie für Repository (Repository) die Option Public Repository (öffentliche Repository) aus.
     - Geben Sie als Repository URL (Repository-URL) die URL **`https://github.com/aws/aws-sdk-ruby.git`** ein.
   - In Environment (Umgebung):
     - Wählen Sie für Environment image (Umgebungs-Abbild) die Option Managed image (Verwaltetes Abbild) aus.
     - Wählen Sie für Operating system (Betriebssystem) die Option Amazon Linux 2 aus.
     - Wählen Sie unter Runtime (Laufzeit) die Option Standard aus.

- Wählen Sie für Image aws/codebuild/amazonlinux2-x86_64-standard:4.0.

3. Wählen Sie unter Build specification (Build-Spezifikationen) die Option Insert build commands (Build-Befehle einfügen) und anschließend Switch to editor (Zum Editor wechseln) aus.
4. Ersetzen Sie unter Build commands (Build-Befehle) den Platzhalter mit dem folgenden Text:

```
version: 0.2

phases:
  install:
    runtime-versions:
      ruby: 2.6
  build:
    commands:
       - echo $CODEBUILD_RESOLVED_SOURCE_VERSION
```

   Der `runtime-versions`-Abschnitt ist erforderlich, wenn Sie das Ubuntu Standard-Image 2.0 verwenden. Hier ist die Ruby Version 2.6-Laufzeit angegeben, Sie können jedoch eine beliebige Laufzeit verwenden. Der `echo`-Befehl zeigt die Version des Quellcodes an, der in der `CODEBUILD_RESOLVED_SOURCE_VERSION`-Umgebungsvariable gespeichert ist.
5. Übernehmen Sie in der Build configuration (Build-Konfiguration) die Standardeinstellungen und wählen Sie dann Start build (Build starten).
6. Geben Sie für Source version (Quellversion) Folgendes ein: **`046e8b67481d53bdc86c3f6affdd5d1afae6d369`**. Dies ist der SHA eines Commits im `https://github.com/aws/aws-sdk-ruby.git`-Repository.
7. Wählen Sie Start build (Build starten).
8. Wenn die Erstellung abgeschlossen ist, sollten Sie Folgendes sehen:

   - Auf der Registerkarte Build logs (Build-Protokolle: die verwendete Version der Projektquelle Ein Beispiel.

     ```
     [Container] Date Time Running command echo $CODEBUILD_RESOLVED_SOURCE_VERSION
     046e8b67481d53bdc86c3f6affdd5d1afae6d369

     [Container] Date Time Phase complete: BUILD State: SUCCEEDED
     ```

   - Auf der Registerkarte Environment variables (Umgebungsvariablen): Die Resolved source version (Aufgelöste Quellversion) stimmt mit der zur Erstellung des Builds verwendeten Commit-ID überein.

- Auf der Registerkarte Phase details (Phasendetails): die Dauer der DOWNLOAD_SOURCE-Phase.

Diese Schritte zeigen, wie Sie einen Build mit derselben Version der Quelle erstellen. Dieses Mal wird die Version der Quelle mithilfe einer Referenz mit der Commit-ID angegeben.

Um eine GitHub Repository-Version mit einer Commit-ID und einer Referenz anzugeben

1. Wählen Sie im linken Navigationsbereich Build projects (Build-Projekte) und anschließend das Projekt, das Sie vorher erstellt haben, aus.
2. Wählen Sie Start build (Build starten).
3. Geben Sie in das Feld Source version (Quellversion) Folgendes ein: **`refs/heads/main^{046e8b67481d53bdc86c3f6affdd5d1afae6d369}`**. Dies ist dieselbe Commit-ID und eine Referenz auf einen Branch im Format *`refs`*`/`*`heads`*`/`*`branchname`*`^{`*`full-commit-SHA`*`}`.
4. Wählen Sie Start build (Build starten).
5. Wenn die Erstellung abgeschlossen ist, sollten Sie Folgendes sehen:

   - Auf der Registerkarte Build logs (Build-Protokolle: die verwendete Version der Projektquelle Ein Beispiel.

     ```
     [Container] Date Time Running command echo $CODEBUILD_RESOLVED_SOURCE_VERSION
     046e8b67481d53bdc86c3f6affdd5d1afae6d369

     [Container] Date Time Phase complete: BUILD State: SUCCEEDED
     ```

   - Auf der Registerkarte Environment variables (Umgebungsvariablen): Die Resolved source version (Aufgelöste Quellversion) stimmt mit der zur Erstellung des Builds verwendeten Commit-ID überein.
   - Auf der Registerkarte Phase details (Phasendetails) sollte die Dauer der DOWNLOAD_SOURCE-Phase kürzer sein als die Dauer, wenn Sie nur die Commit-ID zum Angeben der Version Ihrer Quelle verwendeten.

Quell-Repository-Beispiele von Drittanbietern für CodeBuild

Themen

- Beispiel für eine Bitbucket-Pull-Anfrage und einen Webhook-Filter für CodeBuild
- GitHub Enterprise Server-Beispiel für CodeBuild
- GitHub Beispiel für einen Pull-Request und einen Webhook-Filter für CodeBuild

Beispiel für eine Bitbucket-Pull-Anfrage und einen Webhook-Filter für CodeBuild

AWS CodeBuild unterstützt Webhooks, wenn das Quell-Repository Bitbucket ist. Das bedeutet, dass bei einem CodeBuild Build-Projekt, dessen Quellcode in einem Bitbucket-Repository gespeichert ist, Webhooks verwendet werden können, um den Quellcode jedes Mal neu zu erstellen, wenn eine Codeänderung in das Repository übertragen wird. Weitere Informationen finden Sie unter Bitbucket-Webhook-Ereignisse.

Dieses Beispiel zeigt, wie Sie eine Pull-Anfrage mit einem Bitbucket-Repository erstellen. Es zeigt dir auch, wie du einen Bitbucket-Webhook als Trigger verwendest, um einen Build eines Projekts CodeBuild zu erstellen.

> **Note**
>
> Bei der Verwendung von Webhooks ist es möglich, dass ein Benutzer einen unerwarteten Build auslöst. Informationen zur Minderung dieses Risikos finden Sie unter. Bewährte Methoden für die Verwendung von Webhooks

Themen

- Voraussetzungen
- Erstellen Sie ein Build-Projekt mit Bitbucket als Quell-Repository und aktivieren Sie Webhooks.
- Auslösen eines Builds mit einem Bitbucket-Webhook

Voraussetzungen

Um dieses Beispiel auszuführen, musst du dein AWS CodeBuild Projekt mit deinem Bitbucket-Konto verbinden.

> **Note**
>
> CodeBuild hat seine Berechtigungen mit Bitbucket aktualisiert. Wenn du dein Projekt zuvor mit Bitbucket verbunden hast und jetzt ein Bitbucket-Verbindungsfehler angezeigt wird, musst

du die Verbindung erneut herstellen, um die CodeBuild Erlaubnis zur Verwaltung deiner Webhooks zu erteilen.

Erstellen Sie ein Build-Projekt mit Bitbucket als Quell-Repository und aktivieren Sie Webhooks.

In den folgenden Schritten wird beschrieben, wie du ein AWS CodeBuild Projekt mit Bitbucket als Quell-Repository erstellst und Webhooks aktivierst.

1. Öffne die AWS CodeBuild Konsole unter https://console.aws.amazon.com/codesuite/codebuild/home.
2. Wenn eine CodeBuild Informationsseite angezeigt wird, wählen Sie Create build project aus. Erweitern Sie andernfalls im Navigationsbereich Build, wählen Sie Build projects und dann Create build project aus.
3. Wählen Sie Create build project (Build-Projekt erstellen) aus.
4. In Project configuration (Projektkonfiguration):

   Project name

   Geben Sie einen Namen für dieses Build-Projekt ein. Die Namen der Build-Projekte müssen für jedes AWS Konto eindeutig sein. Sie können auch eine optionale Beschreibung des Build-Projekts hinzufügen, damit andere Benutzer verstehen, wofür dieses Projekt verwendet wird.
5. In Source (Quelle):

   Quellanbieter

   Wähle Bitbucket. Folge den Anweisungen, um dich mit Bitbucket zu verbinden (oder erneut zu verbinden) und wähle dann Autorisieren.

   Repository

   Wähle in meinem Bitbucket-Konto Repository aus.

   Falls du dich noch nicht mit deinem Bitbucket-Konto verbunden hast, gib deinen Bitbucket-Nutzernamen und dein App-Passwort ein und wähle Bitbucket-Anmeldeinformationen speichern aus.

   Bitbucket-Repository

   Gib die URL für dein Bitbucket-Repository ein.

6. Wähle unter Primäre Quell-Webhook-Ereignisse Folgendes aus.

> **Note**
>
> Der Abschnitt „Primäre Webhook-Ereignisse“ ist nur sichtbar, wenn du im vorherigen Schritt „Repository in meinem Bitbucket-Konto“ ausgewählt hast.

1. Wählen Sie beim Erstellen Ihres Projekts Rebuild every time a code change is pushed to this repository (Erneut erstellen, wenn eine Codeänderung an dieses Repository übergeben wird) aus.
2. Wählen Sie unter Event type (Ereignistyp) eines oder mehrere Ereignisse aus.
3. Wenn Sie Fälle filtern möchten, in denen ein Ereignis einen Build auslöst, fügen Sie unter Start a build under these conditions (Unter diesen Bedingungen Build starten) einen oder mehrere optionale Filter hinzu.
4. Wenn Sie Fälle filtern möchten, in denen kein Ereignis ausgelöst wird, fügen Sie unter Don't start a build under these conditions (Unter diesen Bedingungen keinen Build starten) einen oder mehrere optionale Filter hinzu.
5. Wähle Filtergruppe hinzufügen, um bei Bedarf eine weitere Filtergruppe hinzuzufügen.

Weitere Informationen zu Bitbucket-Webhook-Ereignistypen und Filtern findest du unter. Bitbucket-Webhook-Ereignisse

7. In Environment (Umgebung):

Bild der Umgebung

Wählen Sie eine der folgenden Optionen aus:

Um ein Docker-Image zu verwenden, das verwaltet wird von AWS CodeBuild:

Wählen Sie Verwaltetes Image und wählen Sie dann Betriebssystem, Runtime (s), Image und Image-Version aus. Treffen Sie eine Auswahl unter Environment type (Umgebungstyp), sofern verfügbar.

Um ein anderes Docker-Image zu verwenden:

Wählen Sie Benutzerdefiniertes Bild. Wählen Sie als Umgebungstyp ARM, Linux, Linux GPU oder Windows aus. Wenn Sie Andere Registrierung wählen, geben Sie für Externe

Registrierungs-URL den Namen und das Tag des Docker-Images in Docker Hub ein. Verwenden Sie dabei das Format. *docker repository*/*docker image name* Wenn Sie sich für Amazon ECR entscheiden, verwenden Sie das Amazon ECR-Repository und das Amazon ECR-Image, um das Docker-Image in Ihrem Konto auszuwählen. AWS

Um ein privates Docker-Image zu verwenden:

Wählen Sie Benutzerdefiniertes Bild. Wählen Sie als Umgebungstyp ARM, Linux, Linux GPU oder Windows aus. Wählen Sie unter Image registry (Abbildregistrierung) die Option Other registry (Andere Registrierung) aus und geben Sie dann den ARN der Anmeldeinformationen für Ihr privates Docker-Image ein. Die Anmeldeinformationen müssen von Secrets Manager erstellt werden. Weitere Informationen finden Sie unter Was ist AWS Secrets Manager? im AWS Secrets Manager Benutzerhandbuch.

Rolle im Dienst

Wählen Sie eine der folgenden Optionen aus:

- Wenn Sie keine CodeBuild Servicerolle haben, wählen Sie Neue Servicerolle. Geben Sie im Feld Rollenname einen Namen für die neue Rolle ein.
- Wenn Sie eine CodeBuild Servicerolle haben, wählen Sie Bestehende Servicerolle aus. Wählen Sie unter Role ARN die Servicerolle aus.

> **Note**
>
> Wenn Sie die Konsole verwenden, um ein Build-Projekt zu erstellen oder zu aktualisieren, können Sie gleichzeitig eine CodeBuild Servicerolle erstellen. In der Standardeinstellung funktioniert diese Rolle ausschließlich mit diesem Projekt. Wenn Sie die Konsole verwenden, um die Servicerolle mit einem anderen Build-Projekt zu verknüpfen, wird die Rolle so aktualisiert, dass sie mit dem anderen Build-Projekt funktioniert. Eine Servicerolle kann in bis zu zehn Build-Projekten verwendet werden.

8. Führen Sie in Buildspec einen der folgenden Schritte aus:

   - Wählen Sie Buildspec-Datei verwenden, um die Datei buildspec.yml im Quellcode-Stammverzeichnis zu verwenden.
   - Wählen Sie Build-Befehle einfügen, um die Konsole zum Einfügen von Build-Befehlen zu verwenden.

   Weitere Informationen hierzu finden Sie unter Build-Spezifikationsreferenz.

9. In Artifacts (Artefakte):

   Typ

   Wählen Sie eine der folgenden Optionen aus:

   - Wenn keine Build-Ausgabeartefakte erstellt werden sollen, klicken Sie auf die Option No artifacts (Keine Artefakte).
   - Um die Build-Ausgabe in einem S3-Bucket zu speichern, wählen Sie Amazon S3 und gehen Sie dann wie folgt vor:
     - Lassen Sie Name leer, wenn Sie den Projektnamen für die ZIP-Datei mit der Build-Ausgabe verwenden möchten. Geben Sie andernfalls den Namen ein. Standardmäßig ist der Artefaktname der Projektname. Wenn Sie einen anderen Namen verwenden möchten, geben Sie diesen in das Feld für den Artefaktnamen ein. Wenn Sie eine ZIP-Datei ausgeben möchten, schließen Sie die ZIP-Erweiterung mit ein.
     - Wählen Sie für Bucket name den Namen des Ausgabe-Buckets aus.
     - Wenn Sie in diesem Vorgang zuvor die Option Insert build commands (Build-Befehle einfügen) verwendet haben, geben Sie für Output files (Ausgabedateien) die Speicherorte der Build-Dateien ein, die in der ZIP-Datei oder in dem Ordner für die Build-Ausgabe enthalten sein sollen. Bei mehreren Speicherorten trennen Sie die einzelnen Speicherorte durch ein Komma, (wie z. B. `appspec.yml, target/my-app.jar`). Weitere Informationen finden Sie in der Beschreibung von `files` in Syntax der Build-Spezifikation.

   Zusätzliche Konfiguration

   Erweitern Sie Additional configuration (Zusätzliche Einstellungen) und legen Sie die entsprechenden Optionen fest.

10. Wählen Sie Create build project (Build-Projekt erstellen) aus. Klicken Sie auf der Seite Review (Überprüfen) auf Start build (Build starten), um den Build auszuführen.

Auslösen eines Builds mit einem Bitbucket-Webhook

AWS CodeBuild Erstellt bei einem Projekt, das Bitbucket-Webhooks verwendet, einen Build, wenn das Bitbucket-Repository eine Änderung in deinem Quellcode feststellt.

1. Öffne die AWS CodeBuild Konsole unter https://console.aws.amazon.com/codesuite/codebuild/home.

2. Wählen Sie im Navigationsbereich Build projects (Build-Projekte) und dann ein Projekt aus, das mit einem Bitbucket-Repository mit Webhooks verknüpft ist. Informationen zum Erstellen eines Bitbucket-Webhook-Projekts findest du unter. the section called “Erstellen Sie ein Build-Projekt mit Bitbucket als Quell-Repository und aktivieren Sie Webhooks.”
3. Nehmen Sie einige Änderungen am Code in Ihrem Bitbucket-Repository des Projekts vor.
4. Erstellen Sie eine Pull-Anfrage in Ihrem Bitbucket-Repository. Weitere Informationen finden Sie auf der Seite zum Durchführen einer Pull-Anfrage.
5. Wählen Sie auf der Seite über Bitbucket-Webhooks View request (Anfrage anzeigen) aus, um eine Liste der jüngsten Ereignisse anzuzeigen.
6. Wähle Details anzeigen, um Details zu der Antwort von zu sehen. CodeBuild Er kann wie folgt aussehen:

```
"response":"Webhook received and build started: https://us-
east-1.console.aws.amazon.com/codebuild/home..."
"statusCode":200
```

7. Navigieren Sie zur Seite über Bitbucket Pull-Anfragen, um den Build-Status anzuzeigen.

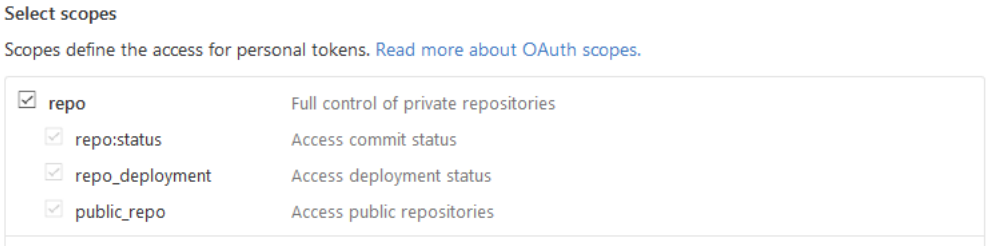
GitHub Enterprise Server-Beispiel für CodeBuild

AWS CodeBuild unterstützt GitHub Enterprise Server als Quell-Repository. Dieses Beispiel zeigt, wie Sie Ihre CodeBuild Projekte einrichten, wenn in Ihrem GitHub Enterprise Server-Repository ein Zertifikat installiert ist. Es zeigt auch, wie Sie Webhooks aktivieren, sodass der Quellcode jedes Mal CodeBuild neu erstellt wird, wenn eine Codeänderung in Ihr GitHub Enterprise Server-Repository übertragen wird.

Voraussetzungen

1. Generieren Sie ein persönliches Zugriffstoken für Ihr CodeBuild Projekt. Wir empfehlen Ihnen, einen GitHub Enterprise-Benutzer zu erstellen und ein persönliches Zugriffstoken für diesen Benutzer zu generieren. Kopieren Sie es in Ihre Zwischenablage, damit Sie es bei der Erstellung Ihres CodeBuild Projekts verwenden können. Weitere Informationen finden Sie auf der GitHub Hilfeseite unter Erstellen eines persönlichen Zugriffstokens für die Befehlszeile.

   Bei der Erstellung der persönlichen Zugriffstoken nehmen Sie den repo-Umfang in die Definition auf.



2. Laden Sie Ihr Zertifikat von GitHub Enterprise Server herunter. CodeBuild verwendet das Zertifikat, um eine vertrauenswürdige SSL-Verbindung zum Repository herzustellen.

   Linux/macOS-Clients:

   Führen Sie in einem Terminalfenster den folgenden Befehl aus:

   ```
   echo -n | openssl s_client -connect HOST:PORTNUMBER \
       | sed -ne '/-BEGIN CERTIFICATE-/,/-END CERTIFICATE-/p' > /folder/filename.pem
   ```

   Ersetzen Sie die Platzhalter im Befehl durch die folgenden Werte:

   HOST. Die IP-Adresse Ihres GitHub Enterprise Server-Repositorys.

   PORTNUMBER. Die Port-Nummer, die Sie für die Verbindung verwenden (z. B. 443).

   folder. Der Ordner, in den Sie das Zertifikat heruntergeladen haben.

   filename. Der Dateiname Ihrer Zertifikatdatei.

   > **Important**
   >
   > Speichern Sie das Zertifikat als .pem-Datei.

   Windows-Clients:

   Verwenden Sie Ihren Browser, um Ihr Zertifikat von GitHub Enterprise Server herunterzuladen. Wenn Sie die Zertifikatdetails der Website sehen, wählen Sie das Vorhängeschlosssymbol. Weitere Informationen zum Exportieren des Zertifikats finden Sie in der Dokumentation zu Ihrem Browser.

> **Important**
>
> Speichern Sie das Zertifikat als .pem-Datei.

3. Laden Sie Ihre Zertifikatdatei in einen S3-Bucket hoch. Weitere Informationen zum Erstellen eines S3-Buckets finden Sie unter Erstellen eines S3-Buckets. Weitere Informationen zum Hochladen von Objekten in einen S3-Bucket finden Sie unter Dateien und Ordner in einen Bucket hochladen.

> **Note**
>
> Dieser Bucket muss sich in derselben AWS Region wie Ihre Builds befinden. Wenn Sie beispielsweise angeben, dass ein Build in der Region USA Ost (Ohio) ausgeführt werden soll, muss sich der Bucket in der Region USA Ost (Ohio) befinden. CodeBuild

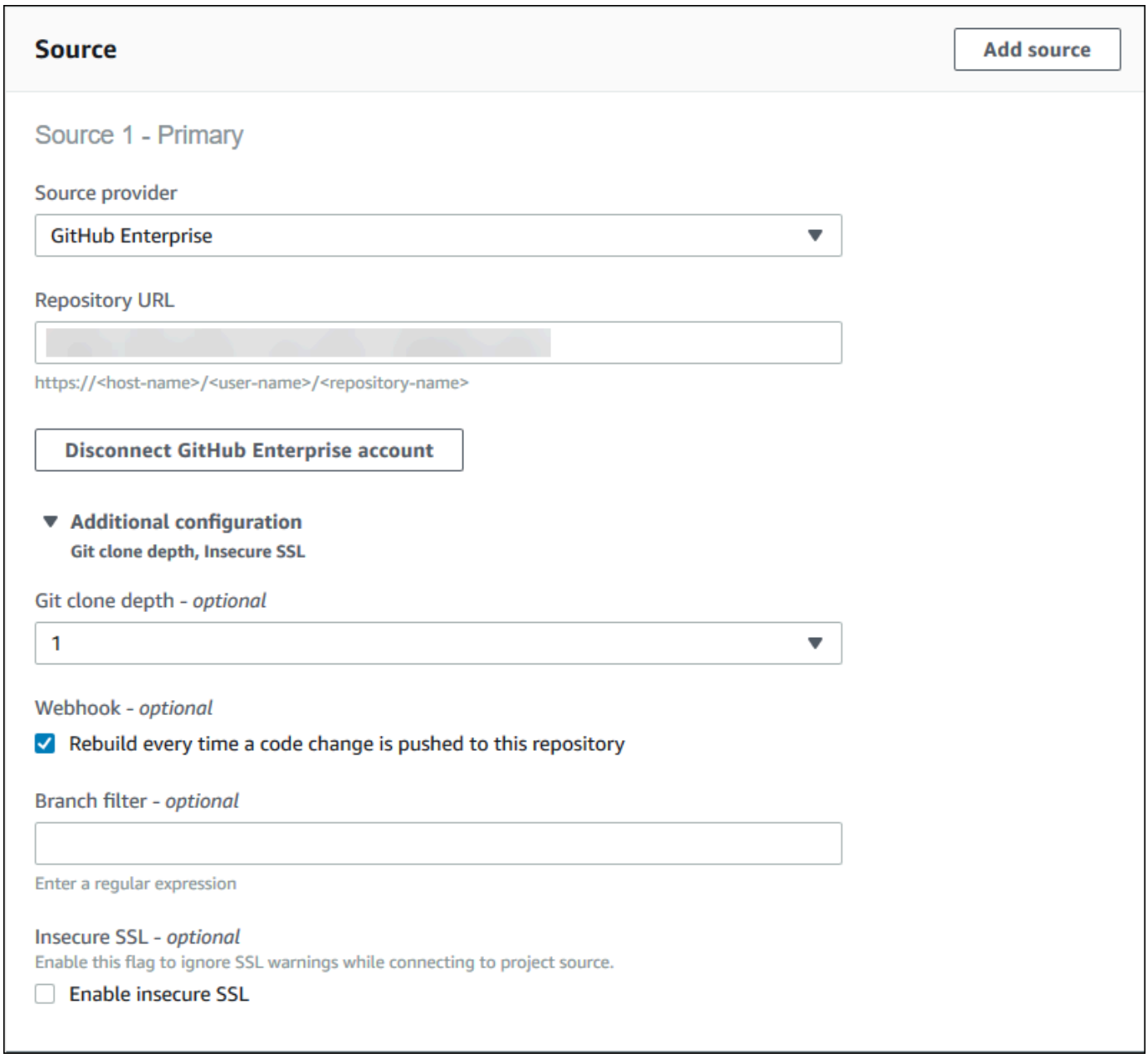
Erstellen Sie ein Build-Projekt mit GitHub Enterprise Server als Quell-Repository und aktivieren Sie Webhooks (Konsole)

1. Öffnen Sie die AWS CodeBuild Konsole unter https://console.aws.amazon.com/codesuite/codebuild/home.
2. Wenn eine CodeBuild Informationsseite angezeigt wird, wählen Sie Create build project aus. Erweitern Sie andernfalls im Navigationsbereich Build, wählen Sie Build projects und dann Create build project aus.
3. Geben Sie unter Project name (Projektname) einen Namen für dieses Build-Projekt ein. Die Namen von Build-Projekten müssen für jedes AWS Konto eindeutig sein. Sie können auch eine optionale Beschreibung des Build-Projekts hinzufügen, damit andere Benutzer verstehen, wofür dieses Projekt verwendet wird.
4. Wählen Sie unter Quelle unter Quellanbieter die Option GitHub Enterprise aus.
   - Für Personal Access Token fügen Sie das Token ein, das Sie in Ihre Zwischenablage kopiert haben, und wählen Save Token. Geben Sie unter Repository-URL die URL für Ihr GitHub Enterprise Server-Repository ein.

> **Note**
>
> Sie müssen das private Zugriffstoken nur einmal eingeben und speichern. Alle future AWS CodeBuild Projekte verwenden dieses Token.

- Geben Sie für Repository URL (Repository-URL) den Pfad zu Ihrem Repository ein, einschließlich des Namens des Repositorys.
- Erweitern Sie Additional configuration (Zusätzliche Konfiguration).
- Wählen Sie Rebuild every time a code change is pushed to this repository (Bei jeder Veröffentlichung einer Codeänderung in diesem Repository neu erstellen) aus, um bei jeder Veröffentlichung einer Codeänderung in diesem Repository einen neuen Build zu erstellen.
- Wählen Sie Unsicheres SSL aktivieren, um SSL-Warnungen zu ignorieren, während Sie eine Verbindung zu Ihrem GitHub Enterprise Server-Projekt-Repository herstellen.

> **Note**
>
> Wir empfehlen, dass Sie Enable insecure SSL (Unsicheres SSL aktivieren) nur für Tests verwenden. Es sollte nicht in einer Produktionsumgebung verwendet werden.



5. In Environment (Umgebung):

   Führen Sie für Environment image (Umgebungs-Image) einen der folgenden Schritte aus:

   - Um ein Docker-Image zu verwenden, das von verwaltet wird AWS CodeBuild, wählen Sie Verwaltetes Image und wählen Sie dann unter Betriebssystem, Runtime (s), Image und Image-Version aus. Treffen Sie eine Auswahl unter Environment type (Umgebungstyp), sofern verfügbar.

- Wenn Sie ein anderes Docker-Image verwenden möchten, wählen Sie Custom image (Benutzerdefiniertes Image) aus. Wählen Sie als Umgebungstyp ARM, Linux, Linux GPU oder Windows aus. Wenn Sie Andere Registrierung wählen, geben Sie für Externe Registrierungs-URL den Namen und das Tag des Docker-Images in Docker Hub ein. Verwenden Sie dabei das Format. *docker repository*/*docker image name* Wenn Sie sich für Amazon ECR entscheiden, verwenden Sie das Amazon ECR-Repository und das Amazon ECR-Image, um das Docker-Image in Ihrem Konto auszuwählen. AWS
- Um ein privates Docker-Image zu verwenden, wählen Sie Benutzerdefiniertes Image. Wählen Sie als Umgebungstyp ARM, Linux, Linux GPU oder Windows aus. Wählen Sie unter Image registry (Abbildregistrierung) die Option Other registry (Andere Registrierung) aus und geben Sie dann den ARN der Anmeldeinformationen für Ihr privates Docker-Image ein. Die Anmeldeinformationen müssen von Secrets Manager erstellt werden. Weitere Informationen finden Sie unter Was ist AWS Secrets Manager? im AWS Secrets Manager - Benutzerhandbuch.

6. Führen Sie unter Service role (Service-Rolle) einen der folgenden Schritte aus:

   - Wenn Sie keine CodeBuild Servicerolle haben, wählen Sie Neue Servicerolle. Geben Sie im Feld Rollenname einen Namen für die neue Rolle ein.
   - Wenn Sie eine CodeBuild Servicerolle haben, wählen Sie Bestehende Servicerolle aus. Wählen Sie unter Role ARN die Servicerolle aus.

   > **Note**
   >
   > Wenn Sie die Konsole verwenden, um ein Build-Projekt zu erstellen oder zu aktualisieren, können Sie gleichzeitig eine CodeBuild Servicerolle erstellen. In der Standardeinstellung funktioniert diese Rolle ausschließlich mit diesem Projekt. Wenn Sie die Konsole verwenden, um die Servicerolle mit einem anderen Build-Projekt zu verknüpfen, wird die Rolle so aktualisiert, dass sie mit dem anderen Build-Projekt funktioniert. Eine Servicerolle kann in bis zu zehn Build-Projekten verwendet werden.

7. Erweitern Sie Additional configuration (Zusätzliche Konfiguration).

   Wenn Sie mit Ihrer VPC arbeiten möchten CodeBuild :

   - Wählen Sie für VPC die VPC-ID aus, die CodeBuild verwendet wird.

- Wählen Sie für VPC-Subnetze die Subnetze aus, die Ressourcen enthalten, die verwendet werden. CodeBuild
- Wählen Sie für VPC-Sicherheitsgruppen die Sicherheitsgruppen aus, die CodeBuild den Zugriff auf Ressourcen in den VPCs ermöglichen.

Weitere Informationen finden Sie unter Verwendung AWS CodeBuild mit Amazon Virtual Private Cloud.

8. Führen Sie in Buildspec einen der folgenden Schritte aus:

   - Wählen Sie Buildspec-Datei verwenden, um die Datei buildspec.yml im Quellcode-Stammverzeichnis zu verwenden.
   - Wählen Sie Build-Befehle einfügen, um die Konsole zum Einfügen von Build-Befehlen zu verwenden.

   Weitere Informationen hierzu finden Sie unter Build-Spezifikationsreferenz.

9. Führen Sie unter Artifacts (Artefakte) für Type (Typ) einen der folgenden Schritte aus:

   - Wenn keine Build-Ausgabeartefakte erstellt werden sollen, klicken Sie auf die Option No artifacts (Keine Artefakte).
   - Um die Build-Ausgabe in einem S3-Bucket zu speichern, wählen Sie Amazon S3 und gehen Sie dann wie folgt vor:
     - Lassen Sie Name leer, wenn Sie den Projektnamen für die ZIP-Datei mit der Build-Ausgabe verwenden möchten. Geben Sie andernfalls den Namen ein. Standardmäßig ist der Artefaktname der Projektname. Wenn Sie einen anderen Namen verwenden möchten, geben Sie diesen in das Feld für den Artefaktnamen ein. Wenn Sie eine ZIP-Datei ausgeben möchten, schließen Sie die ZIP-Erweiterung mit ein.
     - Wählen Sie für Bucket name den Namen des Ausgabe-Buckets aus.
     - Wenn Sie in diesem Vorgang zuvor die Option Insert build commands (Build-Befehle einfügen) verwendet haben, geben Sie für Output files (Ausgabedateien) die Speicherorte der Build-Dateien ein, die in der ZIP-Datei oder in dem Ordner für die Build-Ausgabe enthalten sein sollen. Bei mehreren Speicherorten trennen Sie die einzelnen Speicherorte durch ein Komma, (wie z. B. `appspec.yml, target/my-app.jar`). Weitere Informationen finden Sie in der Beschreibung von `files` in Syntax der Build-Spezifikation.

10. Wählen Sie für Cache type (Cache-Typ) eine der folgenden Optionen aus:

- Wenn Sie keinen Cache verwenden möchten, wählen Sie No cache.
- Wenn Sie einen Amazon S3-Cache verwenden möchten, wählen Sie Amazon S3 und gehen Sie dann wie folgt vor:
  - Wählen Sie für Bucket den Namen des S3-Buckets, in dem der Cache gespeichert wird.
  - (Optional) Geben Sie für das Cache-Pfadpräfix ein Amazon S3 S3-Pfadpräfix ein. Der Wert für Cache path prefix (Cache-Pfadpräfix) ist mit einem Verzeichnisnamen vergleichbar. Er ermöglicht Ihnen das Speichern des Cache in demselben Verzeichnis eines Buckets.

    Important

    Fügen Sie am Ende des Pfadpräfix keinen abschließenden Schrägstrich (/) an.

- Wenn Sie einen lokalen Cache verwenden möchten, wählen Sie Local (Lokal) und dann mindestens einen lokalen Cache-Modus aus.

  Note

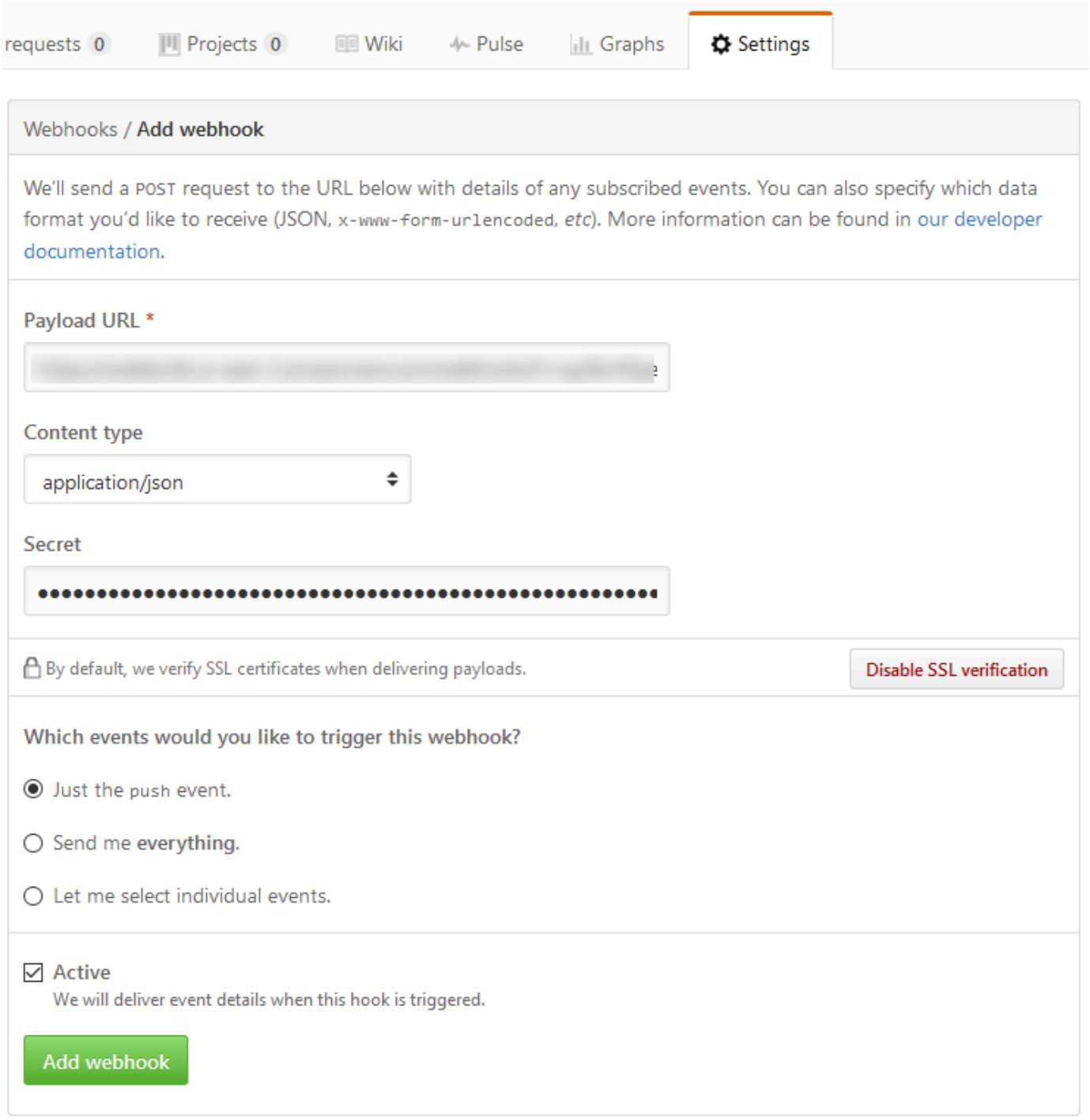
  Der Modus Docker layer cache (Docker-Ebenen-Cache) ist nur für Linux verfügbar. Wenn Sie diesen Modus auswählen, muss Ihr Projekt im privilegierten Modus ausgeführt werden.

Durch die Verwendung eines Caches wird eine erhebliche Ersparnis bei der Erstellungszeit erzielt, da wiederverwendbare Teile der Build-Umgebung im Cache gespeichert und über Builds hinweg verwendet werden. Weitere Informationen über die Angabe eines Cache in der Build-Spezifikationsdatei finden Sie unter Syntax der Build-Spezifikation. Weitere Informationen zum Caching finden Sie unter Build-Caching in AWS CodeBuild.

11. Wählen Sie Create build project (Build-Projekt erstellen) aus. Wählen Sie auf der Build-Projekt-Seite Start build (Build starten) aus.
12. Wenn Sie unter Source (Quelle) Webhooks aktiviert haben, wird das Dialogfeld Create webhook (Webhook erstellen) geöffnet und es werden Werte für Payload URL (Nutzlast-URL) und Secret (Geheim) angezeigt.

> **Important**
>
> Das Dialogfeld Create webhook wird nur einmal angezeigt. Kopieren Sie die Nutzlast-URL und den geheimen Schlüssel. Sie benötigen sie, wenn Sie einen Webhook in GitHub Enterprise Server hinzufügen.
> Wenn Sie erneut eine Payload-URL und einen geheimen Schlüssel generieren müssen, müssen Sie zuerst den Webhook aus Ihrem GitHub Enterprise Server-Repository löschen. Deaktivieren Sie in Ihrem CodeBuild Projekt das Kontrollkästchen Webhook und wählen Sie dann Speichern. Sie können dann ein CodeBuild Projekt erstellen oder aktualisieren, wenn das Webhook-Kontrollkästchen aktiviert ist. Das Dialogfeld Create webhook wird erneut angezeigt.

13. Wählen Sie in GitHub Enterprise Server das Repository aus, in dem Ihr CodeBuild Projekt gespeichert ist.
14. Wählen Sie Settings (Einstellungen), Hooks & services (Hooks und Services) und anschließend Add webhook (Webhook hinzufügen) aus.
15. Geben Sie die Nutzlast-URL und den geheimen Schlüssels ein, übernehmen Sie die Standardeinstellungen für die anderen Felder, und wählen Sie dann Add webhook.



16. Kehren Sie zu Ihrem CodeBuild Projekt zurück. Schließen Sie das Dialogfeld Create webhook und wählen Sie Start build.

GitHub Beispiel für einen Pull-Request und einen Webhook-Filter für CodeBuild

AWS CodeBuild unterstützt Webhooks, wenn das Quell-Repository GitHub Das bedeutet, dass bei einem CodeBuild Build-Projekt, dessen Quellcode in einem GitHub Repository gespeichert ist, Webhooks verwendet werden können, um den Quellcode jedes Mal neu zu erstellen, wenn eine Codeänderung in das Repository übertragen wird. CodeBuild Beispiele finden Sie unter AWS CodeBuild Beispiele.

> **Note**
>
> Bei der Verwendung von Webhooks kann ein Benutzer einen unerwarteten Build auslösen. Informationen zur Minderung dieses Risikos finden Sie unter. Bewährte Methoden für die Verwendung von Webhooks

Erstellen Sie ein Build-Projekt mit GitHub dem Quell-Repository und aktivieren Sie Webhooks (Konsole)

1. Öffnen Sie die AWS CodeBuild Konsole unter https://console.aws.amazon.com/codesuite/codebuild/home.
2. Wenn eine CodeBuild Informationsseite angezeigt wird, wählen Sie Create build project aus. Erweitern Sie andernfalls im Navigationsbereich Build, wählen Sie Build projects und dann Create build project aus.
3. Wählen Sie Create build project (Build-Projekt erstellen) aus.
4. In Project configuration (Projektkonfiguration):

   Project name

   Geben Sie einen Namen für dieses Build-Projekt ein. Die Namen der Build-Projekte müssen für jedes AWS Konto eindeutig sein. Sie können auch eine optionale Beschreibung des Build-Projekts hinzufügen, damit andere Benutzer verstehen, wofür dieses Projekt verwendet wird.
5. In Source (Quelle):

   Quellanbieter

   Wähle GitHub. Folgen Sie den Anweisungen, um eine Verbindung herzustellen (oder erneut herzustellen), GitHub und wählen Sie dann Autorisieren.

Repository

Wählen Sie unter „Mein GitHub Konto“ die Option „Repository“.

GitHub Repositorium

Geben Sie die URL für Ihr GitHub Repository ein.

6. Wählen Sie unter Webhook-Ereignisse der Primärquelle die folgenden Optionen aus.

> **Note**
>
> Der Abschnitt Webhook-Ereignisse der Primärquelle ist nur sichtbar, wenn Sie im vorherigen Schritt Repository in meinem GitHub Konto ausgewählt haben.

1. Wählen Sie beim Erstellen Ihres Projekts Rebuild every time a code change is pushed to this repository (Erneut erstellen, wenn eine Codeänderung an dieses Repository übergeben wird) aus.
2. Wählen Sie unter Event type (Ereignistyp) eines oder mehrere Ereignisse aus.
3. Wenn Sie Fälle filtern möchten, in denen ein Ereignis einen Build auslöst, fügen Sie unter Start a build under these conditions (Unter diesen Bedingungen Build starten) einen oder mehrere optionale Filter hinzu.
4. Wenn Sie Fälle filtern möchten, in denen kein Ereignis ausgelöst wird, fügen Sie unter Don't start a build under these conditions (Unter diesen Bedingungen keinen Build starten) einen oder mehrere optionale Filter hinzu.
5. Wählen Sie Filtergruppe hinzufügen, um bei Bedarf eine weitere Filtergruppe hinzuzufügen.

Weitere Informationen zu GitHub Webhook-Ereignistypen und Filtern finden Sie unterGitHub Webhook-Ereignisse.

7. In Environment (Umgebung):

Bild der Umgebung

Wählen Sie eine der folgenden Optionen aus:

Um ein Docker-Image zu verwenden, das verwaltet wird von AWS CodeBuild:

Wählen Sie Verwaltetes Image und wählen Sie dann Betriebssystem, Runtime (s), Image und Image-Version aus. Treffen Sie eine Auswahl unter Environment type (Umgebungstyp), sofern verfügbar.

Um ein anderes Docker-Image zu verwenden:

Wählen Sie Benutzerdefiniertes Bild. Wählen Sie als Umgebungstyp ARM, Linux, Linux GPU oder Windows aus. Wenn Sie Andere Registrierung wählen, geben Sie für Externe Registrierungs-URL den Namen und das Tag des Docker-Images in Docker Hub ein. Verwenden Sie dabei das Format. *`docker repository`*/*`docker image name`* Wenn Sie sich für Amazon ECR entscheiden, verwenden Sie das Amazon ECR-Repository und das Amazon ECR-Image, um das Docker-Image in Ihrem Konto auszuwählen. AWS

Um ein privates Docker-Image zu verwenden:

Wählen Sie Benutzerdefiniertes Bild. Wählen Sie als Umgebungstyp ARM, Linux, Linux GPU oder Windows aus. Wählen Sie unter Image registry (Abbildregistrierung) die Option Other registry (Andere Registrierung) aus und geben Sie dann den ARN der Anmeldeinformationen für Ihr privates Docker-Image ein. Die Anmeldeinformationen müssen von Secrets Manager erstellt werden. Weitere Informationen finden Sie unter Was ist AWS Secrets Manager? im AWS Secrets Manager Benutzerhandbuch.

Rolle im Dienst

Wählen Sie eine der folgenden Optionen aus:

- Wenn Sie keine CodeBuild Servicerolle haben, wählen Sie Neue Servicerolle. Geben Sie im Feld Rollenname einen Namen für die neue Rolle ein.
- Wenn Sie eine CodeBuild Servicerolle haben, wählen Sie Bestehende Servicerolle aus. Wählen Sie unter Role ARN die Servicerolle aus.

> **Note**
>
> Wenn Sie die Konsole verwenden, um ein Build-Projekt zu erstellen oder zu aktualisieren, können Sie gleichzeitig eine CodeBuild Servicerolle erstellen. In der Standardeinstellung funktioniert diese Rolle ausschließlich mit diesem Projekt. Wenn Sie die Konsole verwenden, um die Servicerolle mit einem anderen Build-Projekt zu

verknüpfen, wird die Rolle so aktualisiert, dass sie mit dem anderen Build-Projekt funktioniert. Eine Servicerolle kann in bis zu zehn Build-Projekten verwendet werden.

8. Führen Sie in Buildspec einen der folgenden Schritte aus:

   - Wählen Sie Buildspec-Datei verwenden, um die Datei buildspec.yml im Quellcode-Stammverzeichnis zu verwenden.
   - Wählen Sie Build-Befehle einfügen, um die Konsole zum Einfügen von Build-Befehlen zu verwenden.

   Weitere Informationen hierzu finden Sie unter Build-Spezifikationsreferenz.

9. In Artifacts (Artefakte):

   Typ

   Wählen Sie eine der folgenden Optionen aus:

   - Wenn keine Build-Ausgabeartefakte erstellt werden sollen, klicken Sie auf die Option No artifacts (Keine Artefakte).
   - Um die Build-Ausgabe in einem S3-Bucket zu speichern, wählen Sie Amazon S3 und gehen Sie dann wie folgt vor:
     - Lassen Sie Name leer, wenn Sie den Projektnamen für die ZIP-Datei mit der Build-Ausgabe verwenden möchten. Geben Sie andernfalls den Namen ein. Standardmäßig ist der Artefaktname der Projektname. Wenn Sie einen anderen Namen verwenden möchten, geben Sie diesen in das Feld für den Artefaktnamen ein. Wenn Sie eine ZIP-Datei ausgeben möchten, schließen Sie die ZIP-Erweiterung mit ein.
     - Wählen Sie für Bucket name den Namen des Ausgabe-Buckets aus.
     - Wenn Sie in diesem Vorgang zuvor die Option Insert build commands (Build-Befehle einfügen) verwendet haben, geben Sie für Output files (Ausgabedateien) die Speicherorte der Build-Dateien ein, die in der ZIP-Datei oder in dem Ordner für die Build-Ausgabe enthalten sein sollen. Bei mehreren Speicherorten trennen Sie die einzelnen Speicherorte durch ein Komma, (wie z. B. `appspec.yml, target/my-app.jar`). Weitere Informationen finden Sie in der Beschreibung von `files` in Syntax der Build-Spezifikation.

Zusätzliche Konfiguration

Erweitern Sie Additional configuration (Zusätzliche Einstellungen) und legen Sie die entsprechenden Optionen fest.

10. Wählen Sie Create build project (Build-Projekt erstellen) aus. Klicken Sie auf der Seite Review (Überprüfen) auf Start build (Build starten), um den Build auszuführen.

Verifizierungsprüfungen

1. Öffnen Sie die AWS CodeBuild Konsole unter https://console.aws.amazon.com/codesuite/codebuild/home.
2. Wählen Sie im linken Navigationsbereich Build projects aus.
3. Führen Sie eine der folgenden Aktionen aus:
   - Klicken Sie auf den Link des Build-Projekts mit Webhooks, das Sie kontrollieren möchten, und klicken Sie dann auf Build details (Build-Details).
   - Klicken Sie auf die Schaltfläche neben dem Build-Projekt mit den Webhooks, die Sie verifizieren möchten, wählen Sie Details anzeigen und dann den Tab Build-Details aus.
4. Wählen Sie unter Webhook-Ereignisse mit primärer Quelle den Webhook-URL-Link aus.
5. Vergewissern Sie sich in Ihrem GitHub Repository auf der Seite Einstellungen unter Webhooks, dass Pull Requests und Pushes ausgewählt sind.
6. In Ihren GitHub Profileinstellungen sollten Sie unter Persönliche Einstellungen, Anwendungen, Autorisierte OAuth-Apps sehen, dass Ihre Anwendung für den Zugriff auf die AWS von Ihnen ausgewählte Region autorisiert wurde.

Beispiel für die Verwendung des semantischen Versionings zum Benennen von Build-Artefakten

Dieses Beispiel enthält buildspec-Beispieldateien, die zeigen, wie ein Artefaktname angegeben wird, der während der Build-Erstellung erstellt wird. Ein in einer buildspec-Datei angegebener Name kann Shell-Befehle und Umgebungsvariablen enthalten, um ihn eindeutig zu gestalten. Ein in einer buildspec-Datei angegebener Name überschreibt einen Name, den Sie beim Erstellen des Projekts in der Konsole angeben.

Wenn Sie den Build mehrmals erstellen, kann mit einem in der buildspec-Datei angegebenen Namen sichergestellt werden, dass die Namen der Ausgabeartefaktdateien eindeutig sind. Sie können beispielsweise einen Datums- und Zeitstempel verwenden, der während der Build-Erstellung in den Artefaktnamen eingefügt wird.

Wenn Sie den in der Konsole eingegebenen Artefaktnamen mit einem Namen in der buildspec-Datei überschreiben möchten, gehen Sie wie folgt vor:

1. Konfigurieren Sie das Build-Projekt so, dass der Artefaktname mit einem Namen in der buildspec-Datei überschrieben wird.

   - Wenn Sie die Konsole zum Erstellen Ihres Build-Projekts verwenden, wählen Sie Enable semantitic versioning (Semantisches Versioning aktivieren) aus. Weitere Informationen finden Sie unter Erstellen Sie ein Build-Projekt (Konsole).
   - Wenn Sie das verwenden AWS CLI, setzen Sie das in der `overrideArtifactName` übergebenen Datei im JSON-Format auf true. `create-project` Weitere Informationen finden Sie unter Erstellen eines Build-Projekts (AWS CLI).
   - Wenn Sie die AWS CodeBuild API verwenden, setzen Sie das `overrideArtifactName` Flag für das `ProjectArtifacts` Objekt, wenn ein Projekt erstellt oder aktualisiert oder ein Build gestartet wird.

2. Geben Sie einen Namen in der buildspec-Datei an. Ziehen Sie die folgenden buildspec-Beispieldateien zur Orientierung heran.

Dieses Linux-Beispiel zeigt, wie Sie einen Artefaktnamen angeben, der das Datum der Build-Erstellung enthält:

```
version: 0.2
phases:
  build:
    commands:
      - rspec HelloWorld_spec.rb
artifacts:
  files:
    - '**/*'
  name: myname-$(date +%Y-%m-%d)
```

Dieses Linux-Beispiel zeigt Ihnen, wie Sie einen Artefaktnamen angeben, der eine CodeBuild Umgebungsvariable verwendet. Weitere Informationen finden Sie unter Umgebungsvariablen in Build-Umgebungen.

```
version: 0.2
phases:
  build:
    commands:
      - rspec HelloWorld_spec.rb
artifacts:
  files:
    - '**/*'
  name: myname-$AWS_REGION
```

Dieses Windows-Beispiel zeigt, wie Sie einen Artefaktnamen angeben, der das Datum und die Zeit der Build-Erstellung enthält:

```
version: 0.2
env:
  variables:
    TEST_ENV_VARIABLE: myArtifactName
phases:
  build:
    commands:
      - cd samples/helloworld
      - dotnet restore
      - dotnet run
artifacts:
  files:
    - '**/*'
  name: $Env:TEST_ENV_VARIABLE-$(Get-Date -UFormat "%Y%m%d-%H%M%S")
```

Dieses Windows-Beispiel zeigt Ihnen, wie Sie einen Artefaktnamen angeben, der eine in der Buildspec-Datei deklarierte Variable und eine Umgebungsvariable verwendet. CodeBuild Weitere Informationen finden Sie unter Umgebungsvariablen in Build-Umgebungen.

```
version: 0.2
env:
  variables:
    TEST_ENV_VARIABLE: myArtifactName
phases:
```

```
  build:
    commands:
      - cd samples/helloworld
      - dotnet restore
      - dotnet run
artifacts:
  files:
    - '**/*'
  name: $Env:TEST_ENV_VARIABLE-$Env:AWS_REGION
```

Weitere Informationen finden Sie unter Referenz zur Build-Spezifikation für CodeBuild.

Microsoft Windows-Beispiele für CodeBuild

In diesen Beispielen wird eine AWS CodeBuild Buildumgebung verwendet, in der Microsoft Windows Server 2019, das.NET Framework und das.NET Core SDK ausgeführt werden, um Laufzeitdateien aus in F# und Visual Basic geschriebenem Code zu erstellen.

> **Important**
>
> Das Ausführen dieser Beispiele kann zu Gebühren für Ihr AWS Konto führen. Dazu gehören mögliche Gebühren für CodeBuild und für AWS Ressourcen und Aktionen im Zusammenhang mit Amazon S3 und CloudWatch Logs. AWS KMS Weitere Informationen finden Sie unter CodeBuildPreise, Amazon S3 S3-Preise, AWS Key Management Service Preise und CloudWatch Amazon-Preise.

Ausführen der Beispiele

So führen Sie diese Beispiele aus

1. Erstellen Sie die Dateien wie in den Abschnitten „Verzeichnisstruktur“ und „Dateien“ dieses Themas beschrieben, und laden Sie sie dann in einen S3-Eingabe-Bucket CodeCommit oder ein GitHub OR-Repository hoch.

   > **Important**
   >
   > Laden Sie nicht *(root directory name)* hoch, sondern nur die Dateien in *(root directory name)*.

Wenn Sie einen S3-Empfangs-Bucket verwenden, sollten Sie eine ZIP-Datei erstellen, die die Dateien enthält, und diese dann in den Empfangs-Bucket hochladen. Fügen Sie *(root directory name)* nicht zur ZIP-Datei hinzu, sondern nur die Dateien in *(root directory name)*.

2. Erstellen Sie ein Build-Projekt. Das Build-Projekt muss das `mcr.microsoft.com/dotnet/framework/sdk:4.8` Image verwenden, um .NET Framework-Projekte zu erstellen.

   Wenn Sie das AWS CLI Build-Projekt mit erstellen, sieht die Eingabe des `create-project` Befehls im JSON-Format möglicherweise ähnlich aus. (Ersetzen Sie die Platzhalter durch Ihre eigenen Werte.)

```
{
  "name": "sample-windows-build-project",
  "source": {
    "type": "S3",
    "location": "codebuild-region-ID-account-ID-input-bucket/windows-build-input-artifact.zip"
  },
  "artifacts": {
    "type": "S3",
    "location": "codebuild-region-ID-account-ID-output-bucket",
    "packaging": "ZIP",
    "name": "windows-build-output-artifact.zip"
  },
  "environment": {
    "type": "WINDOWS_SERVER_2019_CONTAINER",
    "image": "mcr.microsoft.com/dotnet/framework/sdk:4.8",
    "computeType": "BUILD_GENERAL1_MEDIUM"
  },
  "serviceRole": "arn:aws:iam::account-ID:role/role-name",
  "encryptionKey": "arn:aws:kms:region-ID:account-ID:key/key-ID"
}
```

3. Führen Sie den Build aus und folgen Sie den Schritten unter. Ausführen eines Build

4. Um das Build-Ausgabeartefakt abzurufen, laden Sie in Ihrem S3-Ausgabe-Bucket die Datei *windows-build-output-artifact*`.zip` auf Ihren lokalen Computer oder Ihre lokale Instance herunter. Extrahieren Sie den Inhalt, um zur Runtime und zu anderen Dateien zu gelangen.

- Die Runtime-Datei für das F#-Beispiel mit dem.NET Framework`FSharpHelloWorld.exe`,, befindet sich im `FSharpHelloWorld\bin\Debug` Verzeichnis.
- Die Runtime-Datei für das Visual Basic-Beispiel, das das.NET Framework verwendet`VBHelloWorld.exe`, befindet sich im `VBHelloWorld\bin\Debug` Verzeichnis.

Verzeichnisstruktur

Diesen Beispiele setzen die folgenden Verzeichnisstrukturen voraus.

F# und das .NET Framework

```
(root directory name)
### buildspec.yml
### FSharpHelloWorld.sln
### FSharpHelloWorld
    ### App.config
    ### AssemblyInfo.fs
    ### FSharpHelloWorld.fsproj
    ### Program.fs
```

Visual Basic und das .NET Framework

```
(root directory name)
### buildspec.yml
### VBHelloWorld.sln
### VBHelloWorld
    ### App.config
    ### HelloWorld.vb
    ### VBHelloWorld.vbproj
    ### My Project
        ### Application.Designer.vb
        ### Application.myapp
        ### AssemblyInfo.vb
        ### Resources.Designer.vb
        ### Resources.resx
        ### Settings.Designer.vb
        ### Settings.settings
```

Dateien

Diese Beispiele verwenden die folgenden Dateien.

F# und das .NET Framework

`buildspec.yml` (in *(root directory name)*):

```
version: 0.2

env:
  variables:
    SOLUTION: .\FSharpHelloWorld.sln
    PACKAGE_DIRECTORY: .\packages
    DOTNET_FRAMEWORK: 4.8

phases:
  build:
    commands:
      - '& nuget restore $env:SOLUTION -PackagesDirectory $env:PACKAGE_DIRECTORY'
      - '& msbuild -p:FrameworkPathOverride="C:\Program Files (x86)\Reference
 Assemblies\Microsoft\Framework\.NETFramework\v$env:DOTNET_FRAMEWORK" $env:SOLUTION'
artifacts:
  files:
    - .\FSharpHelloWorld\bin\Debug\*
```

`FSharpHelloWorld.sln` (in *(root directory name)*):

```
Microsoft Visual Studio Solution File, Format Version 12.00
# Visual Studio 14
VisualStudioVersion = 14.0.25420.1
MinimumVisualStudioVersion = 10.0.40219.1
Project("{F2A71F9B-5D33-465A-A702-920D77279786}") = "FSharpHelloWorld",
 "FSharpHelloWorld\FSharpHelloWorld.fsproj", "{D60939B6-526D-43F4-9A89-577B2980DF62}"
EndProject
Global
  GlobalSection(SolutionConfigurationPlatforms) = preSolution
    Debug|Any CPU = Debug|Any CPU
    Release|Any CPU = Release|Any CPU
  EndGlobalSection
  GlobalSection(ProjectConfigurationPlatforms) = postSolution
    {D60939B6-526D-43F4-9A89-577B2980DF62}.Debug|Any CPU.ActiveCfg = Debug|Any CPU
    {D60939B6-526D-43F4-9A89-577B2980DF62}.Debug|Any CPU.Build.0 = Debug|Any CPU
```

```
    {D60939B6-526D-43F4-9A89-577B2980DF62}.Release|Any CPU.ActiveCfg = Release|Any CPU
    {D60939B6-526D-43F4-9A89-577B2980DF62}.Release|Any CPU.Build.0 = Release|Any CPU
  EndGlobalSection
  GlobalSection(SolutionProperties) = preSolution
    HideSolutionNode = FALSE
  EndGlobalSection
EndGlobal
```

App.config (in *(root directory name)*\FSharpHelloWorld):

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
  <startup>
    <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.8" />
  </startup>
</configuration>
```

AssemblyInfo.fs (in *(root directory name)*\FSharpHelloWorld):

```
namespace FSharpHelloWorld.AssemblyInfo

open System.Reflection
open System.Runtime.CompilerServices
open System.Runtime.InteropServices

// General Information about an assembly is controlled through the following
// set of attributes. Change these attribute values to modify the information
// associated with an assembly.
[<assembly: AssemblyTitle("FSharpHelloWorld")>]
[<assembly: AssemblyDescription("")>]
[<assembly: AssemblyConfiguration("")>]
[<assembly: AssemblyCompany("")>]
[<assembly: AssemblyProduct("FSharpHelloWorld")>]
[<assembly: AssemblyCopyright("Copyright ©  2017")>]
[<assembly: AssemblyTrademark("")>]
[<assembly: AssemblyCulture("")>]

// Setting ComVisible to false makes the types in this assembly not visible
// to COM components.  If you need to access a type in this assembly from
// COM, set the ComVisible attribute to true on that type.
[<assembly: ComVisible(false)>]

// The following GUID is for the ID of the typelib if this project is exposed to COM
```

```
[<assembly: Guid("d60939b6-526d-43f4-9a89-577b2980df62")>]

// Version information for an assembly consists of the following four values:
//
// Major Version
// Minor Version
// Build Number
// Revision
//
// You can specify all the values or you can default the Build and Revision Numbers
// by using the '*' as shown below:
// [<assembly: AssemblyVersion("1.0.*")>]
[<assembly: AssemblyVersion("1.0.0.0")>]
[<assembly: AssemblyFileVersion("1.0.0.0")>]

do
  ()
```

FSharpHelloWorld.fsproj (in *(root directory name)*\FSharpHelloWorld):

```
<?xml version="1.0" encoding="utf-8"?>
<Project ToolsVersion="14.0" DefaultTargets="Build" xmlns="http://
schemas.microsoft.com/developer/msbuild/2003">
  <Import Project="$(MSBuildExtensionsPath)\
$(MSBuildToolsVersion)\Microsoft.Common.props"
 Condition="Exists('$(MSBuildExtensionsPath)\
$(MSBuildToolsVersion)\Microsoft.Common.props')" />
  <PropertyGroup>
    <Configuration Condition=" '$(Configuration)' == '' ">Debug</Configuration>
    <Platform Condition=" '$(Platform)' == '' ">AnyCPU</Platform>
    <SchemaVersion>2.0</SchemaVersion>
    <ProjectGuid>d60939b6-526d-43f4-9a89-577b2980df62</ProjectGuid>
    <OutputType>Exe</OutputType>
    <RootNamespace>FSharpHelloWorld</RootNamespace>
    <AssemblyName>FSharpHelloWorld</AssemblyName>
    <TargetFrameworkVersion>v4.8</TargetFrameworkVersion>
    <AutoGenerateBindingRedirects>true</AutoGenerateBindingRedirects>
    <TargetFSharpCoreVersion>4.4.0.0</TargetFSharpCoreVersion>
    <Name>FSharpHelloWorld</Name>
  </PropertyGroup>
  <PropertyGroup Condition=" '$(Configuration)|$(Platform)' == 'Debug|AnyCPU' ">
    <DebugSymbols>true</DebugSymbols>
    <DebugType>full</DebugType>
```

```
    <Optimize>false</Optimize>
    <Tailcalls>false</Tailcalls>
    <OutputPath>bin\Debug\</OutputPath>
    <DefineConstants>DEBUG;TRACE</DefineConstants>
    <WarningLevel>3</WarningLevel>
    <PlatformTarget>AnyCPU</PlatformTarget>
    <DocumentationFile>bin\Debug\FSharpHelloWorld.XML</DocumentationFile>
    <Prefer32Bit>true</Prefer32Bit>
  </PropertyGroup>
  <PropertyGroup Condition=" '$(Configuration)|$(Platform)' == 'Release|AnyCPU' ">
    <DebugType>pdbonly</DebugType>
    <Optimize>true</Optimize>
    <Tailcalls>true</Tailcalls>
    <OutputPath>bin\Release\</OutputPath>
    <DefineConstants>TRACE</DefineConstants>
    <WarningLevel>3</WarningLevel>
    <PlatformTarget>AnyCPU</PlatformTarget>
    <DocumentationFile>bin\Release\FSharpHelloWorld.XML</DocumentationFile>
    <Prefer32Bit>true</Prefer32Bit>
  </PropertyGroup>
  <ItemGroup>
    <Reference Include="mscorlib" />
    <Reference Include="FSharp.Core, Version=$(TargetFSharpCoreVersion),
 Culture=neutral, PublicKeyToken=b03f5f7f11d50a3a">
      <Private>True</Private>
    </Reference>
    <Reference Include="System" />
    <Reference Include="System.Core" />
    <Reference Include="System.Numerics" />
  </ItemGroup>
  <ItemGroup>
    <Compile Include="AssemblyInfo.fs" />
    <Compile Include="Program.fs" />
    <None Include="App.config" />
  </ItemGroup>
  <PropertyGroup>
    <MinimumVisualStudioVersion Condition="'$(MinimumVisualStudioVersion)' == ''">11</
MinimumVisualStudioVersion>
  </PropertyGroup>
  <Choose>
    <When Condition="'$(VisualStudioVersion)' == '11.0'">
      <PropertyGroup Condition="Exists('$(MSBuildExtensionsPath32)\..\Microsoft SDKs\F#
\3.0\Framework\v4.0\Microsoft.FSharp.Targets')">
```

```
        <FSharpTargetsPath>$(MSBuildExtensionsPath32)\..\Microsoft SDKs\F#
\3.0\Framework\v4.0\Microsoft.FSharp.Targets</FSharpTargetsPath>
      </PropertyGroup>
    </When>
    <Otherwise>
      <PropertyGroup Condition="Exists('$(MSBuildExtensionsPath32)\Microsoft
\VisualStudio\v$(VisualStudioVersion)\FSharp\Microsoft.FSharp.Targets')">
        <FSharpTargetsPath>$(MSBuildExtensionsPath32)\Microsoft\VisualStudio\v
$(VisualStudioVersion)\FSharp\Microsoft.FSharp.Targets</FSharpTargetsPath>
      </PropertyGroup>
    </Otherwise>
  </Choose>
  <Import Project="$(FSharpTargetsPath)" />
  <!-- To modify your build process, add your task inside one of the targets below and
 uncomment it.
       Other similar extension points exist, see Microsoft.Common.targets.
  <Target Name="BeforeBuild">
  </Target>
  <Target Name="AfterBuild">
  </Target>
  -->
</Project>
```

`Program.fs` (in *`(root directory name)`*`\FSharpHelloWorld`):

```
// Learn more about F# at http://fsharp.org
// See the 'F# Tutorial' project for more help.

[<EntryPoint>]
let main argv =
  printfn "Hello World"
  0 // return an integer exit code
```

Visual Basic und das .NET Framework

`buildspec.yml` (in *`(root directory name)`*):

```
version: 0.2

env:
  variables:
    SOLUTION: .\VBHelloWorld.sln
    PACKAGE_DIRECTORY: .\packages
```

```
    DOTNET_FRAMEWORK: 4.8

phases:
  build:
    commands:
      - '& "C:\ProgramData\chocolatey\bin\NuGet.exe" restore $env:SOLUTION -
PackagesDirectory $env:PACKAGE_DIRECTORY'
      - '& "C:\Program Files (x86)\MSBuild\14.0\Bin\MSBuild.exe" -
p:FrameworkPathOverride="C:\Program Files (x86)\Reference Assemblies\Microsoft
\Framework\.NETFramework\v$env:DOTNET_FRAMEWORK" $env:SOLUTION'
artifacts:
  files:
    - .\VBHelloWorld\bin\Debug\*
```

`VBHelloWorld.sln` (in *`(root directory name)`*):

```
Microsoft Visual Studio Solution File, Format Version 12.00
# Visual Studio 14
VisualStudioVersion = 14.0.25420.1
MinimumVisualStudioVersion = 10.0.40219.1
Project("{F184B08F-C81C-45F6-A57F-5ABD9991F28F}") = "VBHelloWorld", "VBHelloWorld
\VBHelloWorld.vbproj", "{4DCEC446-7156-4FE6-8CCC-219E34DD409D}"
EndProject
Global
  GlobalSection(SolutionConfigurationPlatforms) = preSolution
    Debug|Any CPU = Debug|Any CPU
    Release|Any CPU = Release|Any CPU
  EndGlobalSection
  GlobalSection(ProjectConfigurationPlatforms) = postSolution
    {4DCEC446-7156-4FE6-8CCC-219E34DD409D}.Debug|Any CPU.ActiveCfg = Debug|Any CPU
    {4DCEC446-7156-4FE6-8CCC-219E34DD409D}.Debug|Any CPU.Build.0 = Debug|Any CPU
    {4DCEC446-7156-4FE6-8CCC-219E34DD409D}.Release|Any CPU.ActiveCfg = Release|Any CPU
    {4DCEC446-7156-4FE6-8CCC-219E34DD409D}.Release|Any CPU.Build.0 = Release|Any CPU
  EndGlobalSection
  GlobalSection(SolutionProperties) = preSolution
    HideSolutionNode = FALSE
  EndGlobalSection
EndGlobal
```

`App.config` (in *`(root directory name)`*`\VBHelloWorld`):

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
```

```
  <startup>
    <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.8" />
  </startup>
</configuration>
```

HelloWorld.vb (in *(root directory name)*\VBHelloWorld):

```
Module HelloWorld

  Sub Main()
    MsgBox("Hello World")
  End Sub

End Module
```

VBHelloWorld.vbproj (in *(root directory name)*\VBHelloWorld):

```
<?xml version="1.0" encoding="utf-8"?>
<Project ToolsVersion="14.0" DefaultTargets="Build" xmlns="http://
schemas.microsoft.com/developer/msbuild/2003">
  <Import Project="$(MSBuildExtensionsPath)\
$(MSBuildToolsVersion)\Microsoft.Common.props"
 Condition="Exists('$(MSBuildExtensionsPath)\
$(MSBuildToolsVersion)\Microsoft.Common.props')" />
  <PropertyGroup>
    <Configuration Condition=" '$(Configuration)' == '' ">Debug</Configuration>
    <Platform Condition=" '$(Platform)' == '' ">AnyCPU</Platform>
    <ProjectGuid>{4DCEC446-7156-4FE6-8CCC-219E34DD409D}</ProjectGuid>
    <OutputType>Exe</OutputType>
    <StartupObject>VBHelloWorld.HelloWorld</StartupObject>
    <RootNamespace>VBHelloWorld</RootNamespace>
    <AssemblyName>VBHelloWorld</AssemblyName>
    <FileAlignment>512</FileAlignment>
    <MyType>Console</MyType>
    <TargetFrameworkVersion>v4.8</TargetFrameworkVersion>
    <AutoGenerateBindingRedirects>true</AutoGenerateBindingRedirects>
  </PropertyGroup>
  <PropertyGroup Condition=" '$(Configuration)|$(Platform)' == 'Debug|AnyCPU' ">
    <PlatformTarget>AnyCPU</PlatformTarget>
    <DebugSymbols>true</DebugSymbols>
    <DebugType>full</DebugType>
    <DefineDebug>true</DefineDebug>
    <DefineTrace>true</DefineTrace>
```

```
    <OutputPath>bin\Debug\</OutputPath>
    <DocumentationFile>VBHelloWorld.xml</DocumentationFile>
    <NoWarn>42016,41999,42017,42018,42019,42032,42036,42020,42021,42022</NoWarn>
  </PropertyGroup>
  <PropertyGroup Condition=" '$(Configuration)|$(Platform)' == 'Release|AnyCPU' ">
    <PlatformTarget>AnyCPU</PlatformTarget>
    <DebugType>pdbonly</DebugType>
    <DefineDebug>false</DefineDebug>
    <DefineTrace>true</DefineTrace>
    <Optimize>true</Optimize>
    <OutputPath>bin\Release\</OutputPath>
    <DocumentationFile>VBHelloWorld.xml</DocumentationFile>
    <NoWarn>42016,41999,42017,42018,42019,42032,42036,42020,42021,42022</NoWarn>
  </PropertyGroup>
  <PropertyGroup>
    <OptionExplicit>On</OptionExplicit>
  </PropertyGroup>
  <PropertyGroup>
    <OptionCompare>Binary</OptionCompare>
  </PropertyGroup>
  <PropertyGroup>
    <OptionStrict>Off</OptionStrict>
  </PropertyGroup>
  <PropertyGroup>
    <OptionInfer>On</OptionInfer>
  </PropertyGroup>
  <ItemGroup>
    <Reference Include="System" />
    <Reference Include="System.Data" />
    <Reference Include="System.Deployment" />
    <Reference Include="System.Xml" />
    <Reference Include="System.Core" />
    <Reference Include="System.Xml.Linq" />
    <Reference Include="System.Data.DataSetExtensions" />
    <Reference Include="System.Net.Http" />
  </ItemGroup>
  <ItemGroup>
    <Import Include="Microsoft.VisualBasic" />
    <Import Include="System" />
    <Import Include="System.Collections" />
    <Import Include="System.Collections.Generic" />
    <Import Include="System.Data" />
    <Import Include="System.Diagnostics" />
    <Import Include="System.Linq" />
```

```
    <Import Include="System.Xml.Linq" />
    <Import Include="System.Threading.Tasks" />
  </ItemGroup>
  <ItemGroup>
    <Compile Include="HelloWorld.vb" />
    <Compile Include="My Project\AssemblyInfo.vb" />
    <Compile Include="My Project\Application.Designer.vb">
      <AutoGen>True</AutoGen>
      <DependentUpon>Application.myapp</DependentUpon>
    </Compile>
    <Compile Include="My Project\Resources.Designer.vb">
      <AutoGen>True</AutoGen>
      <DesignTime>True</DesignTime>
      <DependentUpon>Resources.resx</DependentUpon>
    </Compile>
    <Compile Include="My Project\Settings.Designer.vb">
      <AutoGen>True</AutoGen>
      <DependentUpon>Settings.settings</DependentUpon>
      <DesignTimeSharedInput>True</DesignTimeSharedInput>
    </Compile>
  </ItemGroup>
  <ItemGroup>
    <EmbeddedResource Include="My Project\Resources.resx">
      <Generator>VbMyResourcesResXFileCodeGenerator</Generator>
      <LastGenOutput>Resources.Designer.vb</LastGenOutput>
      <CustomToolNamespace>My.Resources</CustomToolNamespace>
      <SubType>Designer</SubType>
    </EmbeddedResource>
  </ItemGroup>
  <ItemGroup>
    <None Include="My Project\Application.myapp">
      <Generator>MyApplicationCodeGenerator</Generator>
      <LastGenOutput>Application.Designer.vb</LastGenOutput>
    </None>
    <None Include="My Project\Settings.settings">
      <Generator>SettingsSingleFileGenerator</Generator>
      <CustomToolNamespace>My</CustomToolNamespace>
      <LastGenOutput>Settings.Designer.vb</LastGenOutput>
    </None>
    <None Include="App.config" />
  </ItemGroup>
  <Import Project="$(MSBuildToolsPath)\Microsoft.VisualBasic.targets" />
  <!-- To modify your build process, add your task inside one of the targets below and
uncomment it.
```

```
        Other similar extension points exist, see Microsoft.Common.targets.
  <Target Name="BeforeBuild">
  </Target>
  <Target Name="AfterBuild">
  </Target>
  -->
</Project>
```

Application.Designer.vb (in *(root directory name)*\VBHelloWorld\My Project):

```
'------------------------------------------------------------------------------
' <auto-generated>
'   This code was generated by a tool.
'   Runtime Version:4.0.30319.42000
'
'   Changes to this file may cause incorrect behavior and will be lost if
'   the code is regenerated.
' </auto-generated>
'------------------------------------------------------------------------------

Option Strict On
Option Explicit On
```

Application.myapp (in *(root directory name)*\VBHelloWorld\My Project):

```
<?xml version="1.0" encoding="utf-8"?>
<MyApplicationData xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <MySubMain>false</MySubMain>
  <SingleInstance>false</SingleInstance>
  <ShutdownMode>0</ShutdownMode>
  <EnableVisualStyles>true</EnableVisualStyles>
  <AuthenticationMode>0</AuthenticationMode>
  <ApplicationType>2</ApplicationType>
  <SaveMySettingsOnExit>true</SaveMySettingsOnExit>
</MyApplicationData>
```

AssemblyInfo.vb (in *(root directory name)*\VBHelloWorld\My Project):

```
Imports System
Imports System.Reflection
Imports System.Runtime.InteropServices
```

```
' General Information about an assembly is controlled through the following
' set of attributes. Change these attribute values to modify the information
' associated with an assembly.

' Review the values of the assembly attributes

<Assembly: AssemblyTitle("VBHelloWorld")>
<Assembly: AssemblyDescription("")>
<Assembly: AssemblyCompany("")>
<Assembly: AssemblyProduct("VBHelloWorld")>
<Assembly: AssemblyCopyright("Copyright ©  2017")>
<Assembly: AssemblyTrademark("")>

<Assembly: ComVisible(False)>

'The following GUID is for the ID of the typelib if this project is exposed to COM
<Assembly: Guid("137c362b-36ef-4c3e-84ab-f95082487a5a")>

' Version information for an assembly consists of the following four values:
'
' Major Version
' Minor Version
' Build Number
' Revision
'
' You can specify all the values or you can default the Build and Revision Numbers
' by using the '*' as shown below:
' <Assembly: AssemblyVersion("1.0.*")>

<Assembly: AssemblyVersion("1.0.0.0")>
<Assembly: AssemblyFileVersion("1.0.0.0")>
```

`Resources.Designer.vb` (in *`(root directory name)`*`\VBHelloWorld\My Project`):

```
'------------------------------------------------------------------------------
' <auto-generated>
'   This code was generated by a tool.
'   Runtime Version:4.0.30319.42000
'
'   Changes to this file may cause incorrect behavior and will be lost if
'   the code is regenerated.
' </auto-generated>
```

```
'------------------------------------------------------------------------------

Option Strict On
Option Explicit On

Namespace My.Resources

  'This class was auto-generated by the StronglyTypedResourceBuilder
  'class via a tool like ResGen or Visual Studio.
  'To add or remove a member, edit your .ResX file then rerun ResGen
  'with the /str option, or rebuild your VS project.
  '''<summary>
  '''  A strongly-typed resource class, for looking up localized strings, etc.
  '''</summary>

 <Global.System.CodeDom.Compiler.GeneratedCodeAttribute("System.Resources.Tools.StronglyTypedRe
 "4.0.0.0"), _
  Global.System.Diagnostics.DebuggerNonUserCodeAttribute(), _
  Global.System.Runtime.CompilerServices.CompilerGeneratedAttribute(), _
  Global.Microsoft.VisualBasic.HideModuleNameAttribute()> _
  Friend Module Resources

    Private resourceMan As Global.System.Resources.ResourceManager

    Private resourceCulture As Global.System.Globalization.CultureInfo

    '''<summary>
    '''  Returns the cached ResourceManager instance used by this class.
    '''</summary>

 <Global.System.ComponentModel.EditorBrowsableAttribute(Global.System.ComponentModel.EditorBrow
 _
    Friend ReadOnly Property ResourceManager() As
 Global.System.Resources.ResourceManager
      Get
        If Object.ReferenceEquals(resourceMan, Nothing) Then
          Dim temp As Global.System.Resources.ResourceManager = New
 Global.System.Resources.ResourceManager("VBHelloWorld.Resources",
 GetType(Resources).Assembly)
          resourceMan = temp
        End If
        Return resourceMan
      End Get
    End Property
```

```
    '''<summary>
    '''  Overrides the current thread's CurrentUICulture property for all
    '''  resource lookups using this strongly typed resource class.
    '''</summary>

 <Global.System.ComponentModel.EditorBrowsableAttribute(Global.System.ComponentModel.EditorBrow
 _
    Friend Property Culture() As Global.System.Globalization.CultureInfo
      Get
        Return resourceCulture
      End Get
      Set(ByVal value As Global.System.Globalization.CultureInfo)
        resourceCulture = value
      End Set
    End Property
  End Module
End Namespace
```

Resources.resx (in *(root directory name)*\VBHelloWorld\My Project):

```
<?xml version="1.0" encoding="utf-8"?>
<root>
  <!--
    Microsoft ResX Schema

    Version 2.0

    The primary goals of this format is to allow a simple XML format
    that is mostly human readable. The generation and parsing of the
    various data types are done through the TypeConverter classes
    associated with the data types.

    Example:

    ... ado.net/XML headers & schema ...
    <resheader name="resmimetype">text/microsoft-resx</resheader>
    <resheader name="version">2.0</resheader>
    <resheader name="reader">System.Resources.ResXResourceReader,
 System.Windows.Forms, ...</resheader>
    <resheader name="writer">System.Resources.ResXResourceWriter,
 System.Windows.Forms, ...</resheader>
```

```
    <data name="Name1"><value>this is my long string</value><comment>this is a
comment</comment></data>
    <data name="Color1" type="System.Drawing.Color, System.Drawing">Blue</data>
    <data name="Bitmap1" mimetype="application/x-microsoft.net.object.binary.base64">
      <value>[base64 mime encoded serialized .NET Framework object]</value>
    </data>
    <data name="Icon1" type="System.Drawing.Icon, System.Drawing"
mimetype="application/x-microsoft.net.object.bytearray.base64">
      <value>[base64 mime encoded string representing a byte array form of the .NET
Framework object]</value>
      <comment>This is a comment</comment>
    </data>

    There are any number of "resheader" rows that contain simple
    name/value pairs.

    Each data row contains a name, and value. The row also contains a
    type or mimetype. Type corresponds to a .NET class that support
    text/value conversion through the TypeConverter architecture.
    Classes that don't support this are serialized and stored with the
    mimetype set.

    The mimetype is used for serialized objects, and tells the
    ResXResourceReader how to depersist the object. This is currently not
    extensible. For a given mimetype the value must be set accordingly:

    Note - application/x-microsoft.net.object.binary.base64 is the format
    that the ResXResourceWriter will generate, however the reader can
    read any of the formats listed below.

    mimetype: application/x-microsoft.net.object.binary.base64
    value   : The object must be serialized with
            : System.Serialization.Formatters.Binary.BinaryFormatter
            : and then encoded with base64 encoding.

    mimetype: application/x-microsoft.net.object.soap.base64
    value   : The object must be serialized with
            : System.Runtime.Serialization.Formatters.Soap.SoapFormatter
            : and then encoded with base64 encoding.

    mimetype: application/x-microsoft.net.object.bytearray.base64
    value   : The object must be serialized into a byte array
            : using a System.ComponentModel.TypeConverter
            : and then encoded with base64 encoding.
```

```
 -->
 <xsd:schema id="root" xmlns="" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:msdata="urn:schemas-microsoft-com:xml-msdata">
   <xsd:element name="root" msdata:IsDataSet="true">
     <xsd:complexType>
       <xsd:choice maxOccurs="unbounded">
         <xsd:element name="metadata">
           <xsd:complexType>
             <xsd:sequence>
               <xsd:element name="value" type="xsd:string" minOccurs="0" />
             </xsd:sequence>
             <xsd:attribute name="name" type="xsd:string" />
             <xsd:attribute name="type" type="xsd:string" />
             <xsd:attribute name="mimetype" type="xsd:string" />
           </xsd:complexType>
         </xsd:element>
         <xsd:element name="assembly">
           <xsd:complexType>
             <xsd:attribute name="alias" type="xsd:string" />
             <xsd:attribute name="name" type="xsd:string" />
           </xsd:complexType>
         </xsd:element>
         <xsd:element name="data">
           <xsd:complexType>
             <xsd:sequence>
               <xsd:element name="value" type="xsd:string" minOccurs="0"
msdata:Ordinal="1" />
               <xsd:element name="comment" type="xsd:string" minOccurs="0"
msdata:Ordinal="2" />
             </xsd:sequence>
             <xsd:attribute name="name" type="xsd:string" msdata:Ordinal="1" />
             <xsd:attribute name="type" type="xsd:string" msdata:Ordinal="3" />
             <xsd:attribute name="mimetype" type="xsd:string" msdata:Ordinal="4" />
           </xsd:complexType>
         </xsd:element>
         <xsd:element name="resheader">
           <xsd:complexType>
             <xsd:sequence>
               <xsd:element name="value" type="xsd:string" minOccurs="0"
msdata:Ordinal="1" />
             </xsd:sequence>
             <xsd:attribute name="name" type="xsd:string" use="required" />
           </xsd:complexType>
         </xsd:element>
```

```
        </xsd:choice>
      </xsd:complexType>
    </xsd:element>
  </xsd:schema>
  <resheader name="resmimetype">
    <value>text/microsoft-resx</value>
  </resheader>
  <resheader name="version">
    <value>2.0</value>
  </resheader>
  <resheader name="reader">
    <value>System.Resources.ResXResourceReader, System.Windows.Forms, Version=2.0.0.0,
 Culture=neutral, PublicKeyToken=b77a5c561934e089</value>
  </resheader>
  <resheader name="writer">
    <value>System.Resources.ResXResourceWriter, System.Windows.Forms, Version=2.0.0.0,
 Culture=neutral, PublicKeyToken=b77a5c561934e089</value>
  </resheader>
</root>
```

`Settings.Designer.vb` (in *`(root directory name)`*`\VBHelloWorld\My Project`):

```
'------------------------------------------------------------------------------
' <auto-generated>
'     This code was generated by a tool.
'     Runtime Version:4.0.30319.42000
'
'     Changes to this file may cause incorrect behavior and will be lost if
'     the code is regenerated.
' </auto-generated>
'------------------------------------------------------------------------------

Option Strict On
Option Explicit On

Namespace My

  <Global.System.Runtime.CompilerServices.CompilerGeneratedAttribute(), _

 Global.System.CodeDom.Compiler.GeneratedCodeAttribute("Microsoft.VisualStudio.Editors.Settings
 "11.0.0.0"), _
```

```
Global.System.ComponentModel.EditorBrowsableAttribute(Global.System.ComponentModel.EditorBrows
_
 Partial Friend NotInheritable Class MySettings
   Inherits Global.System.Configuration.ApplicationSettingsBase

   Private Shared defaultInstance As MySettings =
CType(Global.System.Configuration.ApplicationSettingsBase.Synchronized(New
MySettings), MySettings)

   #Region "My.Settings Auto-Save Functionality"
     #If _MyType = "WindowsForms" Then
       Private Shared addedHandler As Boolean

       Private Shared addedHandlerLockObject As New Object

       <Global.System.Diagnostics.DebuggerNonUserCodeAttribute(),
Global.System.ComponentModel.EditorBrowsableAttribute(Global.System.ComponentModel.EditorBrows
_
       Private Shared Sub AutoSaveSettings(ByVal sender As Global.System.Object, ByVal
e As Global.System.EventArgs)
         If My.Application.SaveMySettingsOnExit Then
           My.Settings.Save()
         End If
       End Sub
     #End If
   #End Region

   Public Shared ReadOnly Property [Default]() As MySettings
     Get

       #If _MyType = "WindowsForms" Then
         If Not addedHandler Then
           SyncLock addedHandlerLockObject
             If Not addedHandler Then
               AddHandler My.Application.Shutdown, AddressOf AutoSaveSettings
               addedHandler = True
             End If
           End SyncLock
         End If
       #End If
       Return defaultInstance
     End Get
   End Property
```

```
  End Class
End Namespace

Namespace My

  <Global.Microsoft.VisualBasic.HideModuleNameAttribute(), _
  Global.System.Diagnostics.DebuggerNonUserCodeAttribute(), _
  Global.System.Runtime.CompilerServices.CompilerGeneratedAttribute()> _
  Friend Module MySettingsProperty

    <Global.System.ComponentModel.Design.HelpKeywordAttribute("My.Settings")> _
    Friend ReadOnly Property Settings() As Global.VBHelloWorld.My.MySettings
      Get
        Return Global.VBHelloWorld.My.MySettings.Default
      End Get
    End Property
  End Module
End Namespace
```

`Settings.settings` (in *`(root directory name)`*`\VBHelloWorld\My Project`):

```
<?xml version='1.0' encoding='utf-8'?>
<SettingsFile xmlns="http://schemas.microsoft.com/VisualStudio/2004/01/settings"
 CurrentProfile="(Default)" UseMySettingsClassName="true">
  <Profiles>
    <Profile Name="(Default)" />
  </Profiles>
  <Settings />
</SettingsFile>
```

Planen eines Builds in AWS CodeBuild

Bevor Sie AWS CodeBuild verwenden, müssen Sie sich folgende Fragen beantworten:

1. Wo wird der Quellcode gespeichert? CodeBuild unterstützt derzeit das Erstellen von aus den folgenden Quellcode-Repository-Anbietern. Der Quellcode muss eine Build-Spezifikationsdatei (buildspec) enthalten. Eine Build-Spezifikation ist eine Sammlung von Build-Befehlen und zugehörigen Einstellungen im YAML-Format, die CodeBuild verwendet, um einen Build auszuführen. Sie können eine Build-Spezifikation in einer Build-Projektdefinition deklarieren.

Repository-Anbieter	Erforderlich	Dokumentation
CodeCommit	Repository-Name. (Optional) Mit dem Quellcode verknüpfte Commit-ID.	Weitere Informationen finden Sie unter den folgenden Themen im AWS CodeCommit-Benutzerhandbuch: Erstellen eines CodeCommit Repositorys Erstellen eines Commit in CodeCommit
Amazon S3	Empfangs-Bucket-Name. Objektname für die Build-Eingabe-ZIP-Datei, die den Quellcode enthält. (Optional) Mit der Build-Eingabe-ZIP-Datei verknüpfte Versions-ID.	Sehen Sie sich diese Themen im Handbuch Erste Schritte mit Amazon S3 an: Erstellen Sie einen Bucket Hinzufügen eines Objekts zu einem Bucket

Repository-Anbieter	Erforderlich	Dokumentation
GitHub	Repository-Name. (Optional) Mit dem Quellcode verknüpfte Commit-ID.	Sehen Sie sich dieses Thema auf der GitHub Hilfe-Website an: Erstellen eines Repository
Bitbucket	Repository-Name. (Optional) Mit dem Quellcode verknüpfte Commit-ID.	Siehe dieses Thema auf der Dokumentationswebsite zur Bitbucket Cloud: Erstellen eines Repositorys

2. Welche Build-Befehle müssen Sie ausführen und in welcher Reihenfolge? Standardmäßig lädt die Build-Eingabe von dem von Ihnen angegebenen Anbieter CodeBuild herunter und lädt die Build-Ausgabe in den von Ihnen angegebenen Bucket hoch. Sie verwenden die Build-Spezifikationen für Anweisungen dazu, wie die heruntergeladene Build-Eingabe in die erwartete Build-Ausgabe umgewandelt werden soll. Weitere Informationen hierzu finden Sie unter Build-Spezifikationsreferenz.
3. Welche Laufzeiten und Tools benötigen Sie zur Build-Ausführung? Erstellen Sie den Build beispielsweise für Java, Ruby, Python oder Node.js? Benötigt der Build Maven oder Ant oder einen Compiler für Java, Ruby oder Python? Erfordert der Build Git, die AWS CLI oder andere Tools?

   CodeBuild führt Builds in Build-Umgebungen aus, die Docker-Images verwenden. Diese Docker-Images müssen in einem Repository-Typ gespeichert sein, der von CodeBuild unterstützt wird. Dazu gehören das CodeBuild Docker-Image-Repository, Docker Hub und Amazon Elastic Container Registry (Amazon ECR). Weitere Informationen zum CodeBuild Docker-Image-Repository finden Sie unter Docker-Images bereitgestellt von CodeBuild.
4. Benötigen Sie AWS Ressourcen, die nicht automatisch von bereitgestellt werden CodeBuild? Wenn ja, welche Sicherheitsrichtlinien benötigen diese Ressourcen? Beispielsweise müssen

Sie möglicherweise die CodeBuild Servicerolle ändern, damit mit diesen Ressourcen CodeBuild arbeiten kann.

5. Möchten Sie mit Ihrer VPC CodeBuild arbeiten? Wenn Ja, benötigen Sie für Ihre VPC-Konfiguration die VPC-ID, die Subnet-IDs und die Sicherheitsgruppen-IDs. Weitere Informationen finden Sie unter Verwendung AWS CodeBuild mit Amazon Virtual Private Cloud.

Nachdem Sie diese Fragen beantwortet haben, sollten Sie über die Einstellungen und Ressourcen verfügen, die Sie benötigen, um einen Build erfolgreich auszuführen. Sie können den Build wie folgt ausführen:

- Verwenden Sie die AWS CodeBuild-Konsole, AWS CLI oder AWS-SDKs. Weitere Informationen finden Sie unter Ausführen eines Build.
- Erstellen oder identifizieren Sie eine Pipeline in AWS CodePipelineund fügen Sie dann eine Build- oder Testaktion hinzu, die anweist, Ihren Code automatisch CodeBuild zu testen, Ihren Build auszuführen oder beides. Weitere Informationen finden Sie unter Verwenden von CodePipeline mit CodeBuild.

Referenz zur Build-Spezifikation für CodeBuild

Dieses Thema stellt wichtige Referenzinformationen über Build-Spezifikationsdateien (buildspecs) bereit. Eine Buildspec ist eine Sammlung von Build-Befehlen und zugehörigen Einstellungen im YAML-Format, die zur Ausführung eines Builds CodeBuild verwendet werden. Sie können eine Buildspec als Teil des Quellcodes einbeziehen oder Sie können eine Buildspec definieren, wenn Sie ein Build-Projekt erstellen. Informationen zur Funktionsweise einer Build-Spezifikation finden Sie unter Funktionsweise von CodeBuild.

Themen

Dateiname der Build-Spezifikation und Speicherort

Wenn Sie eine Build-Spezifikation als Teil des Quellcodes einbinden, muss die Build-Spezifikationsdatei standardmäßig als `buildspec.yml` benannt und im Stammverzeichnis Ihres Quellverzeichnisses abgelegt werden.

Sie können den Standard-Namen und -Speicherort der Build-Spezifikationsdatei überschreiben. Beispielsweise ist Folgendes möglich:

- Verwenden Sie eine andere Build-Spezifikationsdatei für verschiedene Builds im selben Repository, z. B. `buildspec_debug.yml` und `buildspec_release.yml`.
- Speichern Sie eine Build-Spezifikationsdatei in einem anderen Verzeichnis als dem Stammverzeichnis des Quellverzeichnisses, also z. B. in `config/buildspec.yml` oder in einem S3-Bucket. Der S3-Bucket muss sich in derselben AWS Region wie Ihr Build-Projekt befinden. Geben Sie die buildspec-Datei mit ihrem ARN an (z. B. `arn:aws:s3:::<my-codebuild-sample2>/buildspec.yml`).

Sie können für ein Build-Projekt nur ein Build-Spezifikation angeben, unabhängig vom Namen der Build-Spezifikationsdatei.

Führen Sie einen der folgenden Schritte aus, um den Standard-Dateinamen, -Speicherort der Build-Spezifikationsdatei oder beides zu überschreiben:

- Führen Sie den `update-project` Befehl AWS CLI `create-project` or aus und setzen Sie den `buildspec` Wert auf den Pfad zur alternativen Buildspec-Datei relativ zum Wert der integrierten Umgebungsvariablen. `CODEBUILD_SRC_DIR` Sie können das Gleiche auch mit der `create project` Operation in den SDKs tun. AWS Weitere Informationen finden Sie unter Erstellen eines Build-Projekts oder Ändern der Einstellungen eines Build-Projekts.
- Führen Sie den AWS CLI `start-build` Befehl aus und legen Sie den `buildspecOverride` Wert auf den Pfad zur alternativen Buildspec-Datei relativ zum Wert der integrierten Umgebungsvariablen fest. `CODEBUILD_SRC_DIR` Sie können das Gleiche auch mit der `start build` Operation in den SDKs tun. AWS Weitere Informationen finden Sie unter Ausführen eines Build.
- Legen Sie in einer AWS CloudFormation Vorlage die `BuildSpec` Eigenschaft von `Source` in einer Ressource vom Typ `AWS::CodeBuild::Project` auf den Pfad zur alternativen Buildspec-Datei relativ zum Wert der integrierten Umgebungsvariablen fest. `CODEBUILD_SRC_DIR` Weitere

Informationen finden Sie unter der BuildSpec Eigenschaft in der AWS CodeBuild Projektquelle im AWS CloudFormation Benutzerhandbuch.

Syntax der Build-Spezifikation

Build-Spezifikationsdateien müssen im Format YAML ausgedrückt werden.

Wenn ein Befehl ein Zeichen oder eine Zeichenfolge enthält, die von YAML nicht unterstützt wird, müssen Sie den Befehl in Anführungszeichen ("") einschließen. Der folgende Befehl ist in Anführungszeichen eingeschlossen, da ein Doppelpunkt (:) gefolgt von einem Leerzeichen in YAML nicht erlaubt ist. Das Anführungszeichen im Befehl wird mit Escape (\") angegeben.

```
"export PACKAGE_NAME=$(cat package.json | grep name | head -1 | awk -F: '{ print $2 }'
 | sed 's/[\",]//g')"
```

Die Build-Spezifikation hat die folgende Syntax:

```
version: 0.2

run-as: Linux-user-name

env:
  shell: shell-tag
  variables:
    key: "value"
    key: "value"
  parameter-store:
    key: "value"
    key: "value"
  exported-variables:
    - variable
    - variable
  secrets-manager:
    key: secret-id:json-key:version-stage:version-id
  git-credential-helper: no | yes

proxy:
  upload-artifacts: no | yes
  logs: no | yes

batch:
```

```
  fast-fail: false | true
  # build-list:
  # build-matrix:
  # build-graph:

phases:
  install:
    run-as: Linux-user-name
    on-failure: ABORT | CONTINUE
    runtime-versions:
      runtime: version
      runtime: version
    commands:
      - command
      - command
    finally:
      - command
      - command
    # steps:
  pre_build:
    run-as: Linux-user-name
    on-failure: ABORT | CONTINUE
    commands:
      - command
      - command
    finally:
      - command
      - command
    # steps:
  build:
    run-as: Linux-user-name
    on-failure: ABORT | CONTINUE
    commands:
      - command
      - command
    finally:
      - command
      - command
    # steps:
  post_build:
    run-as: Linux-user-name
    on-failure: ABORT | CONTINUE
    commands:
      - command
```

```
      - command
    finally:
      - command
      - command
    # steps:
reports:
  report-group-name-or-arn:
    files:
      - location
      - location
    base-directory: location
    discard-paths: no | yes
    file-format: report-format
artifacts:
  files:
    - location
    - location
  name: artifact-name
  discard-paths: no | yes
  base-directory: location
  exclude-paths: excluded paths
  enable-symlinks: no | yes
  s3-prefix: prefix
  secondary-artifacts:
    artifactIdentifier:
      files:
        - location
        - location
      name: secondary-artifact-name
      discard-paths: no | yes
      base-directory: location
    artifactIdentifier:
      files:
        - location
        - location
      discard-paths: no | yes
      base-directory: location
cache:
  paths:
    - path
    - path
```

Die Build-Spezifikation enthält Folgendes:

version

Erforderliche Zuweisung. Diese steht für die Version der Build-Spezifikation. Wir empfehlen Ihnen, `0.2` zu verwenden.

> **Note**
>
> Obwohl Version 0.1 wird weiterhin unterstützt wird, empfehlen wir Ihnen, nach Möglichkeit Version 0.2 zu verwenden. Weitere Informationen finden Sie unter Versionen der Build-Spezifikationen.

run-as

Optionale Sequenz. Nur für Linux-Benutzer verfügbar. Gibt einen Linux-Benutzer an, der Befehle in dieser Buildspec-Datei ausführt. `run-as`gewährt dem angegebenen Benutzer Lese- und Ausführungsberechtigungen. Wenn Sie `run-as` zu Beginn der buildspec-Datei angeben, gilt die Einstellung global für alle Befehle. Wenn Sie nicht möchten, dass ein Benutzer Berechtigungen für sämtliche Befehle in der buildspec-Datei erhält, können Sie die Berechtigungen auf eine Phase beschränken, indem Sie `run-as` in einem der `phases`-Blöcke angeben. Wird `run-as` nicht angegeben, werden alle Befehle als Root-Benutzer ausgeführt.

env

Optionale Sequenz. Diese steht für die Informationen von einer oder mehrerer benutzerdefinierter Umgebungsvariablen.

> **Note**
>
> Um vertrauliche Informationen zu schützen, sind die folgenden Informationen in CodeBuild Protokollen versteckt:
>
> - AWS Zugriffstasten-IDs. Weitere Informationen finden Sie unter Verwaltung von Zugriffsschlüsseln für IAM-Benutzer im AWS Identity and Access Management Benutzerhandbuch.
> - Mit dem Parameter Store angegebene Zeichenfolgen. Weitere Informationen finden Sie unter Systems Manager Parameter Store und Systems Manager Parameter Store Console Walkthrough im Amazon EC2 Systems Manager Manager-Benutzerhandbuch.

- Zeichenketten, die angegeben wurden mit. AWS Secrets Manager Weitere Informationen finden Sie unter Schlüsselverwaltung.

env/Shell

Optionale Sequenz. Gibt die unterstützte Shell für Linux- oder Windows-Betriebssysteme an.

Für Linux-Betriebssysteme werden folgende Shell-Tags unterstützt:

- `bash`
- `/bin/sh`

Für Windows-Betriebssysteme werden folgende Shell-Tags unterstützt:

- `powershell.exe`
- `cmd.exe`

env/variables

Erforderlich, wenn `env` angegeben ist und Sie benutzerdefinierte Umgebungsvariablen im Klartext definieren möchten. Enthält eine Zuweisung von *`Schlüssel`*/*`Wert-`* Skalaren, wobei jede Zuweisung für eine einzige, benutzerdefinierte Umgebungsvariable in Klartext steht. *`key (Schlüssel)`* ist der Name der benutzerdefinierten Umgebungsvariablen und *`value (Wert)`* ist der Wert dieser Variablen.

> **Important**
>
> Wir raten dringend davon ab, sensible Werte in Umgebungsvariablen zu speichern. Umgebungsvariablen können mit Tools wie der CodeBuild Konsole und dem im Klartext angezeigt werden. AWS CLI Für vertrauliche Werte sollte stattdessen die Zuweisung per `parameter-store` oder `secrets-manager` verwendet werden (siehe unten in diesem Abschnitt).
> Jede festgelegte Umgebungsvariable ersetzt vorhandene Umgebungsvariablen. Wenn das Docker-Image beispielsweise bereits eine Umgebungsvariable mit dem Namen `MY_VAR` und einem Wert von `my_value` enthält und Sie eine Umgebungsvariable mit dem Namen `MY_VAR` und einem Wert von `other_value` festlegen, wird `my_value` durch `other_value` ersetzt. Wenn das Docker-Image demgegenüber bereits eine Umgebungsvariable mit dem Namen `PATH` und einem Wert von `/usr/local/sbin:/usr/local/bin` enthält und Sie eine Umgebungsvariable mit dem Namen `PATH` und

einem Wert von `$PATH:/usr/share/ant/bin` festlegen, wird `/usr/local/sbin:/usr/local/bin` durch den Literalwert `$PATH:/usr/share/ant/bin` ersetzt.
Legen Sie keine Umgebungsvariable mit einem Namen fest, der mit `CODEBUILD_` beginnt. Dieses Präfix ist zur -internen Verwendung reserviert.
Wenn eine Umgebungsvariable mit identischem Namen an mehreren Orten definiert ist, wird der Wert folgendermaßen bestimmt:

- Der Wert im Aufruf zum Starten des Build-Vorgangs hat den höchsten Vorrang. Beim Erstellen eines Builds können Sie Umgebungsvariablen hinzufügen oder überschreiben. Weitere Informationen finden Sie unter Ausführen eines Build in AWS CodeBuild.
- Der Wert in der Build-Projektdefinition folgt darauf. Beim Erstellen oder Bearbeiten eines Projekts können Sie Umgebungsvariablen auf Projektebene hinzufügen. Weitere Informationen finden Sie unter Erstellen eines Build-Projekts in AWS CodeBuild und Ändern der Einstellungen eines Build-Projekts in AWS CodeBuild.
- Der Wert in der buildspec-Deklaration hat die niedrigste Priorität.

env/parameter-store

Erforderlich, wenn `env` angegeben, und Sie benutzerdefinierte Umgebungsvariablen abrufen möchten, die im Amazon EC2 Systems Manager Parameter Store gespeichert sind. Enthält eine Zuordnung von *`Schlüssel-/Wert-Skalaren`*, wobei jede Zuordnung eine einzelne benutzerdefinierte Umgebungsvariable darstellt, die im Amazon EC2 Systems Manager Parameter Store gespeichert ist. *`key`* ist der Name, den Sie später in Ihren Build-Befehlen verwenden, um auf diese benutzerdefinierte Umgebungsvariable zu verweisen, und *`value`* ist der Name der benutzerdefinierten Umgebungsvariablen, die im Amazon EC2 Systems Manager Parameter Store gespeichert ist. Informationen zum Speichern sensibler Werte finden Sie unter Systems Manager Parameter Store and Walkthrough: Create and test a String parameters (console) im Amazon EC2 Systems Manager Manager-Benutzerhandbuch.

> **Important**
>
> Um das Abrufen von benutzerdefinierten Umgebungsvariablen CodeBuild zu ermöglichen, die im Amazon EC2 Systems Manager Parameter Store gespeichert sind, müssen Sie die `ssm:GetParameters` Aktion zu Ihrer CodeBuild Servicerolle hinzufügen. Weitere Informationen finden Sie unter Erstellen Sie eine CodeBuild Servicerolle.
> Alle Umgebungsvariablen, die Sie aus dem Amazon EC2 Systems Manager Parameter Store abrufen, ersetzen vorhandene Umgebungsvariablen. Wenn das Docker-Image

beispielsweise bereits eine Umgebungsvariable mit dem Namen `MY_VAR` und einem Wert von `my_value` enthält und Sie eine Umgebungsvariable mit dem Namen `MY_VAR` und einem Wert von `other_value` abrufen, wird `my_value` durch `other_value` ersetzt. Wenn das Docker-Image demgegenüber bereits eine Umgebungsvariable mit dem Namen `PATH` und einem Wert von `/usr/local/sbin:/usr/local/bin` enthält und Sie eine Umgebungsvariable mit dem Namen `PATH` und einem Wert von `$PATH:/usr/share/ant/bin` abrufen, wird `/usr/local/sbin:/usr/local/bin` durch den Literalwert `$PATH:/usr/share/ant/bin` ersetzt.
Speichern Sie keine Umgebungsvariable mit einem Namen, der mit `CODEBUILD_` beginnt. Dieses Präfix ist zur -internen Verwendung reserviert.
Wenn eine Umgebungsvariable mit identischem Namen an mehreren Orten definiert ist, wird der Wert folgendermaßen bestimmt:

- Der Wert im Aufruf zum Starten des Build-Vorgangs hat den höchsten Vorrang. Beim Erstellen eines Builds können Sie Umgebungsvariablen hinzufügen oder überschreiben. Weitere Informationen finden Sie unter Ausführen eines Build in AWS CodeBuild.
- Der Wert in der Build-Projektdefinition folgt darauf. Beim Erstellen oder Bearbeiten eines Projekts können Sie Umgebungsvariablen auf Projektebene hinzufügen. Weitere Informationen finden Sie unter Erstellen eines Build-Projekts in AWS CodeBuild und Ändern der Einstellungen eines Build-Projekts in AWS CodeBuild.
- Der Wert in der buildspec-Deklaration hat die niedrigste Priorität.

env/secrets-manager

Erforderlich, wenn Sie benutzerdefinierte Umgebungsvariablen abrufen möchten, die in AWS Secrets Manager gespeichert sind. Geben Sie einen Secrets Manager `reference-key` mit dem folgenden Muster an:

`<key>`: *`<secret-id>`*:*`<json-key>`*:*`<version-stage>`*:*`<version-id>`*

<key>

(Erforderlich) Der Name der lokalen Umgebungsvariablen. Verwenden Sie diesen Namen, um während des Builds auf die Variable zuzugreifen.

<secret-id>

(Erforderlich) Der Name oder der Amazon-Ressourcenname (ARN), der als eindeutige Kennung für das Geheimnis dient. Um auf ein Secret in Ihrem AWS -Konto zuzugreifen, geben

Sie einfach den Secret-Namen an. Um auf ein Geheimnis in einem anderen AWS Konto zuzugreifen, geben Sie den geheimen ARN an.

<json-key>

(Optional) Gibt den Schlüsselnamen des Secrets Manager Manager-Schlüssel-Wert-Paares an, dessen Wert Sie abrufen möchten. Wenn Sie kein a angeben`json-key`, wird der gesamte CodeBuild geheime Text abgerufen.

<version-stage>

(Optional) Gibt die geheime Version, die Sie abrufen möchten, anhand des an die Version angehängten Staging-Labels an. Staging-Kennzeichnungen werden verwendet, um während des Rotationsprozesses den Überblick über verschiedene Versionen zu behalten. Wenn Sie `version-stage` verwenden, geben Sie `version-id` nicht an. Wenn Sie keine Versionsstufe oder Versions-ID angeben, wird standardmäßig die Version mit dem Versionsstufenwert `AWSCURRENT` abgerufen.

<version-id>

(Optional) Gibt den eindeutigen Bezeichner der Version des Geheimnisses an, die Sie verwenden möchten. Wenn Sie `version-id` angeben, dürfen Sie `version-stage` nicht angeben. Wenn Sie keine Versionsstufe oder Versions-ID angeben, wird standardmäßig die Version mit dem Versionsstufenwert `AWSCURRENT` abgerufen.

Im folgenden Beispiel `TestSecret` ist der Name des Schlüssel-Wert-Paares in Secrets Manager gespeichert. Der Schlüssel für `TestSecret` ist. `MY_SECRET_VAR` Sie greifen während des Builds über den `LOCAL_SECRET_VAR` Namen auf die Variable zu.

```
env:
  secrets-manager:
    LOCAL_SECRET_VAR: "TestSecret:MY_SECRET_VAR"
```

Weitere Informationen finden Sie unter Was ist AWS Secrets Manager? im AWS Secrets Manager -Benutzerhandbuch.

env/exported-variables

Optionale Zuweisung. Wird verwendet, um Umgebungsvariablen aufzulisten, die Sie exportieren möchten. Geben Sie den Namen jeder Variable, die Sie exportieren möchten, unter `exported-variables` in einer separaten Zeile an. Die Variable, die Sie exportieren möchten, muss während des Builds in Ihrem Container verfügbar sein. Die Variable, die Sie exportieren, kann eine Umgebungsvariable sein.

Exportierte Umgebungsvariablen werden in Verbindung mit verwendet AWS CodePipeline , um Umgebungsvariablen aus der aktuellen Buildphase in nachfolgende Phasen der Pipeline zu exportieren. Weitere Informationen finden Sie im AWS CodePipeline Benutzerhandbuch unter Arbeiten mit Variablen.

Während eines Builds ist der Wert einer Variablen ab der `install`-Phase verfügbar. Er kann zwischen dem Beginn der `install`-Phase und dem Ende der `post_build`-Phase aktualisiert werden. Nach dem Ende der `post_build`-Phase kann sich der Wert der exportierten Variablen nicht ändern.

> **Note**
>
> Folgendes kann nicht exportiert werden:
>
> - Amazon EC2 Systems Manager Parameter Speichert die im Build-Projekt angegebenen Geheimnisse.
> - Secrets Manager Manager-Geheimnisse, die im Build-Projekt angegeben wurden
> - Umgebungsvariablen, die mit `AWS_` beginnen.

env/ git-credential-helper

Optionale Zuweisung. Wird verwendet, um anzugeben, ob sein Git-Helper für Anmeldeinformationen CodeBuild verwendet wird, um Git-Anmeldeinformationen bereitzustellen. `yes`ob es verwendet wird. Andernfalls `no` oder nicht angegeben. Weitere Informationen finden Sie unter gitcredentials auf der Git-Website.

> **Note**
>
> `git-credential-helper` wird für Builds, die von einem Webhook für ein öffentliches Git-Repository ausgelöst werden, nicht unterstützt.

Proxy

Optionale Sequenz. Wird verwendet, um Einstellungen darzustellen, wenn Sie Ihren Build in einem expliziten Proxy-Server ausführen. Weitere Informationen finden Sie unter Führen Sie CodeBuild in einem expliziten Proxy-Server aus.

proxy/upload-artifacts

Optionale Zuweisung. Legen Sie diese auf `yes` fest, wenn Ihr Build in einem expliziten Proxy-Server Artefakte hochladen soll. Der Standardwert ist `no`.

proxy/logs

Optionale Zuweisung. Stellen Sie auf ein, `yes` damit Ihr Build einen expliziten Proxyserver zur Erstellung von CloudWatch Protokollen einfügt. Der Standardwert ist `no`.

phases

Erforderliche Sequenz. Stellt die Befehle dar, die in jeder Phase des Builds CodeBuild ausgeführt werden.

> **Note**
>
> CodeBuild Führt in Buildspec-Version 0.1 jeden Befehl in einer separaten Instanz der Standard-Shell in der Build-Umgebung aus. d. h., dass jeder Befehl unabhängig von allen anderen Befehlen ausgeführt wird. Daher können Sie standardmäßig keinen Einzelbefehl ausführen, der auf dem Status eines vorherigen Befehls basiert (beispielsweise beim Ändern von Verzeichnissen oder beim Einrichten von Variablen). Um diese Einschränkung zu umgehen, empfehlen wir die Nutzung von Version 0.2, die dieses Problem löst. Wenn Sie die Build-Spezifikation Version 0.1 verwenden müssen, empfehlen wir die Ansätze in Shells und Befehle in Build-Umgebungen.

phases/*/run-as

Optionale Sequenz. Verwenden Sie den Befehl in einer Build-Phase, um den Linux-Benutzer festzulegen, der die zugehörigen Befehle ausführt. Wenn zusätzlich zu Beginn der buildspec-Datei `run-as` global für alle Befehle angegeben wird, wird dem Benutzer auf Phasenebene Vorrang eingeräumt. Beispiel: Wenn `run-as` global User-1 angibt und in der `install`-Phase eine `run-as`-Anweisung User-2 definiert, werden alle Befehle in der buildspec-Datei als User-1 ausgeführt, bis auf die Befehle in der `install`-Phase, die als User-2 ausgeführt werden.

phasen/*/ bei einem Fehler

Optionale Sequenz. Gibt die Aktion an, die ergriffen werden soll, wenn während der Phase ein Fehler auftritt. Dabei kann es sich um einen der folgenden Werte handeln:

- `ABORT`- Bricht den Build ab.
- `CONTINUE`- Fahren Sie mit der nächsten Phase fort.

Wenn diese Eigenschaft nicht angegeben ist, folgt der Fehlerprozess den Übergangsphasen, wie unter beschriebenÜbergang von Build-Phasen.

Phasen/*/ schließlich

Optionaler Block. In einem `finally` Block angegebene Befehle werden nach Befehlen im Block ausgeführt. `commands` Die Befehle in einem `finally` Block werden auch dann ausgeführt, wenn ein Befehl im `commands` Block fehlschlägt. Wenn der `commands` Block beispielsweise drei Befehle enthält und der erste fehlschlägt, werden die verbleibenden zwei Befehle CodeBuild übersprungen und alle Befehle im `finally` Block ausgeführt. Die Phase ist erfolgreich, wenn alle Befehle in den Blöcken `commands` und `finally` erfolgreich ausgeführt werden. Wenn ein Befehl in einer Phase fehlschlägt, schlägt die Phase fehl.

Die zulässigen Build-Phasennamen sind:

phases/install

Optionale Sequenz. Stellt die Befehle dar, falls vorhanden, die während der Installation CodeBuild ausgeführt werden. Wir empfehlen Ihnen, die Phase `install` nur für Installationspakete in der Build-Umgebung einzusetzen. Beispielsweise könnten sie diese Phase nutzen, um ein Code-Testframework wie Mocha oder RSpec zu installieren.

phases/install/runtime-versions

Optionale Sequenz. Eine Runtime-Version wird mit dem Ubuntu-Standard-Image 5.0 oder höher und dem Amazon Linux 2-Standard-Image 4.0 oder höher unterstützt. Wenn dieses Argument angegeben wird, muss mindestens eine Laufzeit in diesen Abschnitt aufgenommen werden. Geben Sie eine Laufzeit mit einer bestimmten Version an, eine Hauptversion, gefolgt von, `.x` um anzugeben, dass diese Hauptversion mit ihrer neuesten Nebenversion CodeBuild verwendet wird, oder `latest` um die neueste Haupt- und Nebenversion zu verwenden (z. B.`ruby: 3.2`,`nodejs: 18.x`, oder`java: latest`). Sie können die Laufzeit unter Verwendung einer Zahl oder einer Umgebungsvariablen angeben. Wenn Sie beispielsweise das Amazon Linux 2-Standard-Image 4.0 verwenden, wird im Folgenden angegeben, dass Version 17 von Java, die neueste Nebenversion von Python Version 3 und eine Version, die in einer Umgebungsvariablen von Ruby enthalten ist, installiert sind. Weitere Informationen finden Sie unter Docker-Images bereitgestellt von CodeBuild.

```
phases:
  install:
    runtime-versions:
      java: corretto8
      python: 3.x
      ruby: "$MY_RUBY_VAR"
```

Sie können eine oder mehrere Laufzeiten im Abschnitt `runtime-versions` Ihrer buildspec-Datei angeben. Wenn Ihre Laufzeit von einer anderen Laufzeit abhängig ist, können Sie auch die abhängige Laufzeit in der buildspec-Datei angeben. Wenn Sie in der Buildspec-Datei keine Laufzeiten angeben, CodeBuild wählt die Standard-Laufzeiten, die in dem von Ihnen verwendeten Image verfügbar sind. Wenn Sie eine oder mehrere Laufzeiten angeben, werden nur diese Laufzeiten verwendet. CodeBuild Wenn keine abhängige Laufzeit angegeben ist, wird CodeBuild versucht, die abhängige Laufzeit für Sie auszuwählen.

Wenn zwei angegebene Laufzeiten unvereinbar sind, schlägt der Build fehl. Beispiel: `android: 29` und `java: openjdk11` sind miteinander unvereinbar. Wenn beide angegeben werden, schlägt der Build fehl.

Weitere Hinweise zu den verfügbaren Laufzeiten finden Sie unterVerfügbare Laufzeiten.

> **Note**
>
> Wenn Sie einen `runtime-versions` Abschnitt angeben und ein anderes Image als Ubuntu Standard Image 2.0 oder höher oder das Amazon Linux 2 (AL2) -Standard-Image 1.0 oder höher verwenden, gibt der Build die Warnung "“ aus`Skipping install of runtimes. Runtime version selection is not supported by this build image`.

phases/install/commands

Optionale Sequenz. Enthält eine Folge von Skalaren, wobei jeder Skalar für einen einzelnen Befehl steht, der während der Installation CodeBuild ausgeführt wird. CodeBuild führt jeden Befehl nacheinander in der angegebenen Reihenfolge von Anfang bis Ende aus.

phases/pre_build

Optionale Sequenz. Stellt die Befehle dar, falls vorhanden, die vor dem Build CodeBuild ausgeführt werden. Sie können diese Phase beispielsweise verwenden, um sich bei Amazon ECR anzumelden, oder Sie können npm-Abhängigkeiten installieren.

phases/pre_build/commands

Erforderliche Sequenz, wenn `pre_build` angegeben ist. Enthält eine Folge von Skalaren, wobei jeder Skalar für einen einzelnen Befehl steht, der vor dem Build CodeBuild ausgeführt wird. CodeBuildführt jeden Befehl nacheinander in der angegebenen Reihenfolge von Anfang bis Ende aus.

phases/build

Optionale Sequenz. Stellt die Befehle dar, falls vorhanden, die während des Builds CodeBuild ausgeführt werden. Beispielsweise könnten sie diese Phase nutzen, um Mocha, RSpec oder sbt auszuführen.

phases/build/commands

Erforderlich, `build` wenn angegeben. Enthält eine Folge von Skalaren, wobei jeder Skalar für einen einzelnen Befehl steht, der während des CodeBuild Builds ausgeführt wird. CodeBuild führt jeden Befehl nacheinander in der angegebenen Reihenfolge von Anfang bis Ende aus.

phases/post_build

Optionale Sequenz. Stellt die Befehle dar, sofern vorhanden, die nach dem Build CodeBuild ausgeführt werden. Sie könnten beispielsweise Maven verwenden, um die Build-Artefakte in eine JAR- oder WAR-Datei zu packen, oder Sie könnten ein Docker-Image in Amazon ECR übertragen. Dann können Sie eine Build-Benachrichtigung über Amazon SNS senden.

phases/post_build/commands

Erforderlich, wenn `post_build` angegeben. Enthält eine Folge von Skalaren, wobei jeder Skalar für einen einzelnen Befehl steht, der nach dem CodeBuild Build ausgeführt wird. CodeBuild führt jeden Befehl nacheinander in der angegebenen Reihenfolge von Anfang bis Ende aus.

Berichte

report-group-name-or-arn

Optionale Sequenz. Gibt die Berichtsgruppe an, an die die Berichte gesendet werden. Ein Projekt kann maximal fünf Berichtsgruppen haben. Geben Sie den ARN für eine vorhandene Berichtsgruppe oder den Namen einer neuen Berichtsgruppe an. Wenn Sie einen Namen angeben, CodeBuild erstellt eine Berichtsgruppe mit Ihrem Projektnamen und dem Namen, den Sie im Format `<project-name>-<report-group-name>` angeben. Der Name der Berichtsgruppe kann auch mithilfe einer Umgebungsvariablen in der Buildspec festgelegt werden, z. B. `$REPORT_GROUP_NAME` Weitere Informationen finden Sie unter Benennung von Berichtsgruppen.

reports/<report-group>/files

Erforderliche Sequenz. Stellt die Speicherorte dar, die die Rohdaten der vom Bericht generierten Testergebnisse enthalten. Enthält eine Folge von Skalaren, wobei jeder Skalar für einen separaten Speicherort steht, an dem Testdateien gefunden CodeBuild werden können, und zwar relativ zum ursprünglichen Build-Speicherort oder, falls festgelegt, zum. `base-directory` Speicherorte können Folgendes enthalten:

- Eine Einzeldatei (z. B. `my-test-report-file.json`).
- Eine einzelne Datei in einem Unterverzeichnis (Beispiel: *`my-subdirectory`*`/my-test-report-file.json` oder *`my-parent-subdirectory`*`/`*`my-subdirectory`*`/my-test-report-file.json`).
- `'**/*'` steht rekursiv für alle Dateien.
- *`my-subdirectory`*`/*` steht für alle Dateien in einem Unterverzeichnis mit Namen *`my-subdirectory`*.
- *`my-subdirectory`*`/**/*` steht rekursiv für alle Dateien beginnend mit einem Unterverzeichnis mit Namen *`my-subdirectory`*.

reports/<report-group>/file-format

Optionale Zuweisung. Stellt das Berichtsdateiformat dar. Wenn nicht angegeben, wird `JUNITXML` verwendet. Bei diesem Wert wird nicht zwischen Groß- und Kleinschreibung unterschieden. Die möglichen Werte sind:

Testberichte

`CUCUMBERJSON`

Cucumber JSON

`JUNITXML`

JUnit XML

`NUNITXML`

NUnit XML

`NUNIT3XML`

Einheit 3 XML

`TESTNGXML`

TestNG XML

`VISUALSTUDIOTRX`

Visual Studio TRX

Berichte über die Codeabdeckung

`CLOVERXML`

Kleeblatt XML

`COBERTURAXML`

XML-Coverage

`JACOCOXML`

JaCoCo XML

`SIMPLECOV`

SimpleCov JSON

> **Note**
>
> CodeBuild akzeptiert Berichte zur JSON-Codeabdeckung, die von simplecov generiert wurden, nicht von simplecov-json.

reports/<report-group>/base-directory

Optionale Zuweisung. Stellt ein oder mehrere Verzeichnisse auf oberster Ebene im Verhältnis zum ursprünglichen Build-Speicherort dar, anhand derer bestimmt wird, wo sich die CodeBuild Rohtestdateien befinden.

reports/<report-group>/discard-paths

Optional. Gibt an, ob die Berichtsdateiverzeichnisse in der Ausgabe abgeflacht werden. Wenn dies nicht angegeben ist oder no enthält, werden Berichtsdateien mit intakter Verzeichnisstruktur ausgegeben. Wenn dies `yes` enthält, werden alle Testdateien im selben Ausgabeverzeichnis abgelegt. Wenn beispielsweise ein Pfad zu einem Testergebnis `com/myapp/mytests/TestResult.xml` lautet, wird die Datei durch Angabe von `yes` in `/TestResult.xml` gespeichert.

Artefakte

Optionale Sequenz. Stellt Informationen darüber dar, wo die Build-Ausgabe zu CodeBuild finden ist und wie sie für den Upload in den S3-Ausgabe-Bucket CodeBuild vorbereitet wird. Diese Reihenfolge ist nicht erforderlich, wenn Sie beispielsweise ein Docker-Image erstellen und an Amazon ECR übertragen oder wenn Sie Komponententests für Ihren Quellcode ausführen, ihn aber nicht erstellen.

> **Note**
>
> Amazon S3-Metadaten haben einen CodeBuild Header mit `x-amz-meta-codebuild-buildarn` dem Namen, `buildArn` der den CodeBuild Build enthält, der Artefakte in Amazon S3 veröffentlicht. Der `buildArn` wurde hinzugefügt, um die Quellenverfolgung für Benachrichtigungen zu ermöglichen und um zu referenzieren, aus welchem Build das Artefakt generiert wurde.

artifacts/files

Erforderliche Sequenz. Diese stellt die Speicherorte dar, die die Build-Ausgabeartefakte in der Build-Umgebung enthalten. Enthält eine Folge von Skalaren, wobei jeder Skalar für einen separaten Speicherort steht, an dem Build-Ausgabeartefakte gefunden werden CodeBuild können, und zwar relativ zum ursprünglichen Build-Speicherort oder, falls festgelegt, zum Basisverzeichnis. Speicherorte können Folgendes enthalten:

- Eine Einzeldatei (z. B. `my-file.jar`).
- Eine einzelne Datei in einem Unterverzeichnis (Beispiel: *`my-subdirectory`*`/my-file.jar` oder *`my-parent-subdirectory`*`/`*`my-subdirectory`*`/my-file.jar`).
- `'**/*'` steht rekursiv für alle Dateien.
- *`my-subdirectory`*`/*` steht für alle Dateien in einem Unterverzeichnis mit Namen *`my-subdirectory`*.
- *`my-subdirectory`*`/**/*` steht rekursiv für alle Dateien beginnend mit einem Unterverzeichnis mit Namen *`my-subdirectory`*.

Wenn Sie die Speicherorte für Build-Ausgabeartefakte angeben, CodeBuild kann der ursprüngliche Build-Speicherort in der Build-Umgebung gefunden werden. Sie müssen die Speicherorte von Build-Ausgabeartefakten mit dem Pfad zu den ursprünglichen Build-Speicherorten nicht voranstellen oder angeben, `./` oder ähnliches. Wenn sie den Pfad zu diesem Speicherort wissen möchten, können Sie während eines Builds ein Befehl ausführen wie `echo $CODEBUILD_SRC_DIR`. Der Speicherort für jede Build-Umgebung kann geringfügig voneinander abweichen.

artifacts/name

Optionaler Name. Gibt einen Namen für Ihr Build-Artefakt an. Dieser Name wird verwendet, wenn eine der folgenden Bedingungen zutrifft.

- Sie verwenden die CodeBuild API, um Ihre Builds zu erstellen, und das `overrideArtifactName` Flag wird auf dem `ProjectArtifacts` Objekt gesetzt, wenn ein Projekt aktualisiert, ein Projekt erstellt oder ein Build gestartet wird.
- Sie verwenden die CodeBuild Konsole, um Ihre Builds zu erstellen, in der Buildspec-Datei wird ein Name angegeben und Sie wählen Semantische Versionierung aktivieren, wenn Sie ein Projekt erstellen oder aktualisieren. Weitere Informationen finden Sie unter Erstellen Sie ein Build-Projekt (Konsole).

Sie können einen Namen in der Build-Spezifikationsdatei angeben, die zur Build-Zeit berechnet wird. Der in einer Build-Spezifikationsdatei angegebene Name verwendet die Shell-Befehlssprache. Beispielsweise können Sie dem Namen Ihres Artefakts ein Datum und eine Uhrzeit anhängen, damit dieser stets eindeutig ist. Eindeutige Artefakt-Namen verhindern, dass Artefakte überschrieben werden. Weitere Informationen finden Sie unter Shell-Befehlssprache.

- Dies ist ein Beispiel für einen Artefakt-Namen, dem das Datum angefügt wurde, an dem das Artefakt erstellt wurde.

```
version: 0.2
phases:
  build:
    commands:
      - rspec HelloWorld_spec.rb
artifacts:
  files:
    - '**/*'
  name: myname-$(date +%Y-%m-%d)
```

- Dies ist ein Beispiel für einen Artefaktnamen, der eine Umgebungsvariable verwendet. CodeBuild Weitere Informationen finden Sie unter Umgebungsvariablen in Build-Umgebungen.

```
version: 0.2
phases:
  build:
    commands:
      - rspec HelloWorld_spec.rb
artifacts:
  files:
    - '**/*'
  name: myname-$AWS_REGION
```

- Dies ist ein Beispiel für einen Artefaktnamen, der eine CodeBuild Umgebungsvariable verwendet, an die das Erstellungsdatum des Artefakts angehängt wird.

```
version: 0.2
phases:
  build:
    commands:
      - rspec HelloWorld_spec.rb
artifacts:
  files:
    - '**/*'
  name: $AWS_REGION-$(date +%Y-%m-%d)
```

Sie können dem Namen Pfadinformationen hinzufügen, sodass die benannten Artefakte auf der Grundlage des Pfads im Namen in Verzeichnissen platziert werden. In diesem Beispiel werden Build-Artefakte in der Ausgabe unter platziert`builds/<build number>/my-artifacts`.

```
version: 0.2
```

```
phases:
  build:
    commands:
      - rspec HelloWorld_spec.rb
artifacts:
  files:
    - '**/*'
  name: builds/$CODEBUILD_BUILD_NUMBER/my-artifacts
```

artifacts/discard-paths

Optional. Gibt an, ob die Build-Artefaktnerzeichnisse in der Ausgabe abgeflacht werden. Wenn dies nicht angegeben ist oder no enthält, werden Build-Artefakte mit intakter Verzeichnisstruktur ausgegeben. Wenn `yes`enthalten ist, werden alle Build-Artefakte im selben Ausgabeverzeichnis platziert. Wenn beispielsweise ein Pfad zu einer Datei im Build-Ausgabeartefakt `com/mycompany/app/HelloWorld.java` ist, wird diese Datei durch Angabe von `yes` in `/HelloWorld.java` gespeichert.

artifacts/base-directory

Optionale Zuweisung. Stellt relativ zum ursprünglichen Build-Speicherort ein oder mehrere Verzeichnisse der obersten Ebene dar, anhand derer bestimmt CodeBuild wird, welche Dateien und Unterverzeichnisse in das Build-Ausgabeartefakt aufgenommen werden sollen. Gültige Werte sind:

- Ein einziges Top-Level-Verzeichnis (z. B. `my-directory`).
- `'my-directory*'` stellt alle Top-Level-Verzeichnisse dar, deren Namen mit `my-directory` beginnen.

Die übereinstimmenden Top-Level-Verzeichnisse werden nicht in den Build-Ausgabeartefakt aufgenommen, sondern nur deren Dateien und Unterverzeichnisse.

Sie können `files` und `discard-paths` verwenden, um weiter zu beschränken, welche Dateien und Unterverzeichnisse aufgenommen werden. Wie zum Beispiel für die folgende Verzeichnisstruktur:

```
.
### my-build-1
#   ### my-file-1.txt
### my-build-2
    ### my-file-2.txt
    ### my-subdirectory
```

```
        ### my-file-3.txt
```

Und für die folgende Sequenz `artifacts` :

```
artifacts:
  files:
    - '*/my-file-3.txt'
  base-directory: my-build-2
```

Das folgende Unterverzeichnis und die Datei werden in den Build-Ausgabeartefakt aufgenommen:

```
.
### my-subdirectory
    ### my-file-3.txt
```

Während für die Sequenz `artifacts` Folgendes gilt:

```
artifacts:
  files:
    - '**/*'
  base-directory: 'my-build*'
  discard-paths: yes
```

Die folgenden Dateien werden in den Build-Ausgabeartefakt aufgenommen:

```
.
### my-file-1.txt
### my-file-2.txt
### my-file-3.txt
```

Artefakte/Ausschlusspfade

Optionale Zuweisung. Steht für einen oder mehrere Pfade, relativ zu`base-directory`, die Artefakte aus dem CodeBuild Build ausschließen. Das Sternchen (*) entspricht einem oder mehreren Zeichen einer Namenskomponente ohne Überschreiten der Ordnergrenzen. Ein doppeltes Sternchen (**) steht für null oder mehr Zeichen einer Namenskomponente in allen Verzeichnissen.

Beispiele für Ausschlusspfade sind die folgenden:

- Um eine Datei aus allen Verzeichnissen auszuschließen: `"**/file-name/**/*"`
- Um alle Punktordner auszuschließen: `"**/.*/**/*"`
- Um alle Punktdateien auszuschließen: `"**/.*"`

Artefakte/ aktivieren-Symlinks

Optional. Wenn der Ausgabetyp ist`ZIP`, gibt er an, ob interne symbolische Links in der ZIP-Datei erhalten bleiben. Wenn dieser Wert enthält`yes`, werden alle internen symbolischen Links in der Quelle in der Artefakt-ZIP-Datei beibehalten.

Artefakte/ s3-Präfix

Optional. Gibt ein Präfix an, das verwendet wird, wenn die Artefakte in einen Amazon S3 S3-Bucket ausgegeben werden, und der Namespace-Typ ist`BUILD_ID`. Bei Verwendung lautet `<s3-prefix>/<build-id>/<name>.zip` der Ausgabepfad im Bucket.

artifacts/secondary-artifacts

Optionale Sequenz. Repräsentiert einzelne oder mehrere Artefaktdefinitionen als Zuordnung zwischen einem Artefaktbezeichner und einer Artefaktdefinition. Jeder Artefaktbezeichner in diesem Block muss einem im Attribut `secondaryArtifacts` des Projekts definierten Artefakt entsprechen. Jede separate Definition hat die gleiche Syntax wie der `artifacts`-Block oben.

> **Note**
>
> Die `artifacts/files`Reihenfolge ist immer erforderlich, auch wenn nur sekundäre Artefakte definiert sind.

Angenommen sei ein Projekt mit folgender Struktur:

```
{
  "name": "sample-project",
  "secondaryArtifacts": [
    {
      "type": "S3",
      "location": "<output-bucket1>",
      "artifactIdentifier": "artifact1",
      "name": "secondary-artifact-name-1"
    },
    {
      "type": "S3",
```

```
      "location": "<output-bucket2>",
      "artifactIdentifier": "artifact2",
      "name": "secondary-artifact-name-2"
    }
  ]
}
```

Anschließend sieht die buildspec-Datei wie folgt aus:

```
version: 0.2

phases:
build:
  commands:
    - echo Building...
artifacts:
  files:
    - '**/*'
  secondary-artifacts:
    artifact1:
      files:
        - directory/file1
      name: secondary-artifact-name-1
    artifact2:
      files:
        - directory/file2
      name: secondary-artifact-name-2
```

Cache

Optionale Sequenz. Stellt Informationen darüber dar, CodeBuild wo die Dateien für das Hochladen des Caches in einen S3-Cache-Bucket vorbereitet werden können. Diese Sequenz ist nicht erforderlich, wenn der Cache-Typ des Projekts `No Cache` ist.

cache/paths

Erforderliche Sequenz. Steht für die Speicherorte des Caches. Enthält eine Folge von Skalaren, wobei jeder Skalar für einen separaten Speicherort steht, an dem Build-Ausgabeartefakte gefunden werden CodeBuild können, und zwar relativ zum ursprünglichen Build-Speicherort oder, falls festgelegt, zum Basisverzeichnis. Speicherorte können Folgendes enthalten:

- Eine Einzeldatei (z. B. `my-file.jar`).

- Eine einzelne Datei in einem Unterverzeichnis (Beispiel: *my-subdirectory*/my-file.jar oder *my-parent-subdirectory*/*my-subdirectory*/my-file.jar).
- '**/*' steht rekursiv für alle Dateien.
- *my-subdirectory*/* steht für alle Dateien in einem Unterverzeichnis mit Namen *my-subdirectory*.
- *my-subdirectory*/**/* steht rekursiv für alle Dateien beginnend mit einem Unterverzeichnis mit Namen *my-subdirectory*.

> **Important**
>
> Da es sich bei der Build-Spezifikationsdeklaration um eine gültige YAML handelt, ist die Formatierung in einer Build-Spezifikationsdeklaration wichtig. Wenn die Anzahl der Leerzeichen in der Build-Spezifikationsdeklaration unzulässig ist, können die Builds sofort fehlschlagen. Verwenden Sie eine YAML-Validierung, um zu testen, ob es sich bei Ihrer Build-Spezifikationsdeklaration um eine gültige YAML handelt.
> Wenn Sie beim Erstellen oder Aktualisieren eines Buildprojekts die oder die AWS SDKs verwenden AWS CLI, um eine Buildspezifikation zu deklarieren, muss es sich bei der Buildspec um eine einzelne Zeichenfolge im YAML-Format handeln, zusammen mit den erforderlichen Leerzeichen und Escape-Zeichen für Zeilenumbrüche. Es folgt ein Beispiel im nächsten Abschnitt.
> Wenn Sie die CodeBuild oder AWS CodePipeline -Konsolen anstelle einer buildspec.yml-Datei verwenden, können Sie Befehle nur für die Phase einfügen. `build` Statt die vorherige Syntax zu verwenden, geben Sie in einer einzigen Ziele sämtliche Befehle an, die Sie während der Build-Phase verwenden möchten. Bei mehreren Befehlen unterteilen Sie die einzelnen Befehle mit &&, (wie z. B. `mvn test && mvn package`).
> Sie können die CodePipeline Konsolen CodeBuild oder anstelle einer buildspec.yml-Datei verwenden, um die Speicherorte der Build-Ausgabeartefakte in der Buildumgebung anzugeben. Statt die vorherige Syntax zu verwenden, geben Sie in einer einzigen Ziele sämtliche Speicherorte an. Bei mehreren Speicherorten trennen Sie die einzelnen Speicherorte durch ein Komma, (wie z. B. `buildspec.yml, target/my-app.jar`).

Beispiel für eine Build-Spezifikation

Hier finden Sie ein Beispiel für eine Datei buildspec.yml.

```
version: 0.2

env:
  variables:
    JAVA_HOME: "/usr/lib/jvm/java-8-openjdk-amd64"
  parameter-store:
    LOGIN_PASSWORD: /CodeBuild/dockerLoginPassword

phases:
  install:
    commands:
      - echo Entered the install phase...
      - apt-get update -y
      - apt-get install -y maven
    finally:
      - echo This always runs even if the update or install command fails
  pre_build:
    commands:
      - echo Entered the pre_build phase...
      - docker login -u User -p $LOGIN_PASSWORD
    finally:
      - echo This always runs even if the login command fails
  build:
    commands:
      - echo Entered the build phase...
      - echo Build started on `date`
      - mvn install
    finally:
      - echo This always runs even if the install command fails
  post_build:
    commands:
      - echo Entered the post_build phase...
      - echo Build completed on `date`

reports:
  arn:aws:codebuild:your-region:your-aws-account-id:report-group/report-group-name-1:
    files:
      - "**/*"
    base-directory: 'target/tests/reports'
    discard-paths: no
  reportGroupCucumberJson:
    files:
      - 'cucumber/target/cucumber-tests.xml'
```

```
    discard-paths: yes
    file-format: CUCUMBERJSON # default is JUNITXML
artifacts:
  files:
    - target/messageUtil-1.0.jar
  discard-paths: yes
  secondary-artifacts:
    artifact1:
      files:
        - target/artifact-1.0.jar
      discard-paths: yes
    artifact2:
      files:
        - target/artifact-2.0.jar
      discard-paths: yes
cache:
  paths:
    - '/root/.m2/**/*'
```

Hier ist ein Beispiel für die vorangegangene Buildspec, ausgedrückt als einzelne Zeichenfolge, zur Verwendung mit den oder den SDKs. AWS CLI AWS

```
"version: 0.2\n\nenv:\n  variables:\n    JAVA_HOME: \"/usr/lib/jvm/java-8-openjdk-
amd64\\"\n  parameter-store:\n    LOGIN_PASSWORD: /CodeBuild/dockerLoginPassword\n
 phases:\n\n  install:\n    commands:\n      - echo Entered the install phase...\n
 - apt-get update -y\n      - apt-get install -y maven\n    finally:\n      - echo This
 always runs even if the update or install command fails \n  pre_build:\n    commands:
\n      - echo Entered the pre_build phase...\n      - docker login -u User -p
 $LOGIN_PASSWORD\n    finally:\n      - echo This always runs even if the login command
 fails \n  build:\n    commands:\n      - echo Entered the build phase...\n      - echo
 Build started on `date`\n      - mvn install\n    finally:\n      - echo This always
 runs even if the install command fails\n  post_build:\n    commands:\n      - echo
 Entered the post_build phase...\n      - echo Build completed on `date`\n\n reports:
\n  reportGroupJunitXml:\n    files:\n      - \"**/*\"\n    base-directory: 'target/
tests/reports'\n    discard-paths: false\n  reportGroupCucumberJson:\n    files:\n
 - 'cucumber/target/cucumber-tests.xml'\n    file-format: CUCUMBERJSON\n\nartifacts:\n
 files:\n    - target/messageUtil-1.0.jar\n  discard-paths: yes\n  secondary-artifacts:
\n    artifact1:\n      files:\n       - target/messageUtil-1.0.jar\n      discard-
paths: yes\n    artifact2:\n      files:\n       - target/messageUtil-1.0.jar\n
 discard-paths: yes\n cache:\n  paths:\n    - '/root/.m2/**/*'"
```

Hier ist ein Beispiel für die Befehle in der `build` Phase zur Verwendung mit den OR-Konsolen. CodeBuild CodePipeline

```
echo Build started on `date` && mvn install
```

In diesen Beispielen gilt:

- Eine benutzerdefinierte Klartext-Umgebungsvariable mit dem Schlüssel `JAVA_HOME` und dem Wert `/usr/lib/jvm/java-8-openjdk-amd64` wird eingerichtet.
- Auf eine benutzerdefinierte Umgebungsvariable mit dem Namen, die `dockerLoginPassword` Sie im Amazon EC2 Systems Manager Parameter Store gespeichert haben, wird später in Build-Befehlen mithilfe des Schlüssels `LOGIN_PASSWORD` verwiesen.
- Sie können diese Build-Phasennamen nicht ändern. Die Befehle, die in diesem Beispiel ausgeführt werden, sind `apt-get update -y` und `apt-get install -y maven` (um Apache Maven zu installieren), `mvn install` (um den Quellcode zu kompilieren, zu testen und in ein Build-Ausgabeartefakt zu packen und das Build-Ausgabeartefakt in seinem internen Repository zu installieren), `docker login` (um sich bei Docker mit dem Passwort anzumelden, das dem Wert der benutzerdefinierten Umgebungsvariablen entspricht, die `dockerLoginPassword` Sie im Amazon EC2 Systems Manager Parameter Store festgelegt haben) und mehrere Befehle. `echo` Die `echo` Befehle sind hier enthalten, um zu zeigen, wie Befehle CodeBuild ausgeführt werden und in welcher Reihenfolge sie ausgeführt werden.
- `files` stellt die Dateien dar, die in den Build-Ausgabespeicherort hochgeladen werden sollen. CodeBuild Lädt in diesem Beispiel die einzelne Datei `messageUtil-1.0.jar` hoch. Die Datei `messageUtil-1.0.jar` ist in dem jeweiligen Verzeichnis mit Namen `target` in der Build-Umgebung zu finden. Da `discard-paths: yes` angegeben wird, wird `messageUtil-1.0.jar` direkt hochgeladen (und nicht an ein intermediäres Verzeichnis mit dem Namen `target`). Der Dateiname `messageUtil-1.0.jar` und der dazugehörige Verzeichnisname von `target` basiert auf der Weise, wie - nur für dieses Beispiel - unter den Apache Maven Build-Ausgabeartefakte erstellt und gespeichert werden. In Ihren eigenen Szenarios lauten diese Dateinamen und Verzeichnisse anders.
- `reports` stellt zwei Berichtsgruppen dar, die während des Builds Berichte generieren:
  - `arn:aws:codebuild:your-region:your-aws-account-id:report-group/report-group-name-1` gibt den ARN einer Berichtsgruppe an. Testergebnisse, die vom Test-Framework generiert werden, befinden sich im Verzeichnis `target/tests/reports`. Das Dateiformat ist `JunitXml` und der Pfad wird nicht aus den Dateien entfernt, die Testergebnisse enthalten.
  - `reportGroupCucumberJson` gibt eine neue Berichtsgruppe an. Wenn der Name des Projekts `my-project` lautet, wird beim Ausführen eines Builds eine Berichtsgruppe mit dem

Namen `my-project-reportGroupCucumberJson` erstellt. Testergebnisse, die vom Test-Framework generiert werden, befinden sich in `cucumber/target/cucumber-tests.xml`. Das Testdateiformat ist `CucumberJson` und der Pfad wird aus den Dateien entfernt, die Testergebnisse enthalten.

Versionen der Build-Spezifikationen

In der folgenden Tabelle werden Versionen von Build-Spezifikationen und die Änderungen zwischen den Versionen aufgeführt.

Version	Änderungen
0.2	• `environment_variables` wurde umbenannt in `env`. • `plaintext` wurde umbenannt in `variables` . • Die `type`-Eigenschaft für `artifacts` ist veraltet. • AWS CodeBuild Führt in Version 0.1 jeden Build-Befehl in einer separaten Instanz der Standard-Shell in der Build-Umgebung aus. CodeBuild Führt in Version 0.2 alle Build-Befehle in derselben Instanz der Standard-Shell in der Build-Umgebung aus.
0.1	Dies ist die erste Definition des Build-Spezifikationsformats.

Batch-Build-Buildspec Referenz

Dieses Thema enthält die Buildspec-Referenz für Batch-Build-Eigenschaften.

Batch

Optionale Zuweisung. Die Batch-Build-Einstellungen für das Projekt.

Chargen/schnelles Scheitern

Optional. Gibt das Verhalten des Batch-Builds an, wenn eine oder mehrere Build-Aufgaben fehlschlagen.

`false`

Der Standardwert. Alle laufenden Builds werden abgeschlossen.

`true`

Alle laufenden Builds werden gestoppt, wenn eine der Build-Aufgaben fehlschlägt.

Standardmäßig werden alle Batch-Build-Aufgaben mit den Build-Einstellungen wie`env`und`phases`, angegeben in der buildspec-Datei. Sie können die Standard-Build-Einstellungen überschreiben, indem Sie`env`Werte oder eine andere buildspec-Datei im`batch/<batch-type>/buildspec`-Parameter.

Der Inhalt der`batch`-Eigenschaft variiert je nach Art des angegebenen Batchbuilds. Die möglichen Batch-Build-Typen sind:

- `batch/build-graph`
- `batch/build-list`
- `batch/build-matrix`

`batch/build-graph`

Definiert ein`Build-Diagramm`aus. Ein Build-Diagramm definiert eine Reihe von Aufgaben, die Abhängigkeiten von anderen Aufgaben im Stapel haben. Weitere Informationen finden Sie unter Diagramm erstellen.

Dieses Element enthält ein Array von Build-Aufgaben. Jede Build-Aufgabe enthält die folgenden Eigenschaften.

identifizieren

Erforderlich. Die ID der Aufgabe.

buildSpec

Optional. Pfad und Dateiname der buildspec-Datei, die für diese Aufgabe verwendet werden soll. Wird dieser Parameter nicht angegeben, wird die aktuelle buildspec-Datei verwendet.

Debug-Sitzung

Optional. Ein boolescher Wert, der angibt, ob das Sitzungsdebuggen für diesen Stapelaufbau aktiviert ist. Weitere Hinweise zum Sitzungsdebuggen finden Sie unterAnzeigen eines laufenden Builds in Session Manageraus.

`false`

Das Debuggen von Sitzungen ist deaktiviert.

`true`

Das Sitzungsdebuggen ist aktiviert.

Abhängig

Optional. Ein Array von Task-Bezeichnern, von denen diese Aufgabe abhängt. Diese Aufgabe wird erst ausgeführt, wenn diese Aufgaben abgeschlossen sind.

env

Optional. Die Build-Umgebung überschreibt die Aufgabe. Dies kann die folgenden Eigenschaften haben:

Datenverarbeitung

Die Kennung des Rechentyps, der für die Aufgabe verwendet werden soll. Siehe .computeTypeinthe section called “Berechnungsmodi und Typen der Build-Umgebung”für mögliche Werte.

Abbild

Die ID des für die Aufgabe zu verwendenden Abbildes. Siehe .Image-Kennunginthe section called “Docker-Images bereitgestellt von CodeBuild”für mögliche Werte.

Privileged-Modus

Ein boolescher Wert, der angibt, ob der Docker-Daemon in einem Docker-Container ausgeführt werden soll. Legen Sie diesen Wert`true`Nur wenn das Build-Projekt zum Erstellen von Docker-Abbildern verwendet wird. Andernfalls schlägt ein Build fehl, der versucht, mit dem Docker-Daemon zu interagieren. Die Standardeinstellung lautet `false`.

type

Die Kennung des Umgebungstyps, der für die Aufgabe verwendet werden soll. Siehe .Umgebungstypinthe section called “Berechnungsmodi und Typen der Build-Umgebung”für mögliche Werte.

Variablen

Die Umgebungsvariablen, die in der Build-Umgebung vorhanden sein werden. Weitere Informationen finden Sie unter env/variables.

ignorieren-scheitern

Optional. Ein boolescher Wert, der angibt, ob ein Fehler dieser Build-Aufgabe ignoriert werden kann.

`false`

Der Standardwert. Wenn diese Build-Aufgabe fehlschlägt, schlägt der Batch-Build fehl.

`true`

Wenn diese Build-Aufgabe fehlschlägt, kann der Batch-Build weiterhin erfolgreich sein.

Im Folgenden ein Beispiel eines Build-Grafik-Build-Spezifikationseintrags:

```
batch:
  fast-fail: false
  build-graph:
    - identifier: build1
      env:
        variables:
          BUILD_ID: build1
      ignore-failure: false
    - identifier: build2
      buildspec: build2.yml
      env:
        variables:
          BUILD_ID: build2
      depend-on:
        - build1
    - identifier: build3
      env:
        variables:
          BUILD_ID: build3
      depend-on:
        - build2
```

batch/build-list

Definiert einBuild-Listeaus. Eine Build-Liste wird verwendet, um eine Reihe von Aufgaben zu definieren, die parallel ausgeführt werden. Weitere Informationen finden Sie unter Liste erstellen.

Dieses Element enthält ein Array von Build-Aufgaben. Jede Build-Aufgabe enthält die folgenden Eigenschaften.

identifizieren

Erforderlich. Die ID der Aufgabe.

buildSpec

Optional. Pfad und Dateiname der buildspec-Datei, die für diese Aufgabe verwendet werden soll. Wird dieser Parameter nicht angegeben, wird die aktuelle buildspec-Datei verwendet.

Debug-Sitzung

Optional. Ein boolescher Wert, der angibt, ob das Sitzungsdebuggen für diesen Stapelaufbau aktiviert ist. Weitere Hinweise zum Sitzungsdebuggen finden Sie unterAnzeigen eines laufenden Builds in Session Manageraus.

`false`

Das Debuggen von Sitzungen ist deaktiviert.

`true`

Das Sitzungsdebuggen ist aktiviert.

env

Optional. Die Build-Umgebung überschreibt die Aufgabe. Dies kann die folgenden Eigenschaften haben:

Datenverarbeitung

Die Kennung des Rechentyps, der für die Aufgabe verwendet werden soll. Siehe .computeTypeinthe section called “Berechnungsmodi und Typen der Build-Umgebung”für mögliche Werte.

Abbild

Die ID des für die Aufgabe zu verwendenden Abbildes. Siehe .Image-Kennunginthe section called “Docker-Images bereitgestellt von CodeBuild”für mögliche Werte.

Privileged-Modus

Ein boolescher Wert, der angibt, ob der Docker-Daemon in einem Docker-Container ausgeführt werden soll. Legen Sie diesen Wert`true`Nur wenn das Build-Projekt zum Erstellen von Docker-Abbildern verwendet wird. Andernfalls schlägt ein Build fehl, der versucht, mit dem Docker-Daemon zu interagieren. Die Standardeinstellung lautet `false`.

type

Die Kennung des Umgebungstyps, der für die Aufgabe verwendet werden soll. Siehe .Umgebungstypinthe section called “Berechnungsmodi und Typen der Build-Umgebung”für mögliche Werte.

Variablen

Die Umgebungsvariablen, die in der Build-Umgebung vorhanden sein werden. Weitere Informationen finden Sie unter env/variables.

ignorieren-scheitern

Optional. Ein boolescher Wert, der angibt, ob ein Fehler dieser Build-Aufgabe ignoriert werden kann.

`false`

Der Standardwert. Wenn diese Build-Aufgabe fehlschlägt, schlägt der Batch-Build fehl.

`true`

Wenn diese Build-Aufgabe fehlschlägt, kann der Batch-Build weiterhin erfolgreich sein.

Im Folgenden ein Beispiel eines Build-Spezifikationseintrags:

```
batch:
  fast-fail: false
  build-list:
    - identifier: build1
      env:
        variables:
          BUILD_ID: build1
      ignore-failure: false
    - identifier: build2
      buildspec: build2.yml
```

```
    env:
      variables:
        BUILD_ID: build2
    ignore-failure: true
```

batch/build-matrix

Definiert einBuild-Matrixaus. Eine Build-Matrix definiert Aufgaben mit verschiedenen Konfigurationen, die parallel ausgeführt werden. CodeBuild erstellt für jede mögliche Konfigurationskombination einen separaten Build. Weitere Informationen finden Sie unter Matrix erstellen.

statische

Die statischen Eigenschaften gelten für alle Build-Aufgaben.

ignorieren-scheitern

Optional. Ein boolescher Wert, der angibt, ob ein Fehler dieser Build-Aufgabe ignoriert werden kann.

`false`

Der Standardwert. Wenn diese Build-Aufgabe fehlschlägt, schlägt der Batch-Build fehl.

`true`

Wenn diese Build-Aufgabe fehlschlägt, kann der Batch-Build weiterhin erfolgreich sein.

env

Optional. Die Build-Umgebung überschreibt alle Aufgaben.

Privileged-Modus

Ein boolescher Wert, der angibt, ob der Docker-Daemon in einem Docker-Container ausgeführt werden soll. Legen Sie diesen Wert`true`Nur wenn das Build-Projekt zum Erstellen von Docker-Abbildern verwendet wird. Andernfalls schlägt ein Build fehl, der versucht, mit dem Docker-Daemon zu interagieren. Die Standardeinstellung lautet `false`.

type

Die Kennung des Umgebungstyps, der für die Aufgabe verwendet werden soll. Siehe .Umgebungstypinthe section called “Berechnungsmodi und Typen der Build-Umgebung”für mögliche Werte.

dynamisch

Die dynamischen Eigenschaften definieren die Build-Matrix.

buildSpec

Optional. Ein Array, das den Pfad und die Dateinamen der BuildSpec-Dateien enthält, die für diese Aufgaben verwendet werden sollen. Wird dieser Parameter nicht angegeben, wird die aktuelle buildspec-Datei verwendet.

env

Optional. Die Build-Umgebung überschreibt diese Aufgaben.

Datenverarbeitung

Ein Array, das die Bezeichner der Berechnungstypen enthält, die für diese Aufgaben verwendet werden sollen. Siehe .computeTypeinthe section called “Berechnungsmodi und Typen der Build-Umgebung”für mögliche Werte.

Abbild

Ein Array, das die Bezeichner der Bilder enthält, die für diese Aufgaben verwendet werden sollen. Siehe .Image-Kennunginthe section called “Docker-Images bereitgestellt von CodeBuild”für mögliche Werte.

Variablen

Ein Array, das die Umgebungsvariablen enthält, die in den Build-Umgebungen für diese Aufgaben vorhanden sind. Weitere Informationen finden Sie unter env/variables.

Im Folgenden ein Beispiel eines Build-Spezifikationseintrags:

```
batch:
  build-matrix:
    static:
      ignore-failure: false
    dynamic:
      buildspec:
        - matrix1.yml
        - matrix2.yml
      env:
        variables:
          MY_VAR:
            - VALUE1
```

```
        - VALUE2
        - VALUE3
```

Weitere Informationen finden Sie unter Matrix erstellen.

Build-Umgebungsreferenz für AWS CodeBuild

Wenn Sie AWS CodeBuild aufrufen, um einen Build auszuführen, müssen Sie Informationen über die Build-Umgebung bereitstellen. Eine Build-Umgebung stellt eine Kombination aus Betriebssystem, Programmiersprache, Runtime und Tools dar, mit denenCodeBuild ein Build ausgeführt wird. Informationen zur Funktionsweise einer BuildUmgebung finden Sie unterFunktionsweise von CodeBuild.

Eine Build-Umgebung enthält ein Docker-Image. Weitere Informationen finden Sie unter the Docker glossary auf der Docker Docs-Website.

Wenn sie CodeBuild Informationen über die Build-Umgebung bereitstellen, geben Sie die Kennung einer Docker-Image in einem unterstützten Repository-Typ an. Dazu gehören dasCodeBuild Docker-Image-Repository, öffentlich verfügbare Images in Docker Hub und Amazon Elastic Container Registry (Amazon ECR) -Repositorys, für die IhrAWS Konto Zugriffsberechtigungen besitzt.

- Wir empfehlen Ihnen, Docker-Images zu verwenden, die im CodeBuild-Docker-Image-Repository gespeichert sind, weil diese auf den Einsatz mit dem Service abgestimmt sind. Weitere Informationen finden Sie unter Docker-Images bereitgestellt von CodeBuild.
- Um die Kennung eines öffentlich verfügbaren Docker-Image zu erhalten, das in Docker Hub gespeichert ist, lesen Sie bitte Searching for Repositories auf der Docker Docs-Website.
- Informationen zum Arbeiten mit Docker-Images, die in Amazon ECR-Repositorys in IhremAWS Konto gespeichert sind, finden Sie unterAmazon ECR-Beispiel.

Zusätzlich zur Kennung für ein Docker-Image geben Sie auch eine Reihe von Datenverarbeitungsressourcen an, die die Build-Umgebung verwendet. Weitere Informationen finden Sie unter Berechnungsmodi und Typen der Build-Umgebung.

Themen

- Umgebungsvariablen in Build-Umgebungen
- Hintergrundaufgaben in Build-Umgebungen

Docker-Images bereitgestellt von CodeBuild

Ein unterstütztes Image ist die neueste Hauptversion eines Images, das in verfügbar ist CodeBuild und mit Updates für Neben- und Patch-Versionen aktualisiert wird. CodeBuild optimiert die Bereitstellungsdauer von Builds mit unterstützten Images, indem sie in den Amazon Machine Images (AMI) der Maschine zwischengespeichert werden. Wenn Sie vom Caching profitieren und die Bereitstellungsdauer Ihres Builds minimieren möchten, wählen Sie im Abschnitt Image-Version der CodeBuild Konsole die Option Immer das neueste Image für diese Runtime-Version verwenden aus, anstatt eine detailliertere Version zu verwenden, z. B. `aws/codebuild/amazonlinux2-x86_64-standard:4.0-1.0.0`

CodeBuild aktualisiert häufig die Liste der Docker-Images, um die neuesten Images hinzuzufügen und alte Images als veraltet anzusehen. Die aktuelle Liste erhalten Sie, wenn Sie einen der folgenden Schritte ausführen:

- Wählen Sie in der CodeBuild Konsole im Assistenten zum Erstellen eines Build-Projekts oder auf der Seite Build-Projekt bearbeiten für Umgebungs-Image die Option Verwaltetes Image aus. Wählen Sie aus den Dropdown-Listen Operating system (Betriebssystem), Runtime (Laufzeit) und Runtime version (Laufzeitversion) aus. Weitere Informationen finden Sie unter Erstellen Sie ein Build-Projekt (Konsole) oder Ändern der Einstellungen eines Build-Projekts (Konsole).
- Führen Sie für den AWS CLI`list-curated-environment-images` folgenden Befehl aus:

```
aws codebuild list-curated-environment-images
```

- Rufen Sie für die AWS SDKs den `ListCuratedEnvironmentImages` Vorgang für Ihre Zielprogrammiersprache auf. Weitere Informationen hierzu finden Sie unter AWS SDKs- und Tools-Referenz.

Das Basis-Image der Windows Server Core 2019-Plattform ist nur in den folgenden Regionen verfügbar:

- USA Ost (Nord-Virginia)
- USA Ost (Ohio)
- USA West (Oregon)

- Europa (Irland)

EC2-Compute-Images

AWS CodeBuild unterstützt die folgenden Docker-Images, die für EC2-Compute-In verfügbar sind. CodeBuild

Plattform	Image-Kennung	Definition
Amazon Linux 2	`aws/codebuild/amazonlinux2-x86_64-standard:4.0`	al2/standard/4.0
Amazon Linux 2023	`aws/codebuild/amazonlinux2-x86_64-standard:5.0`	al2/standard/5,0
Amazon Linux 2	`aws/codebuild/amazonlinux2-x86_64-standard:corretto8`	al2/standard/corretto 8
Amazon Linux 2	`aws/codebuild/amazonlinux2-x86_64-standard:corretto11`	al2/standard/corretto11
Amazon Linux 2	`aws/codebuild/amazonlinux2-aarch64-standard:2.0`	al2/aarch64/standard/2.0
Amazon Linux 2023	`aws/codebuild/amazonlinux2-aarch64-standard:3.0`	al2/aarch64/standard/3.0
Ubuntu 20.04	`aws/codebuild/standard:5.0`	ubuntu/standard/5.0
Ubuntu 22.04	`aws/codebuild/standard:6.0`	Ubuntu/Standard/6.0

Plattform	Image-Kennung	Definition
Ubuntu 22.04	`aws/codebuild/standard:7.0`	Ubuntu/Standard/7.0
Windows Server Core 2019	`aws/codebuild/windows-base:2019-1.0`	N/A
Windows Server Core 2019	`aws/codebuild/windows-base:2019-2.0`	N/A
Windows Server Core 2019	`aws/codebuild/windows-base:2019-3.0`	N/A
Windows Server Core 2022	`aws/codebuild/windows-base:2022-1.0`	N/A

Lambda-Computing-Bilder

AWS CodeBuild unterstützt die folgenden Docker-Images, die für die AWS Lambda Datenverarbeitung verfügbar sind. CodeBuild

`aarch64`-Architektur

Plattform	Image-Kennung	Definition
Amazon Linux 2	`aws/codebuild/amazonlinux-aarch64-lambda-standard:dotnet6`	al-lambda/aarch64/dotnet6
Amazon Linux 2023	`aws/codebuild/amazonlinux-aarch64-lambda-standard:dotnet8`	al-lambda/aarch64/dotnet8
Amazon Linux 2	`aws/codebuild/amazonlinux-aarch64-la`	al-lambda/aarch64/go1.21

Plattform	Image-Kennung	Definition
	`mbda-standard:go1.21`	
Amazon Linux 2	`aws/codebuild/amazonlinux-aarch64-lambda-standard:corretto11`	al-lambda/aarch64/corretto11
Amazon Linux 2	`aws/codebuild/amazonlinux-aarch64-lambda-standard:corretto17`	al-lambda/aarch64/corretto17
Amazon Linux 2023	`aws/codebuild/amazonlinux-aarch64-lambda-standard:corretto21`	al-lambda/aarch64/corretto21
Amazon Linux 2	`aws/codebuild/amazonlinux-aarch64-lambda-standard:nodejs18`	al-lambda/aarch64/nodejs18
Amazon Linux 2023	`aws/codebuild/amazonlinux-aarch64-lambda-standard:nodejs20`	al-lambda/aarch64/nodejs20
Amazon Linux 2	`aws/codebuild/amazonlinux-aarch64-lambda-standard:python3.11`	al-lambda/aarch64/python3.11

Plattform	Image-Kennung	Definition
Amazon Linux 2023	`aws/codebuild/amazonlinux-aarch64-lambda-standard:python3.12`	al-lambda/aarch64/python3.12
Amazon Linux 2	`aws/codebuild/amazonlinux-aarch64-lambda-standard:ruby3.2`	al-lambda/aarch64/ruby3.2

x86_64-Architektur

Plattform	Image-Kennung	Definition
Amazon Linux 2	`aws/codebuild/amazonlinux-x86_64-lambda-standard:dotnet6`	al-lambda/x86_64/dotnet6
Amazon Linux 2023	`aws/codebuild/amazonlinux-x86_64-lambda-standard:dotnet8`	al-lambda/x86_64/dotnet8
Amazon Linux 2	`aws/codebuild/amazonlinux-x86_64-lambda-standard:go1.21`	al-lambda/x86_64/go1.21
Amazon Linux 2	`aws/codebuild/amazonlinux-x86_64-lambda-standard:corretto11`	al-lambda/x86_64/corretto11
Amazon Linux 2	`aws/codebuild/amazonlinux-x86_64-lam`	al-lambda/x86_64/corretto17

Plattform	Image-Kennung	Definition
	`bda-standard:corretto17`	
Amazon Linux 2023	`aws/codebuild/amazonlinux-x86_64-lambda-standard:corretto21`	al-lambda/x86_64/corretto21
Amazon Linux 2	`aws/codebuild/amazonlinux-x86_64-lambda-standard:nodejs18`	al-lambda/x86_64/nodejs18
Amazon Linux 2023	`aws/codebuild/amazonlinux-x86_64-lambda-standard:nodejs20`	al-lambda/x86_64/nodejs20
Amazon Linux 2	`aws/codebuild/amazonlinux-x86_64-lambda-standard:python3.11`	al-lambda/x86_64/python3.11
Amazon Linux 2023	`aws/codebuild/amazonlinux-x86_64-lambda-standard:python3.12`	al-lambda/x86_64/python3.12
Amazon Linux 2	`aws/codebuild/amazonlinux-x86_64-lambda-standard:ruby3.2`	al-lambda/x86_64/ruby3.2

Veraltete Bilder

Ein veraltetes Bild ist ein Bild, das nicht mehr zwischengespeichert oder aktualisiert wird. CodeBuild Ein veraltetes Image erhält keine kleineren Versionsupdates oder Patch-Versionsupdates mehr, und da sie nicht mehr aktualisiert werden, ist ihre Verwendung möglicherweise nicht sicher. Wenn Ihr CodeBuild Projekt für die Verwendung einer älteren Image-Version konfiguriert ist, lädt der Bereitstellungsprozess dieses Docker-Image herunter und verwendet es, um die containerisierte Laufzeitumgebung zu erstellen, wodurch die Bereitstellungsdauer und die Gesamtdauer der Erstellung verlängert werden können.

CodeBuild hat die folgenden Docker-Images als veraltet eingestuft. Sie können diese Images weiterhin verwenden, sie werden jedoch nicht auf dem Build-Host zwischengespeichert, was zu längeren Bereitstellungszeiten führt.

Plattform	Image-Kennung	Definition	Datum der Veraltung
Amazon Linux 2	`aws/codebuild/amazonlinux2-x86_64-standard:3.0`	al2/standard/3.0	09. Mai 2023
Ubuntu 18.04	`aws/codebuild/standard:4.0`	ubuntu/standard/4.0	31. März 2023
Amazon Linux 2	`aws/codebuild/amazonlinux2-aarch64-standard:1.0`	al2/aarch64/standard/1.0	31. März 2023
Ubuntu 18.04	`aws/codebuild/standard:3.0`	ubuntu/standard/3.0	30. Juni 2022
Amazon Linux 2	`aws/codebuild/amazonlinux2-x86_64-standard:2.0`	al2/standard/2.0	30. Juni 2022

Themen

Verfügbare Laufzeiten

Sie können eine oder mehrere Laufzeiten im Abschnitt `runtime-versions` Ihrer buildspec-Datei angeben. Wenn Ihre Laufzeit von einer anderen Laufzeit abhängig ist, können Sie auch die abhängige Laufzeit in der buildspec-Datei angeben. Wenn Sie in der Buildspec-Datei keine Laufzeiten angeben, CodeBuild wählt die Standard-Laufzeiten aus, die in dem von Ihnen verwendeten Image verfügbar sind. Wenn Sie eine oder mehrere Laufzeiten angeben, werden nur diese Laufzeiten verwendet. CodeBuild Wenn keine abhängige Laufzeit angegeben ist, wird CodeBuild versucht, die abhängige Laufzeit für Sie auszuwählen. Weitere Informationen finden Sie unter Specify runtime versions in the buildspec file.

Themen

Linux-Image-Laufzeiten

Die folgende Tabelle enthält die verfügbaren Laufzeiten und die Standard-Linux-Images, die sie unterstützen.

Laufzeiten für Ubuntu- und Amazon Linux-Plattformen

Laufzeitname	Version	Bilder
dotnet	3.1	Amazon Linux 2 AArch64-Standard: 2.0 Ubuntu-Standard: 5.0
	5.0	Ubuntu-Standard: 5.0
	6.0	Amazon Linux 2 x86_64 Lambda-Standard: dotnet6 Amazon Linux 2 AArch64 Lambda-Standard: dotnet6

Laufzeitname	Version	Bilder
		Amazon Linux 2 x86_64 Standard: 4.0 Amazon Linux 2023 x86_64 Standard:5.0 Amazon Linux 2023 AArch64-Standard:3.0 Ubuntu-Standard: 6.0 Ubuntu-Standard: 7.0
	8.0	Amazon Linux 2023 x86_64 Standard:5.0 Amazon Linux 2023 AArch64-Standard:3.0 Ubuntu-Standard: 7.0
Golang	1.12	Amazon Linux 2 AArch64-Standard: 2.0
	1.13	Amazon Linux 2 AArch64-Standard: 2.0
	1.14	Amazon Linux 2 AArch64-Standard: 2.0
	1.15	Ubuntu-Standard: 5.0
	1.16	Ubuntu-Standard: 5.0
	1,18	Amazon Linux 2 x86_64 Standard: 4.0 Ubuntu-Standard: 6.0

Laufzeitname	Version	Bilder
	1.20	Amazon Linux 2023 x86_64 Standard:5.0 Amazon Linux 2023 AArch64-Standard:3.0 Ubuntu-Standard: 7.0
	1.21	Amazon Linux 2 x86_64 Lambda-Standard: go1.21 Amazon Linux 2 AArch64 Lambda-Standard: go1.21 Amazon Linux 2023 x86_64 Standard:5.0 Amazon Linux 2023 AArch64-Standard:3.0 Ubuntu-Standard: 7.0
	1.22	Amazon Linux 2023 x86_64 Standard:5.0 Ubuntu-Standard: 7.0
Java	corretto8	Amazon Linux 2 x86_64 standard: corretto 8 Amazon Linux 2023 x86_64 Standard:5.0 Amazon Linux 2 AArch64-S tandard: 2.0 Ubuntu-Standard: 5.0 Ubuntu-Standard: 7.0

Laufzeitname	Version	Bilder
	corretto11	Amazon Linux 2 x86_64 Standard: Corretto 11 Amazon Linux 2 x86_64 Lambda-Standard:Corretto11 Amazon Linux 2023 x86_64 Standard:5.0 Amazon Linux 2 AArch64 Lambda-Standard: korrekt 11 Amazon Linux 2 AArch64-S tandard: 2.0 Ubuntu-Standard: 5.0 Ubuntu-Standard: 7.0
	Korretto 17	Amazon Linux 2 x86_64 Lambda-Standard: korrekt 17 Amazon Linux 2 AArch64 Lambda-Standard: korrekt 17 Amazon Linux 2 x86_64 Standard: 4.0 Amazon Linux 2023 x86_64 Standard:5.0 Amazon Linux 2023 AArch64-Standard:3.0 Ubuntu-Standard: 6.0 Ubuntu-Standard: 7.0

Laufzeitname	Version	Bilder
	Corretto 21	Amazon Linux 2 x86_64 Lambda-Standard: corretto 21 Amazon Linux 2 AArch64 Lambda-Standard: korrekt 21 Amazon Linux 2023 x86_64 Standard:5.0 Amazon Linux 2023 AArch64-Standard:3.0 Ubuntu-Standard: 7.0
nodejs	10	Amazon Linux 2 AArch64-Standard: 2.0
	12	Amazon Linux 2 AArch64-Standard: 2.0 Ubuntu-Standard: 5.0
	14	Ubuntu-Standard: 5.0
	16	Amazon Linux 2 x86_64 Standard: 4.0 Ubuntu-Standard: 6.0

Laufzeitname	Version	Bilder
	18	Amazon Linux 2 x86_64 Lambda-Standard: nodejs18 Amazon Linux 2 AArch64 Lambda-Standard: nodejs18 Amazon Linux 2023 x86_64 Standard:5.0 Amazon Linux 2023 AArch64-Standard:3.0 Ubuntu-Standard: 7.0
	20	Amazon Linux 2 x86_64 Lambda-Standard: nodejs20 Amazon Linux 2 AArch64 Lambda-Standard: nodejs20 Amazon Linux 2023 x86_64 Standard:5.0 Amazon Linux 2023 AArch64-Standard:3.0 Ubuntu-Standard: 7.0
php	7.3	Amazon Linux 2 AArch64-Standard: 2.0 Ubuntu-Standard: 5.0
	7.4	Amazon Linux 2 AArch64-Standard: 2.0 Ubuntu-Standard: 5.0
	8.0	Ubuntu-Standard: 5.0

Laufzeitname	Version	Bilder
	8.1	Amazon Linux 2 x86_64 Standard: 4.0 Amazon Linux 2023 AArch64-Standard:3.0 Ubuntu-Standard: 6.0
	8.2	Amazon Linux 2023 x86_64 Standard:5.0 Amazon Linux 2023 AArch64-Standard:3.0 Ubuntu-Standard: 7.0
	8.3	Amazon Linux 2023 x86_64 Standard:5.0 Amazon Linux 2023 AArch64-Standard:3.0 Ubuntu-Standard: 7.0
python	3.7	Amazon Linux 2 AArch64-S tandard: 2.0 Ubuntu-Standard: 5.0
	3.8	Amazon Linux 2 AArch64-S tandard: 2.0 Ubuntu-Standard: 5.0

Laufzeitname	Version	Bilder
	3.9	Amazon Linux 2 x86_64 Standard: 4.0 Amazon Linux 2023 x86_64 Standard:5.0 Amazon Linux 2 AArch64-S tandard: 2.0 Ubuntu-Standard: 5.0 Ubuntu-Standard: 7.0
	3.10	Amazon Linux 2023 x86_64 Standard:5.0 Ubuntu-Standard: 6.0 Ubuntu-Standard: 7.0
	3.11	Amazon Linux 2 x86_64 Lambda-Standard: Python 3.11 Amazon Linux 2 AArch64 Lambda-Standard: Python 3.11 Amazon Linux 2023 x86_64 Standard:5.0 Amazon Linux 2023 AArch64-Standard:3.0 Ubuntu-Standard: 7.0

Laufzeitname	Version	Bilder
	3.12	Amazon Linux 2 x86_64 Lambda-Standard: Python 3.12 Amazon Linux 2 AArch64 Lambda-Standard: Python 3.12 Amazon Linux 2023 x86_64 Standard:5.0 Amazon Linux 2023 AArch64-Standard:3.0 Ubuntu-Standard: 7.0
ruby	2.6	Amazon Linux 2 AArch64-Standard: 2.0 Ubuntu-Standard: 5.0
	2.7	Amazon Linux 2 AArch64-Standard: 2.0 Ubuntu-Standard: 5.0
	3.1	Amazon Linux 2 x86_64 Standard: 4.0 Amazon Linux 2023 x86_64 Standard:5.0 Ubuntu-Standard: 6.0 Ubuntu-Standard: 7.0

Laufzeitname	Version	Bilder
	3.2	Amazon Linux 2 x86_64 Lambda-Standard: Ruby3.2 Amazon Linux 2 AArch64 Lambda-Standard: Ruby3.2 Amazon Linux 2023 x86_64 Standard:5.0 Amazon Linux 2023 AArch64-Standard:3.0 Ubuntu-Standard: 7.0
	3.3	Amazon Linux 2023 x86_64 Standard:5.0 Ubuntu-Standard: 7.0

Windows-Image-Laufzeiten

Das Basisimage von Windows Server Core 2019 enthält die folgenden Laufzeiten.

Laufzeiten der Windows-Plattform

Laufzeitname	Windows Server Core 2019-Standard: 1.0-Versionen	Windows Server Core 2019-Standard:2.0-Versionen	Windows Server Core 2019-Standard:3.0-Versionen
dotnet	3.1 5.0	3.1 6.0 7.0	6.0 7.0 8.0
Dotnet-SDK	3.1 5.0	3.1 6.0	8.0

Laufzeitname	Windows Server Core 2019-Standard: 1.0-Versionen	Windows Server Core 2019-Standard:2.0-Versionen	Windows Server Core 2019-Standard:3.0-Versionen
		7.0	
Golang	1.14	1,18	1,21
Gradel	6.7	7.6	8.5
Java	Korretto 11	Korretto 11 Korretto 17	Korretto 21
Maven	3.6	3.8	3.9
nodejs	14,15	16,19	20,11
php	7.4	8.1	8.3
powershell	7.1	7.2	7.4
python	3.8	3,10	3,12
ruby	2.7	3.1	3.3

Laufzeitversionen

Wenn Sie eine Laufzeit im Abschnitt `runtime-versions` Ihrer BuildSpec-Datei angeben, können Sie eine bestimmte Version, eine spezifische Hauptversion und die neueste Unterversion oder die neueste Version angeben. In der folgenden Tabelle sind die verfügbaren Laufzeiten und deren Angabe aufgeführt. Nicht alle Runtime-Versionen sind auf allen Images verfügbar. Die Auswahl der Runtime-Version wird für die benutzerdefinierten Images ebenfalls nicht unterstützt. Weitere Informationen finden Sie unter Verfügbare Laufzeiten. Wenn Sie anstelle der vorinstallierten Runtime-Versionen eine benutzerdefinierte Runtime-Version installieren und verwenden möchten, finden Sie weitere Informationen unterBenutzerdefinierte Runtime-Versionen.

Laufzeitversionen der Plattformen Ubuntu und Amazon Linux 2

Laufzeitname	Version	Spezifische Version	Spezifische Haupt- und neueste Unterversion	Aktuelle Version
android	28	`android: 28`	`android: 28.x`	`android: latest`
	29	`android: 29`	`android: 29.x`	
dotnet	3.1	`dotnet: 3.1`	`dotnet: 3.x`	`dotnet: latest`
	5.0	`dotnet: 5.0`	`dotnet: 5.x`	
	6.0	`dotnet: 6.0`	`dotnet: 6.x`	
	8.0	`dotnet: 8.0`	`dotnet: 8.x`	
Golang	1.12	`golang: 1.12`	`golang: 1.x`	`golang: latest`
	1.13	`golang: 1.13`		
	1.14	`golang: 1.14`		
	1.15	`golang: 1.15`		
	1.16	`golang: 1.16`		
	1.18	`golang: 1.18`		
	1,20	`golang: 1.20`		
	1,21	`golang: 1.21`		
	1,22	`golang: 1.22`		
Java	corretto8	`java: corretto`	`java: corretto.x`	`java: latest`
	corretto11	`java: corretto11`	`java: corretto1.x`	

Laufzeitname	Version	Spezifische Version	Spezifische Haupt- und neueste Unterversion	Aktuelle Version
	Korretto 17	`java: corretto 7`	`java: corretto 7.x`	
	Corretto 21	`java: corretto 1`	`java: corretto 1.x`	
nodejs	10	`nodejs: 10`	`nodejs: 10.x`	`nodejs: latest`
	12	`nodejs: 12`	`nodejs: 12.x`	
	14	`nodejs: 14`	`nodejs: 14.x`	
	16	`nodejs: 16`	`nodejs: 16.x`	
	18	`nodejs: 18`	`nodejs: 18.x`	
	20	`nodejs: 20`	`nodejs: 20.x`	
php	7.3	`php: 7.3`	`php: 7.x`	`php: latest`
	7.4	`php: 7.4`		
	8.0	`php: 8.0`	`php: 8.x`	
	8.1	`php: 8.1`		
	8.2	`php: 8.2`		
	8.3	`php: 8.3`		
python	3.7	`python: 3.7`	`python: 3.x`	`python: latest`
	3.8	`python: 3.8`		
	3.9	`python: 3.9`		

Laufzeitname	Version	Spezifische Version	Spezifische Haupt- und neueste Unterversion	Aktuelle Version
ruby	3,10	`python: 3.10`	`ruby: 2.x`	`ruby: latest`
	3,11	`python: 3.11`		
	3,12	`python: 3.12`		
	2.6	`ruby: 2.6`		
	2.7	`ruby: 2.7`		
	3.1	`ruby: 3.1`	`ruby: 3.x`	
	3.2	`ruby: 3.2`		
	3.3	`ruby: 3.3`		

Sie können eine Build-Spezifikation verwenden, um andere Komponenten (z. B. Apache Maven AWS CLI, Apache Ant, Mocha, RSpec oder ähnliche) während der Buildphase zu installieren. `install` Weitere Informationen finden Sie unter Beispiel für eine Build-Spezifikation.

Benutzerdefinierte Runtime-Versionen

Anstatt die vorinstallierten Laufzeitversionen in CodeBuild -verwalteten Images zu verwenden, können Sie benutzerdefinierte Versionen Ihrer Wahl installieren und verwenden. In der folgenden Tabelle sind die verfügbaren benutzerdefinierten Laufzeiten und deren Angabe aufgeführt.

> **Note**
>
> Die Auswahl einer benutzerdefinierten Runtime-Version wird nur für Ubuntu- und Amazon Linux-Images unterstützt.

Benutzerdefinierte Runtime-Versionen

Laufzeitname	Syntax	Beispiel
dotnet	`<major>.<minor>.<patch>`	`5.0.408`
Golang	`<major>.<minor>` `<major>.<minor>.<patch>`	`1.19` `1.19.1`
Java	`corretto<major>`	`corretto15`

nodejs	`<major>`
	`<major>.<minor>`
	`<major>.<minor>.<patch>`
		`14`
		`14.21`
		`14.21.3`

php	`<major>.<minor>.<patch>`	`8.0.30`

python	`<major>`
	`<major>.<minor>`
	`<major>.<minor>.<patch>`
		`3`
		`3.7`
		`3.7.16`

ruby	`<major>.<minor>.<patch>`	`3.0.6`

Beispiel für eine benutzerdefinierte Runtime-Buildspec

Hier ist ein Beispiel für eine Buildspec, die benutzerdefinierte Laufzeitversionen spezifiziert.

```
version: 0.2
phases:
  install:
    runtime-versions:
      java: corretto15
      php: 8.0.30
      ruby: 3.0.6
      golang: 1.19
      python: 3.7
      nodejs: 14
```

```
      dotnet: 5.0.408
```

Berechnungsmodi und Typen der Build-Umgebung

In CodeBuild können Sie das Image der Rechen- und Laufzeitumgebung angeben, das zum Ausführen Ihrer Builds CodeBuild verwendet wird. Compute bezieht sich auf die Computing-Engine (die CPU, den Arbeitsspeicher und das Betriebssystem), die von verwaltet und gewartet wird CodeBuild. Ein Laufzeitumgebungs-Image ist ein Container-Image, das auf der von Ihnen ausgewählten Rechenplattform ausgeführt wird und zusätzliche Tools enthält, die Ihr Build möglicherweise benötigt, z. B. das AWS CLI.

Themen

Über Rechenmodi

CodeBuild bietet die folgenden Rechenmodi:

- EC2
- AWS Lambda

EC2 bietet optimierte Flexibilität beim Erstellen und AWS Lambda bietet optimierte Startgeschwindigkeiten. AWS Lambda unterstützt schnellere Builds aufgrund einer geringeren Startlatenz. AWS Lambda skaliert außerdem automatisch, sodass Builds nicht in der Warteschlange warten, bis sie ausgeführt werden. Weitere Informationen finden Sie unter Arbeiten mit AWS Lambda Compute in AWS CodeBuild.

Über Umgebungstypen

AWS CodeBuild bietet Build-Umgebungen mit dem folgenden verfügbaren Arbeitsspeicher, den folgenden vCPUs und dem folgenden Festplattenspeicher für den EC2-Rechenmodus:

Datenverarbeitung	ComputeType-Wert für die Umgebung	Wert für den Umgebungstyp	Arbeitsspeicher	vCPUs	Festplattenkapazität
ARM Klein	`BUILD_GENERAL1_SMALL`	`ARM_CONTAINER`	4 GB	2	64 GB
ARM Groß	`BUILD_GENERAL1_LARGE`	`ARM_CONTAINER`	16 GB	8	128 GB
Linux Klein [1]	`BUILD_GENERAL1_SMALL`	`LINUX_CONTAINER`	3 GB	2	64 GB
Linux Medium [1]	`BUILD_GENERAL1_MEDIUM`	`LINUX_CONTAINER`	7 GB	4	128 GB
Linux Large [1]	`BUILD_GENERAL1_LARGE`	`LINUX_CONTAINER`	15 GB	8	128 GB
Linux XLarge	`BUILD_GENERAL1_XLARGE`	`LINUX_CONTAINER`	70 GB	36	256 GB
Linux 2 x groß	`BUILD_GENERAL1_2XLARGE`	`LINUX_CONTAINER`	145 GB	72	824 GB (SSD)
Kleine Linux-GPU	`BUILD_GENERAL1_SMALL`	`LINUX_GPU_CONTAINER`	16 GB	4	220 GB

Datenverarbeitung	ComputeType-Wert für die Umgebung	Wert für den Umgebungstyp	Arbeitsspeicher	vCPUs	Festplattenkapazität
Große Linux-GPU	`BUILD_GENERAL1_LARGE`	`LINUX_GPU_CONTAINER`	255 GB	32	50 GB
Windows Medium	`BUILD_GENERAL1_MEDIUM`	`WINDOWS_SERVER_2019_CONTAINER`	7 GB	4	128 GB
Windows groß	`BUILD_GENERAL1_LARGE`	`WINDOWS_SERVER_2019_CONTAINER`	15 GB	8	128 GB

[1] Die neueste Version dieses Bildtyps wird zwischengespeichert. Wenn Sie eine spezifischere Version angeben, wird CodeBuild diese Version anstelle der zwischengespeicherten Version bereitgestellt. Dies kann die Build-Zeiten verlängern. Um von der Zwischenspeicherung zu profitieren, geben Sie beispielsweise `aws/codebuild/amazonlinux2-x86_64-standard:5.0` anstelle einer granularen Version wie `aws/codebuild/amazonlinux2-x86_64-standard:5.0-1.0.0` an.

AWS CodeBuild stellt Build-Umgebungen mit dem folgenden verfügbaren Arbeitsspeicher und Festplattenspeicher für den AWS Lambda Rechenmodus bereit:

Datenverarbeitung	ComputeType-Wert für die Umgebung	Wert für den Umgebungstyp	Arbeitsspeicher	Festplattenkapazität
ARM Lambda 1 GB	`BUILD_LAMBDA_1GB`	`ARM_LAMBDA_CONTAINER`	1 GB	10 GB

Datenverarbeitung	ComputeType-Wert für die Umgebung	Wert für den Umgebungstyp	Arbeitsspeicher	Festplattenkapazität
ARM Lambda 2 GB	`BUILD_LAMBDA_2GB`	`ARM_LAMBDA_CONTAINER`	2 GB	10 GB
ARM Lambda 4 GB	`BUILD_LAMBDA_4GB`	`ARM_LAMBDA_CONTAINER`	4 GB	10 GB
ARM Lambda 8 GB	`BUILD_LAMBDA_8GB`	`ARM_LAMBDA_CONTAINER`	8 GB	10 GB
ARM Lambda 10 GB	`BUILD_LAMBDA_10GB`	`ARM_LAMBDA_CONTAINER`	10 GB	10 GB
Linux Lambda 1 GB	`BUILD_LAMBDA_1GB`	`LINUX_LAMBDA_CONTAINER`	1 GB	10 GB
Linux Lambda 2 GB	`BUILD_LAMBDA_2GB`	`LINUX_LAMBDA_CONTAINER`	2 GB	10 GB
Linux Lambda 4 GB	`BUILD_LAMBDA_4GB`	`LINUX_LAMBDA_CONTAINER`	4 GB	10 GB
Linux Lambda 8 GB	`BUILD_LAMBDA_8GB`	`LINUX_LAMBDA_CONTAINER`	8 GB	10 GB

Datenverarbeitung	ComputeType-Wert für die Umgebung	Wert für den Umgebungstyp	Arbeitsspeicher	Festplattenkapazität
Linux Lambda 10 GB	`BUILD_LAMBDA_10GB`	`LINUX_LAMBDA_CONTAINER`	10 GB	10 GB

Wenn Sie andere Umgebungstypen verwenden, wird empfohlen, ein zwischengespeichertes Image zu verwenden, um die Build-Zeiten zu reduzieren.

Der für jede Build-Umgebung aufgelistete Speicherplatz ist nur in dem durch die `CODEBUILD_SRC_DIR`-Umgebungsvariable angegebenen Verzeichnis verfügbar.

So wählen Sie einen Datenverarbeitungstyp aus:

- Erweitern Sie in der CodeBuild Konsole im Assistenten „Build-Projekt erstellen“ oder auf der Seite „Build-Projekt bearbeiten“ unter Umgebung die Option Zusätzliche Konfiguration und wählen Sie dann eine der Optionen unter Berechnungstyp aus. Weitere Informationen finden Sie unter Erstellen Sie ein Build-Projekt (Konsole) oder Ändern der Einstellungen eines Build-Projekts (Konsole).
- Führen Sie für den AWS CLI den `update-project` Befehl `create-project` oder aus und geben Sie dabei den `computeType` Wert des `environment` Objekts an. Weitere Informationen finden Sie unter Erstellen eines Build-Projekts (AWS CLI) oder Ändern der Einstellungen eines Build-Projekts (AWS CLI).
- Rufen Sie für die AWS SDKs das Äquivalent der `UpdateProject` Operation `CreateProject` or für Ihre Zielprogrammiersprache auf und geben Sie dabei das Äquivalent zum `computeType` Wert des `environment` Objekts an. Weitere Informationen hierzu finden Sie unter AWS SDKs- und Tools-Referenz.

Für einige Umgebungs- und Rechenarten gelten regionale Verfügbarkeitsbeschränkungen:

- Der Compute-Typ Linux GPU Small (`LINUX_GPU_CONTAINER`) ist nur in diesen Regionen verfügbar:
  - USA Ost (Nord-Virginia)
  - USA West (Oregon)

- Asien-Pazifik (Tokio)
- Canada (Central)
- Europe (Frankfurt)
- Europa (Irland)
- Europe (London)

- Der Compute-Typ Linux GPU Large (`LINUX_GPU_CONTAINER`) ist nur in diesen Regionen verfügbar:
  - US East (Ohio)
  - USA Ost (Nord-Virginia)
  - USA West (Oregon)
  - Asien-Pazifik (Seoul)
  - Asien-Pazifik (Singapur)
  - Asien-Pazifik (Sydney)
  - Asien-Pazifik (Tokio)
  - Canada (Central)
  - China (Peking)
  - China (Ningxia)
  - Europe (Frankfurt)
  - Europa (Irland)
  - Europe (London)
- Der Umgebungstyp `ARM_CONTAINER` ist nur in diesen Regionen verfügbar:
  - US East (Ohio)
  - USA Ost (Nord-Virginia)
  - USA West (Nordkalifornien)
  - USA West (Oregon)
  - Asien-Pazifik (Hongkong)
  - Asien-Pazifik (Jakarta)
  - Asien-Pazifik (Hyderabad)
  - Asien-Pazifik (Mumbai)
  - Asia Pacific (Osaka)

- Asia Pacific (Seoul)
- Asien-Pazifik (Singapur)
- Asien-Pazifik (Sydney)
- Asien-Pazifik (Tokio)
- Canada (Central)
- China (Peking)
- China (Ningxia)
- Europe (Frankfurt)
- Europa (Irland)
- Europa (London)
- Europa (Milan)
- Europa (Paris)
- Europa (Spain)
- Europa (Stockholm)
- Israel (Tel Aviv)
- Naher Osten (Bahrain)
- Naher Osten (VAE)
- Südamerika (São Paulo)

- Der Compute-Typ `BUILD_GENERAL1_2XLARGE` ist nur in diesen Regionen verfügbar:
  - US East (Ohio)
  - USA Ost (Nord-Virginia)
  - USA West (Nordkalifornien)
  - USA West (Oregon)
  - Asien-Pazifik (Hyderabad)
  - Asien-Pazifik (Hongkong)
  - Asien-Pazifik (Jakarta)
  - Asien-Pazifik (Melbourne)
  - Asien-Pazifik (Mumbai)
  - Asia Pacific (Seoul)
  - Asien-Pazifik (Singapur)

- Asien-Pazifik (Sydney)
- Asien-Pazifik (Tokio)
- Canada (Central)
- China (Peking)
- China (Ningxia)
- Europe (Frankfurt)
- Europa (Irland)
- Europe (London)
- Europa (Paris)
- Europa (Spain)
- Europa (Stockholm)
- Europa (Zürich)
- Israel (Tel Aviv)
- Naher Osten (Bahrain)
- Naher Osten (VAE)
- Südamerika (São Paulo)

- Der Rechenmodus AWS Lambda (`ARM_LAMBDA_CONTAINER`und`LINUX_LAMBDA_CONTAINER`) ist nur in diesen Regionen verfügbar:
  - USA Ost (Nord-Virginia)
  - USA Ost (Ohio)
  - USA West (Oregon)
  - Asia Pacific (Mumbai)
  - Asien-Pazifik (Singapur)
  - Asien-Pazifik (Sydney)
  - Asien-Pazifik (Tokio)
  - Europe (Frankfurt)
  - Europa (Irland)
  - Südamerika (São Paulo)

Für den Compute-Typ `BUILD_GENERAL1_2XLARGE` werden Docker-Images bis zu 100 GB unkomprimiert unterstützt.

> **Note**
>
> CodeBuild Unterstützt für benutzerdefinierte Build-Umgebungsimages unabhängig vom Rechnertyp Docker-Images mit bis zu 50 GB unkomprimiert unter Linux und Windows. Zum Prüfen der Größe des Build-Image nutzen Sie Docker und führen den Befehl `docker images REPOSITORY:TAG` aus.

Sie können Amazon EFS verwenden, um auf mehr Speicherplatz in Ihrem Build-Container zuzugreifen. Weitere Informationen finden Sie unter Amazon Elastic File System-Beispiel für AWS CodeBuild. Wenn Sie während eines Builds Container-Festplattenspeicher bearbeiten möchten, dann muss der Build im privilegierten Modus ausgeführt werden.

> **Note**
>
> Standardmäßig ist der Docker-Daemon für Nicht-VPC-Builds aktiviert. Wenn Sie Docker-Container für VPC-Builds verwenden möchten, lesen Sie auf der Docker Docs-Website unter Runtime Privilege and Linux Capabilities nach und aktivieren Sie den privilegierten Modus. Außerdem unterstützt Windows den privilegierten Modus nicht.

Shells und Befehle in Build-Umgebungen

Sie geben eine Reihe von Befehlen für AWS CodeBuild an, die in einer Build-Umgebung während des Build-Lebenszyklus ausgeführt werden (z. B. Installieren der Build-Abhängigkeiten sowie Testen und Kompilieren Ihres Quellcodes). Es gibt mehrere Möglichkeiten, diese Befehle anzugeben:

- Erstellen Sie eine Build-Spezifikationsdatei und schließen Sie diese in Ihren Quellcode ein. In dieser Datei geben Sie die Befehle an, die Sie in jeder Phase des Build-Lebenszyklus ausführen möchten. Weitere Informationen hierzu finden Sie unter Referenz zur Build-Spezifikation für CodeBuild.
- Verwenden Sie zur Erstellung eines Build-Projekts die CodeBuild-Konsole. Geben Sie unter Insert build commands (Build-Befehle eingeben) für Build commands (Build-Befehle) die Befehle ein, die Sie in der `build`-Phase ausführen möchten. Weitere Informationen finden Sie unter Erstellen Sie ein Build-Projekt (Konsole).
- Verwenden Sie die CodeBuild-Konsole, um die Einstellungen eines Build-Projekts zu ändern. Geben Sie unter Insert build commands (Build-Befehle eingeben) für Build commands (Build-

Befehle) die Befehle ein, die Sie in der `build`-Phase ausführen möchten. Weitere Informationen finden Sie unter Ändern der Einstellungen eines Build-Projekts (Konsole).

- Nutzen Sie die AWS CLI oder AWS-SDKs zur Erstellung eines Build-Projekts oder zum Ändern der Einstellungen eines solchen. Greifen Sie auf den Quellcode zu, der eine Build-Spezifikationsdatei mit Ihren Befehlen enthält, oder geben Sie eine einzelne Zeichenfolge ein, die die Inhalte einer äquivalenten Build-Spezifikationsdatei enthält. Weitere Informationen finden Sie unter Erstellen eines Build-Projekts oder Ändern der Einstellungen eines Build-Projekts.
- Verwenden Sie die AWS CLI- oder AWS-SDKs, um einen Build zu starten, der eine Build-Spezifikationsdatei oder eine einzelne Zeichenfolge angibt, die die Inhalte einer äquivalenten Build-Spezifikationsdatei enthält. Weitere Informationen finden Sie in der Beschreibung für den Wert `buildspecOverride` in Ausführen eines Build.

Sie können einen beliebigen Shell Command Language (sh)-Befehl angeben. In der Build-Spezifikationsversion 0.1 führt CodeBuild jeden Shell-Befehl in einer separaten Instance in der Build-Umgebung aus. d. h., dass jeder Befehl unabhängig von allen anderen Befehlen ausgeführt wird. Daher können Sie standardmäßig keinen Einzelbefehl ausführen, der auf dem Status eines vorherigen Befehls basiert (beispielsweise beim Ändern von Verzeichnissen oder beim Einrichten von Variablen). Um diese Einschränkung zu umgehen, empfehlen wir die Nutzung von Version 0.2, die dieses Problem löst. Wenn Sie Version 0.1 verwenden müssen, empfehlen wir folgenden Ansätze:

- Schließen Sie ein Shell-Skript in Ihren Quellcode ein, das die Befehle, die Sie ausführen möchten, in einer einzelnen Instance der Standard-Shell enthält. Sie können beispielsweise eine Datei mit Namen `my-script.sh` in Ihren Quellcode einschließen, der Befehle enthält, wie `cd MyDir; mkdir -p mySubDir; cd mySubDir; pwd;`. Geben Sie dann in Ihrer Build-Spezifikation den Befehl `./my-script.sh` an.
- Geben Sie in Ihrer Build-Spezifikationsdatei oder für die Build commands (Build-Befehle)-Einstellung ausschließlich für die `build`-Phase einen einzelnen Befehl an, der alle Befehle enthält, die Sie in einer einzelnen Instance der Standard-Shell ausführen möchten (z. B `cd MyDir && mkdir -p mySubDir && cd mySubDir && pwd`).

Wenn bei CodeBuild-Fehler auftritt, ist der Fehler schwerer zu beheben. Es ist vergleichsweise einfacher, einen einzelnen Befehl eigenständig in seiner eigenen Instance der Standard-Shell auszuführen.

Befehle, die in einem Windows Server Core-Abbild ausgeführt werden, verwenden die PowerShell.

Umgebungsvariablen in Build-Umgebungen

AWS CodeBuild bietet mehrere Umgebungsvariablen, die Sie in Ihren Build-Befehlen verwenden können:

AWS_DEFAULT_REGION

DieAWS Region, in der der Build ausgeführt wird (z. B.`us-east-1`). Diese Umgebungsvariable wird hauptsächlich bei der AWS CLI verwendet.

AWS_REGION

DieAWS Region, in der der Build ausgeführt wird (z. B.`us-east-1`). Diese Umgebungsvariable wird hauptsächlich von den AWS SDKs verwendet.

CODEBUILD_BATCH_BUILD_IDENTIFIER

Die ID des Builds in einem Batch-Build. Dies ist in der Batch-Buildspec angegeben. Weitere Informationen finden Sie unter the section called “Batch−Build-Spezifikationsreferenz”.

CODEBUILD_BUILD_ARN

Der Amazon-Ressourcenname (ARN ARN ARN N N N ARN N N ARN N ARN N ARN N ARN N ARNarn:aws:codebuild:*region-ID*:*account-ID*:build/codebuild-demo-project:b1e6661e-e4f2-4156-9ab9-82a19EXAMPLE

CODEBUILD_BUILD_ID

DieCodeBuild ID des Builds (zum Beispielcodebuild-demo-project:b1e6661e-e4f2-4156-9ab9-82a19EXAMPLE).

CODEBUILD_BUILD_IMAGE

DieCodeBuild Build-Image-ID (z. B.aws/codebuild/standard:2.0).

CODEBUILD_BUILD_NUMBER

Build-Nummer des Projekt.

CODEBUILD_BUILD_SUCCESS

Ob der aktuelle Build erfolgreich ist. Legen Sie den Wert auf `0` fest, wenn der Build fehlschlägt, und auf 1, wenn er erfolgreich ausgeführt wird.

CODEBUILD-INITIATOR

Die Entität, die den Build gestartet hat. Wenn CodePipeline den Build gestartet hat, ist dies der Name der Pipeline (beispielsweise `codepipeline/my-demo-pipeline`). Wenn ein Benutzer den Build gestartet hat, ist dies der Name des Benutzers (z. B.`MyUserName`). Wenn das Jenkins-Plugin für CodeBuild den Build gestartet hat, lautet die Zeichenfolge `CodeBuild-Jenkins-Plugin`.

CODEBUILD_KMS_KEY_ID

Die Kennung desAWS KMS Schlüssels, der zum Verschlüsseln des Build-Ausgabeartefakts verwendetCodeBuild wird (z. B.`arn:aws:kms:`*`region-ID`*`:`*`account-ID`*`:key/`*`key-ID`* oder`alias/`*`key-alias`*).

CODEBUILD_LOG-PFAD

Der Name des Log-Streams inCloudWatch Logs für den Build.

CODEBUILD_PUBLIC_BUILD_URL

Die URL der Build-Ergebnisse für diesen Build auf der öffentlichen Build-Website. Diese Variable wird nur gesetzt, wenn für das Build-Projekt öffentliche Builds aktiviert sind. Weitere Informationen finden Sie unter Öffentliche Build-Projekte inAWS CodeBuild.

CODEBUILD_RESOLVED_QUELLVERSION

Die Versionskennung des Quellcodes eines Builds. Der Inhalt hängt vom Quellcode-Repository ab:

CodeCommitGitHub,GitHub Enterprise Server und Bitbucket

Diese Variable enthält die Commit-ID.

CodePipeline

Diese Variable enthält die Quellversion, die von bereitgestellt wurdeCodePipeline.

WennCodePipeline die Quellversion nicht aufgelöst werden kann, z. B. wenn es sich bei der Quelle um einen Amazon S3 S3-Bucket handelt, für den die Versionierung nicht aktiviert ist, wird diese Umgebungsvariable nicht gesetzt.

Amazon S3

Diese Variable ist nicht gesetzt.

Falls zutreffend, ist die`CODEBUILD_RESOLVED_SOURCE_VERSION` Variable erst nach der`DOWNLOAD_SOURCE` Phase verfügbar.

CODEBUILD_SOURCE_REPO_URL

Die URL zum Eingabeartefakt oder Quellcode-Repository. Für Amazon S3`s3://` folgen darauf der Bucket-Name und der Pfad zum Eingabeartefakt. FürCodeCommit undGitHub ist dies die Klon-URL des Repositorys. Wenn ein Build von stammtCodePipeline, kann diese Umgebungsvariable leer sein.

Für sekundäre Quellen lautet die Umgebungsvariable für die URL des sekundären Quell-Repositorys`CODEBUILD_SOURCE_REPO_URL_`*`<sourceIdentifier>`*, wo*`<sourceIdentifier>`* ist die Quell-ID, die Sie erstellen.

CODEBUILD-QUELLVERSION

Das Format des Werts hängt vom Quell-Repository ab.

- Für Amazon S3 ist dies die Versions-ID, die dem Eingabeartefakt zugeordnet ist.
- Bei CodeCommit handelt es sich um die Commit-ID oder den Branch-Namen, der zur Version des zu erstellenden Quellcodes gehört.
- FürGitHubGitHub Enterprise Server und Bitbucket ist es Commit-ID, Branch-Name oder Tag-Name, die/der der Version des Quellcodes entspricht, die Sie erstellen möchten.

> **Note**
>
> Für einenGitHub oderGitHub Enterprise Server-Build, der durch ein Webhook-Pull-Request-Ereignis ausgelöst wird, ist dies der Fall`pr/`*`pull-request-number`*.

Für sekundäre Quellen lautet die Umgebungsvariable für die sekundäre Quellversion`CODEBUILD_SOURCE_VERSION_`*`<sourceIdentifier>`*, wo*`<sourceIdentifier>`* ist die Quell-ID, die Sie erstellen. Weitere Informationen finden Sie unter Beispiel für mehrere Eingabequellen und Ausgabeartefakte.

CODEBUILD_SRC_DIR

Der Verzeichnispfad, der für den BuildCodeBuild verwendet wird (z. B.`/tmp/src123456789/src`).

Für sekundäre Quellen lautet die Umgebungsvariable für den sekundären Quellverzeichnispfad`CODEBUILD_SRC_DIR_`*`<sourceIdentifier>`*,

wo*<sourceIdentifier>* ist die Quell-ID, die Sie erstellen. Weitere Informationen finden Sie unter Beispiel für mehrere Eingabequellen und Ausgabeartefakte.

START_TIME

Die als Unix-Zeitstempel angegebene Startzeit des Builds in Millisekunden.

CODEBUILD_WEBHOOK_ACTOR_ACCOUNT_ID

Die Konto-ID des Benutzers, der das Webhook-Ereignis ausgelöst hat.

CODEBUILD_WEBHOOK_BASE_REF

Der Basisreferenzname des Webhook-Ereignisses, das den aktuellen Build auslöst. Bei einer Pull-Anforderung handelt es sich hierbei um die Verzweigungsreferenz.

CODEBUILD_WEBHOOK_EVENT

Das Webhook-Ereignis, das den aktuellen Build auslöst.

CODEBUILD_WEBHOOK_MERGE_COMMIT

Die ID des Merge-Commits, der für den Build verwendet wurde. Diese Variable wird gesetzt, wenn ein Bitbucket-Pull-Request mit der Squash-Strategie zusammengeführt wird und der Pull-Request-Zweig geschlossen wird. In diesem Fall existiert der ursprüngliche Pull-Request-Commit nicht mehr, daher enthält diese Umgebungsvariable den Identifier des gequetschten Merge-Commits.

CODEBUILD_WEBHOOK_PREV_COMMIT

Die ID des letzten Commits vor dem Webhook-Push-Ereignis, das den aktuellen Build auslöst.

CODEBUILD_WEBHOOK_HEAD_REF

Der Hauptreferenzname des Webhook-Ereignisses, das den aktuellen Build auslöst. Hierbei kann es sich um eine Verzweigungsreferenz oder um eine Tag-Referenz handeln.

CODEBUILD_WEBHOOK_TRIGGER

Zeigt das Webhook-Ereignis an, das den Build ausgelöst hat. Diese Variable ist nur für Builds verfügbar, die von einem Webhook ausgelöst wurden. Der Wert wird anhand der Payload analysiertGitHub, dieCodeBuild vonGitHub Enterprise Server oder Bitbucket gesendet wurde. Der Wert des Formats hängt davon ab, welche Art von Ereignis den Build ausgelöst hat.

- Für Builds, die von einer Pull-Anforderung ausgelöst wurden, handelt es sich um `pr/`*`pull-request-number`*.

- Für Builds, die durch das Erstellen eines neuen Branches oder durch Pushen eines Commits für einen Branch ausgelöst wurden, handelt es sich um `branch/`*`branch-name`*.
- Für Builds, die durch das Pushen eines Tags in ein Repository ausgelöst wurden, handelt es sich um `tag/`*`tag-name`*.

ZUHAUSE

Diese Umgebungsvariable ist immer auf gesetzt`/root`.

Sie können auch Build-Umgebungen mit Ihren eigenen Umgebungsvariablen liefern. Weitere Informationen finden Sie unter den folgenden Themen:

- Verwenden von CodePipeline mit CodeBuild
- Erstellen eines Build-Projekts
- Ändern der Einstellungen eines Build-Projekts
- Ausführen eines Build
- Build-Spezifikationsreferenz

Zur Auflistung der verfügbaren Umgebungsvariablen in einer Build-Umgebung können Sie während eines Builds den Befehl `printenv` ausführen (für eine Linux-basierte Build-Umgebung), oder `"Get-ChildItem Env:"` (für Windows-basierte Build-Umgebungen). Mit Ausnahme der vorstehend aufgeführten Umgebungsvariablen sind die mit `CODEBUILD_` beginnenden Umgebungsvariablen nur für die interne Nutzung in CodeBuild bestimmt. Diese sollten nicht in Ihrem Build-Befehlen eingesetzt werden.

> **Important**
>
> Wir raten dringend von der Verwendung von Umgebungsvariablen zum Speichern sensibler Werte, insbesondere vonAWS Zugriffsschlüssel-IDs, ab. Umgebungsvariablen können mit Tools wie der CodeBuild-Konsole und über die AWS CLI im Klartext angezeigt werden. Wir empfehlen Ihnen, vertrauliche Werte im Amazon EC2 Systems Manager Parameter Store zu speichern und sie dann aus Ihrer Buildspec abzurufen. Informationen zum Speichern vertraulicher Werte finden Sie unter Systems Manager Parameter Store and Walkthrough: Create and test a String-Parameter (Konsole) im Amazon EC2 Systems Manager Manager-Benutzerhandbuch. Informationen zum Abrufen dieser Variablen finden Sie unter der `parameter-store`-Zuordnung in Syntax der Build-Spezifikation.

Hintergrundaufgaben in Build-Umgebungen

Sie können Hintergrundaufgaben in Build-Umgebungen ausführen. Dazu verwenden Sie in Ihrer Build-Spezifikation den Befehl nohup, um einen Befehl auch dann als Aufgabe im Hintergrund auszuführen, wenn der Build-Prozess die Shell verlässt. Mit dem Befehl disown können Sie eine laufende Hintergrundaufgabe zwangsweise stoppen.

Beispiele:

- Starten Sie einen Hintergrundprozess und warten Sie, bis dieser später abgeschlossen ist:

```
|
nohup sleep 30 & echo $! > pidfile
…
wait $(cat pidfile)
```

- Starten Sie einen Hintergrundprozess und warten Sie nicht darauf, dass dieser jemals abgeschlossen ist:

```
|
nohup sleep 30 & disown $!
```

- Starten Sie einen Hintergrundprozess und beenden Sie ihn später:

```
|
nohup sleep 30 & echo $! > pidfile
…
kill $(cat pidfile)
```

Führen Sie Builds lokal aus mit demAWS CodeBuildAgentin

Sie können das verwendenAWS CodeBuildauszuführender AgentCodeBuildbaut auf einem lokalen Computer auf. Es sind Agenten für x86_64- und ARM-Plattformen verfügbar.

Sie können sich auch anmelden, um Benachrichtigungen zu erhalten, wenn neue Versionen des Agenten veröffentlicht werden.

Voraussetzungen

Bevor Sie beginnen, müssen Sie Folgendes tun:

- Installieren Sie Git auf Ihrem lokalen Computer.
- Installieren und einrichtenAndockstationauf Ihrem lokalen Computer.

Richten Sie das Build-Image ein

Sie müssen das Build-Image nur einrichten, wenn Sie den Agenten zum ersten Mal ausführen oder wenn sich das Image geändert hat.

Um das Build-Image einzurichten

1. Wenn Sie ein kuratiertes Amazon Linux 2-Image verwenden möchten, können Sie es aus demCodeBuildöffentliches Amazon ECR-Repository unterhttps://gallery.ecr.aws/codebuild/amazonlinux2-x86_64-standardmit dem folgenden Befehl:

   ```
   $ docker pull public.ecr.aws/codebuild/amazonlinux2-x86_64-standard:4.0
   ```

   Wenn Sie ein anderes Linux-Image verwenden möchten, führen Sie alternativ die folgenden Schritte aus:

   a. Klonen Sie dieCodeBuildBild-Repo:

      ```
      $ git clone https://github.com/aws/aws-codebuild-docker-images.git
      ```

   b. Wechseln Sie in das Bildverzeichnis. Verwenden Sie für dieses Beispiel den`aws/codebuild/standard:5.0`bild:

      ```
      $ cd aws-codebuild-docker-images/ubuntu/standard/5.0
      ```

   c. Erstellen Sie das Bild. Dies dauert einige Minuten.

      ```
      $ docker build -t aws/codebuild/standard:5.0 .
      ```

2. Herunterladen des CodeBuild-Agents

   Führen Sie den folgenden Befehl aus, um die x86_64-Version des Agenten herunterzuladen:

   ```
   $ docker pull public.ecr.aws/codebuild/local-builds:latest
   ```

   Führen Sie den folgenden Befehl aus, um die ARM-Version des Agenten herunterzuladen:

```
$ docker pull public.ecr.aws/codebuild/local-builds:aarch64
```

3. DerCodeBuildAgent ist verfügbar vonhttps://gallery.ecr.aws/codebuild/local-builds.

   Die Signatur des Secure Hash Algorithm (SHA) für die x86_64-Version des Agenten lautet:

   ```
   sha256:fac17c6d6c3cb500f6e7975887de1e41d29a9e70a86d6f49f76a2beacfcf967e
   ```

   Die SHA-Signatur für die ARM-Version des Agenten lautet:

   ```
   sha256:57a5dfda63be50edce13dea16dcd5e73e8d8559029658ba08b793c9a7adc68c7
   ```

   Sie können den SHA verwenden, um die Version des Agenten zu identifizieren. Um die SHA-Signatur des Agenten zu sehen, führen Sie den folgenden Befehl aus und suchen Sie nach dem SHA unter`RepoDigests`:

   ```
   $ docker inspect public.ecr.aws/codebuild/local-builds:latest
   ```

Führen Sie denCodeBuildAgentin

Um das auszuführenCodeBuildAgentin

1. Wechseln Sie in das Verzeichnis, das die Quelle Ihres Build-Projekts enthält.
2. Laden Sie das herundercodebuild_build.shskript:

   ```
   $ curl -O  https://raw.githubusercontent.com/aws/aws-codebuild-docker-images/
   master/local_builds/codebuild_build.sh
   $ chmod +x codebuild_build.sh
   ```

3. Führen Sie den`codebuild_build.sh`Skript und spezifizieren Sie Ihr Container-Image und das Ausgabeverzeichnis.

   Führen Sie den folgenden Befehl aus, um einen x86_64-Build auszuführen:

   ```
   $ ./codebuild_build.sh -i <container-image> -a <output directory>
   ```

   Führen Sie den folgenden Befehl aus, um einen ARM-Build auszuführen:

```
$ ./codebuild_build.sh -i <container-image> -a <output directory> -l
 public.ecr.aws/codebuild/local-builds:aarch64
```

Ersetzen*<container-image>*mit dem Namen des Container-Images, wie`aws/codebuild/standard:5.0`oder`public.ecr.aws/codebuild/amazonlinux2-x86_64-standard:4.0`.

Das Skript startet das Build-Image und führt den Build für das Projekt im aktuellen Verzeichnis aus. Um den Speicherort des Build-Projekts anzugeben, fügen Sie den`-s` *<build project directory>*Option zum Skriptbefehl.

Erhalten von Benachrichtigungen über neue CodeBuild-Agenten-Versionen

Sie können Amazon SNS-Benachrichtigungen abonnieren, sodass Sie benachrichtigt werden, wenn neue Versionen vonAWS CodeBuildAgenten werden freigelassen.

Um zu abonnierenCodeBuildAgentenbenachrichtigungen

1. Öffnen Sie die Amazon SNS-Konsole unter https://console.aws.amazon.com/sns/v3/home.
2. Ändern Sie in der Navigationsleiste, falls sie noch nicht ausgewählt ist, dieAWSRegion bisUSA Ost (Nord-Virginia). Sie müssen dies auswählenAWSRegion, weil die Amazon SNS-Benachrichtigungen, die Sie abonnieren, in dieser Region erstellt werden.
3. Wählen Sie im Navigationsbereich Subscriptions aus.
4. Wählen Sie Create subscription.
5. InAbonnement erstellen, gehen Sie wie folgt vor:

   a. Verwenden Sie für Topic ARN den folgenden Amazon-Ressourcennamen (ARN):

   ```
   arn:aws:sns:us-east-1:850632864840:AWS-CodeBuild-Local-Agent-Updates
   ```

   b. Wählen Sie für Protocol (Protokoll) die Option Email (E-Mail) oder SMS.

   c. Wählen Sie unter Endpoint (Endpunkt) aus, worüber (per E-Mail oder SMS) die Benachrichtigungen empfangen werden sollen. Geben Sie eine E-Mail-Adresse, Adresse oder Telefonnummer einschließlich Vorwahl ein.

   d. Wählen Sie Create subscription (Abonnement erstellen) aus.

e. WähleE-Mailum eine E-Mail zu erhalten, in der Sie aufgefordert werden, Ihr Abonnement zu bestätigen. Befolgen Sie die Anweisungen in der E-Mail zum Abschließen Ihres Abonnements.

   Wenn Sie diese Benachrichtigungen nicht mehr erhalten möchten, führen Sie die folgenden Schritte aus, um sich abzumelden.

So melden Sie sich von den CodeBuild-Agenten-Benachrichtigungen ab

1. Öffnen Sie die Amazon SNS-Konsole unter https://console.aws.amazon.com/sns/v3/home.
2. Wählen Sie im Navigationsbereich Subscriptions aus.
3. Wählen Sie das Abonnement und unter Actions (Aktionen) die Option Delete subscriptions (Abonnements löschen) aus. Wenn Sie aufgefordert werden, Ihre Entscheidung zu bestätigen, wählen Sie Delete aus.

Verwendung AWS CodeBuild mit Amazon Virtual Private Cloud

In der Regel AWS CodeBuild kann nicht auf Ressourcen in einer VPC zugegriffen werden. Um den Zugriff zu aktivieren, müssen Sie in Ihrer CodeBuild Projektkonfiguration zusätzliche VPC-spezifische Konfigurationsinformationen angeben. Dazu gehören die VPC-ID, die VPC-Subnetz-IDs und die VPC-Sicherheitsgruppen-IDs. VPC-fähige Builds können dann auf Ressourcen in Ihrer VPC zugreifen. Weitere Informationen zur Einrichtung einer VPC in Amazon VPC finden Sie im Amazon VPC-Benutzerhandbuch.

Themen

Anwendungsfälle

Die VPC-Konnektivität von AWS CodeBuild Builds ermöglicht:

- Führen Sie Integrationstests von Ihrem Build aus anhand von Daten in einer Amazon RDS-Datenbank durch, die in einem privaten Subnetz isoliert ist.
- Fragen Sie Daten in einem ElastiCache Amazon-Cluster direkt aus Tests ab.
- Interagieren Sie mit internen Webservices, die auf Amazon EC2, Amazon ECS gehostet werden, oder mit Services, die internes Elastic Load Balancing verwenden.
- Abhängigkeiten von selbst gehosteten, internen Artefakt-Repositorys, wie PyPI für Python, Maven für Java und npm für Node.js abzurufen,
- Greifen Sie auf Objekte in einem S3-Bucket zu, der so konfiguriert ist, dass der Zugriff nur über einen Amazon VPC-Endpunkt möglich ist.

- externe Webservices, die feste IP-Adressen benötigen, über die elastische IP-Adresse des NAT-Gateways oder der NAT-Instance abzufragen, die mit Ihrem Subnetz verknüpft ist.

Ihre Builds können auf jede Ressource zugreifen, die in Ihrer VPC gehostet wird.

Erlauben Sie Amazon VPC-Zugriff in Ihren Projekten CodeBuild

Nehmen Sie diese Einstellungen in Ihre VPC-Konfiguration auf:

- Wählen Sie für VPC-ID die VPC-ID aus, die verwendet wird. CodeBuild
- Wählen Sie für Subnetze ein privates Subnetz mit NAT-Übersetzung aus, das Routen zu den von verwendeten Ressourcen enthält oder Routen zu diesen enthält. CodeBuild
- Wählen Sie unter Sicherheitsgruppen die Sicherheitsgruppen aus, die CodeBuild den Zugriff auf Ressourcen in den VPCs ermöglichen.

Informationen über das Verwenden der Konsole zum Erstellen eines Build-Projekts finden Sie unter Erstellen Sie ein Build-Projekt (Konsole). Wenn Sie Ihr CodeBuild Projekt in VPC erstellen oder ändern, wählen Sie Ihre VPC-ID, Subnetze und Sicherheitsgruppen aus.

Informationen zur Verwendung von AWS CLI zum Erstellen eines Build-Projekts finden Sie unter. Erstellen eines Build-Projekts (AWS CLI) Wenn Sie with verwenden CodeBuild, muss der AWS CLI Servicerolle, die für CodeBuild die Interaktion mit Diensten im Namen des IAM-Benutzers verwendet wird, eine Richtlinie angehängt sein. Weitere Informationen finden Sie unter Erlauben Sie den CodeBuild Zugriff auf AWS Dienste, die zum Erstellen einer VPC-Netzwerkschnittstelle erforderlich sind.

Das VPCConfig-Objekt sollte Ihre vpcId und Subnetze securityGroupIdsenthalten.

- *vpcId*: Erforderlich. Die VPC-ID, die CodeBuild verwendet wird. Führen Sie diesen Befehl aus, um eine Liste aller Amazon VPC-IDs in Ihrer Region abzurufen:

```
aws ec2 describe-vpcs
```

- *subnets*: Erforderlich. Die Subnetz-IDs, die Ressourcen enthalten, die von verwendet werden. CodeBuild Führen Sie folgenden Befehl aus, um diese IDs zu erhalten:

```
aws ec2 describe-subnets --filters "Name=vpc-id,Values=<vpc-id>" --region us-east-1
```

> **Note**
>
> Ersetzen Sie `us-east-1` durch Ihre Region.

- *`securityGroupIds`*: Erforderlich. Die Sicherheitsgruppen-IDs, die von verwendet werden CodeBuild , um den Zugriff auf Ressourcen in den VPCs zu ermöglichen. Führen Sie diesen Befehl aus, um diese IDs zu erhalten:

```
aws ec2 describe-security-groups --filters "Name=vpc-id,Values=<vpc-id>" --region us-
east-1
```

> **Note**
>
> Ersetzen Sie `us-east-1` durch Ihre Region.

Bewährte Methoden für VPCs

Verwenden Sie diese Checkliste, wenn Sie eine VPC einrichten, mit der Sie arbeiten möchten. CodeBuild

- Richten Sie Ihre VPC mit öffentlichen und privaten Subnetzen und einem NAT-Gateway ein. Das NAT-Gateway muss sich in einem öffentlichen Subnetz befinden. Weitere Informationen finden Sie unter VPC mit öffentlichen und privaten Subnetzen (NAT) im Amazon VPC Benutzerhandbuch.

> **Important**
>
> Sie benötigen ein NAT-Gateway oder eine NAT-Instance zur Verwendung CodeBuild mit Ihrer VPC, sodass öffentliche Endpunkte erreicht werden CodeBuild können (z. B. um CLI-Befehle auszuführen, wenn Builds ausgeführt werden). Sie können das Internet-Gateway nicht anstelle eines NAT-Gateways oder einer NAT-Instance verwenden, da die Zuweisung von Elastic IP-Adressen zu den von ihm erstellten Netzwerkschnittstellen CodeBuild nicht unterstützt wird und Amazon EC2 die automatische Zuweisung einer öffentlichen IP-

Adresse für Netzwerkschnittstellen, die außerhalb von Amazon EC2 EC2-Instance-Starts erstellt wurden, nicht unterstützt.

- Binden Sie mehrere Availability Zones in Ihre VPC ein.
- Stellen Sie sicher, dass für Ihre Sicherheitsgruppen kein eingehender (eingehender) Datenverkehr zu Ihren Builds zugelassen ist. CodeBuild hat keine spezifischen Anforderungen für ausgehenden Datenverkehr, aber Sie müssen den Zugriff auf alle Internetressourcen zulassen, die für Ihren Build erforderlich sind, z. GitHub B. Amazon S3.

  Weitere Informationen finden Sie unter Sicherheitsgruppenregeln im Amazon VPC-Benutzerhandbuch.
- Richten Sie für Ihre Builds separate Subnetze ein.
- Wenn Sie Ihre CodeBuild Projekte für den Zugriff auf Ihre VPC einrichten, wählen Sie nur private Subnetze.

Weitere Informationen zur Einrichtung einer VPC in Amazon VPC finden Sie im Amazon VPC-Benutzerhandbuch.

Weitere Informationen AWS CloudFormation zur Konfiguration einer VPC für die Verwendung der CodeBuild VPC-Funktion finden Sie unter. AWS CloudFormation VPC-Vorlage

Fehlerbehebung für Ihre VPC-Einrichtung

Verwenden Sie die in der Fehlermeldung angegebenen Informationen, um Probleme zu identifizieren, zu diagnostizieren und zu beheben.

Im Folgenden finden Sie einige Richtlinien, die Sie bei der Behebung eines häufigen CodeBuild VPC-Fehlers unterstützen sollen:`Build does not have internet connectivity. Please check subnet network configuration`.

1. Stellen Sie sicher, dass Ihr Internet-Gateway an die VPC angeschlossen ist..
2. Stellen Sie sicher, dass die Routing-Tabelle für Ihr öffentliches Subnetz auf das Internet-Gateway verweist..
3. Stellen Sie sicher, dass Ihre Netzwerk-ACLs den Datenverkehr zulassen.
4. Stellen Sie sicher, dass Ihre Sicherheitsgruppen den Datenverkehr zulassen.
5. Fehlerbehebung für Ihr NAT-Gateway.

6. Vergewissern Sie sich, dass die Routing-Tabelle für private Subnetze auf das NAT-Gateway verweist..
7. Stellen Sie sicher, dass die Servicerolle, die CodeBuild für die Interaktion mit Diensten im Namen des IAM-Benutzers verwendet wird, über die in dieser Richtlinie festgelegten Berechtigungen verfügt. Weitere Informationen finden Sie unter Erstellen Sie eine CodeBuild Servicerolle.

   Wenn CodeBuild Berechtigungen fehlen, erhalten Sie möglicherweise die folgende Fehlermeldung:. `Unexpected EC2 error: UnauthorizedOperation` Dieser Fehler kann auftreten, wenn Sie CodeBuild nicht über die Amazon EC2 EC2-Berechtigungen verfügen, die für die Arbeit mit einer VPC erforderlich sind.

Einschränkungen von VPCs

- VPC-Konnektivität von CodeBuild wird für gemeinsam genutzte VPCs nicht unterstützt.
- VPC-Konnektivität von CodeBuild wird für Windows Server 2019 und Windows Server 2022 nicht unterstützt. Verwenden Sie Amazon Linux oder Ubuntu, um eine VPC mit einer On-Demand-Flotte zu verknüpfen.
- VPC-Konnektivität von CodeBuild wird für Flotten mit reservierter Kapazität unter Windows nicht unterstützt. Verwenden Sie Amazon Linux, um eine VPC einer Flotte mit reservierter Kapazität zuzuordnen.

Verwenden von VPC-Endpunkten

Sie können die Sicherheit von Builds erhöhen, indem Sie AWS CodeBuild so konfigurieren, dass ein VPC-Schnittstellenendpunkt verwendet wird. Schnittstellen-Endpunkte werden betrieben vonPrivateLink, eine Technologie, mit der Sie privat auf Amazon EC2 zugreifen können undCodeBuilddurch die Verwendung von privaten IP-Adressen. PrivateLinkschränkt den gesamten Netzwerkverkehr zwischen Ihren verwalteten Instanzen ein,CodeBuild, und Amazon EC2 zum Amazon-Netzwerk. (Verwaltete Instances haben keinen Zugriff auf das Internet.) Zudem benötigen Sie kein Internet-Gateway, kein NAT-Gerät und kein virtuelles privates Gateway. Es ist nicht erforderlich, PrivateLink zu konfigurieren, aber wir empfehlen dies. Für weitere Informationen überPrivateLinkund VPC-Endpunkte finden Sie unterWas istAWS PrivateLink?.

Vorbereitungen für das Erstellen von VPC-Endpunkten

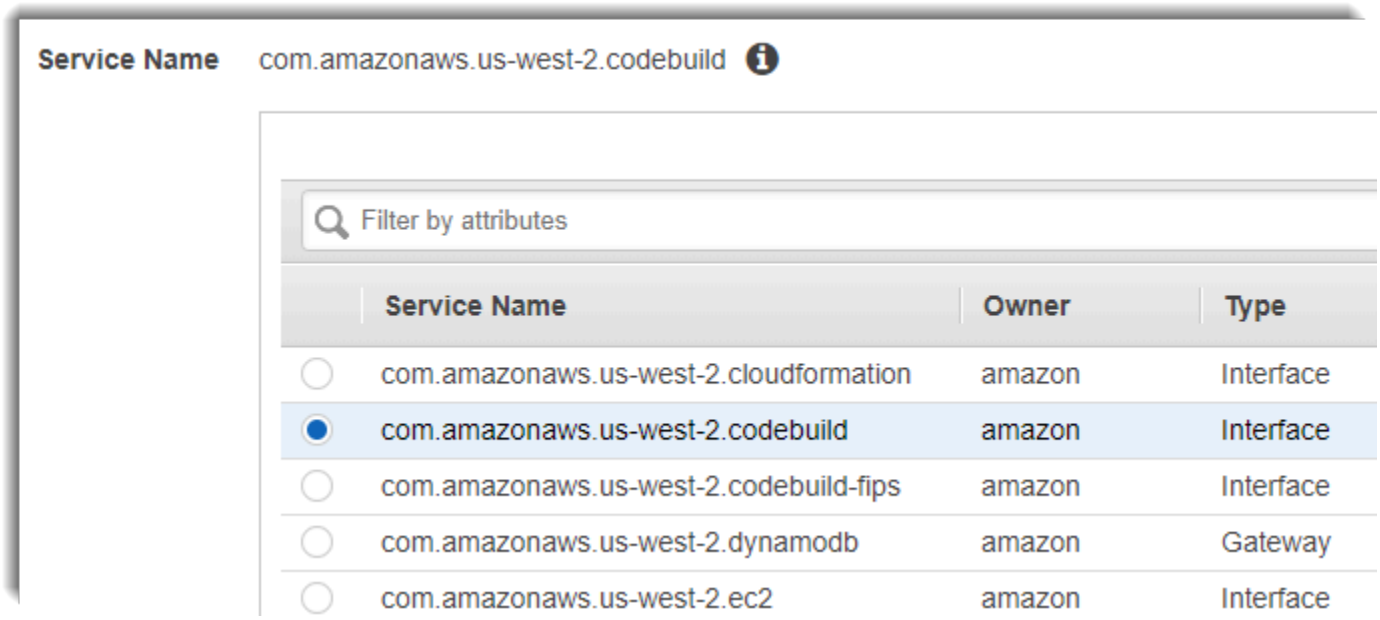
Seien Sie sich der folgenden Einschränkungen bewusst, bevor Sie VPC-Endpunkte für AWS CodeBuild konfigurieren.

> **Note**
>
> Benutze einNAT-Gatewaywenn du benutzen willstCodeBuildmitAWSDienste, die Amazon VPC nicht unterstützenPrivateLinkVerbindungen.

- VPC-Endpunkte unterstützen von Amazon bereitgestelltes DNS nur über Amazon Route 53. Wenn Sie Ihre eigene DNS verwenden möchten, können Sie die bedingte DNS-Weiterleitung nutzen. Weitere Informationen finden Sie unter DHCP-Optionssätze im Amazon-VPC-Benutzerhandbuch.
- VPC-Endpunkte unterstützen derzeit keine regionsübergreifenden Anforderungen. Stellen Sie sicher, dass Sie Ihren Endpunkt in derselben Datei erstellenAWSRegion wie alle S3-Buckets, die Ihre Build-Eingabe und -Ausgabe speichern. Sie können die Amazon S3-Konsole verwenden oderget-bucket-locationBefehl, um den Standort Ihres Buckets zu ermitteln. Verwenden Sie einen regionsspezifischen Amazon S3-Endpunkt, um auf Ihren Bucket zuzugreifen (z. B.`<bucket-name>.s3-us-west-2.amazonaws.com`). Weitere Informationen zu regionsspezifischen Endpunkten für Amazon S3 finden Sie unterAmazon Simple Storage Servicein derAllgemeine Amazon Web Services-Referenz. Wenn du das verwendestAWS CLIum Anfragen an Amazon S3 zu stellen, legen Sie als Standardregion dieselbe Region fest, in der Ihr Bucket erstellt wurde, oder verwenden Sie`--region`Parameter in Ihren Anfragen.

Erstellen von VPC-Endpunkten für CodeBuild

Folgen Sie den Anweisungen unter Schnittstellendpunkt erstellen, um den Endpunkt `com.amazonaws.region.codebuild` zu erstellen. Dies ist ein VPC-Endpunkt für AWS CodeBuild.



region repräsentiert die Regions-ID für eine von CodeBuild unterstützte AWS-Region, z. B. `us-east-2` für die Region USA Ost (Ohio). Für eine Liste der unterstütztenAWSRegionen finden Sie unterCodeBuildin der AWSAllgemeine Referenz. Der Endpunkt ist bereits mit der Region belegt, die Sie beim Anmelden bei AWS angegeben haben. Wenn Sie Ihre Region ändern, wird der VPC-Endpunkt entsprechend aktualisiert.

Erstellen einer VPC-Endpunktrichtlinie für CodeBuild

Sie können eine Richtlinie für Amazon VPC-Endpunkte erstellen fürAWS CodeBuildin dem Sie Folgendes angeben können:

- Prinzipal, der die Aktionen ausführen kann.
- Aktionen, die ausgeführt werden können
- Ressourcen, für die Aktionen ausgeführt werden können

Die folgende Beispielrichtlinie gibt an, dass alle Prinzipale ausschließlich Builds für das Projekt `project-name` starten und einsehen können.

```
{
    "Statement": [
        {
            "Action": [
                "codebuild:ListBuildsForProject",
                "codebuild:StartBuild",
```

```
                "codebuild:BatchGetBuilds"
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:codebuild:region-ID:account-ID:project/project-name",
            "Principal": "*"
        }
    ]
}
```

Weitere Informationen finden Sie unter Steuerung des Zugriffs auf Services mit VPC-Endpunkten im Amazon-VPC-Benutzerhandbuch.

AWS CloudFormation VPC-Vorlage

AWS CloudFormationermöglicht es Ihnen zu erstellen und bereitzustellenAWS-Infrastrukturen vorhersagbar und wiederholt, indem Sie Vorlagendateien einsetzen und eine Sammlung von Ressourcen gemeinsam als einzelne Einheit erstellen und löschen (astapeln) enthalten. Weitere Informationen finden Sie im AWS CloudFormation-Leitfaden.

Im Folgenden finden Sie eine AWS CloudFormation YAML-Vorlage zur Konfiguration einer VPC für die Verwendung von AWS CodeBuild. Diese Datei ist auch in verfügbar.samples.zipaus.

```
Description:  This template deploys a VPC, with a pair of public and private subnets
 spread
  across two Availability Zones. It deploys an internet gateway, with a default
  route on the public subnets. It deploys a pair of NAT gateways (one in each AZ),
  and default routes for them in the private subnets.

Parameters:
  EnvironmentName:
    Description: An environment name that is prefixed to resource names
    Type: String

  VpcCIDR:
    Description: Please enter the IP range (CIDR notation) for this VPC
    Type: String
    Default: 10.192.0.0/16

  PublicSubnet1CIDR:
    Description: Please enter the IP range (CIDR notation) for the public subnet in the
 first Availability Zone
    Type: String
```

```
    Default: 10.192.10.0/24

  PublicSubnet2CIDR:
    Description: Please enter the IP range (CIDR notation) for the public subnet in the
 second Availability Zone
    Type: String
    Default: 10.192.11.0/24

  PrivateSubnet1CIDR:
    Description: Please enter the IP range (CIDR notation) for the private subnet in
 the first Availability Zone
    Type: String
    Default: 10.192.20.0/24

  PrivateSubnet2CIDR:
    Description: Please enter the IP range (CIDR notation) for the private subnet in
 the second Availability Zone
    Type: String
    Default: 10.192.21.0/24

Resources:
  VPC:
    Type: AWS::EC2::VPC
    Properties:
      CidrBlock: !Ref VpcCIDR
      EnableDnsSupport: true
      EnableDnsHostnames: true
      Tags:
        - Key: Name
          Value: !Ref EnvironmentName

  InternetGateway:
    Type: AWS::EC2::InternetGateway
    Properties:
      Tags:
        - Key: Name
          Value: !Ref EnvironmentName

  InternetGatewayAttachment:
    Type: AWS::EC2::VPCGatewayAttachment
    Properties:
      InternetGatewayId: !Ref InternetGateway
      VpcId: !Ref VPC
```

```
  PublicSubnet1:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref VPC
      AvailabilityZone: !Select [ 0, !GetAZs '' ]
      CidrBlock: !Ref PublicSubnet1CIDR
      MapPublicIpOnLaunch: true
      Tags:
        - Key: Name
          Value: !Sub ${EnvironmentName} Public Subnet (AZ1)

  PublicSubnet2:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref VPC
      AvailabilityZone: !Select [ 1, !GetAZs  '' ]
      CidrBlock: !Ref PublicSubnet2CIDR
      MapPublicIpOnLaunch: true
      Tags:
        - Key: Name
          Value: !Sub ${EnvironmentName} Public Subnet (AZ2)

  PrivateSubnet1:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref VPC
      AvailabilityZone: !Select [ 0, !GetAZs  '' ]
      CidrBlock: !Ref PrivateSubnet1CIDR
      MapPublicIpOnLaunch: false
      Tags:
        - Key: Name
          Value: !Sub ${EnvironmentName} Private Subnet (AZ1)

  PrivateSubnet2:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref VPC
      AvailabilityZone: !Select [ 1, !GetAZs  '' ]
      CidrBlock: !Ref PrivateSubnet2CIDR
      MapPublicIpOnLaunch: false
      Tags:
        - Key: Name
          Value: !Sub ${EnvironmentName} Private Subnet (AZ2)
```

```
  NatGateway1EIP:
    Type: AWS::EC2::EIP
    DependsOn: InternetGatewayAttachment
    Properties:
      Domain: vpc

  NatGateway2EIP:
    Type: AWS::EC2::EIP
    DependsOn: InternetGatewayAttachment
    Properties:
      Domain: vpc

  NatGateway1:
    Type: AWS::EC2::NatGateway
    Properties:
      AllocationId: !GetAtt NatGateway1EIP.AllocationId
      SubnetId: !Ref PublicSubnet1

  NatGateway2:
    Type: AWS::EC2::NatGateway
    Properties:
      AllocationId: !GetAtt NatGateway2EIP.AllocationId
      SubnetId: !Ref PublicSubnet2

  PublicRouteTable:
    Type: AWS::EC2::RouteTable
    Properties:
      VpcId: !Ref VPC
      Tags:
        - Key: Name
          Value: !Sub ${EnvironmentName} Public Routes

  DefaultPublicRoute:
    Type: AWS::EC2::Route
    DependsOn: InternetGatewayAttachment
    Properties:
      RouteTableId: !Ref PublicRouteTable
      DestinationCidrBlock: 0.0.0.0/0
      GatewayId: !Ref InternetGateway

  PublicSubnet1RouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PublicRouteTable
```

```
      SubnetId: !Ref PublicSubnet1

  PublicSubnet2RouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PublicRouteTable
      SubnetId: !Ref PublicSubnet2


  PrivateRouteTable1:
    Type: AWS::EC2::RouteTable
    Properties:
      VpcId: !Ref VPC
      Tags:
        - Key: Name
          Value: !Sub ${EnvironmentName} Private Routes (AZ1)

  DefaultPrivateRoute1:
    Type: AWS::EC2::Route
    Properties:
      RouteTableId: !Ref PrivateRouteTable1
      DestinationCidrBlock: 0.0.0.0/0
      NatGatewayId: !Ref NatGateway1

  PrivateSubnet1RouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PrivateRouteTable1
      SubnetId: !Ref PrivateSubnet1

  PrivateRouteTable2:
    Type: AWS::EC2::RouteTable
    Properties:
      VpcId: !Ref VPC
      Tags:
        - Key: Name
          Value: !Sub ${EnvironmentName} Private Routes (AZ2)

  DefaultPrivateRoute2:
    Type: AWS::EC2::Route
    Properties:
      RouteTableId: !Ref PrivateRouteTable2
      DestinationCidrBlock: 0.0.0.0/0
      NatGatewayId: !Ref NatGateway2
```

```
  PrivateSubnet2RouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PrivateRouteTable2
      SubnetId: !Ref PrivateSubnet2

  NoIngressSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupName: "no-ingress-sg"
      GroupDescription: "Security group with no ingress rule"
      VpcId: !Ref VPC

Outputs:
  VPC:
    Description: A reference to the created VPC
    Value: !Ref VPC

  PublicSubnets:
    Description: A list of the public subnets
    Value: !Join [ ",", [ !Ref PublicSubnet1, !Ref PublicSubnet2 ]]

  PrivateSubnets:
    Description: A list of the private subnets
    Value: !Join [ ",", [ !Ref PrivateSubnet1, !Ref PrivateSubnet2 ]]

  PublicSubnet1:
    Description: A reference to the public subnet in the 1st Availability Zone
    Value: !Ref PublicSubnet1

  PublicSubnet2:
    Description: A reference to the public subnet in the 2nd Availability Zone
    Value: !Ref PublicSubnet2

  PrivateSubnet1:
    Description: A reference to the private subnet in the 1st Availability Zone
    Value: !Ref PrivateSubnet1

  PrivateSubnet2:
    Description: A reference to the private subnet in the 2nd Availability Zone
    Value: !Ref PrivateSubnet2

  NoIngressSecurityGroup:
```

```
    Description: Security group with no ingress rule
    Value: !Ref NoIngressSecurityGroup
```

Verwenden von AWS CodeBuild mit einem Proxy-Server

Sie können AWS CodeBuild mit einem Proxy-Server verwenden, um den HTTP- und HTTPS-Datenverkehr in und aus dem Internet zu regeln. Um CodeBuild mit einem Proxy-Server auszuführen, installieren Sie einen Proxy-Server in einem öffentlichen Subnetz und CodeBuild in einem privaten Subnetz in einer VPC.

Es gibt zwei primäre Anwendungsfälle für die Ausführung von CodeBuild in einem Proxy-Server:

- Damit entfällt die Nutzung eines NAT-Gateways oder einer NAT-Instance in Ihrer VPC.
- Sie können die URLs angeben, auf welche die Instances in dem Proxy-Server zugreifen können, und die URLs, für die der Proxy-Server den Zugriff verweigert.

Sie können CodeBuild mit zwei Arten von Proxy-Servern verwenden. In beiden Fällen wird der Proxy-Server in einem öffentlichen Subnetz und CodeBuild wird in einem privaten Subnetz ausgeführt.

- Expliziter Proxy: Wenn Sie einen expliziten Proxy-Server verwenden, müssen Sie konfigurierenNO_PROXY,HTTP_PROXY, undHTTPS_PROXYUmgebungsvariablen in CodeBuild auf Projektebene. Weitere Informationen finden Sie unter Ändern der Einstellungen eines Build-Projekts in AWS CodeBuild und Erstellen eines Build-Projekts in AWS CodeBuild.
- Transparenter Proxy: Wenn Sie einen transparenten Proxy-Server verwenden, ist keine spezielle Konfiguration erforderlich.

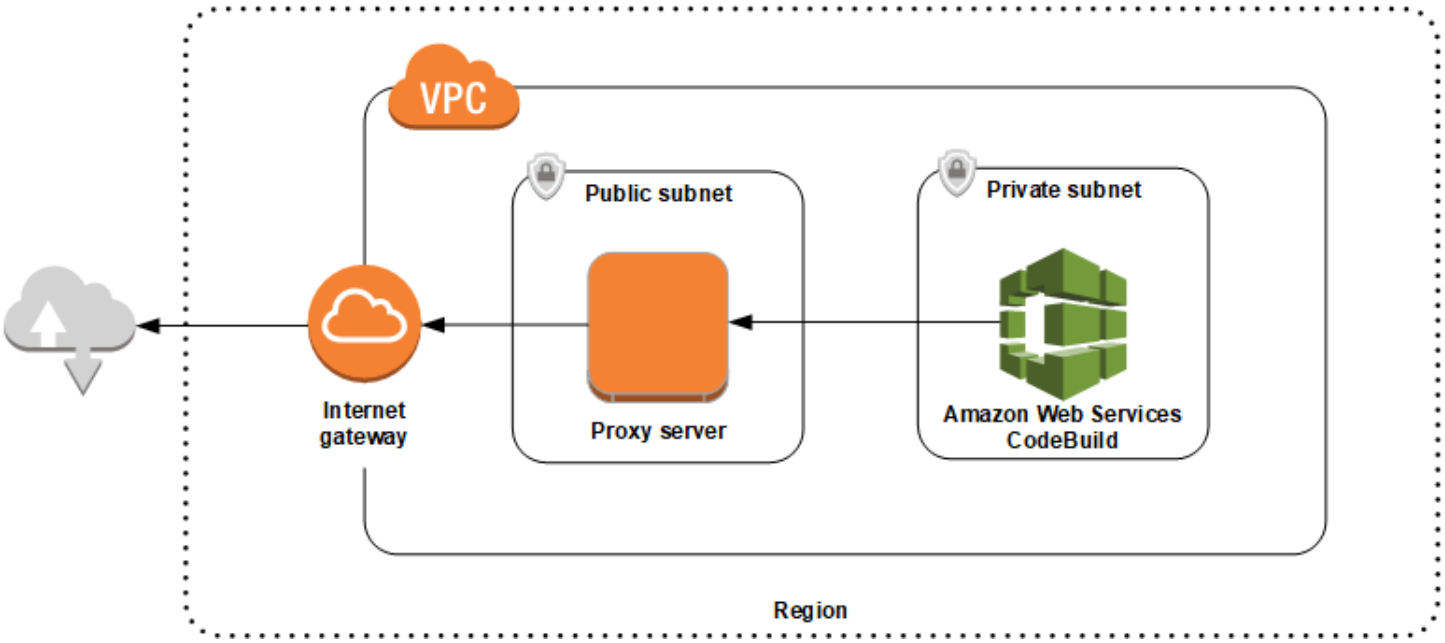
Themen

Erforderliche Komponenten zur Ausführung von CodeBuild in einem Proxy-Server

Für die Ausführung von AWS CodeBuild in einem transparenten oder expliziten Proxy-Server benötigen Sie diese Komponenten:

- Eine VPC.
- Ein öffentliches Subnetz in Ihrer VPC für den Proxy-Server
- Ein privates Subnetz in Ihrer VPC für CodeBuild.
- Ein Internet-Gateway, das die Kommunikation zwischen der VPC und dem Internet ermöglicht

Das folgende Diagramm zeigt, wie die Komponenten interagieren.



Einrichten eines VPC, von Subnetzen und eines Netzwerk-Gateways

Die folgenden Schritte sind für die Ausführung von AWS CodeBuild in einem transparenten oder expliziten Proxy-Server erforderlich.

1. Erstellen Sie eine VPC. Weitere Informationen finden Sie unter Erstellen einer VPC im Amazon-VPC-Benutzerhandbuch.
2. Erstellen Sie zwei Subnetze in Ihrer VPC. Eines ist ein öffentliches Subnetz mit dem Namen `Public Subnet`, in dem der Proxy-Server ausgeführt wird. Das andere ist ein privates Subnetz mit dem Namen`Private Subnet`in dem CodeBuild läuft.

Informationen hierzu finden Sie im Abschnitt Erstellen eines Subnetzes in Ihrer VPC.

3. Erstellen Sie ein Internet-Gateway und ordnen Sie es Ihrer VPC zu. Weitere Informationen finden Sie unter Erstellen und Anfügen eines Internet-Gateways.
4. Fügen Sie eine Regel zu der Standard-Routing-Tabelle hinzu, die ausgehenden Datenverkehr von der VPC (0.0.0.0/0) an das Internet-Gateway weiterleitet. Informationen hierzu finden Sie unter Hinzufügen und Entfernen von Routen zu und aus einer Routing-Tabelle.
5. Fügen Sie eine Regel zu der Standard-Sicherheitsgruppe Ihrer VPC hinzu, die eingehenden SSH-Datenverkehr (TCP 22) von Ihrer VPC (0.0.0.0/0) zulässt.
6. Folgen Sie den Anweisungen inStarten einer Instance mit dem Assistenten zum Starten von InstancesimBenutzerhandbuch für Amazon EC2um eine Amazon Linux-Instance zu starten. Wählen Sie bei Ausführung des Assistenten die folgenden Optionen aus:
   - In :Wählen eines Instance-TypsWählen Sie ein Amazon Linux-Amazon-Systemabbild (AMI) aus.
   - Wählen Sie unter Subnet (Subnetz) das öffentliche Subnetz aus, das Sie zuvor in diesem Thema erstellt haben. Wenn Sie den vorgeschlagenen Namen verwendet haben, heißt dieses Subnetz Public Subnet (Öffentliches Subnetz).
   - Klicken Sie in Auto-assign Public IP auf Enable.
   - Wählen Sie auf der Seite Configure Security Group (Sicherheitsgruppe konfigurieren) für Assign a security group (Sicherheitsgruppe zuweisen) die Option Select an existing security group (Vorhandene Sicherheitsgruppe auswählen) aus. Wählen Sie nun die Standard-Sicherheitsgruppe aus.
   - Nachdem Sie Launch (Starten) ausgewählt haben, wählen Sie ein vorhandenes Schlüsselpaar aus oder erstellen Sie ein neues.

   Wählen Sie für alle anderen Optionen die Standardeinstellungen aus.
7. Nachdem die EC2-Instance ausgeführt wird, deaktivieren Sie Quell-/Zielprüfungen. Weitere Informationen finden Sie unter Deaktivieren von Quell-/Zielprüfungen im Amazon VPC-Benutzerhandbuch.
8. Erstellen Sie eine Routing-Tabelle in Ihrer VPC. Fügen Sie eine Regel zu der Routing-Tabelle hinzu, die für das Internet bestimmten Datenverkehr zu Ihrem Proxy-Server leitet. Ordnen Sie diese Routing-Tabelle Ihrem privaten Subnetz zu. Dies ist erforderlich, damit ausgehende Anfragen von Instances in Ihrem privaten Subnetz, in dem CodeBuild ausgeführt wird, immer über den Proxy-Server geleitet werden.

Installieren und Konfigurieren eines Proxy-Servers

Es stehen viele Proxy-Server zur Auswahl. Um die Ausführung von AWS CodeBuild in einem Proxy-Server zu demonstrieren, wird hier ein Open-Source-Proxy-Server, Squid, verwendet. Dasselbe Konzept gilt auch für andere Proxy-Server.

Verwenden Sie für die Installation von Squid ein yum-Repository. Führen Sie dazu die folgenden Befehle aus:

```
sudo yum update -y
sudo yum install -y squid
```

Nachdem Sie Squid installiert haben, bearbeiten Sie die `squid.conf`-Datei mit den Anweisungen weiter unten in diesem Thema.

Konfigurieren von Squid für HTTPS-Datenverkehr

Bei HTTPS ist der HTTP-Datenverkehr in einer Transport Layer Security (TLS)-Verbindung gekapselt. Squid verwendet eine Funktion namens SslPeekAndSplice zum Abrufen der SNI (Server Name Indication, Servernamensanzeige) von der TLS-Initiierung, die den angeforderten Internet-Host enthält. Dies ist erforderlich, damit Squid den HTTPS-Datenverkehr nicht entschlüsseln muss. Zum Aktivieren von SslPeekAndSplice benötigt Squid ein Zertifikat. Erstellen Sie dieses Zertifikat mit OpenSSL:

```
sudo mkdir /etc/squid/ssl
cd /etc/squid/ssl
sudo openssl genrsa -out squid.key 2048
sudo openssl req -new -key squid.key -out squid.csr -subj "/C=XX/ST=XX/L=squid/O=squid/
CN=squid"
sudo openssl x509 -req -days 3650 -in squid.csr -signkey squid.key -out squid.crt
sudo cat squid.key squid.crt | sudo tee squid.pem
```

> **Note**
>
> Für HTTP muss Squid nicht konfiguriert werden. Von allen HTTP/1.1-Anforderungsnachrichten kann die Funktion das Host-Header-Feld abrufen, das den angeforderten Internet-Host angibt.

Führen Sie CodeBuild in einem expliziten Proxy-Server aus

Themen

Um AWS CodeBuild in einem expliziten Proxy-Server auszuführen, müssen Sie den Proxy-Server so konfigurieren, dass Datenverkehr zu und von externen Standorten zugelassen oder abgelehnt wird, und anschließend die Umgebungsvariablen `HTTPS_PROXY` und `HTTP_PROXY` konfigurieren.

Konfigurieren von Squid als expliziter Proxy-Server

Um Squid als expliziten Proxy-Server zu konfigurieren, müssen Sie die folgenden Änderungen an der Datei `/etc/squid/squid.conf` vornehmen:

- Entfernen Sie die folgenden Standard-ACL-Regeln (ACL = Access Control List, Zugriffskontrollliste).

```
acl localnet src 10.0.0.0/8
acl localnet src 172.16.0.0/12
acl localnet src 192.168.0.0/16
acl localnet src fc00::/7
acl localnet src fe80::/10
```

  Fügen Sie anstelle der von Ihnen entfernten Standard-ACL-Regeln Folgendes hinzu. Die erste Zeile lässt Anforderungen von Ihrer VPC zu. Die nächsten beiden Zeilen gewähren Ihrem Proxy-Server Zugriff auf die Ziel-URLs, die von AWS CodeBuildverwendet werden könnten. Ändern Sie den regulären Ausdruck in der letzten Zeile, um S3-Buckets oder ein CodeCommit--Repository in einerAWSRegion : z. B.:

  - Wenn Ihre Quelle Amazon S3 ist, gehen Sie nach dem Befehl voracl download_src dstdom_regex .*s3\.us-west-1\.amazonaws\.comum Zugriff auf S3-Buckets in der`us-west-1`Region :
  - Wenn Ihre QuelleAWS CodeCommitVerwendung von Verwendung von`git-codecommit.<`*`your-region`*`>.amazonaws.com`So fügen Sie eine hinzuAWSRegion einer Genehmigungsliste.

```
acl localnet src 10.1.0.0/16 #Only allow requests from within the VPC
acl allowed_sites dstdomain .github.com #Allows to download source from GitHub
acl allowed_sites dstdomain .bitbucket.com #Allows to download source from Bitbucket
acl download_src dstdom_regex .*\.amazonaws\.com #Allows to download source from
 Amazon S3 or CodeCommit
```

- Ersetzen Sie `http_access allow localnet` durch Folgendes:

```
http_access allow localnet allowed_sites
http_access allow localnet download_src
```

- Wenn Sie möchten, dass Ihr Build Protokolle und Artefakte hochlädt, führen Sie einen der folgenden Schritte aus:

  1. Fügen Sie vor der Anweisung `http_access deny all` die folgenden Anweisungen ein. Sie ermöglichen CodeBuild den Zugriff auf CloudWatch und Amazon S3. Der Zugriff auf CloudWatch ist erforderlich, damit CodeBuild CloudWatch-Protokolle erstellen kann. Zugriff auf Amazon S3 ist für das Hochladen von Artefakten und Amazon S3 S3-Caching erforderlich.

     - ```
       https_port 3130 cert=/etc/squid/ssl/squid.pem ssl-bump intercept
       acl SSL_port port 443
       http_access allow SSL_port
       acl allowed_https_sites ssl::server_name .amazonaws.com
       acl step1 at_step SslBump1
       acl step2 at_step SslBump2
       acl step3 at_step SslBump3
       ssl_bump peek step1 all
       ssl_bump peek step2 allowed_https_sites
       ssl_bump splice step3 allowed_https_sites
       ssl_bump terminate step2 all
       ```

     - Nach dem Speichern`squid.conf`Führen Sie den folgenden Befehl aus:

       ```
       sudo iptables -t nat -A PREROUTING -p tcp --dport 443 -j REDIRECT --to-port 3130
       sudo service squid restart
       ```

  2. Fügen Sie Ihrer Buildspec-Datei `proxy` hinzu. Weitere Informationen finden Sie unter Syntax der Build-Spezifikation.

     ```
     version: 0.2
     proxy:
       upload-artifacts: yes
     ```
     ```

```
  logs: yes
phases:
  build:
    commands:
      - command
```

> **Note**
>
> Wenn Sie einen RequestError-Timeout-Fehler erhalten, beachten Sie die Informationen im Abschnitt RequestError Timeout-Fehler beim Ausführen von CodeBuild auf einem Proxy-Server.

Weitere Informationen finden Sie unter Expliziter Proxy-Server – `squid.conf`-Beispieldatei an späterer Stelle in diesem Thema.

Erstellen eines CodeBuild-Projekts

Wenn Sie AWS CodeBuild mit Ihrem expliziten Proxy-Server ausführen möchten, legen Sie für die Umgebungsvariablen `HTTP_PROXY` und `HTTPS_PROXY` die private IP-Adresse der EC2-Instance fest, die Sie für Ihren Proxy-Server und Port 3128 auf Projektebene erstellt haben. Die private IP-Adresse sieht folgendermaßen aus: `http://`*`your-ec2-private-ip-address`*`:3128`. Weitere Informationen finden Sie unter Erstellen eines Build-Projekts in AWS CodeBuild und Ändern der Einstellungen eines Build-Projekts in AWS CodeBuild.

Verwenden Sie den folgenden Befehl, um das Zugriffsprotokoll des Squid-Proxys anzuzeigen:

```
sudo tail -f /var/log/squid/access.log
```

Expliziter Proxy-Server – **`squid.conf`**-Beispieldatei

Im Folgenden sehen Sie ein Beispiel für eine `squid.conf`-Datei, die für einen expliziten Proxy-Server konfiguriert wurde.

```
 acl localnet src 10.0.0.0/16 #Only allow requests from within the VPC
 # add all URLS to be whitelisted for download source and commands to be run in build
environment
 acl allowed_sites dstdomain .github.com    #Allows to download source from github
 acl allowed_sites dstdomain .bitbucket.com #Allows to download source from bitbucket
```

```
 acl allowed_sites dstdomain ppa.launchpad.net #Allows to run apt-get in build
environment
 acl download_src dstdom_regex .*\.amazonaws\.com #Allows to download source from S3
or CodeCommit
 acl SSL_ports port 443
 acl Safe_ports port 80  # http
 acl Safe_ports port 21  # ftp
 acl Safe_ports port 443  # https
 acl Safe_ports port 70  # gopher
 acl Safe_ports port 210  # wais
 acl Safe_ports port 1025-65535 # unregistered ports
 acl Safe_ports port 280  # http-mgmt
 acl Safe_ports port 488  # gss-http
 acl Safe_ports port 591  # filemaker
 acl Safe_ports port 777  # multiling http
 acl CONNECT method CONNECT
 #
 # Recommended minimum Access Permission configuration:
 #
 # Deny requests to certain unsafe ports
 http_access deny !Safe_ports
 # Deny CONNECT to other than secure SSL ports
 http_access deny CONNECT !SSL_ports
 # Only allow cachemgr access from localhost
 http_access allow localhost manager
 http_access deny manager
 # We strongly recommend the following be uncommented to protect innocent
 # web applications running on the proxy server who think the only
 # one who can access services on "localhost" is a local user
 #http_access deny to_localhost
 #
 # INSERT YOUR OWN RULE(S) HERE TO ALLOW ACCESS FROM YOUR CLIENTS
 #
 # Example rule allowing access from your local networks.
 # Adapt localnet in the ACL section to list your (internal) IP networks
 # from where browsing should be allowed
 http_access allow localnet allowed_sites
 http_access allow localnet download_src
 http_access allow localhost
 # Add this for CodeBuild to access CWL end point, caching and upload artifacts S3
bucket end point
 https_port 3130 cert=/etc/squid/ssl/squid.pem ssl-bump intercept
 acl SSL_port port 443
 http_access allow SSL_port
```

```
acl allowed_https_sites ssl::server_name .amazonaws.com
acl step1 at_step SslBump1
acl step2 at_step SslBump2
acl step3 at_step SslBump3
ssl_bump peek step1 all
ssl_bump peek step2 allowed_https_sites
ssl_bump splice step3 allowed_https_sites
ssl_bump terminate step2 all
# And finally deny all other access to this proxy
http_access deny all
# Squid normally listens to port 3128
http_port 3128
# Uncomment and adjust the following to add a disk cache directory.
#cache_dir ufs /var/spool/squid 100 16 256
# Leave coredumps in the first cache dir
coredump_dir /var/spool/squid
#
# Add any of your own refresh_pattern entries above these.
#
refresh_pattern ^ftp:  1440 20% 10080
refresh_pattern ^gopher: 1440 0% 1440
refresh_pattern -i (/cgi-bin/|\?) 0 0% 0
refresh_pattern .  0 20% 4320
```

Führen Sie CodeBuild in einem transparenten Proxy-Server aus

Um AWS CodeBuild in einem transparenten Proxy-Server auszuführen, müssen Sie den Proxy-Server mit Zugriff auf die Websites und Domänen konfigurieren, mit denen er interagiert.

Konfigurieren von Squid als transparenter Proxy-Server

Um einen transparenten Proxy-Server zu konfigurieren, müssen Sie ihm Zugriff auf die Domänen und Websites erteilen, auf die er zugreifen soll. Um AWS CodeBuild mit einem transparenten Proxy-Server auszuführen, müssen Sie Zugriff auf `amazonaws.com` erteilen. Sie müssen auch Zugriff auf andere Websites erteilen, die CodeBuild verwendet. Diese hängen davon ab, wie Sie Ihre CodeBuild--Projekte erstellen. Beispiele sind die Websites für Repositorys wie GitHub, Bitbucket, Yum und Maven. Um Squid Zugriff auf bestimmte Domains und Websites zu erteilen, aktualisieren Sie die `squid.conf`-Datei mit einem Befehl ähnlich dem folgenden: Dieser Beispielbefehl gewährt Zugriff auf `amazonaws.com`, `github.com` und `bitbucket.com`. Sie können dieses Beispiel bearbeiten, um Zugriff auf andere Websites zu erteilen.

```
cat | sudo tee /etc/squid/squid.conf #EOF
visible_hostname squid
#Handling HTTP requests
http_port 3129 intercept
acl allowed_http_sites dstdomain .amazonaws.com
#acl allowed_http_sites dstdomain domain_name [uncomment this line to add another
 domain]
http_access allow allowed_http_sites
#Handling HTTPS requests
https_port 3130 cert=/etc/squid/ssl/squid.pem ssl-bump intercept
acl SSL_port port 443
http_access allow SSL_port
acl allowed_https_sites ssl::server_name .amazonaws.com
acl allowed_https_sites ssl::server_name .github.com
acl allowed_https_sites ssl::server_name .bitbucket.com
#acl allowed_https_sites ssl::server_name [uncomment this line to add another website]
acl step1 at_step SslBump1
acl step2 at_step SslBump2
acl step3 at_step SslBump3
ssl_bump peek step1 all
ssl_bump peek step2 allowed_https_sites
ssl_bump splice step3 allowed_https_sites
ssl_bump terminate step2 all
http_access deny all
EOF
```

Eingehende Anforderungen von Instances im privaten Subnetz muss zu den Squid-Ports umgeleitet werden. Squid ist an Port 3129 für HTTP-Datenverkehr (anstelle von 80) und an Port 3130 für HTTPS-Datenverkehr (anstelle von 443) empfangsbereit. Verwenden Sie den Befehl iptables, um Datenverkehr weiterzuleiten:

```
sudo iptables -t nat -A PREROUTING -p tcp --dport 80 -j REDIRECT --to-port 3129
sudo iptables -t nat -A PREROUTING -p tcp --dport 443 -j REDIRECT --to-port 3130
sudo service iptables save
sudo service squid start
```

Erstellen eines CodeBuild-Projekts

Nach der Konfiguration können Sie Ihren Proxy-Server ohne weitere Konfiguration mit AWS CodeBuild in einem privaten Subnetz verwenden. Jede HTTP- und HTTPS-Anforderung durchläuft

den öffentlichen Proxy-Server. Verwenden Sie den folgenden Befehl, um das Zugriffsprotokoll des Squid-Proxys anzuzeigen:

```
sudo tail -f /var/log/squid/access.log
```

Ausführung eines Paket-Managers und anderer Tools in einem Proxy-Server

So führen Sie ein Tool wie z. B. einen Paket-Manager in einem Proxy-Server aus

1. Fügen Sie das Tool der Genehmigungsliste auf Ihrem Proxy-Server hinzu, indem Sie Ihrer `squid.conf`-Datei Anweisungen hinzufügen.
2. Fügen Sie eine Zeile zu Ihrer buildspec-Datei hinzu, die auf den privaten Endpunkt Ihres Proxy-Servers verweist.

Die folgenden Beispiele veranschaulichen dies für `apt-get`, `curl` und `maven`. Wenn Sie ein anderes Tool verwenden, gelten die gleichen Prinzipien. Fügen Sie es einer Zulassungsliste in der`squid.conf`Datei und fügen Sie Ihrer Build-Spezifikationsdatei (buildspec) einen Befehl hinzu, um CodeBuild auf den Endpunkt Ihres Proxy-Servers zu hinzuweisen.

Ausführen von **`apt-get`** in einem Proxy-Server

1. Fügen Sie der `squid.conf`-Datei die folgenden Anweisungen hinzu, um `apt-get` einer Genehmigungsliste in Ihrem Proxy-Server hinzuzufügen. Die ersten drei Zeilen erlauben`apt-get`um in der Build-Umgebung zu laufen.

   ```
   acl allowed_sites dstdomain ppa.launchpad.net # Required for apt-get to run in the
    build environment
   acl apt_get dstdom_regex .*\.launchpad.net # Required for CodeBuild to run apt-get
    in the build environment
   acl apt_get dstdom_regex .*\.ubuntu.com    # Required for CodeBuild to run apt-get
    in the build environment
   http_access allow localnet allowed_sites
   http_access allow localnet apt_get
   ```

2. Nehmen Sie die folgende Anweisung in Ihre buildspec Datei auf, damit `apt-get`-Befehle in `/etc/apt/apt.conf.d/00proxy` nach der Proxy-Konfiguration suchen.

```
echo 'Acquire::http::Proxy "http://<private-ip-of-proxy-server>:3128";
 Acquire::https::Proxy "http://<private-ip-of-proxy-server>:3128";
 Acquire::ftp::Proxy "http://<private-ip-of-proxy-server>:3128";' > /etc/apt/
apt.conf.d/00proxy
```

Ausführen von **`curl`** in einem Proxy-Server

1. Fügen Sie der Datei `squid.conf` Folgendes hinzu, um `curl` einer Genehmigungsliste in Ihrer Build-Umgebung hinzuzufügen.

   ```
   acl allowed_sites dstdomain ppa.launchpad.net # Required to run apt-get in the
    build environment
   acl allowed_sites dstdomain google.com # Required for access to a webiste. This
    example uses www.google.com.
   http_access allow localnet allowed_sites
   http_access allow localnet apt_get
   ```

2. Nehmen Sie die folgende Anweisung in Ihre buildspec-Datei auf, damit `curl` den privaten Proxy-Server für den Zugriff auf die Website verwendet, die Sie `squid.conf` hinzugefügt haben. In diesem Beispiel ist die Website `google.com`.

   ```
   curl -x <private-ip-of-proxy-server>:3128 https://www.google.com
   ```

Ausführen von **`maven`** in einem Proxy-Server

1. Fügen Sie der Datei `squid.conf` Folgendes hinzu, um `maven` einer Genehmigungsliste in Ihrer Build-Umgebung hinzuzufügen.

   ```
   acl allowed_sites dstdomain ppa.launchpad.net # Required to run apt-get in the
    build environment
   acl maven dstdom_regex .*\.maven.org # Allows access to the maven repository in the
    build environment
   http_access allow localnet allowed_sites
   http_access allow localnet maven
   ```

2. Fügen Sie ihrer buildspec-Datei die folgende Anweisung hinzu.

```
maven clean install -DproxySet=true -DproxyHost=<private-ip-of-proxy-server> -
DproxyPort=3128
```

Arbeiten mit Build-Projekten und Builds in AWS CodeBuild

Führen Sie zum Beginnen die unter beschriebenen Schritte ausErstellen eines Build-Projektsund führen Sie anschließend die Schritte in ausAusführen eines Buildaus. Weitere Informationen zu Build-Projekten und Builds finden Sie in den folgenden Themen.

Themen

- Arbeit mit Build-Projekten
- Arbeiten mit Builds in AWS CodeBuild

Arbeit mit Build-Projekten

Ein Build-Projekt enthält Informationen darüber, wie ein Build ausgeführt wird, einschließlich der Herkunft des Quellcodes, der zu verwendenden Build-Umgebung, der auszuführenden Build-Befehle und des Speicherorts der Build-Ausgabe.

Sie können diese Aufgaben bei der Arbeit mit Build-Projekten durchführen:

Themen

- Erstellen eines Build-Projekts in AWS CodeBuild
- Erstellen einer Benachrichtigungsregel
- Anzeigen einer Liste mit Build-Projektnamen in AWS CodeBuild
- Anzeigen der Details eines Build-Projekts in AWS CodeBuild
- Build-Caching in AWS CodeBuild
- Trigger einbauen AWS CodeBuild
- GitLab Verbindungen
- Webhooks verwenden mit AWS CodeBuild
- Ändern der Einstellungen eines Build-Projekts in AWS CodeBuild
- Löschen eines Build-Projekts in AWS CodeBuild
- Arbeiten mit freigegebenen Projekten
- Projekte taggen in AWS CodeBuild
- Batch integriert AWS CodeBuild
- GitHub Action-Runner in AWS CodeBuild

- Öffentliche Build-Projekte inAWS CodeBuild

Erstellen eines Build-Projekts in AWS CodeBuild

Sie können zum Erstellen eines Build-Projekts die AWS CodeBuild-Konsole, AWS CLI oder AWS-SDKs verwenden.

Voraussetzungen

Bevor Sie ein Build-Projekt erstellen, beantworten Sie die Fragen unterPlanen eines Buildsaus.

Themen

- Erstellen Sie ein Build-Projekt (Konsole)
- Erstellen eines Build-Projekts (AWS CLI)
- Erstellen eines Build-Projekts (AWS SDKs)
- Erstellen eines Build-Projekts (AWS CloudFormation)

Erstellen Sie ein Build-Projekt (Konsole)

Öffnen Sie die AWS CodeBuild Konsole unter https://console.aws.amazon.com/codesuite/codebuild/home.

Wenn eine CodeBuild Informationsseite angezeigt wird, wählen Sie Create build project aus. Erweitern Sie andernfalls im Navigationsbereich Build, wählen Sie Build projects und dann Create build project aus.

Wählen Sie Create build project (Build-Projekt erstellen) aus.

Füllen Sie die folgenden Abschnitte aus. Wenn Sie fertig sind, wählen Sie unten auf der Seite die Option Build-Projekt erstellen aus.

Abschnitte:

- Konfiguration des Projekts
- Quelle
- Umgebung
- Buildspec
- Batch-Konfiguration

- -Artefakte
- Logs (Protokolle)

Konfiguration des Projekts

Project name

Geben Sie einen Namen für dieses Build-Projekt ein. Die Namen der Build-Projekte müssen für jedes AWS Konto eindeutig sein.

Beschreibung

Geben Sie optional eine Beschreibung des Build-Projekts ein, damit andere Benutzer verstehen, wofür dieses Projekt verwendet wird.

Badge erstellen

(Optional) Wählen Sie Build-Badge aktivieren aus, um den Build-Status Ihres Projekts sichtbar und einbettbar zu machen. Weitere Informationen finden Sie unter Build Badges-Beispiel.

> **Note**
>
> Das Build-Badge gilt nicht, wenn Ihr Quellanbieter Amazon S3 ist.

Aktivieren Sie das Limit für gleichzeitige Builds

(Optional) Wenn Sie die Anzahl der gleichzeitigen Builds für dieses Projekt einschränken möchten, führen Sie die folgenden Schritte aus:

1. Wählen Sie „Anzahl gleichzeitiger Builds einschränken“, die mit diesem Projekt gestartet werden können.
2. Geben Sie im Feld Limit für gleichzeitige Builds die maximale Anzahl gleichzeitiger Builds ein, die für dieses Projekt zulässig sind. Dieses Limit darf nicht höher sein als das Limit für gleichzeitige Builds, das für das Konto festgelegt wurde. Wenn Sie versuchen, eine Zahl einzugeben, die über dem Kontolimit liegt, wird eine Fehlermeldung angezeigt.

Neue Builds werden nur gestartet, wenn die aktuelle Anzahl der Builds dieses Limit unterschreitet oder ihm entspricht. Wenn die aktuelle Build-Anzahl dieses Limit erreicht, werden neue Builds gedrosselt und nicht ausgeführt.

Zusätzliche Informationen

(Optional) Geben Sie für Tags den Namen und den Wert aller Tags ein, die von den unterstützenden AWS Diensten verwendet werden sollen. Verwenden Sie Add row, um ein Tag hinzuzufügen. Sie können bis zu 50 Tags hinzufügen.

Quelle

Quellanbieter

Wählen Sie den Typ des Quellcode-Anbieters. Verwenden Sie die folgenden Listen, um die für Ihren Quellanbieter geeignete Auswahl zu treffen:

> **Note**
>
> CodeBuild unterstützt Bitbucket Server nicht.

Amazon S3

Bucket

Wähle den Namen des Eingabe-Buckets, der den Quellcode enthält.

S3-Objektschlüssel oder S3-Ordner

Geben Sie den Namen der ZIP-Datei oder den Pfad zu dem Ordner ein, der den Quellcode enthält. Geben Sie einen Vorwärtsschrägstrich (/) ein, um den gesamten Inhalt im S3-Bucket herunterzuladen.

Quellversion

Geben Sie die Versions-ID des Objekts ein, das den Build Ihrer Eingabedatei darstellt. Weitere Informationen finden Sie unter Beispiel für eine Quellversion mit AWS CodeBuild.

CodeCommit

Repository

Wählen Sie das Repository aus, das Sie verwenden möchten.

Art der Referenz

Wählen Sie Branch, Git-Tag oder Commit ID, um die Version Ihres Quellcodes anzugeben. Weitere Informationen finden Sie unter Beispiel für eine Quellversion mit AWS CodeBuild.

> **Note**
>
> Wir empfehlen, Git-Branchnamen zu wählen, die nicht wie Commit-IDs aussehen, wie zum Beispiel `811dd1ba1aba14473856cee38308caed7190c0d` oder`5392f7`. Dies hilft dir dabei, Git-Checkout-Kollisionen mit tatsächlichen Commits zu vermeiden.

Tiefe des Git-Klonens

Wählen Sie aus, ob Sie einen flachen Klon erstellen möchten, dessen Verlauf auf die angegebene Anzahl von Commits gekürzt ist. Wenn Sie einen vollständigen Klon erstellen möchten, wählen Sie Full (Vollständig) aus.

Git-Submodule

Wählen Sie Use Git submodules (Git-Untermodule verwenden) aus, wenn Ihr Repository Git-Untermodule enthalten soll.

Bitbucket

Repository

Wähle Mit OAuth Connect oder Mit einem Bitbucket-App-Passwort verbinden und befolge die Anweisungen, um dich mit Bitbucket zu verbinden (oder erneut zu verbinden).

Wähle ein öffentliches Repository oder ein Repository in deinem Konto.

Quellversion

Geben Sie einen Zweig, eine Commit-ID, ein Tag oder eine Referenz und eine Commit-ID ein. Weitere Informationen finden Sie unter Beispiel für eine Quellversion mit AWS CodeBuild.

> **Note**
>
> Wir empfehlen, Git-Branchnamen zu wählen, die nicht wie Commit-IDs aussehen, wie zum Beispiel `811dd1ba1aba14473856cee38308caed7190c0d` oder`5392f7`. Dies hilft dir dabei, Git-Checkout-Kollisionen mit tatsächlichen Commits zu vermeiden.

Tiefe des Git-Klonens

Wählen Sie Git clone depth (Git-Klontiefe) aus, um einen flachen Klon mit einem Verlauf zu erstellen, der auf die angegebene Anzahl von Commits gekürzt ist. Wenn Sie einen vollständigen Klon erstellen möchten, wählen Sie Full (Vollständig) aus.

Git-Submodule

Wählen Sie Use Git submodules (Git-Untermodule verwenden) aus, wenn Ihr Repository Git-Untermodule enthalten soll.

Build-Status

Wählen Sie Build-Status beim Start und Ende Ihrer Builds an den Quellanbieter melden, wenn Sie möchten, dass der Status des Beginns und Abschlusses Ihres Builds Ihrem Quellanbieter gemeldet wird.

Um dem Quellanbieter den Buildstatus melden zu können, muss der mit dem Quellanbieter verknüpfte Benutzer Schreibzugriff auf das Repository haben. Wenn der Benutzer keinen Schreibzugriff hat, kann der Build-Status nicht aktualisiert werden. Weitere Informationen finden Sie unter Zugriff auf den Quellanbieter.

Geben Sie für den Statuskontext den Wert ein, der für den `name` Parameter im Bitbucket-Commit-Status verwendet werden soll. Weitere Informationen finden Sie unter Build in der Bitbucket-API-Dokumentation.

Geben Sie unter Ziel-URL den Wert ein, der für den `url` Parameter im Bitbucket-Commit-Status verwendet werden soll. Weitere Informationen finden Sie unter Build in der Bitbucket-API-Dokumentation.

Der Status eines durch einen Webhook ausgelösten Builds wird immer an den Quellanbieter gemeldet. Damit der Status eines Builds, der von der Konsole aus gestartet wird, oder eines API-Aufrufs an den Quellanbieter gemeldet wird, müssen Sie diese Einstellung auswählen.

Wenn die Builds Ihres Projekts durch einen Webhook ausgelöst werden, müssen Sie einen neuen Commit an das Repo übertragen, damit eine Änderung an dieser Einstellung wirksam wird.

Wählen Sie unter Primäre Quell-Webhook-Ereignisse die Option Jedes Mal neu erstellen, wenn eine Codeänderung in dieses Repository übertragen wird, wenn Sie den Quellcode jedes Mal erstellen CodeBuild möchten, wenn eine Codeänderung in dieses Repository übertragen wird. Weitere Hinweise zu Webhooks und Filtergruppen finden Sie unter. Bitbucket-Webhook-Ereignisse

GitHub

Repository

Wählen Sie Mit OAuth Connect oder Mit einem GitHub persönlichen Zugriffstoken verbinden und folgen Sie den Anweisungen, um eine Verbindung herzustellen (oder erneut herzustellen) GitHub und den Zugriff zu autorisieren. AWS CodeBuild

Wählen Sie ein öffentliches Repository oder ein Repository in Ihrem Konto.

Quellversion

Geben Sie einen Zweig, eine Commit-ID, ein Tag oder eine Referenz und eine Commit-ID ein. Weitere Informationen finden Sie unter Beispiel für eine Quellversion mit AWS CodeBuild.

> **Note**
>
> Wir empfehlen, Git-Branchnamen zu wählen, die nicht wie Commit-IDs aussehen, wie zum Beispiel `811dd1ba1aba14473856cee38308caed7190c0d` oder`5392f7`. Dies hilft dir dabei, Git-Checkout-Kollisionen mit tatsächlichen Commits zu vermeiden.

Tiefe des Git-Klonens

Wählen Sie Git clone depth (Git-Klontiefe) aus, um einen flachen Klon mit einem Verlauf zu erstellen, der auf die angegebene Anzahl von Commits gekürzt ist. Wenn Sie einen vollständigen Klon erstellen möchten, wählen Sie Full (Vollständig) aus.

Git-Submodule

Wählen Sie Use Git submodules (Git-Untermodule verwenden) aus, wenn Ihr Repository Git-Untermodule enthalten soll.

Build-Status

Wählen Sie Build-Status beim Start und Ende Ihrer Builds an den Quellanbieter melden, wenn Sie möchten, dass der Status des Beginns und Abschlusses Ihres Builds Ihrem Quellanbieter gemeldet wird.

Um dem Quellanbieter den Buildstatus melden zu können, muss der mit dem Quellanbieter verknüpfte Benutzer Schreibzugriff auf das Repository haben. Wenn der Benutzer keinen Schreibzugriff hat, kann der Build-Status nicht aktualisiert werden. Weitere Informationen finden Sie unter Zugriff auf den Quellanbieter.

Geben Sie für den Statuskontext den Wert ein, der für den `context` Parameter im GitHub Commit-Status verwendet werden soll. Weitere Informationen finden Sie im GitHub Entwicklerhandbuch unter Einen Commit-Status erstellen.

Geben Sie für Ziel-URL den Wert ein, der für den `target_url` Parameter im GitHub Commit-Status verwendet werden soll. Weitere Informationen finden Sie im GitHub Entwicklerhandbuch unter Einen Commit-Status erstellen.

Der Status eines durch einen Webhook ausgelösten Builds wird immer an den Quellanbieter gemeldet. Damit der Status eines Builds, der von der Konsole aus gestartet wird, oder eines API-Aufrufs an den Quellanbieter gemeldet wird, müssen Sie diese Einstellung auswählen.

Wenn die Builds Ihres Projekts durch einen Webhook ausgelöst werden, müssen Sie einen neuen Commit an das Repo übertragen, damit eine Änderung an dieser Einstellung wirksam wird.

Wählen Sie unter Primäre Quell-Webhook-Ereignisse die Option Jedes Mal neu erstellen, wenn eine Codeänderung in dieses Repository übertragen wird, wenn Sie den Quellcode jedes Mal erstellen CodeBuild möchten, wenn eine Codeänderung in dieses Repository übertragen wird. Weitere Hinweise zu Webhooks und Filtergruppen finden Sie unter. GitHub Webhook-Ereignisse

GitHub Enterprise Server

GitHub Persönliches Zugriffstoken für Unternehmen

Informationen darüber, wie Sie GitHub Beispiel für einen Enterprise Server ein persönliches Zugriffstoken in Ihre Zwischenablage kopieren, finden Sie unter. Fügen Sie das Token in das Textfeld ein und wählen Sie anschließend Save Token (Token speichern) aus.

> **Note**
>
> Sie müssen das private Zugriffstoken nur einmal eingeben und speichern. CodeBuild verwendet dieses Token in allen future Projekten.

Quellversion

Geben Sie eine Pull-Anfrage, einen Branch, eine Commit-ID, ein Tag oder eine Referenz und eine Commit-ID ein. Weitere Informationen finden Sie unter Beispiel für eine Quellversion mit AWS CodeBuild.

> **Note**
>
> Wir empfehlen, Git-Branchnamen zu wählen, die nicht wie Commit-IDs aussehen, wie zum Beispiel `811dd1ba1aba14473856cee38308caed7190c0d` oder`5392f7`. Dies hilft dir dabei, Git-Checkout-Kollisionen mit tatsächlichen Commits zu vermeiden.

Tiefe des Git-Klonens

Wählen Sie Git clone depth (Git-Klontiefe) aus, um einen flachen Klon mit einem Verlauf zu erstellen, der auf die angegebene Anzahl von Commits gekürzt ist. Wenn Sie einen vollständigen Klon erstellen möchten, wählen Sie Full (Vollständig) aus.

Git-Submodule

Wählen Sie Use Git submodules (Git-Untermodule verwenden) aus, wenn Ihr Repository Git-Untermodule enthalten soll.

Build-Status

Wählen Sie Build-Status beim Start und Ende Ihrer Builds an den Quellanbieter melden, wenn Sie möchten, dass der Status des Beginns und Abschlusses Ihres Builds Ihrem Quellanbieter gemeldet wird.

Um dem Quellanbieter den Buildstatus melden zu können, muss der mit dem Quellanbieter verknüpfte Benutzer Schreibzugriff auf das Repository haben. Wenn der Benutzer keinen Schreibzugriff hat, kann der Build-Status nicht aktualisiert werden. Weitere Informationen finden Sie unter Zugriff auf den Quellanbieter.

Geben Sie für den Statuskontext den Wert ein, der für den `context` Parameter im GitHub Commit-Status verwendet werden soll. Weitere Informationen finden Sie im GitHub Entwicklerhandbuch unter Einen Commit-Status erstellen.

Geben Sie für Ziel-URL den Wert ein, der für den `target_url` Parameter im GitHub Commit-Status verwendet werden soll. Weitere Informationen finden Sie im GitHub Entwicklerhandbuch unter Einen Commit-Status erstellen.

Der Status eines durch einen Webhook ausgelösten Builds wird immer an den Quellanbieter gemeldet. Damit der Status eines Builds, der von der Konsole aus gestartet wird, oder eines API-Aufrufs an den Quellanbieter gemeldet wird, müssen Sie diese Einstellung auswählen.

Wenn die Builds Ihres Projekts durch einen Webhook ausgelöst werden, müssen Sie einen neuen Commit an das Repo übertragen, damit eine Änderung an dieser Einstellung wirksam wird.

Unsicheres SSL

Wählen Sie Unsicheres SSL aktivieren, um SSL-Warnungen zu ignorieren, während Sie eine Verbindung zu Ihrem GitHub Enterprise-Projekt-Repository herstellen.

Wählen Sie unter Primäre Quell-Webhook-Ereignisse die Option Jedes Mal neu erstellen, wenn eine Codeänderung in dieses Repository übertragen wird, wenn Sie den Quellcode jedes Mal neu erstellen CodeBuild möchten, wenn eine Codeänderung in dieses Repository übertragen wird. Weitere Hinweise zu Webhooks und Filtergruppen finden Sie unter. GitHub Webhook-Ereignisse

GitLab

Connection (Verbindung)

Connect dein GitLab Konto über und verwende die Verbindung AWS CodeConnections, um dein Drittanbieter-Repository als Quelle für dein Build-Projekt zu verknüpfen.

Wählen Sie Standardverbindung oder Benutzerdefinierte Verbindung.

Standardverbindung wendet eine GitLab Standardverbindung für alle Projekte an. Benutzerdefinierte Verbindung wendet eine benutzerdefinierte GitLab Verbindung an, die die Standardeinstellungen Ihres Kontos überschreibt.

Standardverbindung

Der Name der Standardverbindung, die mit Ihrem Konto verknüpft ist.

Wenn Sie noch keine Verbindung zu Ihrem Anbieter hergestellt haben, finden Sie Stellen Sie eine Verbindung zu GitLab (Konsole) her eine Anleitung unter.

Benutzerdefinierte Verbindung

Wählen Sie den Namen der benutzerdefinierten Verbindung, die Sie verwenden möchten.

Wenn Sie noch keine Verbindung zu Ihrem Anbieter hergestellt haben, finden Sie Stellen Sie eine Verbindung zu GitLab (Konsole) her eine Anleitung unter.

Repository

Wählen Sie das Repository aus, das Sie verwenden möchten.

Quellversion

Geben Sie eine Pull-Request-ID, einen Branch, eine Commit-ID, ein Tag oder eine Referenz und eine Commit-ID ein. Weitere Informationen finden Sie unter Beispiel für eine Quellversion mit AWS CodeBuild.

> **Note**
>
> Wir empfehlen, Git-Branchnamen zu wählen, die nicht wie Commit-IDs aussehen, wie zum Beispiel `811dd1ba1aba14473856cee38308caed7190c0d` oder`5392f7`. Dies hilft dir dabei, Git-Checkout-Kollisionen mit tatsächlichen Commits zu vermeiden.

Tiefe des Git-Klonens

Wählen Sie Git clone depth (Git-Klontiefe) aus, um einen flachen Klon mit einem Verlauf zu erstellen, der auf die angegebene Anzahl von Commits gekürzt ist. Wenn Sie einen vollständigen Klon erstellen möchten, wählen Sie Full (Vollständig) aus.

Build-Status

Wählen Sie Build-Status beim Start und Ende Ihrer Builds an den Quellanbieter melden, wenn Sie möchten, dass der Status des Beginns und Abschlusses Ihres Builds Ihrem Quellanbieter gemeldet wird.

Um dem Quellanbieter den Buildstatus melden zu können, muss der mit dem Quellanbieter verknüpfte Benutzer Schreibzugriff auf das Repository haben. Wenn der Benutzer keinen Schreibzugriff hat, kann der Build-Status nicht aktualisiert werden. Weitere Informationen finden Sie unter Zugriff auf den Quellanbieter.

GitLab Self Managed

Connection (Verbindung)

Connect dein GitLab Konto über und verwende die Verbindung AWS CodeConnections, um dein Drittanbieter-Repository als Quelle für dein Build-Projekt zu verknüpfen.

Wählen Sie Standardverbindung oder Benutzerdefinierte Verbindung.

Standardverbindung wendet eine GitLab selbstverwaltete Standardverbindung für alle Projekte an. Benutzerdefinierte Verbindung wendet eine benutzerdefinierte GitLab selbstverwaltete Verbindung an, die die Standardeinstellungen Ihres Kontos überschreibt.

Standardverbindung

Der Name der Standardverbindung, die mit Ihrem Konto verknüpft ist.

Falls Sie noch keine Verbindung zu Ihrem Anbieter hergestellt haben, finden Sie im Benutzerhandbuch der Developer Tools Console unter Erstellen einer Verbindung zu GitLab Self-Managed eine Anleitung.

Benutzerdefinierte Verbindung

Wählen Sie den Namen der benutzerdefinierten Verbindung, die Sie verwenden möchten.

Wenn Sie noch keine Verbindung zu Ihrem Anbieter hergestellt haben, finden Sie im Benutzerhandbuch der Developer Tools Console unter Erstellen einer Verbindung zu GitLab Self-Managed eine Anleitung.

Repository

Wählen Sie das Repository aus, das Sie verwenden möchten.

Quellversion

Geben Sie eine Pull-Request-ID, einen Branch, eine Commit-ID, ein Tag oder eine Referenz und eine Commit-ID ein. Weitere Informationen finden Sie unter Beispiel für eine Quellversion mit AWS CodeBuild.

> **Note**
>
> Wir empfehlen, Git-Branchnamen zu wählen, die nicht wie Commit-IDs aussehen, wie zum Beispiel `811dd1ba1aba14473856cee38308caed7190c0d` oder`5392f7`. Dies hilft dir dabei, Git-Checkout-Kollisionen mit tatsächlichen Commits zu vermeiden.

Tiefe des Git-Klonens

Wählen Sie Git clone depth (Git-Klontiefe) aus, um einen flachen Klon mit einem Verlauf zu erstellen, der auf die angegebene Anzahl von Commits gekürzt ist. Wenn Sie einen vollständigen Klon erstellen möchten, wählen Sie Full (Vollständig) aus.

Build-Status

Wählen Sie Build-Status beim Start und Ende Ihrer Builds an den Quellanbieter melden, wenn Sie möchten, dass der Status des Beginns und Abschlusses Ihres Builds Ihrem Quellanbieter gemeldet wird.

Um dem Quellanbieter den Buildstatus melden zu können, muss der mit dem Quellanbieter verknüpfte Benutzer Schreibzugriff auf das Repository haben. Wenn der Benutzer keinen Schreibzugriff hat, kann der Build-Status nicht aktualisiert werden. Weitere Informationen finden Sie unter Zugriff auf den Quellanbieter.

Umgebung

Bereitstellungsmodell

Führen Sie eine der folgenden Aktionen aus:

- Um On-Demand-Flotten zu verwenden, die von verwaltet werden AWS CodeBuild, wählen Sie On-Demand. CodeBuild Bietet mit On-Demand-Flotten Rechenleistung für Ihre Builds. Die Maschinen werden zerstört, wenn der Bau abgeschlossen ist. On-Demand-Flotten werden vollständig verwaltet und verfügen über automatische Skalierungsfunktionen zur Bewältigung von Nachfragespitzen.
- Um Flotten mit reservierter Kapazität zu verwenden, die von verwaltet werden AWS CodeBuild, wählen Sie Reservierte Kapazität und dann einen Flottennamen aus. Bei Flotten mit reservierter Kapazität konfigurieren Sie eine Reihe von dedizierten Instances für Ihre Build-Umgebung. Diese Maschinen bleiben inaktiv und sind bereit, Builds oder Tests sofort zu verarbeiten, wodurch die Build-Dauer reduziert wird. Mit Flotten mit reservierter Kapazität sind Ihre Maschinen immer in Betrieb und es fallen weiterhin Kosten an, solange sie bereitgestellt werden.

Weitere Informationen finden Sie unter Arbeiten mit reservierter Kapazität in AWS CodeBuild.

Bild der Umgebung

Führen Sie eine der folgenden Aktionen aus:

- Um ein Docker-Image zu verwenden, das von verwaltet wird AWS CodeBuild, wählen Sie Verwaltetes Image aus und treffen Sie dann eine Auswahl unter Betriebssystem, Runtime (s), Image und Image-Version. Treffen Sie eine Auswahl unter Environment type (Umgebungstyp), sofern verfügbar.
- Wenn Sie ein anderes Docker-Image verwenden möchten, wählen Sie Custom image (Benutzerdefiniertes Image) aus. Wählen Sie als Umgebungstyp ARM, Linux, Linux GPU oder Windows aus. Wenn Sie Andere Registrierung wählen, geben Sie für Externe Registrierungs-URL den Namen und das Tag des Docker-Images in Docker Hub ein. Verwenden Sie dabei das Format. *`docker repository`*/*`docker image name`* Wenn Sie sich für Amazon ECR entscheiden, verwenden Sie das Amazon ECR-Repository und das Amazon ECR-Image, um das Docker-Image in Ihrem Konto auszuwählen. AWS
- Um ein privates Docker-Image zu verwenden, wählen Sie Benutzerdefiniertes Image. Wählen Sie als Umgebungstyp ARM, Linux, Linux GPU oder Windows aus. Wählen Sie unter Image registry (Abbildregistrierung) die Option Other registry (Andere Registrierung) aus und geben Sie dann den ARN der Anmeldeinformationen für Ihr privates Docker-Image ein. Die Anmeldeinformationen müssen von Secrets Manager erstellt werden. Weitere Informationen finden Sie unter Was ist AWS Secrets Manager? im AWS Secrets Manager - Benutzerhandbuch.

> **Note**
>
> CodeBuild überschreibt die `ENTRYPOINT` für benutzerdefinierte Docker-Images.

Datenverarbeitung

Führen Sie eine der folgenden Aktionen aus:

- Um EC2 Compute zu verwenden, wählen Sie EC2. EC2 Compute bietet optimierte Flexibilität bei Aktionsläufen.
- Um Lambda Compute zu verwenden, wählen Sie Lambda. Lambda Compute bietet optimierte Startgeschwindigkeiten für Ihre Builds. Lambda unterstützt schnellere Builds aufgrund einer geringeren Startlatenz. Lambda skaliert auch automatisch, sodass Builds nicht in der Warteschlange warten, bis sie ausgeführt werden. Weitere Informationen finden Sie unter Arbeiten mit AWS Lambda Compute in AWS CodeBuild.

Servicerolle

Führen Sie eine der folgenden Aktionen aus:

- Wenn Sie keine CodeBuild Servicerolle haben, wählen Sie Neue Servicerolle. Geben Sie im Feld Rollenname einen Namen für die neue Rolle ein.
- Wenn Sie eine CodeBuild Servicerolle haben, wählen Sie Bestehende Servicerolle aus. Wählen Sie unter Role ARN die Servicerolle aus.

> **Note**
>
> Wenn Sie die Konsole verwenden, um ein Build-Projekt zu erstellen, können Sie gleichzeitig eine CodeBuild Servicerolle erstellen. In der Standardeinstellung funktioniert diese Rolle ausschließlich mit diesem Projekt. Wenn Sie die Konsole verwenden, um die Servicerolle mit einem anderen Build-Projekt zu verknüpfen, wird die Rolle so aktualisiert, dass sie mit dem anderen Build-Projekt funktioniert. Eine Servicerolle kann in bis zu zehn Build-Projekten verwendet werden.

Zusätzliche Konfiguration

Timeout (Zeitüberschreitung)

Geben Sie einen Wert zwischen 5 Minuten und 36 Stunden an. Danach wird der Build CodeBuild gestoppt, falls er nicht abgeschlossen ist. Wenn Sie die Felder hours und minutes leer lassen, wird der Standardwert von 60 Minuten verwendet.

Privilegiert

(Optional) Wählen Sie Dieses Flag aktivieren aus, wenn Sie Docker-Images erstellen möchten oder wenn Sie möchten, dass Ihre Builds nur dann erweiterte Rechte erhalten, wenn Sie dieses Build-Projekt zum Erstellen von Docker-Images verwenden möchten. Andernfalls schlagen alle zugehörigen Builds fehl, die versuchen, mit dem Docker-Daemon zu interagieren. Sie müssen zudem den Docker-Daemon müssen, damit Ihre Builds interagieren können. Eine Möglichkeit, dies durchzuführen, besteht darin, den Docker-Daemon in der `install`-Phase Ihrer Build-Spezifikation zu initialisieren, indem Sie die folgenden Build-Befehle ausführen. Führen Sie diese Befehle nicht aus, wenn Sie ein Image für die Build-Umgebung ausgewählt haben, das von CodeBuild mit Docker-Unterstützung bereitgestellt wird.

> **Note**
>
> Standardmäßig ist der Docker-Daemon für Nicht-VPC-Builds aktiviert. Wenn Sie Docker-Container für VPC-Builds verwenden möchten, lesen Sie auf der Docker Docs-

Website unter Runtime Privilege and Linux Capabilities nach und aktivieren Sie den privilegierten Modus. Außerdem unterstützt Windows den privilegierten Modus nicht.

```
- nohup /usr/local/bin/dockerd --host=unix:///var/run/docker.sock --
host=tcp://127.0.0.1:2375 --storage-driver=overlay2 &
- timeout 15 sh -c "until docker info; do echo .; sleep 1; done"
```

VPC

Wenn Sie mit Ihrer VPC arbeiten möchten CodeBuild :

- Wählen Sie für VPC die VPC-ID aus, die CodeBuild verwendet wird.
- Wählen Sie für VPC-Subnetze die Subnetze aus, die Ressourcen enthalten, die verwendet werden. CodeBuild
- Wählen Sie für VPC-Sicherheitsgruppen die Sicherheitsgruppen aus, die CodeBuild den Zugriff auf Ressourcen in den VPCs ermöglichen.

Weitere Informationen finden Sie unter Verwendung AWS CodeBuild mit Amazon Virtual Private Cloud.

Datenverarbeitung

Wählen Sie eine der verfügbaren Optionen.

Umgebungsvariablen

Geben Sie den Namen und den Wert ein, und wählen Sie dann den Typ der einzelnen Umgebungsvariablen aus, die von Builds verwendet werden sollen.

> **Note**
>
> CodeBuild legt die Umgebungsvariable für Ihre AWS Region automatisch fest. Sie müssen die folgenden Umgebungsvariablen festlegen, wenn Sie sie nicht zu Ihrer buildspec.yml hinzugefügt haben:
>
> - AWS_ACCOUNT_ID
> - IMAGE_REPO_NAME
> - IMAGE_TAG

Konsole und AWS CLI Benutzer können Umgebungsvariablen sehen. Wenn Sie keine Bedenken hinsichtlich der Sichtbarkeit Ihrer Umgebungsvariablen haben, stellen Sie die Felder Name und Value ein und legen Sie dann den Type auf Plaintext fest.

Wir empfehlen, dass Sie eine Umgebungsvariable mit einem sensiblen Wert wie einer AWS Zugriffsschlüssel-ID, einem AWS geheimen Zugriffsschlüssel oder einem Passwort als Parameter in Amazon EC2 Systems Manager Parameter Store oder AWS Secrets Manager speichern.

Wenn Sie Amazon EC2 Systems Manager Parameter Store verwenden, wählen Sie als Typ die Option Parameter. Geben Sie unter Name einen Bezeichner ein, CodeBuild auf den verwiesen werden soll. Geben Sie für Value den Namen des Parameters ein, wie er im Amazon EC2 Systems Manager Parameter Store gespeichert ist. Verwenden Sie beispielsweise einen Parameter mit der Bezeichnung `/CodeBuild/dockerLoginPassword` und wählen Sie für Type (Typ) Parameter Store aus. Geben Sie unter Name `LOGIN_PASSWORD` ein. Geben Sie für Wert `/CodeBuild/dockerLoginPassword` ein.

> **Important**
>
> Wenn Sie Amazon EC2 Systems Manager Parameter Store verwenden, empfehlen wir, Parameter mit Parameternamen zu speichern, die mit `/CodeBuild/` (z. B. `/CodeBuild/dockerLoginPassword`) beginnen. Sie können die CodeBuild Konsole verwenden, um einen Parameter in Amazon EC2 Systems Manager zu erstellen. Wählen Sie Create a parameter (Parameter erstellen) aus und befolgen Sie dann die Anweisungen im Dialogfeld. (In diesem Dialogfeld können Sie für KMS-Schlüssel den ARN eines AWS KMS Schlüssels in Ihrem Konto angeben. Amazon EC2 Systems Manager verwendet diesen Schlüssel, um den Wert des Parameters beim Speichern zu verschlüsseln und beim Abrufen zu entschlüsseln.) Wenn Sie die CodeBuild Konsole verwenden, um einen Parameter zu erstellen, beginnt die Konsole den Parameternamen mit dem, `/CodeBuild/` wie er gespeichert wird. Weitere Informationen finden Sie unter Systems Manager Parameter Store und Systems Manager Parameter Store Console Walkthrough im Amazon EC2 Systems Manager Manager-Benutzerhandbuch.
>
> Wenn sich Ihr Build-Projekt auf Parameter Store von Amazon EC2 Systems Manager bezieht, muss die Service-Rolle des Build-Projekts die `ssm:GetParameters` Aktion zulassen. Wenn Sie zuvor Neue Servicerolle ausgewählt haben, wird CodeBuild diese Aktion in die Standard-Servicerolle für Ihr Build-Projekt aufgenommen. Wenn Sie

jedoch Existing service role (Vorhandene Servicerolle) ausgewählt haben, müssen Sie diese Aktion separat in Ihre Servicerolle aufnehmen.
Wenn sich Ihr Build-Projekt auf Parameter bezieht, die im Amazon EC2 Systems Manager Parameter Store mit Parameternamen gespeichert sind, die nicht mit beginnen`/CodeBuild/`, und Sie Neue Servicerolle wählen, müssen Sie diese Servicerolle aktualisieren, um Zugriff auf Parameternamen zu gewähren, die nicht mit `/CodeBuild/` beginnen. Dies liegt daran, dass diese Service-Rolle nur auf Parameternamen zugreift, die mit `/CodeBuild/` beginnen.
Wenn Sie Neue Servicerolle wählen, beinhaltet die Servicerolle die Berechtigung, alle Parameter unter dem `/CodeBuild/` Namespace im Amazon EC2 Systems Manager Parameter Store zu entschlüsseln.
Von Ihnen gesetzte Umgebungsvariablen ersetzen vorhandene Umgebungsvariablen. Wenn das Docker-Image beispielsweise bereits eine Umgebungsvariable mit dem Namen `MY_VAR` und einem Wert von `my_value` enthält und Sie eine Umgebungsvariable mit dem Namen `MY_VAR` und einem Wert von `other_value` festlegen, wird `my_value` durch `other_value` ersetzt. Wenn das Docker-Image demgegenüber bereits eine Umgebungsvariable mit dem Namen `PATH` und einem Wert von `/usr/local/sbin:/usr/local/bin` enthält und Sie eine Umgebungsvariable mit dem Namen `PATH` und einem Wert von `$PATH:/usr/share/ant/bin` festlegen, wird `/usr/local/sbin:/usr/local/bin` durch den Literalwert `$PATH:/usr/share/ant/bin` ersetzt.
Legen Sie keine Umgebungsvariable mit einem Namen fest, der mit `CODEBUILD_` beginnt. Dieses Präfix ist zur -internen Verwendung reserviert.
Wenn eine Umgebungsvariable mit identischem Namen an mehreren Orten definiert ist, wird der Wert folgendermaßen bestimmt:

- Der Wert im Aufruf zum Starten des Build-Vorgangs hat den höchsten Vorrang.
- Der Wert in der Build-Projektdefinition folgt darauf.
- Der Wert in der buildspec-Deklaration hat die niedrigste Priorität.

Wenn Sie Secrets Manager verwenden, wählen Sie für Type Secrets Manager. Geben Sie unter Name einen Bezeichner ein, auf CodeBuild den verwiesen werden soll. Geben Sie unter Wert einen `reference-key` mit dem Muster *`secret-id`*`:`*`json-key`*`:`*`version-stage`*`:`*`version-id`* ein. Weitere Informationen finden Sie unter Secrets Manager reference-key in the buildspec file.

> **Important**
>
> Wenn Sie Secrets Manager verwenden, empfehlen wir, Secrets mit Namen zu speichern, die mit `/CodeBuild/` (z. B.`/CodeBuild/dockerLoginPassword`) beginnen. Weitere Informationen finden Sie unter Was ist AWS Secrets Manager? im AWS Secrets Manager -Benutzerhandbuch.
> Wenn sich Ihr Build-Projekt auf Geheimnisse bezieht, die in Secrets Manager gespeichert sind, muss die Service-Rolle des Build-Projekts die `secretsmanager:GetSecretValue` Aktion zulassen. Wenn Sie zuvor Neue Servicerolle ausgewählt haben, wird CodeBuild diese Aktion in die Standard-Servicerolle für Ihr Build-Projekt aufgenommen. Wenn Sie jedoch Existing service role (Vorhandene Servicerolle) ausgewählt haben, müssen Sie diese Aktion separat in Ihre Servicerolle aufnehmen.
> Wenn sich Ihr Build-Projekt auf Geheimnisse bezieht, die in Secrets Manager mit geheimen Namen gespeichert sind, die nicht mit beginnen`/CodeBuild/`, und Sie Neue Dienstrolle ausgewählt haben, müssen Sie die Servicerolle aktualisieren, um Zugriff auf geheime Namen zu ermöglichen, die nicht mit beginnen`/CodeBuild/`. Dies liegt daran, dass die Servicerolle nur den Zugriff auf geheime Namen ermöglicht, die mit beginnen`/CodeBuild/`.
> Wenn Sie Neue Dienstrolle wählen, beinhaltet die Dienstrolle die Berechtigung, alle Geheimnisse unter dem `/CodeBuild/` Namespace im Secrets Manager zu entschlüsseln.

Buildspec

Spezifikationen erstellen

Führen Sie eine der folgenden Aktionen aus:

- Wenn Ihr Quellcode eine buildspec-Datei enthält, wählen Sie Use a buildspec file (Eine buildspec-Datei verwenden) aus. Sucht standardmäßig nach einer Datei mit dem Namen `buildspec.yml` im Quellcode-Stammverzeichnis. CodeBuild Wenn Ihre Buildspec-Datei einen anderen Namen oder Speicherort verwendet, geben Sie ihren Pfad vom Quellstammverzeichnis aus im Feld Buildspec-Name ein (zum Beispiel oder. `buildspec-two.yml configuration/buildspec.yml` Wenn sich die Buildspec-Datei in einem S3-Bucket befindet, muss sie sich in derselben Region wie Ihr Build-Projekt befinden. AWS Geben Sie die Buildspec-Datei mit ihrem ARN an (z. B.). `arn:aws:s3:::<my-codebuild-sample2>/buildspec.yml`

- Wenn der Quellcode keine Build-Spezifikationsdatei enthält oder Sie andere Build-Befehle ausführen möchten, als für die `build`-Phase in der `buildspec.yml`-Datei im Stammverzeichnis des Quellcodes angegeben wurden, wählen Sie Insert build commands (Build-Befehle einfügen) aus. Geben Sie für Build commands (Build-Befehle) die Befehle ein, die in der `build`-Phase ausgeführt werden sollen. Bei mehreren Befehlen unterteilen Sie die einzelnen Befehle mit &&, (wie z. B. `mvn test && mvn package`). Um Befehle in anderen Phasen auszuführen, oder wenn Sie eine lange Liste von Befehlen für die `build` Phase haben, fügen Sie dem Quellcode-Stammverzeichnis eine `buildspec.yml` Datei hinzu, fügen Sie die Befehle zur Datei hinzu und wählen Sie dann Use the buildspec.yml im Quellcode-Stammverzeichnis.

Weitere Informationen hierzu finden Sie unter Build-Spezifikationsreferenz.

Batch-Konfiguration

Sie können eine Gruppe von Builds als einen einzigen Vorgang ausführen. Weitere Informationen finden Sie unter Batch integriert AWS CodeBuild.

Definieren Sie die Batch-Konfiguration

Wählen Sie diese Option, um Batch-Builds in diesem Projekt zuzulassen.

Batch-Servicerolle

Stellt die Servicerolle für Batch-Builds bereit.

Wählen Sie eine der folgenden Optionen aus:

- Wenn Sie keine Batch-Servicerolle haben, wählen Sie Neue Servicerolle aus. Geben Sie im Feld Servicerolle einen Namen für die neue Rolle ein.
- Wenn Sie eine Batch-Servicerolle haben, wählen Sie Bestehende Servicerolle aus. Wählen Sie unter Servicerolle die Servicerolle aus.

Batch-Builds führen eine neue Sicherheitsrolle in der Batch-Konfiguration ein. Diese neue Rolle ist erforderlich, da sie in der Lage sein CodeBuild muss`StartBuild`, die `RetryBuild` Aktionen`StopBuild`, und in Ihrem Namen aufzurufen, um Builds als Teil eines Batches auszuführen. Kunden sollten aus zwei Gründen eine neue Rolle und nicht dieselbe Rolle verwenden, die sie in ihrem Build verwenden:

- Die Zuweisung der Build-Rolle `StartBuild` und der `RetryBuild` Berechtigungen würde es einem einzelnen Build ermöglichen, mehrere Builds über die Buildspec zu starten. `StopBuild`

- CodeBuild Batch-Builds bieten Einschränkungen, die die Anzahl der Builds und Berechnungstypen einschränken, die für die Builds im Batch verwendet werden können. Wenn die Build-Rolle über diese Berechtigungen verfügt, ist es möglich, dass die Builds selbst diese Einschränkungen umgehen.

Zulässige Berechnungstypen für Batch

Wählen Sie die für den Stapel zulässigen Berechnungstypen aus. Wählen Sie alle zutreffenden Antworten aus.

Maximal zulässige Anzahl von Builds pro Batch

Geben Sie die maximale Anzahl von Builds ein, die im Stapel zulässig sind. Wenn ein Stapel diesen Grenzwert überschreitet, schlägt der Batch fehl.

Batch-Timeout

Geben Sie die maximale Zeit für den Abschluss des Batch-Builds ein.

Kombinieren Sie Artefakte

Wählen Sie Alle Artefakte aus dem Stapel an einem einzigen Ort kombinieren aus, um alle Artefakte aus dem Stapel an einem einzigen Ort zusammenzufassen.

Batch-Berichtsmodus

Wählen Sie den gewünschten Modus für den Build-Statusbericht für Batch-Builds aus.

> **Note**
>
> Dieses Feld ist nur verfügbar, GitHub wenn die Projektquelle Bitbucket oder GitHub Enterprise ist und unter Quelle die Option Buildstatus an den Quellanbieter melden, wenn deine Builds beginnen und enden, ausgewählt ist.

Aggregierte Builds

Wählen Sie diese Option aus, um die Status aller Builds im Batch in einem einzigen Statusbericht zusammenzufassen.

Einzelne Builds

Wählen Sie diese Option aus, damit der Build-Status für alle Builds im Batch separat gemeldet wird.

-Artefakte

Typ

Führen Sie eine der folgenden Aktionen aus:

- Wenn keine Build-Ausgabeartefakte erstellt werden sollen, klicken Sie auf die Option No artifacts. Möglicherweise möchten Sie dies tun, wenn Sie nur Build-Tests ausführen oder ein Docker-Image in ein Amazon ECR-Repository übertragen möchten.
- Um die Build-Ausgabe in einem S3-Bucket zu speichern, wählen Sie Amazon S3 und gehen Sie dann wie folgt vor:
  - Lassen Sie Name leer, wenn Sie den Projektnamen für die ZIP-Datei mit der Build-Ausgabe verwenden möchten. Geben Sie andernfalls den Namen ein. (Wenn eine ZIP-Datei mit einer Dateierweiterung ausgegeben werden soll, vergewissern Sie sich, dass Sie die Dateierweiterung an den Namen der ZIP-Datei anfügen.)
  - Wählen Sie Enable semantitic versioning (Semantisches Versioning aktivieren) aus, wenn Sie möchten, dass ein Name in der buildspec-Datei jeden beliebigen in der Konsole angegebenen Namen überschreibt. Der Name in einer buildspec-Datei wird zur Erstellungszeit berechnet und verwendet die Shell-Befehlssprache. Beispielsweise können Sie dem Namen Ihres Artefakts ein Datum und eine Uhrzeit anhängen, damit dieser stets eindeutig ist. Eindeutige Artefakt-Namen verhindern, dass Artefakte überschrieben werden. Weitere Informationen finden Sie unter Syntax der Build-Spezifikation.
  - Wählen Sie für Bucket name den Namen des Ausgabe-Buckets aus.
  - Wenn Sie in diesem Vorgang zuvor die Option Insert build commands (Build-Befehle eingeben) verwendet haben, geben Sie für Output files (Ausgabedateien) die Speicherorte der Build-Dateien ein, die in der ZIP-Datei oder dem Ordner für die Build-Ausgabe enthalten sein sollen. Bei mehreren Speicherorten trennen Sie die einzelnen Speicherorte durch ein Komma, (wie z. B. `appspec.yml, target/my-app.jar`). Weitere Informationen finden Sie in der Beschreibung von `files` in Syntax der Build-Spezifikation.
  - Wenn Sie nicht wollen, dass Ihre Build-Artefakte verschlüsselt werden, wählen Sie Remove artifacts encryption (Verschlüsselung von Artefakten entfernen) aus.

Für jede Gruppe sekundärer Artefakte:

1. Geben Sie für Artifact identifier (Artefakt-ID) einen Wert mit weniger als 128 Zeichen ein, der nur alphanumerische Zeichen und Unterstriche enthält.
2. Wählen Sie Add artifact (Artefakt hinzufügen) aus.

3. Führen Sie die vorherigen Schritte aus, um die sekundären Artefakte zu konfigurieren.
4. Wählen Sie Save artifact (Artefakt speichern) aus.

Zusätzliche Konfiguration

Verschlüsselungsschlüssel

(Optional) Führen Sie eine der folgenden Optionen aus:

- Um das Von AWS verwalteter Schlüssel für Amazon S3 in Ihrem Konto zur Verschlüsselung der Build-Ausgabeartefakte zu verwenden, lassen Sie den Verschlüsselungsschlüssel leer. Dies ist die Standardeinstellung.
- Um einen vom Kunden verwalteten Schlüssel zum Verschlüsseln der Build-Ausgabeartefakte zu verwenden, geben Sie im Feld Verschlüsselungsschlüssel den ARN des KMS-Schlüssels ein. Verwenden Sie dabei das Format `arn:aws:kms:`*`region-ID`*`:`*`account-ID`*`:key/`*`key-ID`*.

Cache-Typ

Wählen Sie für Cache type (Cache-Typ) eine der folgenden Optionen aus:

- Wenn Sie keinen Cache verwenden möchten, wählen Sie No cache.
- Wenn Sie einen Amazon S3-Cache verwenden möchten, wählen Sie Amazon S3 und gehen Sie dann wie folgt vor:
  - Wählen Sie für Bucket den Namen des S3-Buckets, in dem der Cache gespeichert wird.
  - (Optional) Geben Sie für das Cache-Pfadpräfix ein Amazon S3 S3-Pfadpräfix ein. Der Wert für Cache path prefix (Cache-Pfadpräfix) ist mit einem Verzeichnisnamen vergleichbar. Er ermöglicht Ihnen das Speichern des Cache in demselben Verzeichnis eines Buckets.

    > **Important**
    >
    > Fügen Sie am Ende des Pfadpräfix keinen abschließenden Schrägstrich (/) an.

- Wenn Sie einen lokalen Cache verwenden möchten, wählen Sie Local (Lokal) und dann mindestens einen lokalen Cache-Modus aus.

> **Note**
>
> Der Modus Docker layer cache (Docker-Ebenen-Cache) ist nur für Linux verfügbar. Wenn Sie diesen Modus auswählen, muss Ihr Projekt im privilegierten Modus ausgeführt werden.

Durch die Verwendung eines Caches wird eine erhebliche Ersparnis bei der Erstellungszeit erzielt, da wiederverwendbare Teile der Build-Umgebung im Cache gespeichert und über Builds hinweg verwendet werden. Weitere Informationen über die Angabe eines Cache in der Build-Spezifikationsdatei finden Sie unter Syntax der Build-Spezifikation. Weitere Informationen zum Caching finden Sie unter Build-Caching in AWS CodeBuild.

Logs (Protokolle)

Wählen Sie die Protokolle aus, die Sie erstellen möchten. Sie können Amazon CloudWatch Logs, Amazon S3 S3-Protokolle oder beides erstellen.

CloudWatch

Wenn Sie Amazon CloudWatch Logs-Protokolle wünschen:

CloudWatch Logs

Wählen Sie CloudWatch logs (CW-Protokolle).

Group name (Gruppenname)

Geben Sie den Namen Ihrer Amazon CloudWatch Logs-Protokollgruppe ein.

Name des Streams

Geben Sie den Namen Ihres Amazon CloudWatch Logs-Log-Streams ein.

S3

Wenn Sie Amazon S3 S3-Protokolle wünschen:

S3-Protokolle

Wählen Sie S3 logs (S3-Protokolle).

Bucket

Wählen Sie den Namen des S3-Buckets für Ihre Logs.

Pfadpräfix

Geben Sie das Präfix für Ihre Logs ein.

Deaktivieren Sie die S3-Protokollverschlüsselung

Wählen Sie diese Option aus, wenn Sie nicht möchten, dass Ihre S3-Protokolle verschlüsselt werden.

Erstellen eines Build-Projekts (AWS CLI)

Weitere Informationen zur Verwendung von AWS CLI with CodeBuild finden Sie unterBefehlszeilenreferenz.

Um ein CodeBuild Build-Projekt mit dem zu erstellen AWS CLI, erstellen Sie eine Projektstruktur im JSON-Format, füllen die Struktur aus und rufen den `create-project`Befehl zum Erstellen des Projekts auf.

Erstellen Sie die JSON-Datei

Erstellen Sie eine JSON-Skelettdatei mit dem `create-project`Befehl und verwenden Sie die `--generate-cli-skeleton` folgende Option:

```
aws codebuild create-project --generate-cli-skeleton > <json-file>
```

Dadurch wird eine JSON-Datei mit dem von angegebenen Pfad und Dateinamen erstellt<json-file>.

Füllen Sie die JSON-Datei aus

Ändern Sie die JSON-Daten wie folgt und speichern Sie Ihre Ergebnisse.

```
{
  "name": "<project-name>",
  "description": "<description>",
  "source": {
    "type": "CODECOMMIT" | "CODEPIPELINE" | "GITHUB" | "GITHUB_ENTERPRISE" | "GITLAB" |
 "GITLAB_SELF_MANAGED" | "BITBUCKET" | "S3" | "NO_SOURCE",
    "location": "<source-location>",
    "gitCloneDepth": "<git-clone-depth>",
    "buildspec": "<buildspec>",
    "InsecureSsl": "<insecure-ssl>",
    "reportBuildStatus": "<report-build-status>",
    "buildStatusConfig": {
```

```
      "context": "<context>",
      "targetUrl": "<target-url>"
    },
    "gitSubmodulesConfig": {
      "fetchSubmodules": "<fetch-submodules>"
    },
    "auth": {
      "type": "<auth-type>",
      "resource": "<auth-resource>"
    },
    "sourceIdentifier": "<source-identifier>"
  },
  "secondarySources": [
    {
        "type": "CODECOMMIT" | "CODEPIPELINE" | "GITHUB" | "GITHUB_ENTERPRISE" |
"GITLAB" | "GITLAB_SELF_MANAGED" | "BITBUCKET" | "S3" | "NO_SOURCE",
        "location": "<source-location>",
        "gitCloneDepth": "<git-clone-depth>",
        "buildspec": "<buildspec>",
        "InsecureSsl": "<insecure-ssl>",
        "reportBuildStatus": "<report-build-status>",
        "auth": {
          "type": "<auth-type>",
          "resource": "<auth-resource>"
        },
        "sourceIdentifier": "<source-identifier>"
    }
  ],
  "secondarySourceVersions": [
    {
      "sourceIdentifier": "<secondary-source-identifier>",
      "sourceVersion": "<secondary-source-version>"
    }
  ],
  "sourceVersion": "<source-version>",
  "artifacts": {
    "type": "CODEPIPELINE" | "S3" | "NO_ARTIFACTS",
    "location": "<artifacts-location>",
    "path": "<artifacts-path>",
    "namespaceType": "<artifacts-namespacetype>",
    "name": "<artifacts-name>",
    "overrideArtifactName": "<override-artifact-name>",
    "packaging": "<artifacts-packaging>"
  },
```

```
  "secondaryArtifacts": [
    {
      "type": "CODEPIPELINE" | "S3" | "NO_ARTIFACTS",
      "location": "<secondary-artifact-location>",
      "path": "<secondary-artifact-path>",
      "namespaceType": "<secondary-artifact-namespaceType>",
      "name": "<secondary-artifact-name>",
      "packaging": "<secondary-artifact-packaging>",
      "artifactIdentifier": "<secondary-artifact-identifier>"
    }
  ],
  "cache": {
    "type": "<cache-type>",
    "location": "<cache-location>",
    "mode": [
      "<cache-mode>"
    ]
  },
  "environment": {
    "type": "LINUX_CONTAINER" | "LINUX_GPU_CONTAINER" | "ARM_CONTAINER" |
"WINDOWS_SERVER_2019_CONTAINER" | "WINDOWS_SERVER_2022_CONTAINER",
    "image": "<image>",
    "computeType": "BUILD_GENERAL1_SMALL" | "BUILD_GENERAL1_MEDIUM" |
"BUILD_GENERAL1_LARGE" | "BUILD_GENERAL1_2XLARGE",
    "certificate": "<certificate>",
    "environmentVariables": [
      {
        "name": "<environmentVariable-name>",
        "value": "<environmentVariable-value>",
        "type": "<environmentVariable-type>"
      }
    ],
    "registryCredential": [
      {
        "credential": "<credential-arn-or-name>",
        "credentialProvider": "<credential-provider>"
      }
    ],
    "imagePullCredentialsType": "CODEBUILD" | "SERVICE_ROLE",
    "privilegedMode": "<privileged-mode>"
  },
  "serviceRole": "<service-role>",
  "timeoutInMinutes": <timeout>,
  "queuedTimeoutInMinutes": <queued-timeout>,
```

```
  "encryptionKey": "<encryption-key>",
  "tags": [
    {
      "key": "<tag-key>",
      "value": "<tag-value>"
    }
  ],
  "vpcConfig": {
    "securityGroupIds": [
         "<security-group-id>"
    ],
    "subnets": [
         "<subnet-id>"
    ],
    "vpcId": "<vpc-id>"
  },
  "badgeEnabled": "<badge-enabled>",
  "logsConfig": {
    "cloudWatchLogs": {
      "status": "<cloudwatch-logs-status>",
      "groupName": "<group-name>",
      "streamName": "<stream-name>"
    },
    "s3Logs": {
      "status": "<s3-logs-status>",
      "location": "<s3-logs-location>",
      "encryptionDisabled": "<s3-logs-encryption-disabled>"
    }
  },
  "fileSystemLocations": [
    {
      "type": "EFS",
      "location": "<EFS-DNS-name-1>:/<directory-path>",
      "mountPoint": "<mount-point>",
      "identifier": "<efs-identifier>",
      "mountOptions": "<efs-mount-options>"
    }
  ],
  "buildBatchConfig": {
    "serviceRole": "<batch-service-role>",
    "combineArtifacts": <combine-artifacts>,
    "restrictions": {
      "maximumBuildsAllowed": <max-builds>,
      "computeTypesAllowed": [
```

```
        "<compute-type>"
      ]
    },
    "timeoutInMins": <batch-timeout>,
    "batchReportMode": "REPORT_AGGREGATED_BATCH" | "REPORT_INDIVIDUAL_BUILDS"
  },
  "concurrentBuildLimit": <concurrent-build-limit>
}
```

Ersetzen Sie Folgendes:

Name

Erforderlich Name dieses Build-Projekts. Dieser Name muss für alle Build-Projekte in Ihrem AWS Konto eindeutig sein.

description

Optional. Beschreibung dieses Build-Projekts.

source

Erforderlich Ein ProjectSourceObjekt, das Informationen zu den Quellcodeeinstellungen dieses Build-Projekts enthält. Nachdem Sie ein `source`-Objekt hinzugefügt haben, können Sie mit bis zu zwölf weitere Quellen hinzufügen. Diese Einstellungen umfassen u. a. folgende:

Quelle/Typ

Erforderlich Der Typ des Repositorys, das den zu erstellenden Quellcode enthält. Gültige Werte sind:

- `CODECOMMIT`
- `CODEPIPELINE`
- `GITHUB`
- `GITHUB_ENTERPRISE`
- `GITLAB`
- `GITLAB_SELF_MANAGED`
- `BITBUCKET`
- `S3`
- `NO_SOURCE`

Wenn Sie `NO_SOURCE` verwenden, kann die Build-Spezifikation keine Datei sein, da das Projekt nicht über eine Quelle verfügt. Stattdessen müssen Sie das `buildspec`-Attribut verwenden, um eine YAML-formatierte Zeichenfolge für Ihre Build-Spezifikation zu verwenden. Weitere Informationen finden Sie unter Beispiel für ein Projekt ohne Quelle.

Quelle/ Ort

<source-type>Erforderlich, sofern Sie nicht auf eingestellt haben. `CODEPIPELINE` Der Speicherort des Quellcodes für den angegebenen Repository-Typ.

- Zum CodeCommit Beispiel die HTTPS-Klon-URL zum Repository, das den Quellcode und die Buildspec-Datei enthält (z. B.). `https://git-codecommit.<region-id>.amazonaws.com/v1/repos/<repo-name>`
- Für Amazon S3 der Name des Build-Eingabe-Buckets, gefolgt vom Pfad und Namen der ZIP-Datei, die den Quellcode und die Buildspec enthält. Beispielsweise:
  - Für eine ZIP-Datei, die sich im Stammverzeichnis des Eingabe-Buckets befindet:. `<bucket-name>/<object-name>.zip`
  - Für eine ZIP-Datei, die sich in einem Unterordner im Eingabe-Bucket befindet:`<bucket-name>/<subfoler-path>/<object-name>.zip`.
- Für GitHub die HTTPS-Klon-URL zum Repository, das den Quellcode und die Buildspec-Datei enthält. Die URL muss „github.com“ enthalten. Sie müssen Ihr Konto mit Ihrem AWS Konto verbinden. GitHub Verwenden Sie dazu die CodeBuild Konsole, um ein Build-Projekt zu erstellen.
  - Wählen Sie Authorize application. (Nachdem Sie sich mit Ihrem GitHub Konto verbunden haben, müssen Sie die Erstellung des Build-Projekts nicht abschließen. Sie können die CodeBuild Konsole schließen.)
- Für GitHub Enterprise Server die HTTP- oder HTTPS-Klon-URL zum Repository, das den Quellcode und die Buildspec-Datei enthält. Sie müssen Ihr Konto auch mit Ihrem GitHub Enterprise AWS Server-Konto verbinden. Verwenden Sie dazu die CodeBuild Konsole, um ein Build-Projekt zu erstellen.
  1. Erstellen Sie ein persönliches Zugriffstoken in GitHub Enterprise Server.
  2. Kopieren Sie dieses Token in Ihre Zwischenablage, damit Sie es bei der Erstellung Ihres CodeBuild Projekts verwenden können. Weitere Informationen finden Sie auf der GitHub Hilfeseite unter Erstellen eines persönlichen Zugriffstokens für die Befehlszeile.
  3. Wenn Sie die Konsole verwenden, um Ihr CodeBuild Projekt zu erstellen, wählen Sie in Source für Source Provider die Option GitHubEnterprise aus.

4. Für Personal Access Token fügen Sie das Token ein, das in Ihre Zwischenablage kopiert wurde. Wählen Sie Save Token. Ihr CodeBuild Konto ist jetzt mit Ihrem GitHub Enterprise Server-Konto verbunden.

- Für GitLab und GitLab selbst verwaltet, die HTTPS-Klon-URL zum Repository, das den Quellcode und die Buildspec-Datei enthält. Beachten Sie, dass die URL bei Verwendung GitLab gitlab.com enthalten muss. Wenn du GitLab Self-managed verwendest, muss die URL nicht gitlab.com enthalten. Du musst dein Konto mit deinem AWS Konto GitLab oder GitLab deinem selbst verwalteten Konto verbinden. Verwenden Sie dazu die CodeBuild Konsole, um ein Build-Projekt zu erstellen.
  - Wählen Sie im Navigationsbereich der Developer Tools die Optionen Einstellungen, Verbindungen und anschließend Verbindung erstellen aus. Erstellen Sie auf dieser Seite entweder eine GitLab oder eine GitLab selbstverwaltete Verbindung und wählen Sie dann Connect aus. GitLab
- Für Bitbucket handelt es sich um die HTTPS-Klon-URL des Repositorys, das den Quellcode und die buildspec-Datei enthält. Die URL muss „bitbucket.org“ enthalten. Du musst dein Konto außerdem mit deinem AWS Bitbucket-Konto verbinden. Verwende dazu die CodeBuild Konsole, um ein Build-Projekt zu erstellen.
  1. Wenn Sie die Konsole zum Verbinden (oder Wiederverbinden) mit Bitbucket verwenden, wählen Sie auf der Seite Confirm access to your account die Option Grant access. (Nachdem du dich mit deinem Bitbucket-Konto verbunden hast, musst du die Erstellung des Build-Projekts nicht abschließen. Du kannst die CodeBuild Konsole schließen.)
- Für AWS CodePipeline, geben Sie keinen `location` Wert für an`source`. CodePipeline ignoriert diesen Wert CodePipeline, da Sie beim Erstellen einer Pipeline in den Quellcodepfad der Pipeline den Speicherort des Quellcodes angeben.

Quelle/ gitCloneDepth

Optional. Die Tiefe des herunterzuladenden Verlaufs. Der Mindestwert ist 0. Ist dieser Wert 0, größer als 25 oder nicht angegeben, wird für jedes Build-Projekt der vollständige Verlauf heruntergeladen. Wenn Ihr Quelltyp Amazon S3 ist, wird dieser Wert nicht unterstützt.

Quelle/ Buildspec

Optional. Die zu verwendende Build-Spezifikationsdefinition oder -datei. Wenn der Wert nicht angegeben oder eine leere Zeichenfolge ist, muss der Quellcode eine `buildspec.yml`-Datei im Stammverzeichnis enthalten. Wenn dieser Wert gesetzt ist, kann es sich entweder um eine Inline-Buildspec-Definition, den Pfad zu einer alternativen Buildspec-Datei relativ zum Stammverzeichnis Ihrer Primärquelle oder um den Pfad zu einem S3-Bucket handeln. Der Bucket muss sich in

derselben Region wie das Build-Projekt befinden AWS . Geben Sie die buildspec-Datei mit ihrem ARN an (z. B. `arn:aws:s3:::<my-codebuild-sample2>/buildspec.yml`). Weitere Informationen finden Sie unter Dateiname der Build-Spezifikation und Speicherort.

Quelle/Authentifizierung

Nicht verwenden. Dieses Objekt wird nur von der CodeBuild Konsole verwendet.

Quelle/ reportBuildStatus

Gibt an, ob Ihr Quell-Anbieter den Status eines Build-Starts und -Abschlusses sendet. Wenn Sie dies mit einem anderen Quellanbieter als GitHub GitHub Enterprise Server oder Bitbucket festlegen, `invalidInputException` wird ein ausgelöst.

Um den Build-Status an den Quell-Provider melden zu können, muss der mit dem Quell-Provider verknüpfte Benutzer Schreibzugriff auf das Repo haben. Wenn der Benutzer keinen Schreibzugriff hat, kann der Build-Status nicht aktualisiert werden. Weitere Informationen finden Sie unter Zugriff auf den Quellanbieter.

Quelle/ buildStatusConfig

Enthält Informationen, die definieren, wie das CodeBuild Build-Projekt den Build-Status an den Quellanbieter meldet. Diese Option wird nur verwendet, wenn der Quelltyp `GITHUBGITHUB_ENTERPRISE`, oder ist`BITBUCKET`.

Quelle/buildStatusConfig/Kontext

Bei Bitbucket-Quellen wird dieser Parameter für den `name` Parameter im Bitbucket-Commit-Status verwendet. Bei GitHub Quellen wird dieser Parameter für den `context` Parameter im GitHub Commit-Status verwendet.

Sie können zum Beispiel die Build-Nummer `context` enthalten und den Webhook-Trigger mithilfe der CodeBuild Umgebungsvariablen auslösen:

```
AWS CodeBuild sample-project Build #$CODEBUILD_BUILD_NUMBER -
 $CODEBUILD_WEBHOOK_TRIGGER
```

Dies führt dazu, dass der Kontext für Build #24, ausgelöst durch ein Webhook-Pull-Request-Ereignis, wie folgt aussieht:

```
AWS CodeBuild sample-project Build #24 - pr/8
```

Quelle/ ZielURL buildStatusConfig

Bei Bitbucket-Quellen wird dieser Parameter für den `url` Parameter im Bitbucket-Commit-Status verwendet. Bei GitHub Quellen wird dieser Parameter für den `target_url` Parameter im GitHub Commit-Status verwendet.

Sie können beispielsweise den Wert `targetUrl` auf festlegen `https://aws.amazon.com/codebuild/`*`<path to build>`* und der Commit-Status wird auf diese URL verweisen.

Sie können auch CodeBuild Umgebungsvariablen in die aufnehmen`targetUrl`, um der URL zusätzliche Informationen hinzuzufügen. Um beispielsweise die Build-Region zur URL hinzuzufügen, setzen Sie den Wert `targetUrl` auf:

```
"targetUrl": "https://aws.amazon.com/codebuild/<path to build>?region=
$AWS_REGION"
```

Wenn die Build-Region ist`us-east-2`, wird dies erweitert auf:

```
https://aws.amazon.com/codebuild/<path to build>?region=us-east-2
```

Quelle/ gitSubmodulesConfig

Optional. Informationen zur Konfiguration der Git-Submodule. Wird nur mit CodeCommit GitHub, GitHub Enterprise Server und Bitbucket verwendet.

quelle///fetchSubmodules gitSubmodulesConfig

Legen Sie für `fetchSubmodules` `true` fest, wenn die Git-Submodule in Ihr Repository aufgenommen werden sollen. Die einbezogenen Git-Submodule müssen als HTTPS konfiguriert werden.

Quelle/ InsecureSsl

Optional. Wird nur mit GitHub Enterprise Server verwendet. Setzen Sie diesen Wert auf, `true` um TLS-Warnungen zu ignorieren, während Sie eine Verbindung zu Ihrem GitHub Enterprise Server-Projekt-Repository herstellen. Der Standardwert ist `false`. `InsecureSsl` sollte nur für Testzwecke verwendet werden. Es sollte nicht in einer Produktionsumgebung verwendet werden.

Quelle/ SourceIdentifier

Ein benutzerdefinierter Bezeichner für die Projektquelle. Optional für die Primärquelle. Für Sekundärquellen erforderlich.

secondarySources

Optional. Eine Reihe von ProjectSourceObjekten, die Informationen zu den Sekundärquellen für ein Build-Projekt enthalten. Sie können bis zu 12 Sekundärquellen hinzufügen. Die `secondarySources` Objekte verwenden dieselben Eigenschaften wie das Objekt. In einem sekundären Quellobjekt `sourceIdentifier` ist der erforderlich.

secondarySourceVersions

Optional. Ein Array von ProjectSourceVersion-Objekten. Wenn `secondarySourceVersions` auf Build-Ebene angegeben ist, haben sie Vorrang vor dieser Angabe.

sourceVersion

Optional. Die Version der Build-Eingabe, die für dieses Projekt erstellt werden soll. Ist dieser Parameter nicht angegeben, wird die neueste Version verwendet. Ist er festgelegt, muss er Folgendes sein:

- Für CodeCommit die zu verwendende Commit-ID, den Branch oder das Git-Tag.
- Für GitHub die Commit-ID, die Pull-Request-ID, den Branch-Namen oder den Tag-Namen, der der Version des Quellcodes entspricht, den Sie erstellen möchten. Wenn eine Pull-Anforderungs-ID angegeben ist, muss diese das Format `pr/pull-request-ID` aufweisen (Beispiel: `pr/25`). Wenn ein Branch-Name angegeben wird, wird die Commit-ID von HEAD verwendet. Wenn sie nicht angegeben ist, wird die Commit-ID von HEAD für den Standard-Branch verwendet.
- Für GitLab die Commit-ID, die Pull-Request-ID, den Branchennamen, den Tag-Namen oder die Referenz und eine Commit-ID. Weitere Informationen finden Sie unter Beispiel für eine Quellversion mit AWS CodeBuild.
- Für Bitbucket, Commit-ID, Branch-Name oder Tag-Name, die/der der Version des Quellcodes entspricht, die Sie erstellen möchten. Wenn ein Branch-Name angegeben wird, wird die Commit-ID von HEAD verwendet. Wenn sie nicht angegeben ist, wird die Commit-ID von HEAD für den Standard-Branch verwendet.
- Für Amazon S3 die Versions-ID des Objekts, das die zu verwendende Build-Eingabe-ZIP-Datei darstellt.

Wenn `sourceVersion` auf Build-Ebene angegeben ist, hat jene Version Vorrang vor dieser Version `sourceVersion` (auf Projektebene). Weitere Informationen finden Sie unter Beispiel für eine Quellversion mit AWS CodeBuild.

Artefakte

Erforderlich Ein ProjectArtifactsObjekt, das Informationen über die Einstellungen für das Ausgabeartefakt dieses Build-Projekts enthält. Nachdem Sie ein `artifacts`-Objekt hinzugefügt haben, können Sie mit bis zu zwölf weitere Artefakte hinzufügen. Diese Einstellungen umfassen u. a. folgende:

Artefakte/Typ

Erforderlich Der Typ des Build-Ausgabeartefakts. Gültige Werte für sind:

- `CODEPIPELINE`
- `NO_ARTIFACTS`
- `S3`

Artefakte/ Standort

Wird nur mit dem Artefakttyp verwendet. `S3` Wird nicht für andere Artefakttypen verwendet.

Der Name des Ausgabe-Buckets, den Sie erstellt oder in den Voraussetzungen identifiziert haben.

Artefakte/Pfad

Wird nur mit dem Artefakttyp verwendet. `S3` Wird nicht für andere Artefakttypen verwendet.

Der Pfad im Ausgabe-Bucket, in den die ZIP-Datei oder der Ordner eingefügt werden sollen. Wenn Sie keinen Wert für angeben`path`, CodeBuild verwendet `namespaceType` (falls angegeben) und, `name` um den Pfad und den Namen der Build-Ausgabe-ZIP-Datei oder des Ordners zu ermitteln. Wenn Sie beispielsweise für `path` und `MyPath` `MyArtifact.zip` für angeben`name`, würden der Pfad und der Name wie folgt lauten`MyPath/MyArtifact.zip`.

Artefakte/ namespaceType

Wird nur mit dem Artefakttyp verwendet. `S3` Wird nicht für andere Artefakttypen verwendet.

Der Namespace der ZIP-Datei oder des Ordners für die Build-Ausgabe. Gültige Werte sind `BUILD_ID` und `NONE`. Verwenden Sie `BUILD_ID`, um die Build-ID in den Pfad und den Namen für die ZIP-Datei oder den Ordner einzubeziehen. Verwenden Sie andernfalls `NONE`. Wenn Sie keinen Wert für angeben`namespaceType`, CodeBuild verwendet `path` (falls angegeben) und, `name` um den Pfad und den Namen der ZIP-Datei oder des Ordners der Build-Ausgabe zu ermitteln. Wenn Sie beispielsweise für, `MyPath` für `path` und `BUILD_ID` `MyArtifact.zip` für

angebennamespaceType, würden der Pfad und der Name wie folgt lautenMyPath/*build-ID*/MyArtifact.zip. name

artifacts/name

Wird nur mit dem S3 Artefakttyp verwendet. Wird nicht für andere Artefakttypen verwendet.

Der Name der ZIP-Datei oder des Ordners in der location Build-Ausgabe. Wenn Sie beispielsweise für path und MyPath MyArtifact.zip für angebenname, würden der Pfad und der Name wie folgt lautenMyPath/MyArtifact.zip.

Artefakte/ overrideArtifactName

Wird nur mit dem Artefakttyp S3 verwendet. Wird nicht für andere Artefakttypen verwendet.

Optional. Wenn auf gesetzttrue, hat der im artifacts Block der Buildspec-Datei angegebene Name Vorrang. name Weitere Informationen finden Sie unter Referenz zur Build-Spezifikation für CodeBuild.

Artefakte/ Verpackung

Wird nur mit dem Artefakttyp verwendet. S3 Wird nicht für andere Artefakttypen verwendet.

Optional. Gibt an, wie die Artefakte verpackt werden sollen. Die zulässigen Werte sind:

NONE

Erstellen Sie einen Ordner, der die Build-Artefakte enthält. Dies ist der Standardwert.

ZIP

Erstellen Sie eine ZIP-Datei, die die Build-Artefakte enthält.

secondaryArtifacts

Optional. Eine Reihe von ProjectArtifactsObjekten, die Informationen zu den Einstellungen für sekundäre Artefakte für ein Build-Projekt enthalten. Sie können bis zu zwölf sekundäre Attribute hinzufügen. secondaryArtifacts verwendet viele der Einstellungen, die vom -Objekt verwendet werden.

Cache

Erforderlich Ein ProjectCacheObjekt, das Informationen zu den Cache-Einstellungen dieses Build-Projekts enthält. Weitere Informationen finden Sie unter Build-Caching.

Umgebung

Erforderlich Ein ProjectEnvironmentObjekt, das Informationen über die Build-Umgebungseinstellungen dieses Projekts enthält. Diese Einstellungen umfassen Folgendes:

Umgebung/Typ

Erforderlich Der Typ der Build-Umgebung. Weitere Informationen finden Sie unter type in der CodeBuild API-Referenz.

Umgebung/Bild

Erforderlich Der Bezeichner des Docker-Images, den diese Build-Umgebung nutzt. Dieser Bezeichner wird normalerweise im Format *image-name*:*tag* aufgeführt. In dem Docker-Repository, das zur Verwaltung seiner Docker-Images CodeBuild verwendet wird, könnte dies beispielsweise der Fall sein. `aws/codebuild/standard:5.0` Im Docker Hub: `maven:3.3.9-jdk-8`. In Amazon ECR,*account-id*`.dkr.ecr.`*region-id*`.amazonaws.com/`*your-Amazon-ECR-repo-name*`:`*tag*. Weitere Informationen finden Sie unter Docker-Images bereitgestellt von CodeBuild.

Umgebung/ ComputeType

Erforderlich Gibt die Rechenressourcen an, die von dieser Build-Umgebung verwendet werden. Weitere Informationen finden Sie unter ComputeType in der CodeBuild API-Referenz.

Umgebung/Zertifikat

Optional. Der ARN des Amazon S3 S3-Buckets, das Pfadpräfix und der Objektschlüssel, der das PEM-kodierte Zertifikat enthält. Der Objektschlüssel kann entweder nur die .pem-Datei oder eine .zip-Datei mit dem PEM-codierten Zertifikat sein. Wenn Ihr Amazon S3 S3-Bucket-Name beispielsweise lautet*<my-bucket>*, Ihr Pfadpräfix ist *<cert>* und Ihr Objektschlüsselname lautet*<certificate.pem>*, dann `certificate` sind die akzeptablen Formate für *<my-bucket/cert/certificate.pem>* oder`arn:aws:s3:::`*<my-bucket/cert/certificate.pem>*.

Umgebung/Umgebungsvariablen

Optional. Ein Array von EnvironmentVariableObjekten, das die Umgebungsvariablen enthält, die Sie für diese Build-Umgebung angeben möchten. Jede Umgebungsvariable wird als Objekt ausgedrückt, das ein `namevalue`, und `type` von `namevalue`, und enthält`type`.

Konsole und AWS CLI Benutzer können alle Umgebungsvariablen sehen. Wenn Sie keine Bedenken hinsichtlich der Sichtbarkeit Ihrer Umgebungsvariablen haben, setzen Sie `name` und und `value` setzen Sie `type` sie auf`PLAINTEXT`.

Wir empfehlen, Umgebungsvariablen mit sensiblen Werten wie einer AWS Zugriffsschlüssel-ID, einem AWS geheimen Zugriffsschlüssel oder einem Passwort als Parameter im Amazon EC2 Systems Manager Parameter Store oder AWS Secrets Manager zu speichern. Geben `name` Sie für diesen gespeicherten Parameter einen Bezeichner ein, CodeBuild auf den verwiesen werden soll.

Wenn Sie Amazon EC2 Systems Manager Parameter Store für verwenden`value`, legen Sie den Namen des Parameters so fest, wie er im Parameter Store gespeichert ist. Setzen Sie `type` auf `PARAMETER_STORE`. Verwenden Sie einen `/CodeBuild/dockerLoginPassword` als Beispiel benannten Parameter und setzen Sie ihn `name` auf`LOGIN_PASSWORD`. Setzen Sie `value` auf `/CodeBuild/dockerLoginPassword`. Setzen Sie `type` auf `PARAMETER_STORE`.

> **Important**
>
> Wenn Sie Amazon EC2 Systems Manager Parameter Store verwenden, empfehlen wir, Parameter mit Parameternamen zu speichern, die mit `/CodeBuild/` (z. B.`/CodeBuild/dockerLoginPassword`) beginnen. Sie können die CodeBuild Konsole verwenden, um einen Parameter in Amazon EC2 Systems Manager zu erstellen. Wählen Sie Create a parameter (Parameter erstellen) aus und befolgen Sie dann die Anweisungen im Dialogfeld. (In diesem Dialogfeld können Sie für KMS-Schlüssel den ARN eines AWS KMS Schlüssels in Ihrem Konto angeben. Amazon EC2 Systems Manager verwendet diesen Schlüssel, um den Wert des Parameters beim Speichern zu verschlüsseln und beim Abrufen zu entschlüsseln.) Wenn Sie die CodeBuild Konsole verwenden, um einen Parameter zu erstellen, beginnt die Konsole den Parameternamen mit dem, `/CodeBuild/` wie er gespeichert wird. Weitere Informationen finden Sie unter Systems Manager Parameter Store und Systems Manager Parameter Store Console Walkthrough im Amazon EC2 Systems Manager Manager-Benutzerhandbuch.
> Wenn sich Ihr Build-Projekt auf Parameter Store von Amazon EC2 Systems Manager bezieht, muss die Service-Rolle des Build-Projekts die `ssm:GetParameters` Aktion zulassen. Wenn Sie zuvor Neue Servicerolle ausgewählt haben, wird CodeBuild diese Aktion in die Standard-Servicerolle für Ihr Build-Projekt aufgenommen. Wenn Sie jedoch Existing service role (Vorhandene Servicerolle) ausgewählt haben, müssen Sie diese Aktion separat in Ihre Servicerolle aufnehmen.

Wenn sich Ihr Build-Projekt auf Parameter bezieht, die im Amazon EC2 Systems Manager Parameter Store mit Parameternamen gespeichert sind, die nicht mit beginnen/`CodeBuild/`, und Sie Neue Servicerolle wählen, müssen Sie diese Servicerolle aktualisieren, um Zugriff auf Parameternamen zu gewähren, die nicht mit `/CodeBuild/` beginnen. Dies liegt daran, dass diese Service-Rolle nur auf Parameternamen zugreift, die mit `/CodeBuild/` beginnen.
Wenn Sie Neue Servicerolle wählen, beinhaltet die Servicerolle die Berechtigung, alle Parameter unter dem `/CodeBuild/` Namespace im Amazon EC2 Systems Manager Parameter Store zu entschlüsseln.
Von Ihnen gesetzte Umgebungsvariablen ersetzen vorhandene Umgebungsvariablen. Wenn das Docker-Image beispielsweise bereits eine Umgebungsvariable mit dem Namen `MY_VAR` und einem Wert von `my_value` enthält und Sie eine Umgebungsvariable mit dem Namen `MY_VAR` und einem Wert von `other_value` festlegen, wird `my_value` durch `other_value` ersetzt. Wenn das Docker-Image demgegenüber bereits eine Umgebungsvariable mit dem Namen `PATH` und einem Wert von `/usr/local/sbin:/usr/local/bin` enthält und Sie eine Umgebungsvariable mit dem Namen `PATH` und einem Wert von `$PATH:/usr/share/ant/bin` festlegen, wird `/usr/local/sbin:/usr/local/bin` durch den Literalwert `$PATH:/usr/share/ant/bin` ersetzt.
Legen Sie keine Umgebungsvariable mit einem Namen fest, der mit `CODEBUILD_` beginnt. Dieses Präfix ist zur -internen Verwendung reserviert.
Wenn eine Umgebungsvariable mit identischem Namen an mehreren Orten definiert ist, wird der Wert folgendermaßen bestimmt:

- Der Wert im Aufruf zum Starten des Build-Vorgangs hat den höchsten Vorrang.
- Der Wert in der Build-Projektdefinition folgt darauf.
- Der Wert in der buildspec-Deklaration hat die niedrigste Priorität.

Wenn Sie Secrets Manager verwenden, geben Sie für den Namen des Parameters an`value`, wie er in Secrets Manager gespeichert ist. Setzen Sie `type` auf `SECRETS_MANAGER`. Verwenden Sie ein `/CodeBuild/dockerLoginPassword` als Beispiel benanntes Geheimnis und legen Sie `name` den Wert auf fest`LOGIN_PASSWORD`. Setzen Sie `value` auf `/CodeBuild/dockerLoginPassword`. Setzen Sie `type` auf `SECRETS_MANAGER`.

> **Important**
>
> Wenn Sie Secrets Manager verwenden, empfehlen wir, Secrets mit Namen zu speichern, die mit `/CodeBuild/` (z. B.`/CodeBuild/dockerLoginPassword`) beginnen. Weitere Informationen finden Sie unter Was ist AWS Secrets Manager? im AWS Secrets Manager -Benutzerhandbuch.
> Wenn sich Ihr Build-Projekt auf Geheimnisse bezieht, die in Secrets Manager gespeichert sind, muss die Service-Rolle des Build-Projekts die `secretsmanager:GetSecretValue` Aktion zulassen. Wenn Sie zuvor Neue Servicerolle ausgewählt haben, wird CodeBuild diese Aktion in die Standard-Servicerolle für Ihr Build-Projekt aufgenommen. Wenn Sie jedoch Existing service role (Vorhandene Servicerolle) ausgewählt haben, müssen Sie diese Aktion separat in Ihre Servicerolle aufnehmen.
> Wenn sich Ihr Build-Projekt auf Geheimnisse bezieht, die in Secrets Manager mit geheimen Namen gespeichert sind, die nicht mit beginnen`/CodeBuild/`, und Sie Neue Dienstrolle ausgewählt haben, müssen Sie die Servicerolle aktualisieren, um Zugriff auf geheime Namen zu ermöglichen, die nicht mit beginnen`/CodeBuild/`. Dies liegt daran, dass die Servicerolle nur den Zugriff auf geheime Namen ermöglicht, die mit beginnen`/CodeBuild/`.
> Wenn Sie Neue Dienstrolle wählen, beinhaltet die Dienstrolle die Berechtigung, alle Geheimnisse unter dem `/CodeBuild/` Namespace im Secrets Manager zu entschlüsseln.

Umgebung/RegistryCredential

Optional. Ein RegistryCredentialObjekt, das die Anmeldeinformationen angibt, die den Zugriff auf eine private Docker-Registrierung ermöglichen.

Umgebung/RegistryCredential/ Credential

Gibt den ARN oder Namen der Anmeldeinformationen an, die mit erstellt wurden AWS Managed Services. Sie können den Namen der Anmeldeinformationen nur verwenden, wenn diese in Ihrer aktuellen Region vorhanden sind.

Environment/RegistryCredential/ CredentialProvider

Der einzige gültige Wert ist `SECRETS_MANAGER`.

Wenn diese Eigenschaft festgelegt ist:

- muss `imagePullCredentials` auf `SERVICE_ROLE` festgelegt sein.
- Das Bild kann kein kuratiertes Bild oder ein Amazon ECR-Bild sein.

Umgebung/Typ imagePullCredentials

Optional. Der Typ der Anmeldeinformationen, die zum Abrufen von Images in Ihrem Build CodeBuild verwendet werden. Es gibt zwei gültige Werte:

CODEBUILD

`CODEBUILD`gibt an, dass es seine eigenen Anmeldeinformationen CodeBuild verwendet. Sie müssen Ihre Amazon ECR-Repository-Richtlinie bearbeiten, um dem CodeBuild Service Principal zu vertrauen.

SERVICE_ROLE

Gibt an, dass die Servicerolle Ihres Build-Projekts CodeBuild verwendet wird.

Wenn Sie ein kontoübergreifendes oder privates Registrierungs-Image verwenden, müssen Sie `SERVICE_ROLE`-Anmeldeinformationen verwenden. Wenn Sie ein CodeBuild kuratiertes Image verwenden, müssen Sie `CODEBUILD` Anmeldeinformationen verwenden.

Umgebung/ PrivilegedMode

`true`Nur festlegen, wenn Sie dieses Build-Projekt zum Erstellen von Docker-Images verwenden möchten. Andernfalls schlagen alle zugehörigen Builds fehl, die versuchen, mit dem Docker-Daemon zu interagieren. Sie müssen zudem den Docker-Daemon müssen, damit Ihre Builds interagieren können. Eine Möglichkeit besteht darin, den Docker-Daemon in der `install`-Phase der buildspec-Datei zu initialisieren, indem Sie die folgenden Build-Befehle ausführen. Führen Sie diese Befehle nicht aus, wenn Sie ein Image für die Build-Umgebung angegeben haben, das von CodeBuild mit Docker-Unterstützung bereitgestellt wird.

> **Note**
>
> Standardmäßig ist der Docker-Daemon für Nicht-VPC-Builds aktiviert. Wenn Sie Docker-Container für VPC-Builds verwenden möchten, lesen Sie auf der Docker Docs-Website unter Runtime Privilege and Linux Capabilities nach und aktivieren Sie den privilegierten Modus. Außerdem unterstützt Windows den privilegierten Modus nicht.

```
- nohup /usr/local/bin/dockerd --host=unix:///var/run/docker.sock --
host=tcp://127.0.0.1:2375 --storage-driver=overlay2 &
```

```
- timeout 15 sh -c "until docker info; do echo .; sleep 1; done"
```

serviceRole

Erforderlich Der ARN der Servicerolle, der CodeBuild verwendet wird, um im Namen des Benutzers mit Diensten zu interagieren (z. B.`arn:aws:iam::`*`account-id`*`:role/`*`role-name`*).

timeoutInMinutes

Optional. Die Anzahl der Minuten zwischen 5 und 2160 (36 Stunden), nach der der Build CodeBuild gestoppt wird, falls er nicht abgeschlossen ist. Wenn Sie keinen anderen Wert angeben, wird der Standardwert von 60 verwendet. Führen Sie den Befehl aus, um festzustellen, ob und wann ein Build aufgrund eines Timeouts CodeBuild gestoppt wurde. `batch-get-builds` Überprüfen Sie die Ausgabe eines `buildStatus`-Werts von `FAILED`, um festzustellen, ob ein Build-Vorgang angehalten wurde. Überprüfen Sie die Ausgabe des `endTime`-Werts in Verbindung mit einem `phaseStatus`-Wert von `TIMED_OUT`, um festzustellen, wann die Zeitbeschränkung bei einem Build-Vorgang überschritten wurde.

queuedTimeoutInMinuten

Optional. Die Anzahl der Minuten zwischen 5 und 480 (8 Stunden), nach denen der Build CodeBuild gestoppt wird, falls er sich noch in der Warteschlange befindet. Wenn Sie keinen anderen Wert angeben, wird der Standardwert von 60 verwendet.

encryptionKey

Optional. Der Alias oder ARN des, der von AWS KMS key verwendet wird CodeBuild , um die Build-Ausgabe zu verschlüsseln. Verwenden Sie zur Angabe eines Alias das Format `arn:aws:kms:`*`region-ID`*`:`*`account-ID`*`:key/`*`key-ID`* oder (bei vorhandenem Alias) `alias/`*`key-alias`*. Falls nicht angegeben, wird der AWS-managed KMS-Schlüssel für Amazon S3 verwendet.

tags

Optional. Ein Array von Tag-Objekten, die die Tags bereitstellen, die Sie diesem Build-Projekt zuordnen möchten. Sie können bis zu 50 Tags angeben. Diese Tags können von jedem AWS Dienst verwendet werden, der CodeBuild Build-Projekt-Tags unterstützt. Jedes Tag wird als Objekt mit a `key` und a ausgedrücktvalue.

vpcConfig

Optional. Ein VpcConfigObjekt, das Informationsinformationen zur VPC-Konfiguration für Ihr Projekt enthält. Weitere Informationen finden Sie unter Verwendung AWS CodeBuild mit Amazon Virtual Private Cloud.

Zu diesen Eigenschaften gehören:

vpcId

Erforderlich Die VPC-ID, die CodeBuild verwendet wird. Führen Sie diesen Befehl aus, um eine Liste aller VPC-IDs in Ihrer Region zu erhalten:

```
aws ec2 describe-vpcs --region <region-ID>
```

Subnetze

Erforderlich Ein Array von Subnetz-IDs, die Ressourcen enthalten, die von verwendet werden. CodeBuild Führen Sie diesen Befehl aus, um diese IDs zu erhalten:

```
aws ec2 describe-subnets --filters "Name=vpc-id,Values=<vpc-id>" --region <region-ID>
```

securityGroupIds

Erforderlich Eine Reihe von Sicherheitsgruppen-IDs, die von verwendet werden CodeBuild , um den Zugriff auf Ressourcen in der VPC zu ermöglichen. Führen Sie diesen Befehl aus, um diese IDs zu erhalten:

```
aws ec2 describe-security-groups --filters "Name=vpc-id,Values=<vpc-id>" --<region-ID>
```

badgeEnabled

Optional. Gibt an, ob Build-Badges in Ihr Projekt aufgenommen werden sollen. CodeBuild Stellen Sie diese Option auf `true` ein, um Build-Badges zu aktivieren, oder `false` auf eine andere Einstellung. Weitere Informationen finden Sie unter Beispiel für Badges erstellen mit CodeBuild.

LogsConfig

Ein LogsConfigObjekt, das Informationen darüber enthält, wo sich die Logs dieses Builds befinden.

LogsConfig/ cloudWatchLogs

Ein CloudWatchLogsConfigObjekt, das Informationen darüber enthält, wie Logs in Logs übertragen werden. CloudWatch

LogsConfig/ S3Logs

Ein LogsConfigS3-Objekt, das Informationen zur Übertragung von Protokollen an Amazon S3 enthält.

fileSystemLocations

Optional. Eine Reihe von ProjectFileSystemsLocationObjekten, die Informationen über Ihre Amazon EFS-Konfiguration enthalten.

buildBatchConfig

Optional. Das `buildBatchConfig` Objekt ist eine ProjectBuildBatchConfigStruktur, die die Batch-Build-Konfigurationsinformationen für das Projekt enthält.

buildBatchConfig/serviceRole

Die Dienstrolle ARN für das Batch-Build-Projekt.

buildBatchConfig/Kombiniere Artefakte

Ein boolescher Wert, der angibt, ob die Build-Artefakte für den Batch-Build an einem einzigen Artefakt-Speicherort kombiniert werden sollen.

buildBatchConfig/Einschränkungen/ maximumBuildsAllowed

Die maximal zulässige Anzahl von Builds.

buildBatchConfig/Einschränkungen/ computeTypesAllowed

Ein Array von Zeichenfolgen, die die Datenverarbeitungstypen angeben, die für den Stapel-Build zulässig sind. Informationen zu diesen Werten finden Sie unter Berechnungstypen für die Build-Umgebung.

buildBatchConfig/timeoutInMinutes

Die maximale Zeitspanne in Minuten, in der der Batch-Build abgeschlossen sein muss.

buildBatchConfig/batchReportMode

Gibt an, wie Build-Statusberichte an den Quellanbieter für den Batch-Build gesendet werden. Gültige Werte sind:

`REPORT_AGGREGATED_BATCH`

(Standard) Aggregieren Sie alle Erstellungsstatus in einem einzigen Statusbericht.

`REPORT_INDIVIDUAL_BUILDS`

Senden Sie für jeden einzelnen Build einen separaten Statusbericht.

concurrentBuildLimit

Die maximale Anzahl gleichzeitiger Builds, die für dieses Projekt zulässig sind.

Neue Builds werden nur gestartet, wenn die aktuelle Anzahl der Builds dieses Limit unterschreitet oder ihm entspricht. Wenn die aktuelle Build-Anzahl dieses Limit erreicht, werden neue Builds gedrosselt und nicht ausgeführt.

Erstellen des Projekts

Um das Projekt zu erstellen, führen Sie den **`create-project`** Befehl erneut aus und übergeben Sie Ihre JSON-Datei:

```
aws codebuild create-project --cli-input-json file://<json-file>
```

Bei Erfolg wird die JSON-Darstellung eines Projektobjekts in der Konsolenausgabe angezeigt. Ein Beispiel für diese Daten finden Sie in der CreateProject Antwortsyntax.

Abgesehen vom Namen des Build-Projekts können Sie alle Einstellungen des Build-Projekts zu einem späteren Zeitpunkt ändern. Weitere Informationen finden Sie unter Ändern der Einstellungen eines Build-Projekts (AWS CLI).

Weitere Informationen zum Starten der Build-Ausführung finden Sie in Ausführen eines Build (AWS CLI).

Wenn Ihr Quellcode in einem GitHub Repository gespeichert ist und Sie den Quellcode jedes Mal neu erstellen CodeBuild möchten, wenn eine Codeänderung in das Repository übertragen wird, finden Sie weitere Informationen unterAusführung von Builds automatisch starten (AWS CLI).

Erstellen eines Build-Projekts (AWS SDKs)

Informationen zur Verwendung von AWS CodeBuild mit den AWS-SDKs finden Sie unter AWS SDKs- und Tools-Referenz.

Erstellen eines Build-Projekts (AWS CloudFormation)

Für weitere Informationen zur Nutzung vonAWS CodeBuildmitAWS CloudFormation, finden Sie unterdieAWS CloudFormationVorlage für CodeBuildimAWS CloudFormation-Benutzerhandbuchaus.

Erstellen einer Benachrichtigungsregel

Sie können Benachrichtigungsregeln verwenden, um Benutzer zu benachrichtigen, wenn wichtige Änderungen wie Build-Erfolge und -Fehler auftreten. In den Benachrichtigungsregeln werden sowohl die Ereignisse als auch das Amazon SNS SNS-Thema festgelegt, das zum Senden von Benachrichtigungen verwendet wird. Weitere Informationen finden Sie unter Was sind Benachrichtigungen?.

Sie können die Konsole oder die AWS CLI verwenden, um Benachrichtigungsregeln für AWS CodeBuild zu erstellen.

So erstellen Sie eine Benachrichtigungsregel (Konsole):

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die CodeBuild Konsole unter https://console.aws.amazon.com/codebuild/.
2. Wählen Sie Build, Build-Projekte und dann ein Build-Projekt aus, dem Sie Benachrichtigungen hinzufügen möchten.
3. Wählen Sie auf der Build-Projektseite Notify (Benachrichtigung) und dann Create notification rule (Benachrichtigungsregel erstellen) aus. Sie können auch die Seite Settings (Einstellungen) für das Build-Projekt aufrufen und Create notification rule (Benachrichtigungsregel erstellen) auswählen.
4. Geben Sie unter Notification name (Benachrichtigungsname) einen Namen für die Regel ein.
5. Wählen Sie unter Detailtyp die Option Basic aus, wenn Sie möchten, dass nur die Informationen, die Amazon zur Verfügung gestellt wurden, in der Benachrichtigung EventBridge enthalten sind. Wählen Sie „Vollständig“, wenn Sie Informationen, die Amazon zur Verfügung gestellt wurden, EventBridge und Informationen, die möglicherweise vom CodeBuild oder vom Benachrichtigungsmanager bereitgestellt wurden, einbeziehen möchten.

Weitere Informationen finden Sie unter Informationen zu Inhalten und Sicherheit von Benachrichtigungen.

6. Wählen Sie unter Events that trigger notifications (Ereignisse, die Benachrichtigungen auslösen) die Ereignisse aus, für die Sie Benachrichtigungen senden möchten. Weitere Informationen finden Sie unter Ereignisse für Benachrichtigungsregeln für Build-Projekte.
7. Führen Sie unter Targets (Ziele) einen der folgenden Schritte aus:

   - Wenn Sie bereits eine Ressource zur Verwendung mit Benachrichtigungen konfiguriert haben, wählen Sie unter Choose target type (Zieltyp wählen) entweder AWS Chatbot(Slack) oder SNS topic (SNS-Thema)aus. Wählen Sie unter Ziel auswählen den Namen des Clients (für einen in konfigurierten Slack-ClientAWS Chatbot) oder den Amazon-Ressourcennamen (ARN) des Amazon SNS-Themas (für Amazon SNS SNS-Themen, die bereits mit der für Benachrichtigungen erforderlichen Richtlinie konfiguriert wurden).
   - Wenn Sie keine Ressource für die Verwendung mit Benachrichtigungen konfiguriert haben, wählen Sie Create target (Ziel erstellen) und dann SNS topic (SNS-Thema) aus. Geben Sie nach codestar-notifications- einen Namen für das Thema an und wählen Sie dann Create (Erstellen).

   > **Note**
   >
   > - Wenn Sie das Amazon-SNS-Thema im Rahmen des Erstellens der Benachrichtigungsregel erstellen, wird die Richtlinie, die es ermöglicht, Ereignisse in dem Thema zu veröffentlichen, für Sie angewendet. Durch die Verwendung eines Themas, das für Benachrichtigungsregeln erstellt wurde, kann sichergestellt werden, dass Sie das Thema nur für die Benutzer abonnieren, die Benachrichtigungen zu dieser Ressource erhalten sollen.
   > - Sie können keinen AWS Chatbot-Client als Teil der Erstellung einer Benachrichtigungsregel erstellen. Wenn Sie AWS Chatbot (Slack) wählen, sehen Sie eine Schaltfläche, die Sie zur Konfiguration eines Clients in AWS Chatbot führt. Wenn Sie diese Option auswählen, wird die AWS Chatbot-Konsole geöffnet. Weitere Informationen finden Sie unter Konfigurieren der Integration zwischen Benachrichtigungen und AWS Chatbot.
   > - Wenn Sie ein vorhandenes Amazon SNS SNS-Thema als Ziel verwenden möchten, müssen Sie die erforderliche Richtlinie für AWS CodeStar Benachrichtigungen

zusätzlich zu allen anderen Richtlinien hinzufügen, die möglicherweise für dieses Thema existieren. Weitere Informationen finden Sie unter Konfigurieren vorhandener Amazon SNS-Themen für Benachrichtigungen und Informationen zu Inhalten und Sicherheit von Benachrichtigungen.

8. Um die Erstellung der Regel abzuschließen, wählen Sie Submit (Absenden) aus.
9. Sie müssen das Amazon SNS SNS-Thema für die Regel abonnieren, bevor sie Benachrichtigungen erhalten können. Weitere Informationen finden Sie unter Amazon SNS SNS-Themen, die Ziele sind für Benutzer abonnieren. Sie können auch die Integration zwischen Benachrichtigungen einrichten und Benachrichtigungen AWS Chatbot an Amazon Chime Chime-Chatrooms senden. Weitere Informationen finden Sie unter Konfigurieren der Integration zwischen Benachrichtigungen und AWS Chatbot.

So erstellen Sie eine Benachrichtigungsregel (AWS CLI):

1. Führen Sie in einem Terminal oder einer Eingabeaufforderung den Befehl create-notification rule aus, um das JSON-Skelett zu generieren:

```
aws codestarnotifications create-notification-rule --generate-cli-skeleton
 > rule.json
```

Sie können die Datei beliebig benennen. In diesem Beispiel trägt die Datei den Namen *rule.json*.

2. Öffnen Sie die JSON-Datei in einem Texteditor, und bearbeiten Sie sie so, dass sie die Ressource, die Ereignistypen und das gewünschte Ziel für die Regel enthält. *Das folgende Beispiel zeigt eine Benachrichtigungsregel, die* ***MyNotificationRule*** *nach einem Build-Projekt benannt ist, das MyBuildProjectin einem AWS Konto mit der ID 123456789012 benannt ist.* Benachrichtigungen werden mit dem vollständigen Detailtyp an ein Amazon SNS SNS-Thema namens *codestar-notifications gesendet – MyNotificationTopic* wenn Builds erfolgreich sind:

```
{
    "Name": "MyNotificationRule",
    "EventTypeIds": [
        "codebuild-project-build-state-succeeded"
    ],
    "Resource": "arn:aws:codebuild:us-east-2:123456789012:MyBuildProject",
```

```
    "Targets": [
        {
            "TargetType": "SNS",
            "TargetAddress": "arn:aws:sns:us-east-2:123456789012:codestar-
notifications-MyNotificationTopic"
        }
    ],
    "Status": "ENABLED",
    "DetailType": "FULL"
}
```

Speichern Sie die Datei.

3. Führen Sie unter Verwendung der soeben bearbeiteten Datei am Terminal oder in der Befehlszeile erneut den Befehl create-notification-rule aus, um die Benachrichtigungsregel zu erstellen:

```
aws codestarnotifications create-notification-rule --cli-input-json
 file://rule.json
```

4. Bei Erfolg gibt der Befehl den ARN der Benachrichtigungsregel zurück, der ähnlich wie im Folgenden dargestellt aussieht:

```
{
    "Arn": "arn:aws:codestar-notifications:us-east-1:123456789012:notificationrule/
dc82df7a-EXAMPLE"
}
```

Anzeigen einer Liste mit Build-Projektnamen in AWS CodeBuild

Sie können dasAWS CodeBuildconsole,AWS CLI, oderAWS-SDKs, um eine Liste von Build-Projekten in CodeBuild anzuzeigen.

Themen

- Anzeigen einer Liste mit Build-Projektnamen (Konsole)
- Anzeigen einer Liste mit Build-Projektnamen (AWS CLI)
- Anzeigen einer Liste mit Build-Projektnamen (AWS SDKs)

Anzeigen einer Liste mit Build-Projektnamen (Konsole)

Sie können eine Liste der Build-Projekte in einer AWS-Region in der Konsole anzeigen. Zu den Informationen gehören der Name, der Quellanbieter, das Repository, der aktuelle Build-Status und gegebenenfalls eine Beschreibung.

1. Öffnen SieAWS CodeBuildconsole beihttps://console.aws.amazon.com/codesuite/codebuild/homeaus.
2. Wählen Sie im linken Navigationsbereich Build projects aus.

> **Note**
>
> Standardmäßig werden nur die letzten 10 Build-Projekte angezeigt. Zur Anzeige von weiteren Build-Projekten wählen Sie das Zahnradsymbol und einen anderen Wert für Projects per page (Projekte je Seite) aus oder verwenden die Vorwärts- und Rückwärtspfeile.

Anzeigen einer Liste mit Build-Projektnamen (AWS CLI)

Führen Sie den Befehl list-projects aus:

```
aws codebuild list-projects --sort-by sort-by --sort-order sort-order --next-
token next-token
```

Ersetzen Sie im Befehl oben die folgenden Platzhalter:

- *sort-by*: Optionale Zeichenfolge, die das Kriterium angibt, das zur Auflistung von Build-Projektnamen verwendet wird. Gültige Werte sind:
  - `CREATED_TIME`: Listet die Build-Projektnamen basierend auf dem Zeitpunkt der Erstellung der einzelnen Build-Projekte auf.
  - `LAST_MODIFIED_TIME`: Listet die Build-Projektnamen basierend auf dem Zeitpunkt der letzten Änderung der Informationen über die einzelnen Build-Projekte auf.
  - `NAME`: Listet die Build-Projekte basierend auf den Namen der einzelnen Build-Projekte auf.
- *sort-order*: Optionale Zeichenfolge, die die Reihenfolge angibt, in der Build-Projekte basierend auf*sort-by*aus. Gültige Werte sind `ASCENDING` und `DESCENDING`.

- *nächstes Token*: Optionale Zeichenfolge. Falls während einer vorherigen Ausführung mehr als 100 Elemente in der Liste enthalten waren, werden nur die ersten 100 Elemente zurückgegeben, zusammen mit einer eindeutigen Zeichenfolge namens next token. Führen Sie diesen Befehl erneut aus, um das nächste Token hinzuzufügen und den nächsten Stapel von Listenelementen abzurufen. Um alle Elemente in der Liste abzurufen, führen Sie diesen Befehl mit jedem nachfolgenden "next token" aus, bis keine weiteren nächsten Token zurückgegeben werden.

Wenn Sie z. B. den folgenden Befehl ausführen:

```
aws codebuild list-projects --sort-by NAME --sort-order ASCENDING
```

Sollte die Ausgabe folgendermaßen oder ähnlich aussehen:

```
{
  "nextToken": "Ci33ACF6...The full token has been omitted for brevity...U+AkMx8=",
  "projects": [
    "codebuild-demo-project",
    "codebuild-demo-project2",
    ... The full list of build project names has been omitted for brevity ...
    "codebuild-demo-project99"
  ]
}
```

Wenn Sie diesen Befehl erneut ausführen:

```
aws codebuild list-projects  --sort-by NAME --sort-order ASCENDING --next-token
 Ci33ACF6...The full token has been omitted for brevity...U+AkMx8=
```

Sollte die Ausgabe folgendermaßen oder ähnlich aussehen:

```
{
  "projects": [
    "codebuild-demo-project100",
    "codebuild-demo-project101",
    ... The full list of build project names has been omitted for brevity ...
    "codebuild-demo-project122"
  ]
}
```

Anzeigen einer Liste mit Build-Projektnamen (AWS SDKs)

Weitere Informationen zur Verwendung von AWS CodeBuild mit den AWS-SDKs finden Sie unter AWS SDKs- und Tools-Referenz.

Anzeigen der Details eines Build-Projekts in AWS CodeBuild

Sie können dasAWS CodeBuild-KonsoleAWS CLI, oderAWS-SDKs, um die Details eines Build-Projekts in CodeBuild anzuzeigen.

Themen

- Anzeigen der Details eines Build-Projekts (Konsole)
- Anzeigen der Details eines Build-Projekts (AWS CLI)
- Anzeigen der Details eines Build-Projekts (AWS SDKs)

Anzeigen der Details eines Build-Projekts (Konsole)

1. Öffnen SieAWS CodeBuild-Konsolehttps://console.aws.amazon.com/codesuite/codebuild/homeaus.
2. Wählen Sie im linken Navigationsbereich Build projects aus.

   > **Note**
   >
   > Standardmäßig werden nur die letzten 10 Build-Projekte angezeigt. Zur Anzeige von weiteren Build-Projekten wählen Sie das Zahnradsymbol und einen anderen Wert für Projects per page (Projekte je Seite) aus oder verwenden die Vorwärts- und Rückwärtspfeile.

3. Wählen Sie in der Liste der Build-Projekte in der Spalte Name den Link für das Build-Projekt aus.
4. Wählen Sie auf der Seite Build project (Build-Projekt): ***project-name*** den Eintrag Build details (Build-Details) aus.

Anzeigen der Details eines Build-Projekts (AWS CLI)

Führen Sie den Befehl batch-get-projects aus:

```
aws codebuild batch-get-projects --names names
```

Ersetzen Sie im Befehl oben den folgenden Platzhalter:

- *Namen*: Erforderliche Zeichenfolge, die verwendet wird, um einen oder mehrere Build-Projektnamen anzugeben, deren Details angezeigt werden sollen. Um mehr als ein Build-Projekt anzugeben, fügen Sie zwischen den einzelnen Build-Projektnamen ein Leerzeichen ein. Sie können bis zu 100 Build-Projektnamen angeben. Informationen zum Abrufen einer Liste von Build-Projekten finden Sie unter Anzeigen einer Liste mit Build-Projektnamen (AWS CLI).

Wenn Sie z. B. den folgenden Befehl ausführen:

```
aws codebuild batch-get-projects --names codebuild-demo-project codebuild-demo-project2
 my-other-demo-project
```

Sollte die Ausgabe folgendermaßen oder ähnlich aussehen. Auslassungspunkte (`...`) stehen für Daten, die zur Abkürzung ausgelassen wurden.

```
{
  "projectsNotFound": [
    "my-other-demo-project"
  ],
  "projects": [
    {
      ...
      "name": codebuild-demo-project,
      ...
    },
    {
      ...
      "name": codebuild-demo-project2",
      ...
    }
  ]
}
```

In der obigen Ausgabe listet das Array `projectsNotFound` alle Build-Projektnamen auf, die angegeben, aber nicht gefunden wurden. Das Array `projects` listet Details für jedes Build-Projekt auf, für das Informationen gefunden wurden. In der obigen Ausgabe wurden aus Gründen der

Übersichtlichkeit Build-Projektdetails ausgelassen. Weitere Informationen finden Sie in der Ausgabe von Erstellen eines Build-Projekts (AWS CLI).

Diebatch-get-projects-Befehl unterstützt das Filtern nach bestimmten Eigenschaftswerten nicht, aber Sie können ein Skript schreiben, das die Eigenschaften für ein Projekt aufzählt. Das folgende Linux-Shell-Skript zählt beispielsweise die Projekte in der aktuellen Region für das Girokonto auf und gibt das von jedem Projekt verwendete Bild aus.

```
#!/usr/bin/sh

# This script enumerates all of the projects for the current account
# in the current region and prints out the image that each project is using.

imageName=""

function getImageName(){
  local environmentValues=(${1//$'\t'/ })
  imageName=${environmentValues[1]}
}

function processProjectInfo() {
  local projectInfo=$1

  while IFS=$'\t' read -r section value; do
    if [[ "$section" == *"ENVIRONMENT"* ]]; then
      getImageName "$value"
    fi
  done <<< "$projectInfo"
}

# Get the list of projects.
projectList=$(aws codebuild list-projects --output=text)

for projectName in $projectList
do
  if [[ "$projectName" != *"PROJECTS"* ]]; then
    echo "============================================="

    # Get the detailed information for the project.
    projectInfo=$(aws codebuild batch-get-projects --output=text --names
 "$projectName")

    processProjectInfo "$projectInfo"
```

```
    printf 'Project "%s" has image "%s"\n' "$projectName" "$imageName"
  fi
done
```

Weitere Informationen zur Verwendung der AWS CLI mit AWS CodeBuild finden Sie unter Befehlszeilenreferenz.

Anzeigen der Details eines Build-Projekts (AWS SDKs)

Weitere Informationen zur Verwendung von AWS CodeBuild mit den AWS-SDKs finden Sie unter AWS SDKs- und Tools-Referenz.

Build-Caching in AWS CodeBuild

Sie können Zeit sparen, wenn beim Erstellen Ihres Projekts ein Cache verwendet wird. In einem Cache können wiederverwendbare Teile Ihrer Build-Umgebung gespeichert werden, die dann für mehrere Builds verwendet werden können. Ihr Build-Projekt kann eine von zwei Arten von Caching verwenden: Amazon S3 oder lokal. Wenn Sie einen lokalen Cache verwenden, müssen Sie mindestens einen von drei Cache-Modi auswählen: Quellcache, Docker-Ebenen-Cache und benutzerdefinierter Cache.

> **Note**
>
> Der Docker-Ebenen-Cache-Modus ist nur für die Linux-Umgebung verfügbar. Wenn Sie diesen Modus wählen, müssen Sie Ihren Build im privilegierten Modus ausführen. CodeBuild Projects gewährt dem privilegierten Modus gewährt seinem Container Zugriff auf alle Geräte. Weitere Informationen finden Sie unter Laufzeitberechtigungen und Linux-Funktionen auf der Docker-Docs-Website.

Themen

- Amazon S3-Caching
- Lokales Caching

Amazon S3-Caching

Amazon S3-Caching speichert den Cache in einem Amazon S3-Bucket, der auf mehreren Build-Hosts verfügbar ist. Dies ist eine gute Option für kleine bis mittelgroße Build-Artefakte, die teurer zu

erstellen sind als zum Herunterladen. Für große Build-Artefakte ist diese Option nicht optimal, da ihre Übertragung über das Netzwerk unter Umständen viel Zeit in Anspruch nimmt, was sich auf die Build-Leistung auswirken kann. Es ist auch nicht die beste Option, wenn Sie Docker-Layer verwenden.

Lokales Caching

Beim lokalen Caching wird der Cache lokal auf einem Build-Host gespeichert und ist nur für diesen Build-Host verfügbar. Dies ist eine gute Option für Zwischen- bis große Build-Artefakte, da der Cache sofort auf dem Build-Host verfügbar ist. Dies ist nicht die beste Option, wenn Ihre Builds selten sind. Die Build-Leistung ist somit nicht durch die Netzwerk-Übertragungsdauer beeinträchtigt.

Wenn Sie sich für lokales Caching entscheiden, müssen Sie mindestens einen der folgenden Cache-Modi auswählen:

- Beim Quellcache-Modus werden Git-Metadaten für primäre und sekundäre Quellen zwischengespeichert. Nachdem der Cache erstellt wurde, wird bei nachfolgenden Builds nur die Änderung zwischen den Commits abgerufen. Dieser Modus ist gut geeignet für Projekte mit einem sauberen Arbeitsverzeichnis und einem großen Git-Repository als Quelle. Wenn Sie diese Option wählen und Ihr Projekt kein Git-Repository (AWS CodeCommit, GitHub, GitHub Enterprise Server oder Bitbucket) verwendet, wird die Option ignoriert.
- Beim Docker-Ebenen-Cache-Modus werden vorhandene Docker-Ebenen zwischengespeichert. Dieser Modus eignet sich für Projekte, bei denen große Docker-Images erstellt oder abgerufen werden. Mit diesem Modus können Leistungsprobleme vermieden werden, die beim Abruf großer Docker-Images aus dem Netzwerk entstehen.

> **Note**
>
> - Ein Docker-Ebenen-Cache kann nur in einer Linux-Umgebung verwendet werden.
> - Das `privileged`-Flag muss so festgelegt sein, dass Ihr Projekt über die erforderlichen Docker-Berechtigungen verfügt.
>
>   Standardmäßig ist Docker-Daemon für Nicht-VPC-Builds aktiviert. Wenn Sie Docker-Container für VPC-Builds verwenden möchten, finden Sie weitere Informationen unter Laufzeitberechtigung und Linux-Funktionen auf der Docker-Docs-Website und aktivieren Sie den privilegierten Modus. Außerdem unterstützt Windows den privilegierten Modus nicht.

- Vor der Verwendung eines Docker-Ebenen-Cache sollten Sie die Auswirkungen auf die Sicherheit berücksichtigen.

- Beim benutzerdefinierten Cache-Modus werden Verzeichnisse zwischengespeichert, die Sie in der buildspec-Datei angeben. Dieser Modus ist gut geeignet, wenn Ihr Build-Szenario nicht für einen der beiden anderen lokalen Cache-Modi in Frage kommt. Bei Verwendung eines benutzerdefinierte Cache gilt Folgendes:
  - Für das Caching können nur Verzeichnisse angegeben werden. Sie können keine einzelnen Dateien angeben.
  - Es werden Symlinks verwendet, um auf zwischengespeicherte Verzeichnisse zu verweisen.
  - Zwischengespeicherte Verzeichnisse werden mit Ihrem Build verknüpft, bevor die Projektquellen heruntergeladen werden. Im Cache gespeicherte Elemente überschreiben Quellelemente, wenn sie denselben Namen haben. Verzeichnisse werden unter Verwendung von Cache-Pfaden in der buildspec-Datei angegeben. Weitere Informationen finden Sie unter Syntax der Build-Spezifikation.
  - Vermeiden Sie Verzeichnisnamen, die in der Quelle und im Cache identisch sind. Lokal zwischengespeicherte Verzeichnisse können die Inhalte von Verzeichnissen in einem Quell-Repository überschreiben oder löschen, das denselben Namen hat.

> **Note**
>
> Lokales Caching wird mit dem `LINUX_GPU_CONTAINER` Umgebungstyp und dem `BUILD_GENERAL1_2XLARGE` Datenverarbeitungstyp nicht unterstützt. Weitere Informationen finden Sie unter Berechnungsmodi und Typen der Build-Umgebung.

> **Note**
>
> Lokales Caching wird nicht unterstützt, wenn Sie für CodeBuild die Arbeit mit einer VPC konfigurieren. Weitere Informationen zur Verwendung von VPCs mit finden Sie CodeBuildunter Verwendung AWS CodeBuild mit Amazon Virtual Private Cloud.

Themen

- Angabe von lokalem Caching (CLI)
- Angabe von lokalem Caching (Konsole)
- Angabe von lokalem Caching (AWS CloudFormation)

Sie können die AWS CLI, die Konsole, das SDK oder verwenden, AWS CloudFormation um einen lokalen Cache anzugeben.

Angabe von lokalem Caching (CLI)

Über den `--cache`-Parameter in der AWS CLI können Sie die drei lokalen Cache-Typen angeben.

- So geben Sie einen Quellcache an:

```
--cache type=LOCAL,mode=[LOCAL_SOURCE_CACHE]
```

- So geben Sie eine Docker-Ebenen-Cache an:

```
--cache type=LOCAL,mode=[LOCAL_DOCKER_LAYER_CACHE]
```

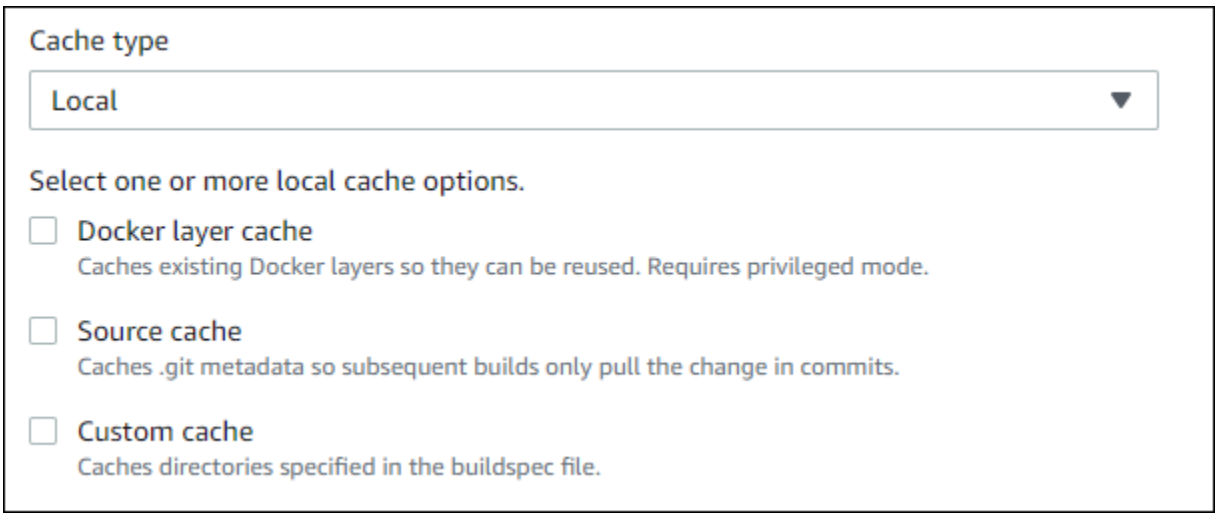
- So geben Sie einen benutzerdefinierten Cache an:

```
--cache type=LOCAL,mode=[LOCAL_CUSTOM_CACHE]
```

Weitere Informationen finden Sie unter Erstellen eines Build-Projekts (AWS CLI).

Angabe von lokalem Caching (Konsole)

Geben Sie den Cache im Abschnitt Artifacts (Artefakte) der Konsole an. Wählen Sie für Cache-Typ Amazon S3 oder Lokal aus. Wenn Sie Local (Lokal) auswählen, wählen Sie mindestens eine der drei lokalen Cache-Optionen aus.



Weitere Informationen finden Sie unter Erstellen Sie ein Build-Projekt (Konsole).

Angabe von lokalem Caching (AWS CloudFormation)

Wenn Sie einen lokalen Cache mittels AWS CloudFormation angeben, geben Sie bei der `Cache`-Eigenschaft für `Type` `LOCAL` an. Der folgende in YAML formatierte AWS CloudFormation-Beispielcode gibt alle drei lokalen Cache-Typen an. Sie können eine beliebige Kombination der Typen angeben. Wenn Sie einen Docker-Ebenen-Cache verwenden, müssen Sie unter `Environment` (Umgebung) für `PrivilegedMode` die Option `true` und für `Type` (Typ) die Option `LINUX_CONTAINER` festlegen.

```
CodeBuildProject:
    Type: AWS::CodeBuild::Project
    Properties:
      Name: MyProject
      ServiceRole: <service-role>
      Artifacts:
        Type: S3
        Location: <bucket-name>
        Name: myArtifact
        EncryptionDisabled: true
        OverrideArtifactName: true
      Environment:
        Type: LINUX_CONTAINER
        ComputeType: BUILD_GENERAL1_SMALL
        Image: aws/codebuild/standard:5.0
        Certificate: <bucket/cert.zip>
        # PrivilegedMode must be true if you specify LOCAL_DOCKER_LAYER_CACHE
        PrivilegedMode: true
```

```
    Source:
      Type: GITHUB
      Location: <github-location>
      InsecureSsl: true
      GitCloneDepth: 1
      ReportBuildStatus: false
    TimeoutInMinutes: 10
    Cache:
      Type: LOCAL
      Modes: # You can specify one or more cache mode,
        - LOCAL_CUSTOM_CACHE
        - LOCAL_DOCKER_LAYER_CACHE
        - LOCAL_SOURCE_CACHE
```

> **Note**
>
> Standardmäßig ist Docker-Daemon für Nicht-VPC-Builds aktiviert. Wenn Sie Docker-Container für VPC-Builds verwenden möchten, finden Sie weitere Informationen unter Laufzeitberechtigung und Linux-Funktionen auf der Docker-Docs-Website und aktivieren Sie den privilegierten Modus. Außerdem unterstützt Windows den privilegierten Modus nicht.

Weitere Informationen finden Sie unter Erstellen eines Build-Projekts (AWS CloudFormation).

Trigger einbauen AWS CodeBuild

Themen

- AWS CodeBuild Trigger erstellen
- AWS CodeBuild Auslöser bearbeiten

AWS CodeBuild Trigger erstellen

AWS CodeBuild Trigger erstellen (Konsole)

Sie können einen Auslöser für ein Projekt erstellen, um eine Erstellung pro Stunde, Tag oder Woche zu planen. Sie können einen Trigger auch mithilfe einer benutzerdefinierten Regel mit einem CloudWatch Amazon-Cron-Ausdruck erstellen. Mit einem cron-Ausdruck können Sie beispielsweise eine Erstellung für eine bestimmte Zeit an jedem Wochentag planen.

> **Note**
>
> Es ist nicht möglich, einen Batch-Build von einem Build-Trigger, einem EventBridge Amazon-Ereignis oder einer AWS Step Functions Aufgabe aus zu starten.

So erstellen Sie einen Auslöser

1. Öffnen Sie die AWS CodeBuild Konsole unter https://console.aws.amazon.com/codesuite/codebuild/home.
2. Wählen Sie im linken Navigationsbereich Build projects aus.
3. Klicken Sie auf den Link des Build-Projekts, dem Sie einen Auslöser hinzufügen möchten, und wählen Sie dann die Registerkarte Build triggers (Auslöser erstellen).

   > **Note**
   >
   > Standardmäßig werden die 100 neuesten Build-Projekte angezeigt. Zur Anzeige von weiteren Build-Projekten wählen Sie das Zahnradsymbol und einen anderen Wert für Projects per page (Projekte je Seite) aus oder verwenden die Vorwärts- und Rückwärtspfeile.

4. Wählen Sie Create trigger.
5. Geben Sie unter Trigger name (Auslösername) einen Namen ein.
6. Wählen Sie in der Dropdown-Liste Frequency (Frequenz) die Frequenz für den Auslöser aus. Wenn Sie mit einem Cron-Ausdruck eine Frequenz erstellen möchten, wählen Sie Custom (Benutzerdefiniert) aus.
7. Legen Sie die Parameter für die Häufigkeit Ihres Auslösers fest. Sie können die ersten paar Zeichen Ihrer Auswahl in das Textfeld eingeben, um die Elemente des Dropdown-Menüs zu filtern.

   > **Note**
   >
   > Die Startstunden und -minuten basieren auf Null. Die Startminute ist eine Zahl zwischen Null und 59. Die Startstunde ist eine Zahl zwischen Null und 23. Ein täglicher Trigger, der jeden Tag um 12:15 Uhr startet, hat beispielsweise eine Startstunde von 12 und eine Startminute von 15. Ein täglicher Trigger, der jeden Tag um Mitternacht beginnt, hat eine

Startstunde von Null und eine Startminute von Null. Ein täglicher Trigger, der jeden Tag um 23:59 Uhr beginnt, hat eine Startstunde von 23 und eine Startminute von 59.

Häufigkeit	Erforderliche Parameter	Details
Stündlich	Startminute	Verwenden Sie das Dropdown-Menü Start minute (Startminute).
Täglich	Startminute Startstunde	Verwenden Sie das Dropdown-Menü Start minute (Startminute). Verwenden Sie das Dropdown-Menü Start hour (Startstunde).
Wöchentlich	Startminute Startstunde Starttag	Verwenden Sie das Dropdown-Menü Start minute (Startminute). Verwenden Sie das Dropdown-Menü Start hour (Startstunde). Verwenden Sie das Dropdown-Menü Start day (Starttag).

Häufigkeit	Erforderliche Parameter	Details
Benutzerdefiniert	Cron-Ausdruck	Geben Sie unter Cron expression (Cron-Ausdruck) einen Cron-Ausdruck ein. Ein Cron-Ausdruck besitzt sechs durch Leerzeichen voneinander getrennte Pflichtfelder. Die Felder geben einen Startwert für Minute, Stunde, Tag, Tag des Monats, Monat, Tag der Woche und Jahr an. Sie können Platzhalterzeichen verwenden, um einen Bereich, zusätzliche Werte und mehr anzugeben. Der Cron-Ausdruck **`0 9 ? * MON-FRI *`** plant beispielsweise an jedem Wochentag um 9:00 Uhr einen Build. Weitere Informationen finden Sie unter Cron-Ausdrücke im Amazon CloudWatch Events-Benutzerhandbuch.

8. Wählen Sie Enable this trigger (Diesen Trigger aktivieren) aus.
9. (Optional) Erweitern Sie den Abschnitt Advanced (Erweitert). Geben Sie in das Feld Source version (Quellversion) eine Version der Quelle ein.

   - Geben Sie für Amazon S3 die Versions-ID ein, die der Version des Eingabeartefakts entspricht, das Sie erstellen möchten. Wenn Sie das Feld Source version (Quellversion) leer lassen, wird die neueste Version verwendet.
   - Geben Sie für AWS CodeCommit eine Commit-ID ein. Wenn das Feld Source version (Quellversion) leer bleibt, wird die HEAD-Commit-ID der Standard-Branch verwendet.
   - Geben Sie für GitHub oder GitHub Enterprise eine Commit-ID, eine Pull-Request-ID, einen Branch-Namen oder einen Tag-Namen ein, der der Version des Quellcodes entspricht, den

Sie erstellen möchten. Wenn Sie eine Pull-Anforderungs-ID angeben, muss diese das Format `pr/pull-request-ID` verwenden (Beispiel: `pr/25`). Wenn Sie einen Branch-Namen angeben wird, wird die Commit-ID von HEAD verwendet. Wenn das Feld Source version (Quellversion) leer ist, wird die HEAD-Commit-ID für die Standard-Branch verwendet.

- Für Bitbucket geben Sie eine Commit-ID, einen Verzweigungsnamen oder einen Tag-Namen, die/der der Version des Quellcodes entspricht, den Sie erstellen möchten. Wenn Sie einen Branch-Namen angeben wird, wird die Commit-ID von HEAD verwendet. Wenn das Feld Source version (Quellversion) leer ist, wird die HEAD-Commit-ID für die Standard-Branch verwendet.

10. (Optional) Geben Sie ein Timeout zwischen 5 Minuten und 2160 Minuten (36 Stunden) an. Dieser Wert gibt an, wie lange ein Build AWS CodeBuild versucht, bevor er beendet wird. Wenn die Felder Hours (Stunden) und Minutes (Minuten) leer bleiben, wird der Timeout-Standardwert im Projekt verwendet.
11. Wählen Sie Create trigger.

AWS CodeBuild Trigger programmgesteuert erstellen

CodeBuild verwendet EventBridge Amazon-Regeln für Build-Trigger. Sie können die EventBridge API verwenden, um programmgesteuert Build-Trigger für Ihre CodeBuild Projekte zu erstellen. Weitere Informationen finden Sie in der Amazon EventBridge API-Referenz.

AWS CodeBuild Auslöser bearbeiten

AWS CodeBuild Trigger bearbeiten (Konsole)

Sie können einen Auslöser für ein Projekt bearbeiten, um eine Erstellung pro Stunde, Tag oder Woche zu planen. Sie können einen Trigger auch bearbeiten, um eine benutzerdefinierte Regel mit einem CloudWatch Amazon-Cron-Ausdruck zu verwenden. Mit einem cron-Ausdruck können Sie beispielsweise eine Erstellung für eine bestimmte Zeit an jedem Wochentag planen. Weitere Informationen zum Erstellen eines Auslösers finden Sie unter AWS CodeBuild Trigger erstellen.

So bearbeiten Sie einen Auslöser

1. Öffnen Sie die AWS CodeBuild Konsole unter https://console.aws.amazon.com/codesuite/codebuild/home.
2. Wählen Sie im linken Navigationsbereich Build projects aus.

3. Wählen Sie den Link des Build-Projekts aus, das Sie bearbeiten möchten, und anschließend die Registerkarte Build triggers (Build-Auslöser).

> **Note**
>
> Standardmäßig werden die 100 neuesten Build-Projekte angezeigt. Zur Anzeige von weiteren Build-Projekten wählen Sie das Zahnradsymbol und einen anderen Wert für Projects per page (Projekte je Seite) aus oder verwenden die Vorwärts- und Rückwärtspfeile.

4. Aktivieren Sie das Optionsfeld neben dem Auslöser, den Sie ändern möchten, und klicken Sie dann auf Edit (Bearbeiten).
5. Wählen Sie in der Dropdown-Liste Frequency (Frequenz) die Frequenz für den Auslöser aus. Wenn Sie mit einem Cron-Ausdruck eine Frequenz erstellen möchten, wählen Sie Custom (Benutzerdefiniert) aus.
6. Legen Sie die Parameter für die Häufigkeit Ihres Auslösers fest. Sie können die ersten paar Zeichen Ihrer Auswahl in das Textfeld eingeben, um die Elemente des Dropdown-Menüs zu filtern.

> **Note**
>
> Die Startstunden und -minuten basieren auf Null. Die Startminute ist eine Zahl zwischen Null und 59. Die Startstunde ist eine Zahl zwischen Null und 23. Ein täglicher Trigger, der jeden Tag um 12:15 Uhr startet, hat beispielsweise eine Startstunde von 12 und eine Startminute von 15. Ein täglicher Trigger, der jeden Tag um Mitternacht beginnt, hat eine Startstunde von Null und eine Startminute von Null. Ein täglicher Trigger, der jeden Tag um 23:59 Uhr beginnt, hat eine Startstunde von 23 und eine Startminute von 59.

Häufigkeit	Erforderliche Parameter	Details
Stündlich	Startminute	Verwenden Sie das Dropdown-Menü Start minute (Startminute).

Häufigkeit	Erforderliche Parameter	Details
Täglich	Startminute Startstunde	Verwenden Sie das Dropdown-Menü Start minute (Startminute). Verwenden Sie das Dropdown-Menü Start hour (Startstunde).
Wöchentlich	Startminute Startstunde Starttag	Verwenden Sie das Dropdown-Menü Start minute (Startminute). Verwenden Sie das Dropdown-Menü Start hour (Startstunde). Verwenden Sie das Dropdown-Menü Start day (Starttag).

Häufigkeit	Erforderliche Parameter	Details
Benutzerdefiniert	Cron-Ausdruck	Geben Sie unter Cron expression (Cron-Ausdruck) einen Cron-Ausdruck ein. Ein Cron-Ausdruck besitzt sechs durch Leerzeichen voneinander getrennte Pflichtfelder. Die Felder geben einen Startwert für Minute, Stunde, Tag, Tag des Monats, Monat, Tag der Woche und Jahr an. Sie können Platzhalterzeichen verwenden, um einen Bereich, zusätzliche Werte und mehr anzugeben. Der Cron-Ausdruck **`0 9 ? * MON-FRI *`** plant beispielsweise an jedem Wochentag um 9:00 Uhr einen Build. Weitere Informationen finden Sie unter Cron-Ausdrücke im Amazon CloudWatch Events-Benutzerhandbuch.

7. Wählen Sie Enable this trigger (Diesen Trigger aktivieren) aus.

> **Note**
>
> Sie können die CloudWatch Amazon-Konsole unter https://console.aws.amazon.com/cloudwatch/ verwenden, um Quellversion, Timeout und andere Optionen zu bearbeiten, die in AWS CodeBuild nicht verfügbar sind.

AWS CodeBuild Trigger programmgesteuert bearbeiten

CodeBuild verwendet EventBridge Amazon-Regeln für Build-Trigger. Sie können die EventBridge API verwenden, um die Build-Trigger für Ihre CodeBuild Projekte programmgesteuert zu bearbeiten. Weitere Informationen finden Sie in der Amazon EventBridge API-Referenz.

GitLab Verbindungen

Mithilfe von Verbindungen können Sie Konfigurationen autorisieren und einrichten, die Ihren Drittanbieter mit Ihren AWS Ressourcen verknüpfen. AWS CodeConnections Um Ihr Drittanbieter-Repository als Quelle für Ihr Build-Projekt zuzuordnen, verwenden Sie eine Verbindung.

Um einen GitLab oder einen GitLab selbst verwalteten Quellanbieter hinzuzufügen CodeBuild, können Sie zwischen folgenden Optionen wählen:

- Verwenden Sie in der CodeBuild Konsole den Assistenten zum Erstellen eines Build-Projekts oder auf der Seite „Quelle bearbeiten“, um die Option GitLaboder GitLab Self Managed Provider auszuwählen. Informationen Stellen Sie eine Verbindung zu GitLab (Konsole) her zum Hinzufügen des Quellanbieters finden Sie unter. Die Konsole hilft Ihnen beim Erstellen einer Verbindungsressource.
- Verwenden Sie die CLI, um Ihre Verbindungsressourcen zu erstellen. Weitere Informationen finden Sie unter Verbindung herstellen zu GitLab (CLI) So erstellen Sie eine Verbindungsressource mit der CLI.

> **Note**
>
> Sie können eine Verbindung auch mithilfe der Developer Tools-Konsole unter Einstellungen herstellen. Weitere Informationen finden Sie unter Verbindung erstellen.

> **Note**
>
> Indem Sie diese Verbindungsinstallation in autorisieren GitLab, gewähren Sie unserem Service die Erlaubnis, Ihre Daten zu verarbeiten, indem Sie auf Ihr Konto zugreifen. Sie können die Berechtigungen jederzeit widerrufen, indem Sie die Anwendung deinstallieren.

Bevor Sie beginnen:

- Sie müssen bereits ein Konto bei erstellt haben. GitLab

> **Note**
>
> Verbindungen bieten nur Zugriff auf Repositorys, die dem Konto gehören, das zum Erstellen und Autorisieren der Verbindung verwendet wurde.

> **Note**
>
> Sie können Verbindungen zu einem Repository herstellen GitLab, in dem Sie die Rolle des Besitzers haben, und dann kann die Verbindung mit dem Repository mit Ressourcen wie verwendet werden CodeBuild. Bei Repositorys in Gruppen müssen Sie nicht der Gruppenbesitzer sein.

- Um eine Quelle für Ihr Build-Projekt anzugeben, müssen Sie bereits ein Repository für erstellt haben GitLab.

Themen

Stellen Sie eine Verbindung zu GitLab (Konsole) her

Gehen Sie wie folgt vor, um mithilfe der CodeBuild Konsole eine Verbindung für Ihr Projekt (Repository) hinzuzufügen GitLab.

Um Ihr Build-Projekt zu erstellen oder zu bearbeiten

1. Melden Sie sich bei der CodeBuild Konsole an.
2. Wählen Sie eine der folgenden Optionen aus.
   - Wählen Sie, ob Sie ein Build-Projekt erstellen möchten. Folgen Sie den Anweisungen unterErstellen Sie ein Build-Projekt (Konsole), um den ersten Bildschirm auszufüllen, und wählen Sie im Abschnitt Quelle unter Quellanbieter die Option GitLab.

- Wählen Sie, ob Sie ein vorhandenes Build-Projekt bearbeiten möchten. Wählen Sie „Bearbeiten“ und dann „Quelle“. Wählen Sie auf der Seite „Quelle bearbeiten“ unter Quellanbieter die Option GitLab.

3. Wählen Sie eine der folgenden Optionen aus:

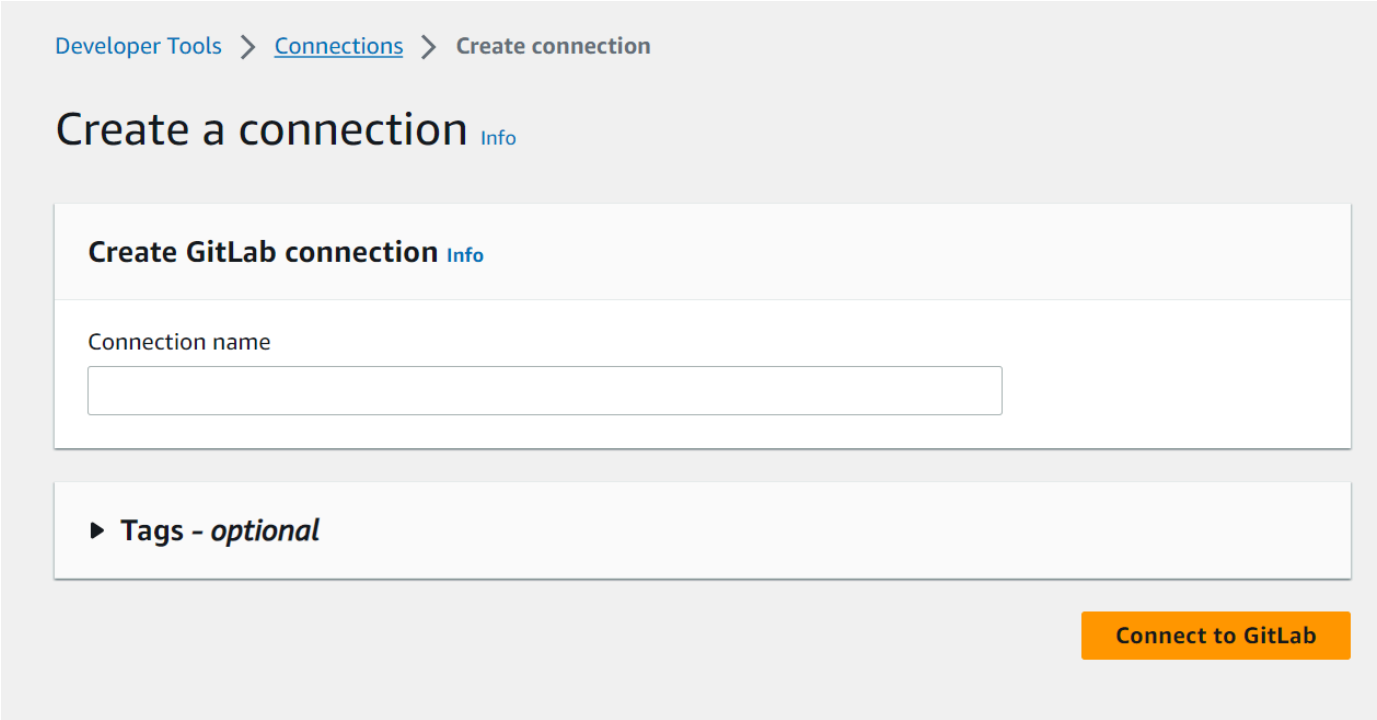
   - Wählen Sie unter Verbindung die Option Standardverbindung aus. Standardverbindung wendet eine GitLab Standardverbindung für alle Projekte an.
   - Wählen Sie unter Verbindung die Option Benutzerdefinierte Verbindung aus. Benutzerdefinierte Verbindung wendet eine benutzerdefinierte GitLab Verbindung an, die die Standardeinstellungen Ihres Kontos überschreibt.

4. Führen Sie eine der folgenden Aktionen aus:

   - Wenn Sie noch keine Verbindung zu Ihrem Anbieter hergestellt haben, wählen Sie unter Standardverbindung oder Benutzerdefinierte Verbindung die Option Neue GitLab Verbindung erstellen aus. Fahren Sie mit Schritt 5 fort, um die Verbindung herzustellen.
   - Wenn Sie unter Verbindung bereits eine Verbindung zu Ihrem Anbieter hergestellt haben, wählen Sie die Verbindung aus. Fahren Sie mit Schritt 10 fort.

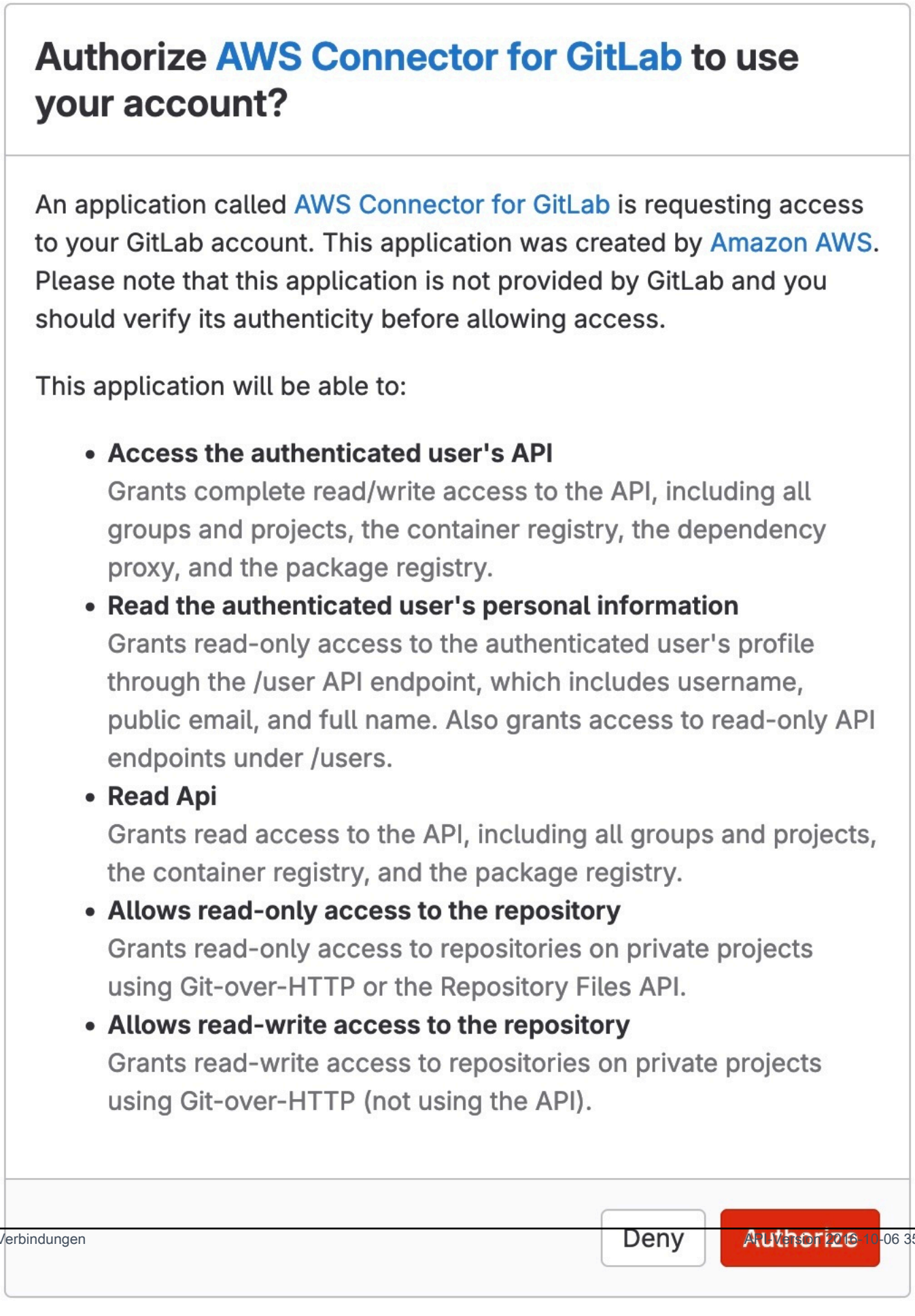
   > **Note**
   >
   > Wenn Sie das Popup-Fenster schließen, bevor eine GitLab Verbindung hergestellt wurde, müssen Sie die Seite aktualisieren.

5. Um eine Verbindung zu einem GitLab Repository herzustellen, wählen Sie unter Anbieter auswählen die Option GitLab. Geben Sie unter Connection name (Verbindungsname) den Namen für die Verbindung ein, die Sie erstellen möchten. Wählen Sie Connect GitLab.



6. Wenn die Anmeldeseite für GitLab angezeigt wird, melden Sie sich mit Ihren Anmeldeinformationen an und wählen Sie dann Anmelden aus.
7. Wenn Sie die Verbindung zum ersten Mal autorisieren, wird eine Autorisierungsseite mit einer Meldung angezeigt, in der Sie aufgefordert werden, die Verbindung für den Zugriff auf Ihr GitLab Konto zu autorisieren.

   Klicken Sie auf Authorize.


Authorize AWS Connector for GitLab to use your account?
An application called AWS Connector for GitLab is requesting access to your GitLab account. This application was created by Amazon AWS. Please note that this application is not provided by GitLab and you should verify its authenticity before allowing access.
This application will be able to:
Access the authenticated user's API
Grants complete read/write access to the API, including all groups and projects, the container registry, the dependency proxy, and the package registry.
Read the authenticated user's personal information
Grants read-only access to the authenticated user's profile through the /user API endpoint, which includes username, public email, and full name. Also grants access to read-only API endpoints under /users.
Read Api
Grants read access to the API, including all groups and projects, the container registry, and the package registry.
Allows read-only access to the repository
Grants read-only access to repositories on private projects using Git-over-HTTP or the Repository Files API.
Allows read-write access to the repository
Grants read-write access to repositories on private projects using Git-over-HTTP (not using the API).
Deny
Authorize

8. Der Browser kehrt zur Seite der Verbindungskonsole zurück. Unter GitLabVerbindungseinstellungen wird die neue Verbindung unter Verbindungsname angezeigt.
9. Wählen Sie Connect aus.

   Nachdem eine GitLab Verbindung erfolgreich hergestellt wurde, wird oben ein Erfolgsbanner angezeigt.
10. Vergewissern Sie sich, dass auf der Seite Build-Projekt erstellen in der Dropdownliste Standardverbindung oder Benutzerdefinierte Verbindung Ihr Verbindungs-ARN aufgeführt ist. Falls nicht, klicken Sie auf die Schaltfläche „Aktualisieren“, damit sie angezeigt wird.
11. Wählen Sie im Repository den Namen Ihres Projekts aus, GitLab indem Sie den Projektpfad mit dem Namespace angeben. Geben Sie beispielsweise für ein Repository auf Gruppenebene den Repository-Namen im folgenden Format ein:. `group-name/repository-name` Weitere Informationen über den Pfad und den Namespace finden Sie in dem `path_with_namespace` Feld unter https://docs.gitlab.com/ee/api/projects.html #. get-single-project Weitere Informationen zum Namespace in finden Sie unter GitLab https://docs.gitlab.com/ee/user/namespace/.

   > **Note**
   >
   > Für Gruppen in GitLab müssen Sie den Projektpfad mit dem Namespace manuell angeben. Geben Sie beispielsweise für ein Repository, das `myrepo` in einer Gruppe benannt ist`mygroup`, Folgendes ein:`mygroup/myrepo`. Sie finden den Projektpfad mit dem Namespace in der URL unter. GitLab

12. Geben Sie im Feld Quellversion — optional eine Pull-Request-ID, einen Branch, eine Commit-ID, ein Tag oder eine Referenz und eine Commit-ID ein. Weitere Informationen finden Sie unter Beispiel für eine Quellversion mit AWS CodeBuild.

    > **Note**
    >
    > Wir empfehlen, Git-Branchnamen zu wählen, die nicht wie Commit-IDs aussehen, wie zum Beispiel `811dd1ba1aba14473856cee38308caed7190c0d` oder`5392f7`. Dies hilft dir, Git-Checkout-Kollisionen mit tatsächlichen Commits zu vermeiden.

13. In Git clone depth — optional kannst du einen Shallow Clone erstellen, dessen Verlauf auf die angegebene Anzahl von Commits gekürzt ist. Wenn Sie einen vollständigen Klon erstellen möchten, wählen Sie Full (Vollständig) aus.
14. Wähle unter Build-Status — optional die Option Build-Status beim Start und Ende deiner Builds an den Quellanbieter melden, wenn du möchtest, dass der Status von Beginn und Abschluss deines Builds deinem Quellanbieter gemeldet wird.

    Um dem Quellanbieter den Buildstatus melden zu können, muss der mit dem Quellanbieter verknüpfte Benutzer Schreibzugriff auf das Repository haben. Wenn der Benutzer keinen Schreibzugriff hat, kann der Build-Status nicht aktualisiert werden. Weitere Informationen finden Sie unter Zugriff auf den Quellanbieter.

Verbindung herstellen zu GitLab (CLI)

Sie können das AWS Command Line Interface (AWS CLI) verwenden, um eine Verbindung herzustellen.

Verwenden Sie dazu den Befehl create-connection.

> **Important**
>
> Eine Verbindung, die über AWS CLI oder AWS CloudFormation erstellt wurde, hat standardmäßig `PENDING` den Status. Nachdem Sie eine Verbindung mit der CLI hergestellt haben oder verwenden Sie die Konsole AWS CloudFormation, um die Verbindung so zu bearbeiten, dass sie ihren Status festlegt`AVAILABLE`.

So stellen Sie eine Verbindung her

- Folgen Sie den Anweisungen im Benutzerhandbuch der Developer Tools-Konsole für Create a connection to GitLab (CLI).

Webhooks verwenden mit AWS CodeBuild

AWS CodeBuild unterstützt die Webhook-Integration mit GitHub GitHub Enterprise Server GitLab, GitLab Self Managed und Bitbucket.

Themen

- Bewährte Methoden für die Verwendung von Webhooks mit AWS CodeBuild
- Bitbucket-Webhook-Ereignisse
- GitHub manuelle Webhooks
- GitHub Webhook-Ereignisse
- GitLab Webhook-Ereignisse

Bewährte Methoden für die Verwendung von Webhooks mit AWS CodeBuild

Für Projekte, die öffentliche Repositorien zur Einrichtung von Webhooks verwenden, empfehlen wir die folgenden Optionen:

Filter einrichten `ACTOR_ACCOUNT_ID`

Fügen Sie `ACTOR_ACCOUNT_ID` Filter zu den Webhook-Filtergruppen Ihres Projekts hinzu, um festzulegen, welche Benutzer einen Build auslösen können. Jedes Webhook-Ereignis, an das gesendet CodeBuild wird, enthält Absenderinformationen, die die Kennung des Akteurs angeben. CodeBuild filtert die Webhooks auf der Grundlage des in den Filtern angegebenen Musters für reguläre Ausdrücke. Sie können die spezifischen Benutzer angeben, die Builds mit diesem Filter auslösen dürfen. Weitere Informationen finden Sie unter GitHub Webhook-Ereignisse und Bitbucket-Webhook-Ereignisse.

`FILE_PATH`Filter einrichten

Fügen Sie `FILE_PATH` Filter zu den Webhook-Filtergruppen Ihres Projekts hinzu, um die Dateien ein- oder auszuschließen, die bei Änderung einen Build auslösen können. Sie können beispielsweise Build-Anfragen für Änderungen an der `buildspec.yml` Datei ablehnen, indem Sie ein reguläres Ausdrucksmuster wie`^buildspec.yml$`, zusammen mit der `excludeMatchedPattern` Eigenschaft verwenden. Weitere Informationen finden Sie unter GitHub Webhook-Ereignisse und Bitbucket-Webhook-Ereignisse.

Schränken Sie die Berechtigungen für Ihre Build-IAM-Rolle ein

Durch einen Webhook ausgelöste Builds verwenden die im Projekt angegebene IAM-Servicerolle. Wir empfehlen, die Berechtigungen in der Servicerolle auf die Mindestanzahl an Berechtigungen festzulegen, die zur Ausführung des Builds erforderlich sind. Erstellen Sie beispielsweise in einem Test- und Bereitstellungsszenario ein Projekt zum Testen und ein anderes Projekt für die Bereitstellung. Das Testprojekt akzeptiert Webhook-Builds aus dem Repository, gewährt jedoch keine Schreibberechtigungen für Ihre Ressourcen. Das Bereitstellungsprojekt gewährt

Schreibberechtigungen für Ihre Ressourcen, und der Webhook-Filter ist so konfiguriert, dass nur vertrauenswürdige Benutzer Builds auslösen können.

Verwenden Sie eine Inline- oder eine in Amazon S3 gespeicherte Buildspec

Wenn Sie Ihre Buildspec-Datei direkt im Projekt selbst definieren oder die Buildspec-Datei in einem Amazon S3 S3-Bucket speichern, ist die Buildspec-Datei nur für den Projekteigentümer sichtbar. Dadurch wird verhindert, dass Pull-Anfragen Codeänderungen an der Buildspec-Datei vornehmen und unerwünschte Builds auslösen. Weitere Informationen finden Sie unter ProjectSource.buildspec in der API-Referenz. CodeBuild

Bitbucket-Webhook-Ereignisse

Sie können Webhook-Filtergruppen verwenden, um anzugeben, welche Bitbucket-Webhook-Ereignisse einen Build auslösen. Du kannst beispielsweise angeben, dass ein Build nur bei Änderungen an bestimmten Branches ausgelöst wird.

Sie können eine oder mehrere Webhook-Filtergruppen erstellen, um anzugeben, welche Webhook-Ereignisse einen Build auslösen. Ein Build wird ausgelöst, wenn eine Filtergruppe als wahr ausgewertet wird. Dies ist der Fall, wenn alle Filter in der Gruppe den Wert true ergeben. Beim Erstellen einer Filtergruppe geben Sie Folgendes an:

Ein Ereignis

Für Bitbucket kannst du eines oder mehrere der folgenden Ereignisse wählen:

- `PUSH`
- `PULL_REQUEST_CREATED`
- `PULL_REQUEST_UPDATED`
- `PULL_REQUEST_MERGED`
- `PULL_REQUEST_CLOSED`

Der Webhook-Ereignistyp befindet sich im Header des Feldes `X-Event-Key`. Aus der folgende Tabelle geht die Zuordnung der `X-Event-Key`-Header-Werte zu Ereignistypen hervor.

> **Note**
>
> Sie müssen das `merged`-Ereignis in Ihren Bitbucket-Webhook-Einstellungen aktivieren, wenn Sie eine Webhook-Filtergruppe erstellen, die den `PULL_REQUEST_MERGED`-Ereignistyp verwendet. Du musst das `declined` Ereignis auch in deiner Bitbucket-

Webhook-Einstellung aktivieren, wenn du eine Webhook-Filtergruppe erstellst, die den Ereignistyp verwendet. PULL_REQUEST_CLOSED

X-Event-Key -Header-Wert	Ereignistyp
repo:push	PUSH
pullrequest:created	PULL_REQUEST_CREATED
pullrequest:updated	PULL_REQUEST_UPDATED
pullrequest:fulfilled	PULL_REQUEST_MERGED
pullrequest:rejected	PULL_REQUEST_CLOSED

Denn PULL_REQUEST_MERGED wenn ein Pull-Request mit der Squash-Strategie zusammengeführt und der Pull-Request-Branch geschlossen wird, ist der ursprüngliche Pull-Request-Commit nicht mehr vorhanden. In diesem Fall enthält die CODEBUILD_WEBHOOK_MERGE_COMMIT Umgebungsvariable den Bezeichner des gequetschten Merge-Commits.

Ein oder mehrere optionale Filter

Verwenden Sie einen regulären Ausdruck, um einen Filter anzugeben. Damit ein Ereignis einen Build auslöst, muss jeder Filter innerhalb der Gruppe, die diesem Ereignis zugeordnet ist, als wahr ausgewertet werden.

ACTOR_ACCOUNT_ID(ACTOR_IDin der Konsole)

Ein Webhook-Ereignis löst einen Build aus, wenn eine Bitbucket-Konto-ID dem Muster für reguläre Ausdrücke entspricht. Dieser Wert wird in der Eigenschaft account_id des Objekts actor in der Webhook-Filternutzlast angezeigt.

HEAD_REF

Ein Webhook-Ereignis löst einen Build aus, wenn die Head-Referenz mit dem Muster für reguläre Ausdrücke übereinstimmt (zum Beispiel undrefs/heads/branch-name). refs/tags/tag-name Ein HEAD_REF-Filter wertet den Git-Referenznamen für den Branch oder Tag aus. Die Branch- oder Tag-Name wird im Feld name des Objekts new im Objekt push der

Webhook-Nutzlast angezeigt. Bei Pull-Anforderungsereignissen wird der Branch-Name im Feld `name` im Objekt `branch` des Objekts `source` in der Webhook-Nutzlast angezeigt.

`BASE_REF`

Ein Webhook-Ereignis löst einen Build aus, wenn die Basisreferenz mit dem Muster des regulären Ausdrucks übereinstimmt. Ein `BASE_REF`-Filter kann nur für Pull-Anfrageereignisse verwendet werden (z. B. `refs/heads/branch-name`). Ein `BASE_REF`-Filter wertet den Git-Referenznamen für die Verzweigung aus. Der Branch-Name wird im Feld `name` des Objekts `branch` im Objekt `destination` in der Webhook-Nutzlast angezeigt.

`FILE_PATH`

Ein Webhook löst einen Build aus, wenn der Pfad einer geänderten Datei dem Muster für reguläre Ausdrücke entspricht.

`COMMIT_MESSAGE`

Ein Webhook löst einen Build aus, wenn die Head-Commit-Nachricht dem Muster des regulären Ausdrucks entspricht.

`WORKFLOW_NAME`

Ein Webhook löst einen Build aus, wenn der Workflow-Name mit dem Muster des regulären Ausdrucks übereinstimmt.

> **Note**
>
> Sie finden die Webhook-Nutzlast in den Webhook-Einstellungen in Ihrem Bitbucket-Repository.

Themen

- Filtern von BitBucket-Webhook-Ereignissen (Konsole)
- Filtern von BitBucket-Webhook-Ereignissen (SDK)
- Filtern von Bitbucket-Webhook-Ereignissen (AWS CloudFormation)

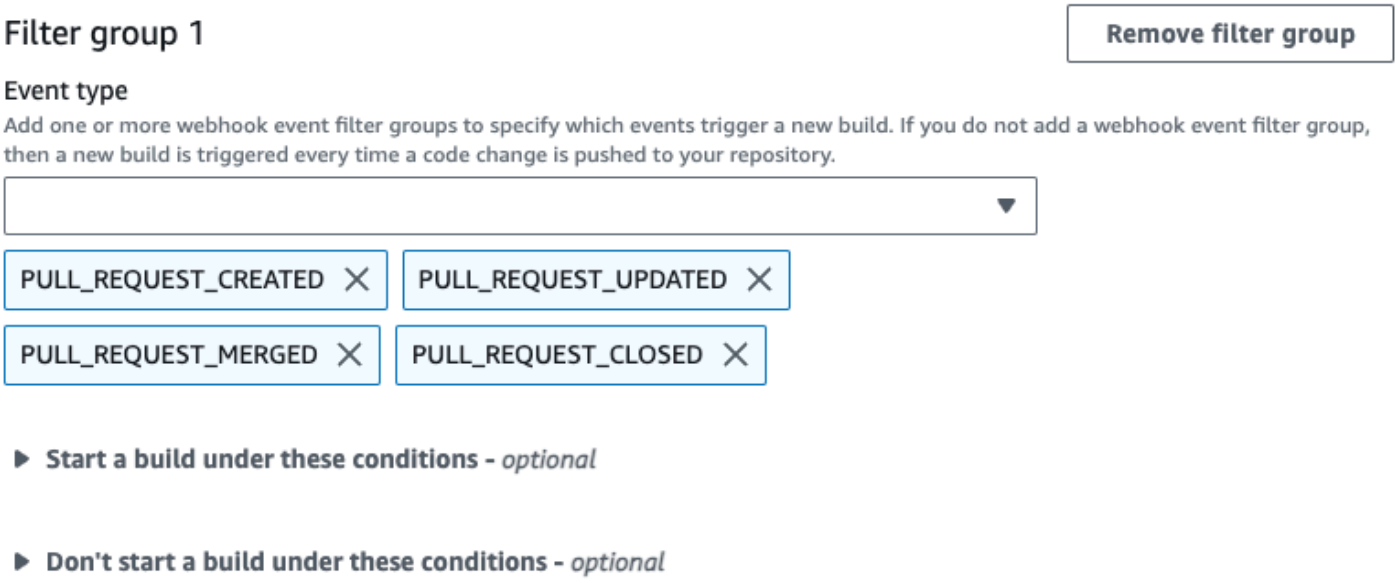
Filtern von BitBucket-Webhook-Ereignissen (Konsole)

Um Webhook-Ereignisse AWS Management Console zu filtern, gehen Sie wie folgt vor:

1. Wählen Sie beim Erstellen Ihres Projekts Rebuild every time a code change is pushed to this repository (Erneut erstellen, wenn eine Codeänderung an dieses Repository übergeben wird) aus.
2. Wählen Sie unter Event type (Ereignistyp) eines oder mehrere Ereignisse aus.
3. Wenn Sie Fälle filtern möchten, in denen ein Ereignis einen Build auslöst, fügen Sie unter Start a build under these conditions (Unter diesen Bedingungen Build starten) einen oder mehrere optionale Filter hinzu.
4. Wenn Sie Fälle filtern möchten, in denen kein Ereignis ausgelöst wird, fügen Sie unter Don't start a build under these conditions (Unter diesen Bedingungen keinen Build starten) einen oder mehrere optionale Filter hinzu.
5. Wählen Sie Add filter group (Filtergruppe hinzufügen) aus, um eine weitere Filtergruppe hinzuzufügen.

Weitere Informationen finden Sie unter Erstellen Sie ein Build-Projekt (Konsole) und WebhookFilterin der AWS CodeBuild API-Referenz.

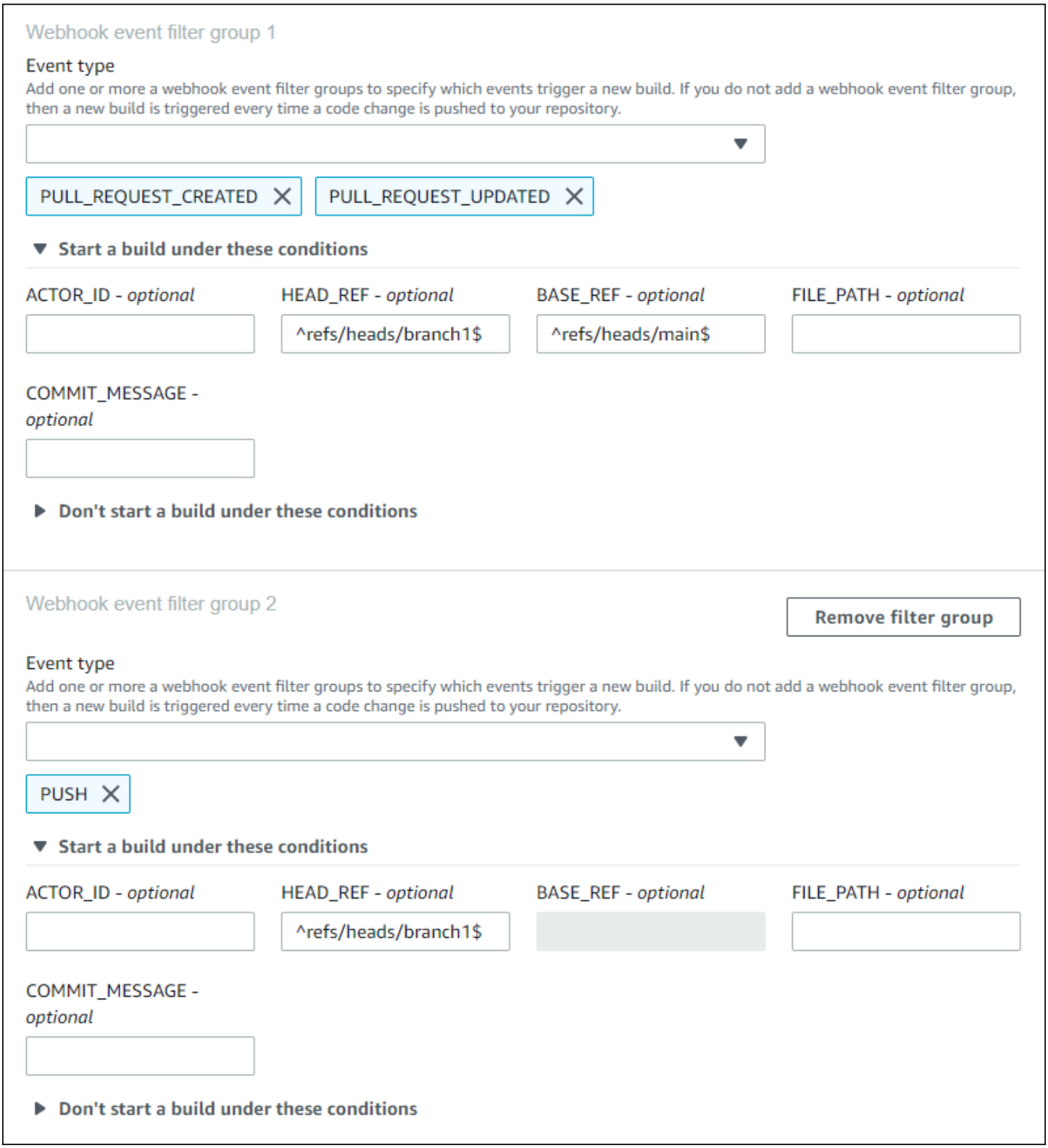
In diesem Beispiel löst eine Webhook-Filtergruppe nur für Pull-Anfragen einen Build aus:



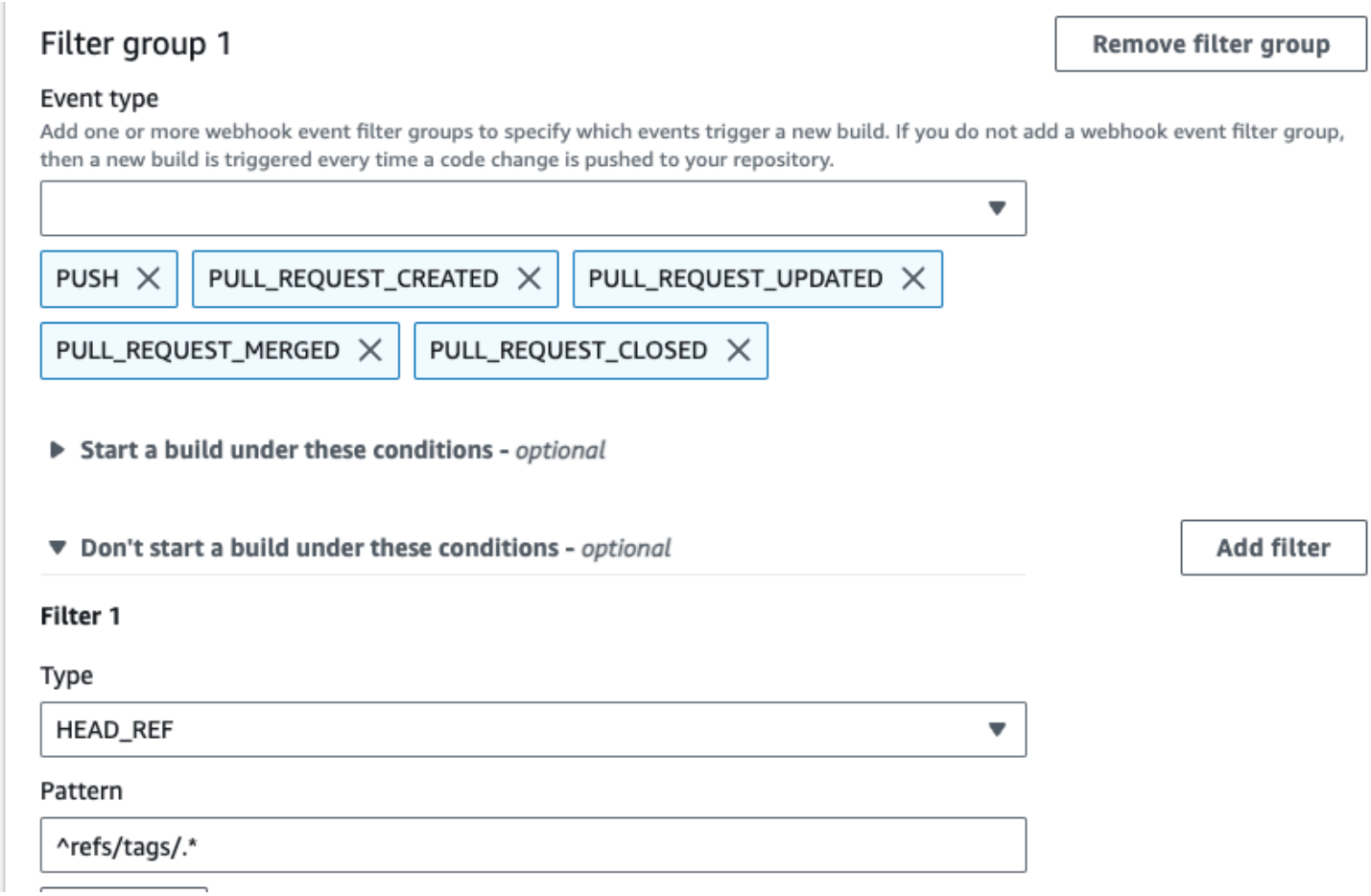
Bei Verwendung eines Beispiels mit zwei Filtergruppen wird ein Build ausgelöst, wenn mindestens eine Gruppe als wahr ausgewertet wird:

- Die erste Filtergruppe gibt Pull-Anfragen an, die in Verzweigungen mit Git-Referenznamen, die dem regulären Ausdruck `^refs/heads/main$` entsprechen, und mit Kopfreferenzen, die `^refs/heads/branch1!` entsprechen, erstellt oder aktualisiert werden.

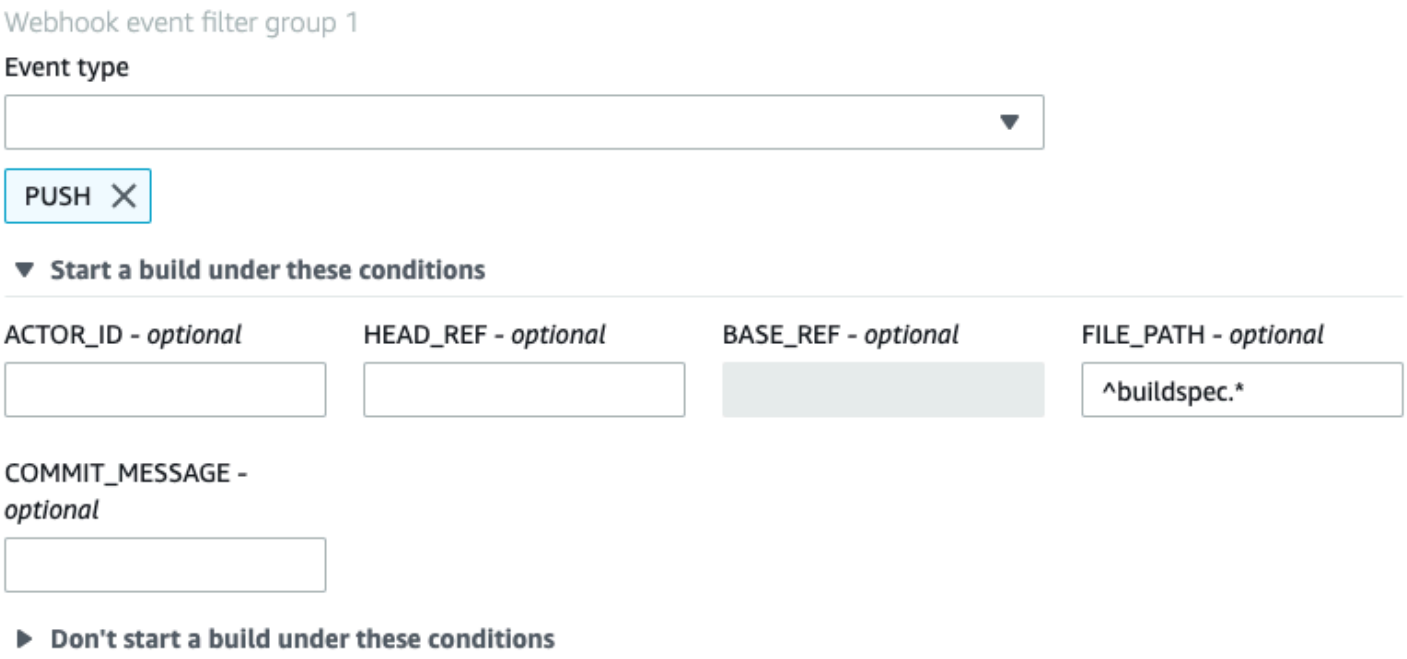
- Die zweite Filtergruppe gibt Push-Anfragen in Verzweigungen mit Git-Referenznamen an, die dem regulären Ausdruck `^refs/heads/branch1$` entsprechen.



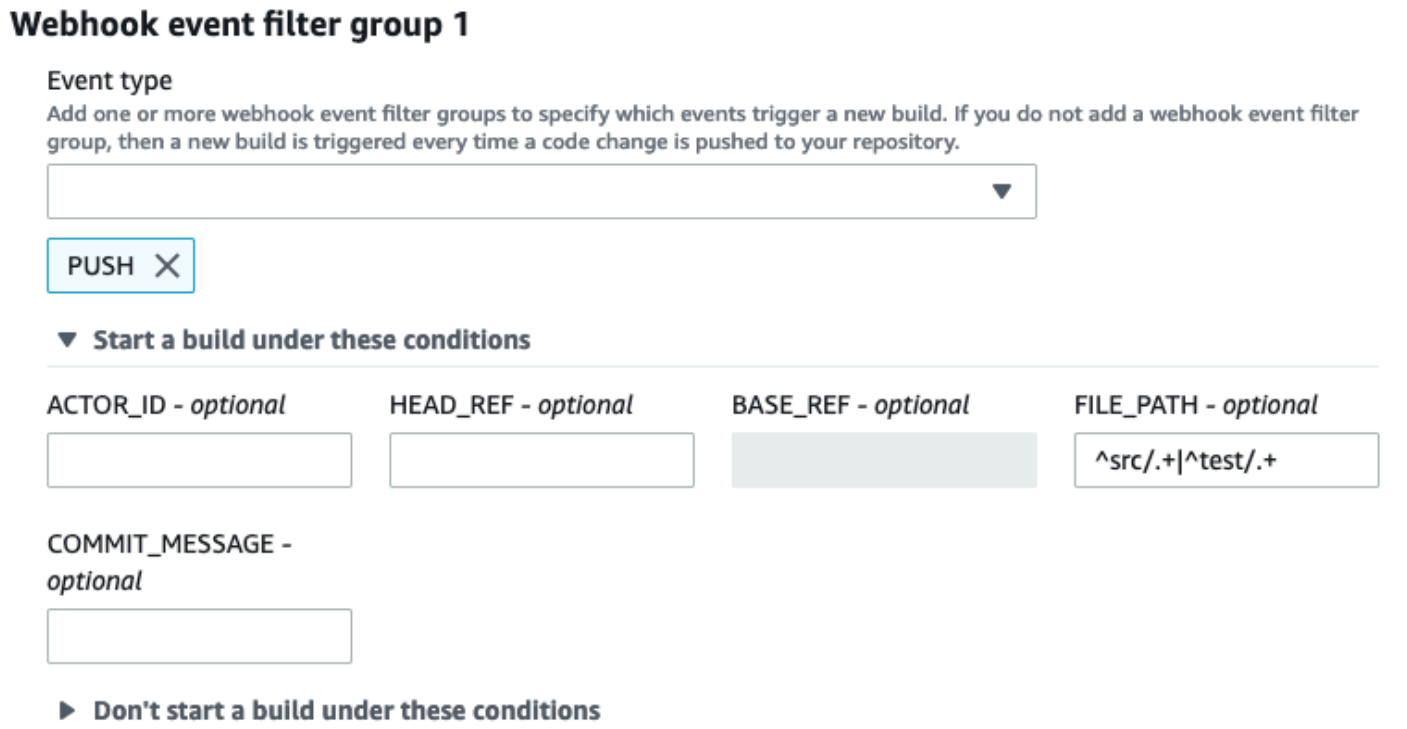
In diesem Beispiel löst eine Webhook-Filtergruppe einen Build für alle Anfragen mit Ausnahme von Tag-Ereignissen aus.



In diesem Beispiel löst eine Webhook-Filtergruppe nur einen Build aus, wenn Dateien geändert werden, deren Namen dem regulären Ausdruck `^buildspec.*` entsprechen.

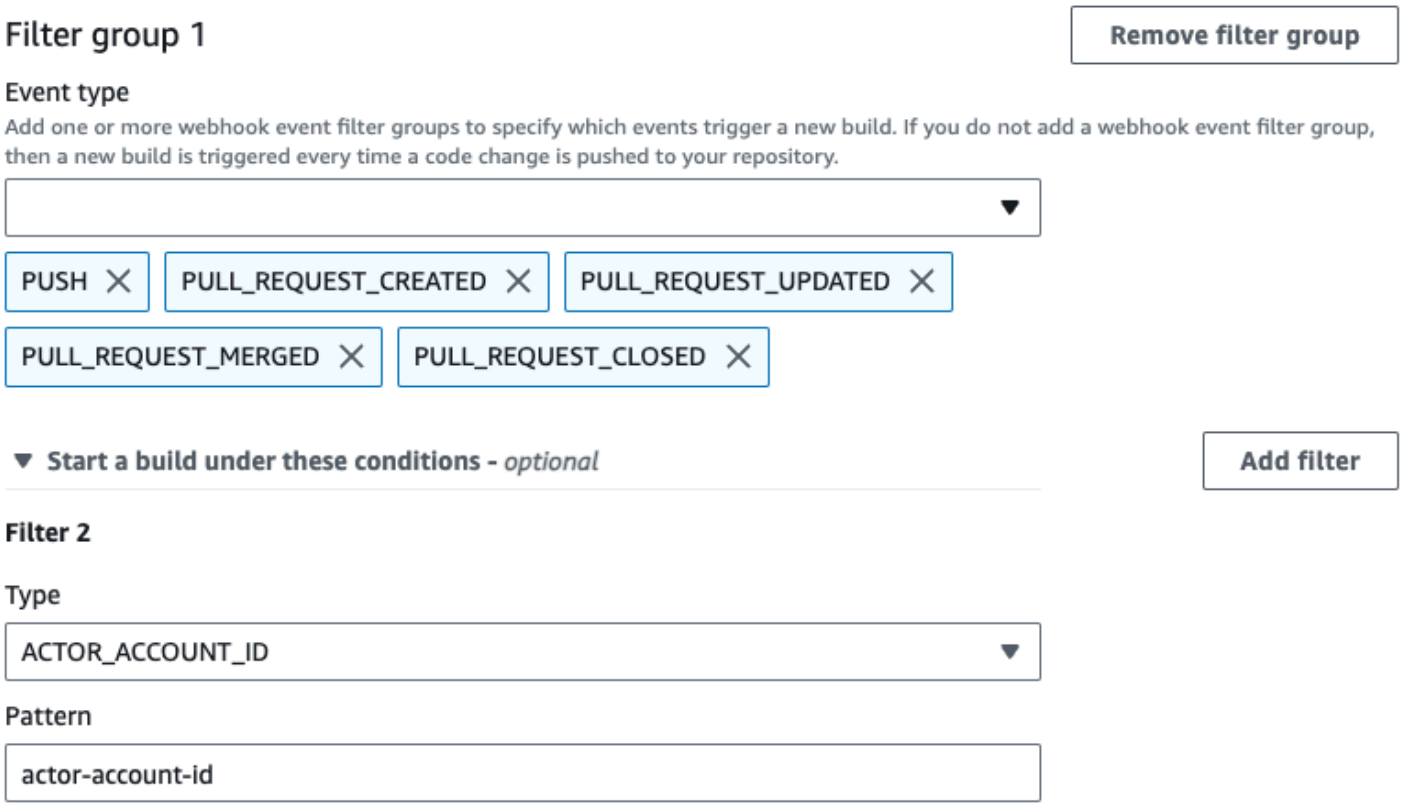


In diesem Beispiel löst eine Webhook-Filtergruppe nur dann einen Build aus, wenn Dateien in `test` Ordnern `src` oder geändert werden.

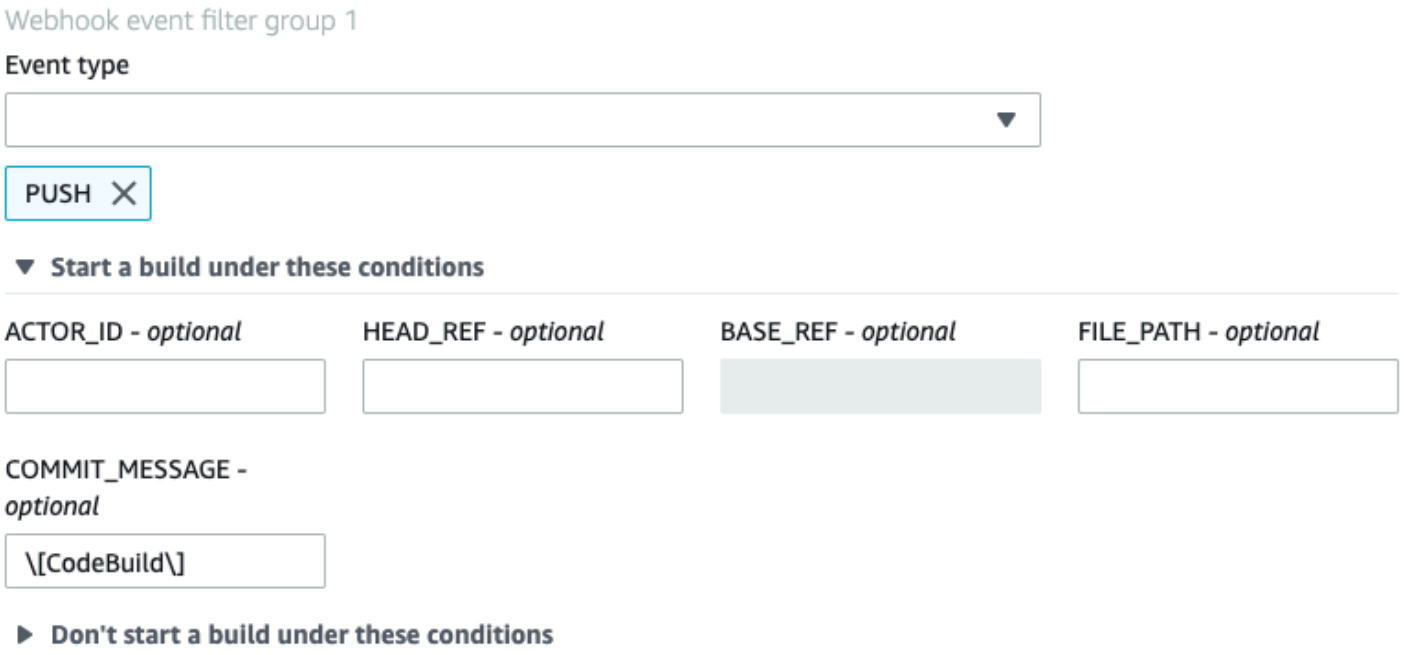


In diesem Beispiel löst eine Webhook-Filtergruppe nur einen Build aus, wenn eine Änderung von einem Bitbucket-Benutzer vorgenommen wird, der nicht über eine Konto-ID verfügt, die dem regulären Ausdruck `actor-account-id` entspricht.

> **Note**
>
> Informationen dazu, wie Sie Ihre Bitbucket-Konto-ID herausfinden können, finden Sie unter https://api.bitbucket.org/2.0/users/*Benutzername*, wobei *Benutzername* Ihr Bitbucket-Benutzername ist.



In diesem Beispiel löst eine Webhook-Filtergruppe einen Build für ein Push-Ereignis aus, wenn die Head-Commit-Nachricht mit dem regulären Ausdruck `\[CodeBuild\]` übereinstimmt.



Filtern von BitBucket-Webhook-Ereignissen (SDK)

Um das AWS CodeBuild SDK zum Filtern von Webhook-Ereignissen zu verwenden, verwenden Sie das `filterGroups` Feld in der Anforderungssyntax der `CreateWebhook` oder `UpdateWebhook` API-Methoden. Weitere Informationen finden Sie WebhookFilterin der CodeBuild API-Referenz.

Um einen Webhook-Filter zu erstellen, der nur für Pull-Anfragen einen Build auslöst, fügen Sie Folgendes in die Anfragesyntax ein:

```
"filterGroups": [
  [
    {
      "type": "EVENT",
      "pattern": "PULL_REQUEST_CREATED, PULL_REQUEST_UPDATED, PULL_REQUEST_MERGED,
 PULL_REQUEST_CLOSED"
    }
  ]
]
```

Um einen Webhook-Filter zu erstellen, der nur für die angegebenen Verzweigungen einen Build auslöst, verwenden Sie den Parameter `pattern`, um einen regulären Ausdruck zum Filtern von Verzweigungsnamen anzugeben. Bei Verwendung eines Beispiels mit zwei Filtergruppen wird ein Build ausgelöst, wenn mindestens eine Gruppe als wahr ausgewertet wird:

- Die erste Filtergruppe gibt Pull-Anfragen an, die in Verzweigungen mit Git-Referenznamen, die dem regulären Ausdruck `^refs/heads/main$` entsprechen, und mit Kopfreferenzen, die `^refs/heads/myBranch$` entsprechen, erstellt oder aktualisiert werden.
- Die zweite Filtergruppe gibt Push-Anfragen in Verzweigungen mit Git-Referenznamen an, die dem regulären Ausdruck `^refs/heads/myBranch$` entsprechen.

```
"filterGroups": [
  [
    {
      "type": "EVENT",
      "pattern": "PULL_REQUEST_CREATED, PULL_REQUEST_UPDATED, PULL_REQUEST_CLOSED"
    },
    {
      "type": "HEAD_REF",
      "pattern": "^refs/heads/myBranch$"
    },
    {
      "type": "BASE_REF",
      "pattern": "^refs/heads/main$"
    }
  ],
  [
    {
      "type": "EVENT",
      "pattern": "PUSH"
    },
    {
      "type": "HEAD_REF",
      "pattern": "^refs/heads/myBranch$"
    }
  ]
]
```

Sie können den Parameter `excludeMatchedPattern` verwenden, um anzugeben, welche Ereignisse keinen Build auslösen. In diesem Beispiel wird für alle Anfragen mit Ausnahme von Tag-Ereignissen ein Build ausgelöst.

```
"filterGroups": [
  [
    {
      "type": "EVENT",
```

```
      "pattern": "PUSH, PULL_REQUEST_CREATED, PULL_REQUEST_UPDATED,
 PULL_REQUEST_MERGED, PULL_REQUEST_CLOSED"
    },
    {
      "type": "HEAD_REF",
      "pattern": "^refs/tags/.*",
      "excludeMatchedPattern": true
    }
  ]
]
```

Sie können einen Filter erstellen, der nur einen Build auslöst, wenn ein Bitbucket-Benutzer mit der Konto-ID `actor-account-id` eine Änderung vornimmt.

> **Note**
>
> Informationen dazu, wie Sie Ihre Bitbucket-Konto-ID herausfinden können, finden Sie unter https://api.bitbucket.org/2.0/users/*Benutzername*, wobei *Benutzername* Ihr Bitbucket-Benutzername ist.

```
"filterGroups": [
  [
    {
      "type": "EVENT",
      "pattern": "PUSH, PULL_REQUEST_CREATED, PULL_REQUEST_UPDATED,
 PULL_REQUEST_MERGED, PULL_REQUEST_CLOSED"
    },
    {
      "type": "ACTOR_ACCOUNT_ID",
      "pattern": "actor-account-id"
    }
  ]
]
```

Sie können einen Filter erstellen, der nur einen Build auslöst, wenn Dateien geändert werden, deren Namen dem regulären Ausdruck in dem Argument `pattern` entsprechen. In diesem Beispiel gibt die Filtergruppe an, dass nur ein Build ausgelöst wird, wenn Dateien geändert werden, deren Name dem regulären Ausdruck `^buildspec.*` entspricht.

```
"filterGroups": [
```

```
  [
    {
      "type": "EVENT",
      "pattern": "PUSH"
    },
    {
      "type": "FILE_PATH",
      "pattern": "^buildspec.*"
    }
  ]
]
```

In diesem Beispiel gibt die Filtergruppe an, dass ein Build nur ausgelöst wird, wenn Dateien in `test` Ordnern `src` oder geändert werden.

```
"filterGroups": [
    [
        {
            "type": "EVENT",
            "pattern": "PUSH"
        },
        {
            "type": "FILE_PATH",
            "pattern": "^src/.+|^test/.+"
        }
    ]
]
```

Sie können einen Filter erstellen, der einen Build nur dann auslöst, wenn die Head-Commit-Nachricht mit dem regulären Ausdruck im Musterargument übereinstimmt. In diesem Beispiel gibt die Filtergruppe an, dass ein Build nur dann ausgelöst wird, wenn die Head-Commit-Nachricht des Push-Ereignisses mit dem regulären Ausdruck `\[CodeBuild\]` übereinstimmt.

```
  "filterGroups": [
    [
      {
        "type": "EVENT",
        "pattern": "PUSH"
      },
      {
        "type": "COMMIT_MESSAGE",
        "pattern": "\[CodeBuild\]"
```

```
      }
    ]
  ]
```

Filtern von Bitbucket-Webhook-Ereignissen (AWS CloudFormation)

Um eine AWS CloudFormation Vorlage zum Filtern von Webhook-Ereignissen zu verwenden, verwenden Sie die `FilterGroups` Eigenschaft des AWS CodeBuild Projekts. Mit dem folgenden in YAML formatierten Teil einer AWS CloudFormation -Vorlage werden zwei Filtergruppen erstellt. Diese lösen zusammen einen Build aus, wenn mindestens eine Gruppe als wahr ausgewertet wird.

- Die erste Filtergruppe gibt an, dass Pull-Anfragen in Verzweigungen mit Git-Referenznamen, die dem regulären Ausdruck `^refs/heads/main$` entsprechen, von einem Bitbucket-Benutzer, der über keine Konto-ID `12345` verfügt, erstellt oder aktualisiert werden.
- Die zweite Filtergruppe gibt an, dass Push-Anfragen in Verzweigungen mit Git-Referenznamen erstellt werden, die dem regulären Ausdruck `^refs/heads/.*` entsprechen.
- Die dritte Filtergruppe gibt eine Push-Anforderung mit einer Head Commit-Nachricht an, die dem regulären Ausdruck `\[CodeBuild\]` entspricht.

```
CodeBuildProject:
  Type: AWS::CodeBuild::Project
  Properties:
    Name: MyProject
    ServiceRole: service-role
    Artifacts:
      Type: NO_ARTIFACTS
    Environment:
      Type: LINUX_CONTAINER
      ComputeType: BUILD_GENERAL1_SMALL
      Image: aws/codebuild/standard:5.0
    Source:
      Type: BITBUCKET
      Location: source-location
    Triggers:
      Webhook: true
      FilterGroups:
        - - Type: EVENT
            Pattern: PULL_REQUEST_CREATED,PULL_REQUEST_UPDATED
          - Type: BASE_REF
            Pattern: ^refs/heads/main$
```

```
            ExcludeMatchedPattern: false
          - Type: ACTOR_ACCOUNT_ID
            Pattern: 12345
            ExcludeMatchedPattern: true
        - - Type: EVENT
            Pattern: PUSH
          - Type: HEAD_REF
            Pattern: ^refs/heads/.*
          - Type: FILE_PATH
            Pattern: READ_ME
            ExcludeMatchedPattern: true
        - - Type: EVENT
            Pattern: PUSH
          - Type: COMMIT_MESSAGE
            Pattern: \[CodeBuild\]
          - Type: FILE_PATH
            Pattern: ^src/.+|^test/.+
```

GitHub manuelle Webhooks

Sie können manuelle GitHub Webhooks so konfigurieren, dass CodeBuild nicht automatisch versucht wird, darin einen Webhook zu erstellen. GitHub CodeBuild gibt im Rahmen des Aufrufs zur Erstellung des Webhooks eine Payload-URL zurück und kann verwendet werden, um den darin enthaltenen Webhook manuell zu erstellen. GitHub Auch wenn CodeBuild das Erstellen eines Webhooks in Ihrem GitHub Konto nicht erlaubt ist, können Sie dennoch manuell einen Webhook für Ihr Build-Projekt erstellen.

Gehen Sie wie folgt vor, um einen GitHub manuellen Webhook zu erstellen.

Um einen GitHub manuellen Webhook zu erstellen

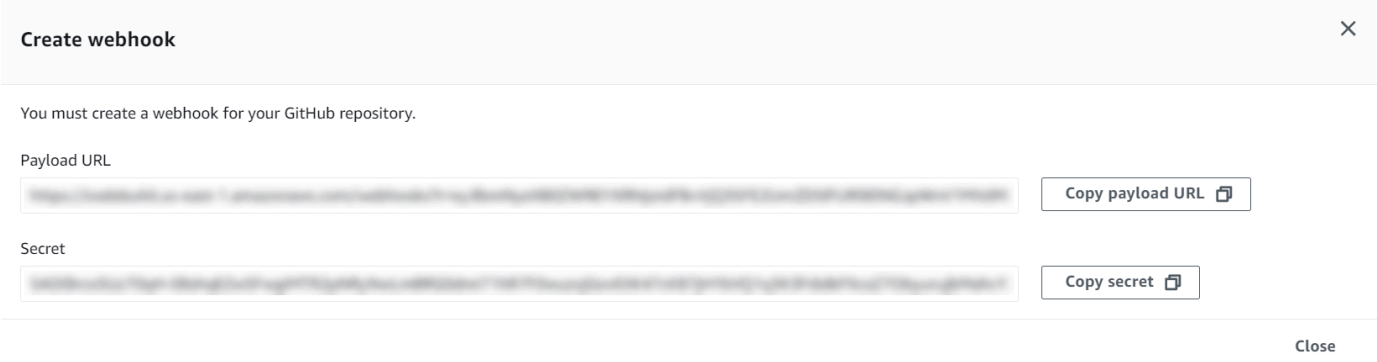
1. Öffnen Sie die AWS CodeBuild Konsole unter https://console.aws.amazon.com/codesuite/codebuild/home.
2. Erstellen Sie ein Build-Projekt. Weitere Informationen finden Sie unter Erstellen Sie ein Build-Projekt (Konsole) und Ausführen eines Build (Konsole).
   - In Source (Quelle):
     - Wählen Sie als Quellanbieter die Option GitHub.
     - Wählen Sie unter Repository die Option Repository in meinem GitHub Konto aus.

- Geben Sie als Repository URL (Repository-URL) die URL **`https://github.com/user-name/repository-name`** ein.

- Gehen Sie unter Webhook-Ereignisse in der Primärquelle wie folgt vor:
  - Wählen Sie für Webhook — optional die Option Jedes Mal neu erstellen, wenn eine Codeänderung in dieses Repository übertragen wird.
  - Wählen Sie Zusätzliche Konfiguration und für Manuelle Erstellung — optional die Option Manuell einen Webhook für dieses Repository in GitHub der Konsole erstellen aus. .

> **Note**
>
> Eine zusätzliche Konfiguration ist nicht verfügbar, wenn Sie GitHub Enterprise als Quellanbieter verwenden.

3. Fahren Sie mit den Standardwerten fort und wählen Sie dann Build-Projekt erstellen. Notieren Sie sich die Werte Payload-URL und Secret, da Sie diese später verwenden werden.



4. Öffnen Sie die GitHub Konsole unter `https://github.com/user-name/repository-name/settings/hooks` und wählen Sie Webhook hinzufügen.

- Geben Sie für Payload-URL den Payload-URL-Wert ein, den Sie sich zuvor notiert haben.
- Wählen Sie für Inhaltstyp die Option application/json aus.
- Geben Sie für Secret den Wert Secret ein, den Sie sich zuvor notiert haben.
- Konfigurieren Sie die einzelnen Ereignisse, an die eine Webhook-Payload gesendet werden soll. CodeBuild Für welche Ereignisse möchten Sie diesen Webhook auslösen? , wählen Sie Lassen Sie mich einzelne Ereignisse auswählen und wählen Sie dann aus den folgenden Ereignissen: Pushes, Pull Requests und Releases. Wenn Sie Builds für `WORKFLOW_JOB_QUEUED` Ereignisse starten möchten, wählen Sie Workflow-Jobs. Weitere Informationen zu GitHub Actions-Runnern finden Sie unterRichten Sie selbst gehostete

GitHub Action-Runner ein in AWS CodeBuild. Weitere Informationen zu Ereignistypen, die von unterstützt werden CodeBuild, finden Sie unterGitHub Webhook-Ereignisse.

5. Wählen Sie Webhook hinzufügen aus.

GitHub Webhook-Ereignisse

Sie können Webhook-Filtergruppen verwenden, um anzugeben, welche GitHub Webhook-Ereignisse einen Build auslösen. Sie können beispielsweise angeben, dass ein Build nur bei Änderungen an bestimmten Branches ausgelöst wird.

Sie können eine oder mehrere Webhook-Filtergruppen erstellen, um anzugeben, welche Webhook-Ereignisse einen Build auslösen. Ein Build wird ausgelöst, wenn eine Filtergruppe als wahr ausgewertet wird. Dies ist der Fall, wenn alle Filter in der Gruppe den Wert true ergeben. Beim Erstellen einer Filtergruppe geben Sie Folgendes an:

Ein Ereignis

Für GitHub können Sie eines oder mehrere der folgenden Ereignisse auswählen:`PUSH`,`PULL_REQUEST_CREATED`,`PULL_REQUEST_UPDATED`,`PULL_REQUEST_REOPENED`,`PU`
`PULL_REQUEST_CLOSEDRELEASED`,`PRERELEASED`, und`WORKFLOW_JOB_QUEUED`. Der Webhook-Ereignistyp ist im `X-GitHub-Event`-Header in der Webhook-Nutzlast zu finden. Im `X-GitHub-Event`-Header befindet sich möglicherweise `pull_request` oder `push`. Bei einem Pull-Anforderungstyp befindet sich der Typ im Feld `action` der Webhook-Ereignisnutzlast. Aus der folgenden Tabelle geht die Zuordnung der `X-GitHub-Event`-Header-Werte und der Werte im Feld `action` der Webhook-Pull-Anforderungsnutzlast zu den verfügbaren Ereignistypen hervor.

`X-GitHub-Event` -Header-Wert	**`action`**-Wert der Webhook-Ereignisnutzlast	Ereignistyp
`pull_request`	`opened`	`PULL_REQUEST_CREATED`
`pull_request`	`reopened`	`PULL_REQUEST_REOPENED`
`pull_request`	`synchronize`	`PULL_REQUEST_UPDATED`

`X-GitHub-Event` -Header-Wert	**`action`**-Wert der Webhook-Ereignisnutzlast	Ereignistyp
`pull_request`	`closed` und das `merged`-Feld sind `true`	`PULL_REQUEST_MERGED`
`pull_request`	`closed` und das `merged`-Feld sind `false`	`PULL_REQUEST_CLOSED`
`push`	–	`PUSH`
`release`	veröffentlicht	`RELEASED`
`release`	vorveröffentlicht	`PRERELEASED`
`workflow_job`	queued	`WORKFLOW_JOB_QUEUED`

> **Note**
>
> Der `PULL_REQUEST_REOPENED` Ereignistyp kann nur mit GitHub GitHub Enterprise Server verwendet werden. Der `WORKFLOW_JOB_QUEUED` Ereignistyp `RELEASEDPRERELEASED`, und kann GitHub nur mit verwendet werden. Weitere Informationen zu `WORKFLOW_JOB_QUEUED` finden Sie unter Tutorial: Einen CodeBuild selbst GitHub gehosteten Actions-Runner konfigurieren.

Ein oder mehrere optionale Filter

Verwenden Sie einen regulären Ausdruck, um einen Filter anzugeben. Damit ein Ereignis einen Build auslöst, muss jeder Filter innerhalb der Gruppe, die diesem Ereignis zugeordnet ist, als wahr ausgewertet werden.

`ACTOR_ACCOUNT_ID`(`ACTOR_ID`in der Konsole)

Ein Webhook-Ereignis löst einen Build aus, wenn eine GitHub oder GitHub Enterprise Server-Konto-ID dem regulären Ausdrucksmuster entspricht. Dieser Wert befindet sich in der Eigenschaft `id` des Objekts `sender` in der Webhook-Nutzlast.

`HEAD_REF`

Ein Webhook-Ereignis löst einen Build aus, wenn die Hauptreferenz mit dem Muster eines regulären Ausdrucks übereinstimmt (z. B. `refs/heads/branch-name` oder`refs/tags/tag-name`). Bei einem Push-Ereignis ist der Referenzname in der Eigenschaft `ref` in der Webhook-Nutzlast zu finden. Bei Pull-Anforderungsereignissen befindet sich der Branch-Name in der Eigenschaft `ref` des Objekts `head` in der Webhook-Nutzlast.

`BASE_REF`

Ein Webhook-Ereignis löst einen Build aus, wenn die Basisreferenz mit dem Muster eines regulären Ausdrucks übereinstimmt (z. B.`refs/heads/branch-name`). Ein `BASE_REF`-Filter kann nur für Pull-Anforderungsereignisse verwendet werden. Der Branch-Name befindet sich in der Eigenschaft `ref` des Objekts `base` in der Webhook-Nutzlast.

`FILE_PATH`

Ein Webhook löst einen Build aus, wenn der Pfad einer geänderten Datei dem Muster regulärer Ausdrücke entspricht. Ein `FILE_PATH` Filter kann für GitHub Push- und Pull-Request-Ereignisse sowie für GitHub Enterprise Server-Push-Ereignisse verwendet werden. Er kann nicht mit GitHub Enterprise Server-Pull-Request-Ereignissen verwendet werden.

`COMMIT_MESSAGE`

Ein Webhook löst einen Build aus, wenn die Head-Commit-Nachricht dem Muster eines regulären Ausdrucks entspricht. Ein `COMMIT_MESSAGE` Filter kann für GitHub Push- und Pull-Request-Ereignisse sowie für GitHub Enterprise Server-Push-Ereignisse verwendet werden. Er kann nicht mit GitHub Enterprise Server-Pull-Request-Ereignissen verwendet werden.

`TAG_NAME`

Ein Webhook löst einen Build aus, wenn der Tag-Name der Version mit dem Muster des regulären Ausdrucks übereinstimmt. Ein `TAG_NAME` Filter kann für GitHub veröffentlichte und vorab veröffentlichte Anforderungsereignisse verwendet werden.

`RELEASE_NAME`

Ein Webhook löst einen Build aus, wenn der Versionsname dem Muster eines regulären Ausdrucks entspricht. Ein `RELEASE_NAME` Filter kann für GitHub veröffentlichte und vorab veröffentlichte Anforderungsereignisse verwendet werden.

`WORKFLOW_NAME`

Ein Webhook löst einen Build aus, wenn der Workflow-Name dem Muster des regulären Ausdrucks entspricht. Ein `WORKFLOW_NAME` Filter kann für Ereignisse in der Warteschlange von Aufträgen im GitHub Aktionsworkflow verwendet werden.

> **Note**
>
> Sie finden die Webhook-Payload in den Webhook-Einstellungen Ihres Repositorys. GitHub

Themen

- GitHub Webhook-Ereignisse filtern (Konsole)
- GitHub Webhook-Ereignisse filtern (SDK)
- GitHub Webhook-Ereignisse filtern ()AWS CloudFormation

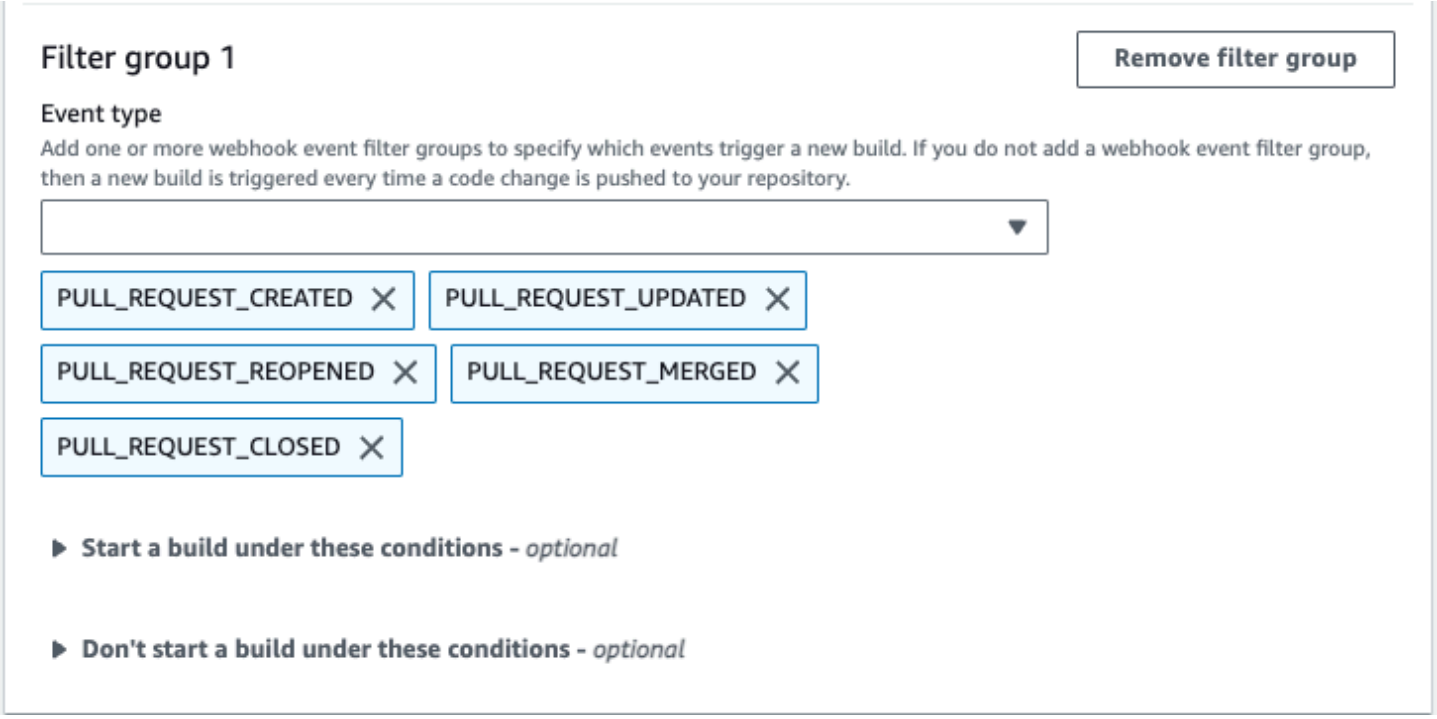
GitHub Webhook-Ereignisse filtern (Konsole)

Wählen Sie unter Webhook-Ereignisse der Primärquelle die folgenden Optionen aus. Dieser Abschnitt ist nur verfügbar, wenn Sie in Mein GitHub Konto für das Quell-Repository die Option Repository ausgewählt haben.

1. Wählen Sie beim Erstellen Ihres Projekts Rebuild every time a code change is pushed to this repository (Erneut erstellen, wenn eine Codeänderung an dieses Repository übergeben wird) aus.
2. Wählen Sie unter Event type (Ereignistyp) eines oder mehrere Ereignisse aus.
3. Wenn Sie Fälle filtern möchten, in denen ein Ereignis einen Build auslöst, fügen Sie unter Start a build under these conditions (Unter diesen Bedingungen Build starten) einen oder mehrere optionale Filter hinzu.
4. Wenn Sie Fälle filtern möchten, in denen kein Ereignis ausgelöst wird, fügen Sie unter Don't start a build under these conditions (Unter diesen Bedingungen keinen Build starten) einen oder mehrere optionale Filter hinzu.
5. Wählen Sie Filtergruppe hinzufügen, um bei Bedarf eine weitere Filtergruppe hinzuzufügen.

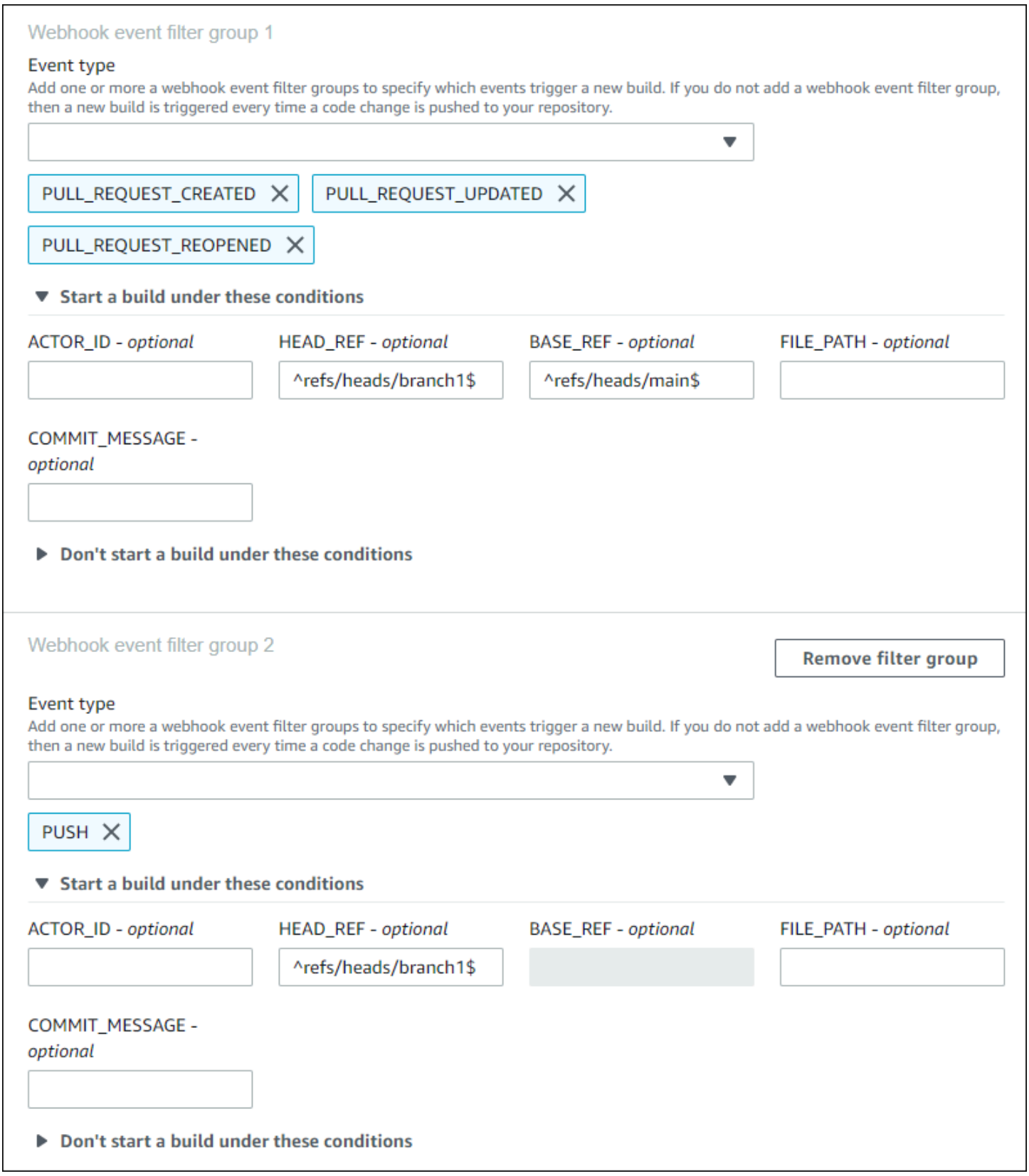
Weitere Informationen finden Sie unter Erstellen Sie ein Build-Projekt (Konsole) und WebhookFilterin der AWS CodeBuild API-Referenz.

In diesem Beispiel löst eine Webhook-Filtergruppe nur für Pull-Anfragen einen Build aus:

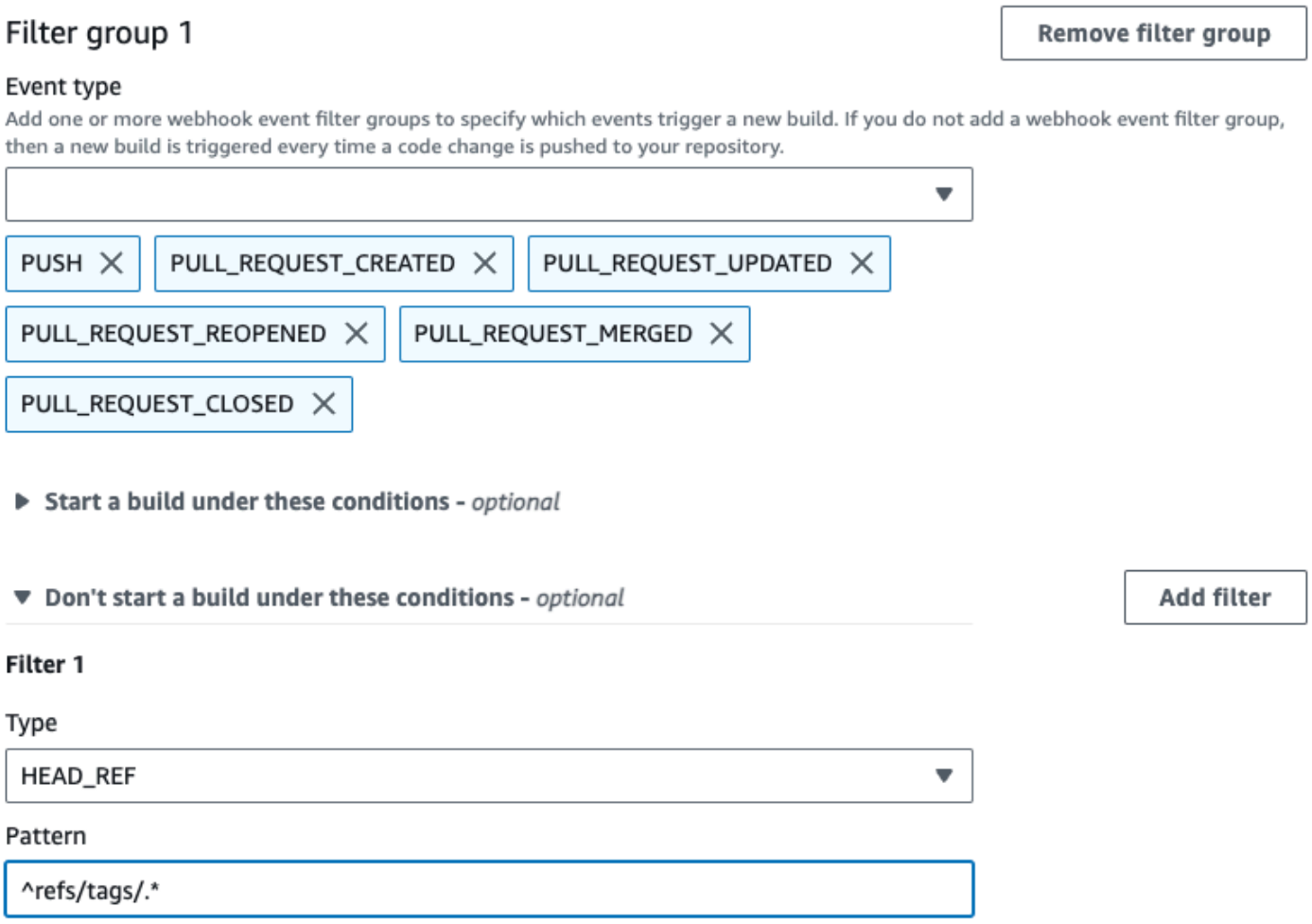


Bei Verwendung eines Beispiels mit zwei Webhook-Filtergruppen wird ein Build ausgelöst, wenn mindestens eine Gruppe als wahr ausgewertet wird:

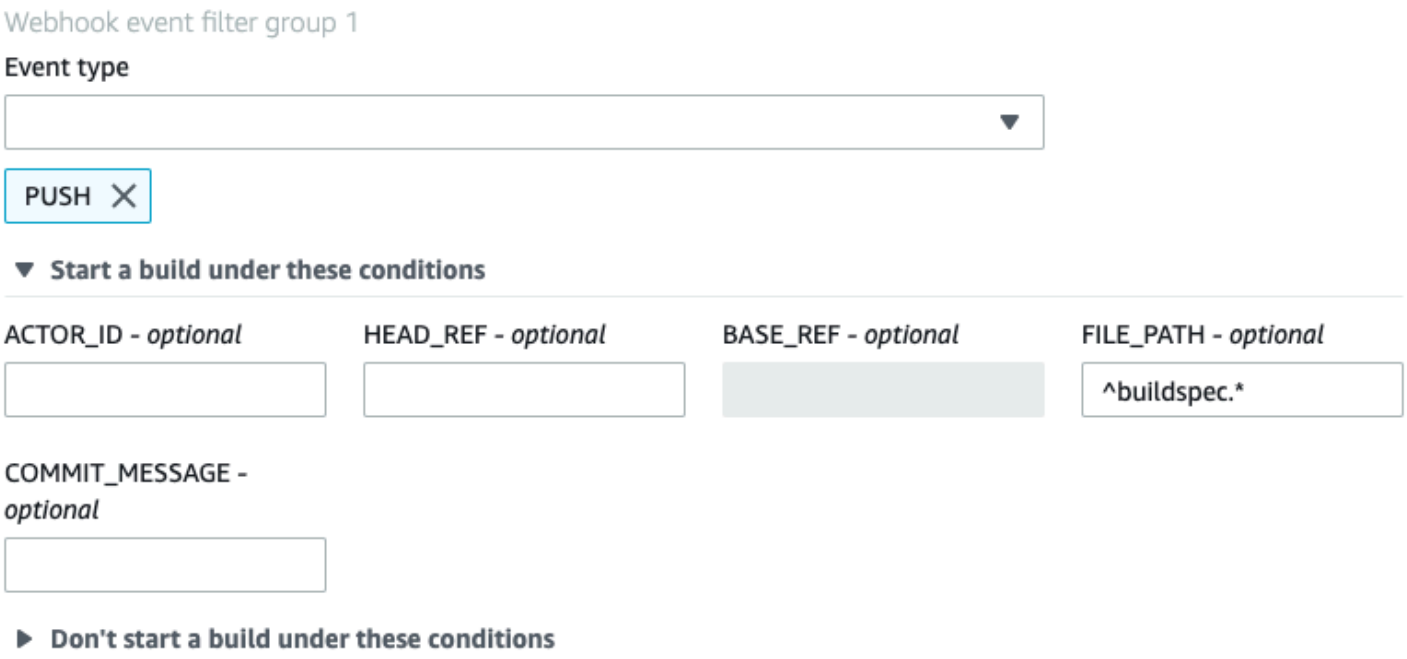
- Die erste Filtergruppe gibt Pull-Anfragen an, die in Verzweigungen mit Git-Referenznamen, die dem regulären Ausdruck `^refs/heads/main$` entsprechen, und mit Kopfreferenzen, die `^refs/heads/branch1$` entsprechen, erstellt, aktualisiert oder erneut geöffnet werden.
- Die zweite Filtergruppe gibt Push-Anfragen in Verzweigungen mit Git-Referenznamen an, die dem regulären Ausdruck `^refs/heads/branch1$` entsprechen.


Webhook event filter group 1
Event type
Add one or more a webhook event filter groups to specify which events trigger a new build. If you do not add a webhook event filter group, then a new build is triggered every time a code change is pushed to your repository.
PULL_REQUEST_CREATED
PULL_REQUEST_UPDATED
PULL_REQUEST_REOPENED
Start a build under these conditions
ACTOR_ID - optional
HEAD_REF - optional
^refs/heads/branch1$
BASE_REF - optional
^refs/heads/main$
FILE_PATH - optional
COMMIT_MESSAGE - optional
Don't start a build under these conditions
Webhook event filter group 2
Remove filter group
Event type
Add one or more a webhook event filter groups to specify which events trigger a new build. If you do not add a webhook event filter group, then a new build is triggered every time a code change is pushed to your repository.
PUSH
Start a build under these conditions
ACTOR_ID - optional
HEAD_REF - optional
^refs/heads/branch1$
BASE_REF - optional
FILE_PATH - optional
COMMIT_MESSAGE - optional
Don't start a build under these conditions

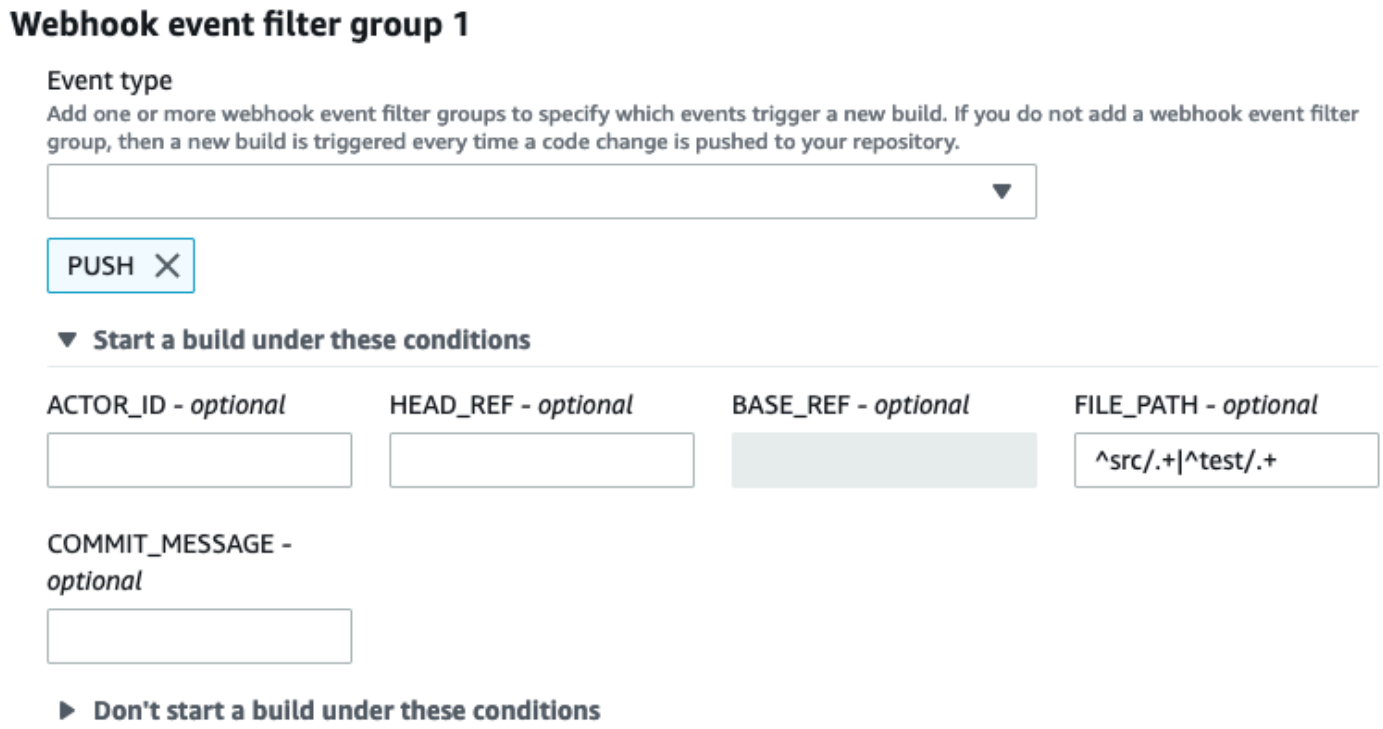
In diesem Beispiel löst eine Webhook-Filtergruppe einen Build für alle Anfragen mit Ausnahme von Tag-Ereignissen aus.



In diesem Beispiel löst eine Webhook-Filtergruppe nur einen Build aus, wenn Dateien geändert werden, deren Namen dem regulären Ausdruck `^buildspec.*` entsprechen.

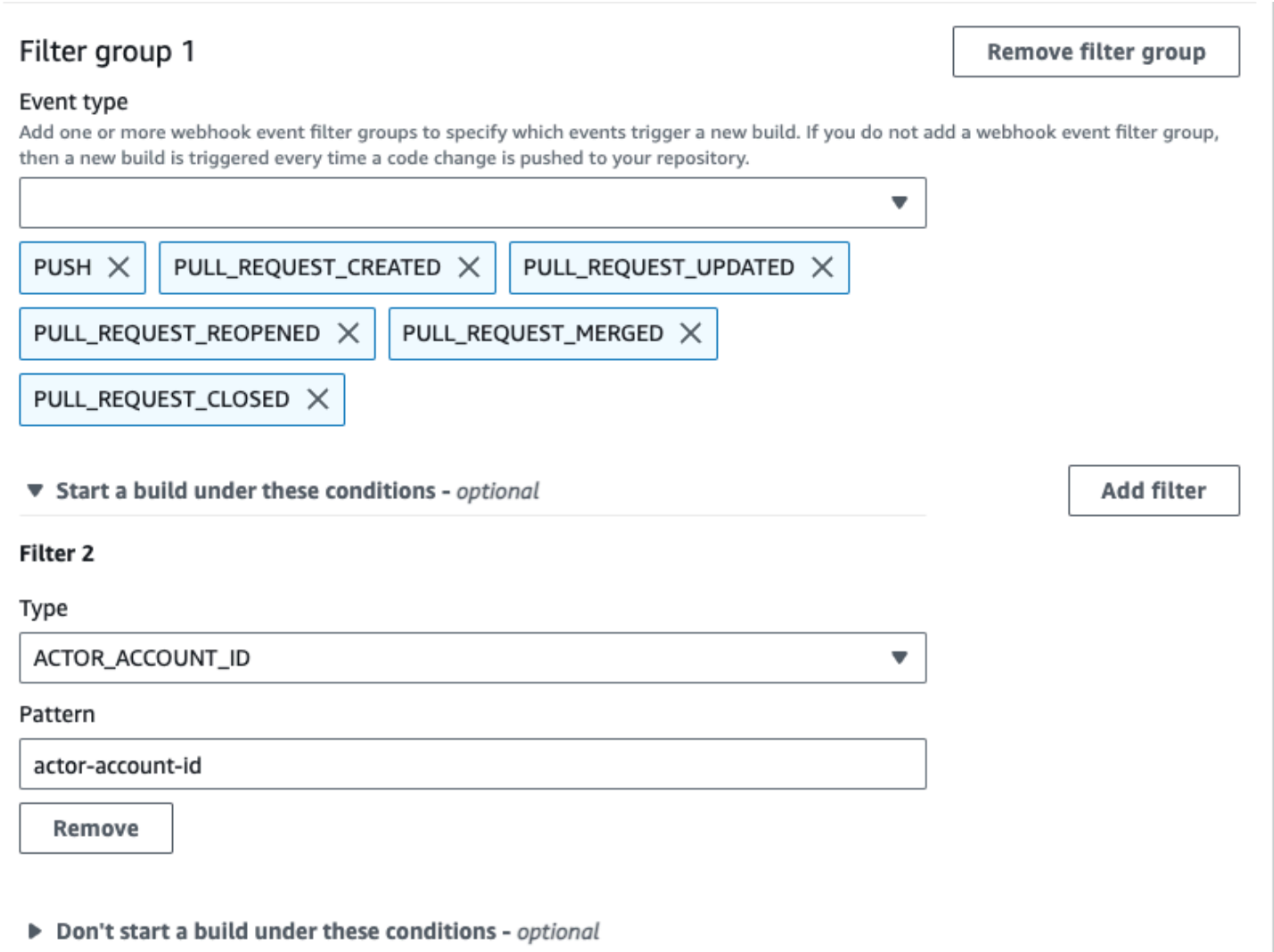


In diesem Beispiel löst eine Webhook-Filtergruppe nur dann einen Build aus, wenn Dateien in `test` Ordnern `src` oder geändert werden.

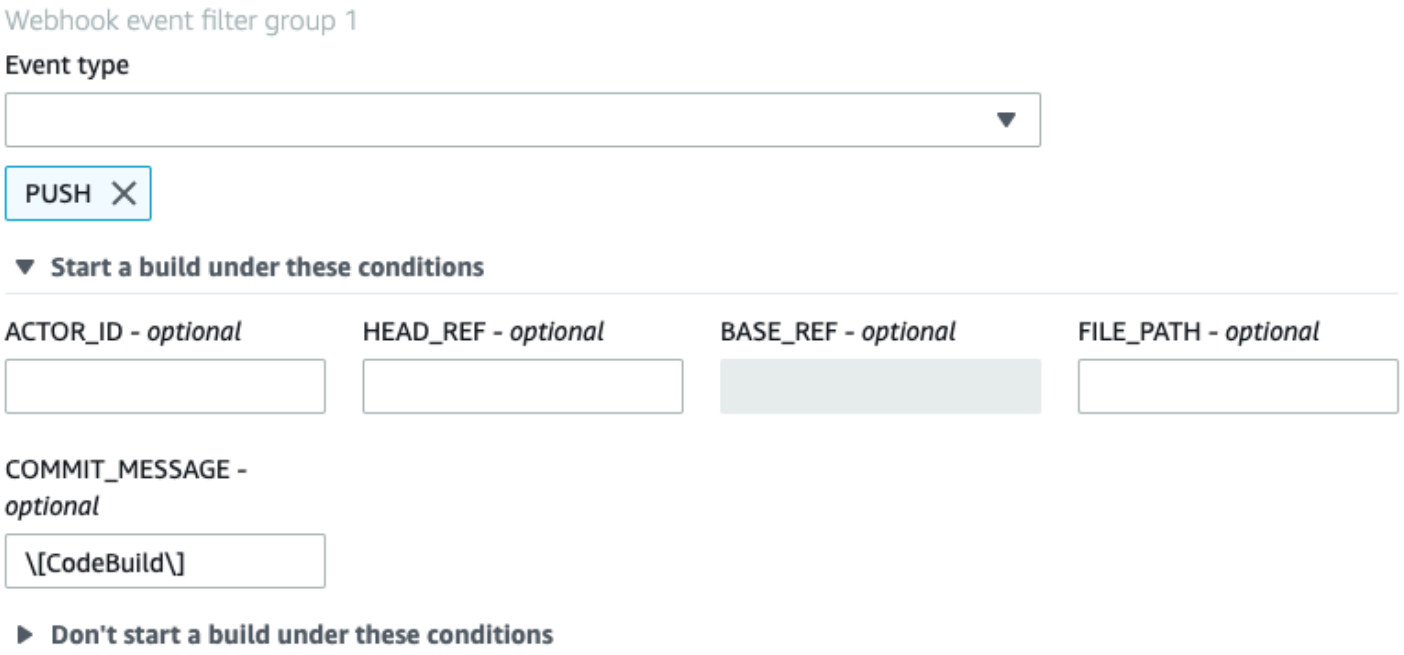


In diesem Beispiel löst eine Webhook-Filtergruppe nur dann einen Build aus, wenn eine Änderung von einem bestimmten Benutzer GitHub oder einem GitHub Enterprise Server-Benutzer mit einer Konto-ID vorgenommen wird, die dem regulären Ausdruck entspricht. `actor-account-id`

> **Note**
>
> Informationen dazu, wie Sie Ihre GitHub Konto-ID finden, finden Sie unter https://api.github.com/users/ *`benutzername`*, wobei *`benutzername Ihr GitHub Benutzername`* ist.

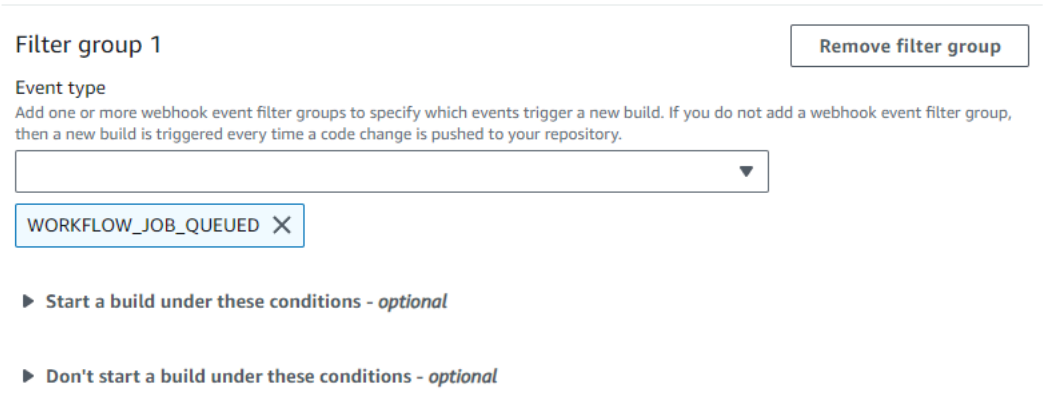


In diesem Beispiel löst eine Webhook-Filtergruppe einen Build für ein Push-Ereignis aus, wenn die Head-Commit-Nachricht mit dem regulären Ausdruck `\[CodeBuild\]` übereinstimmt.

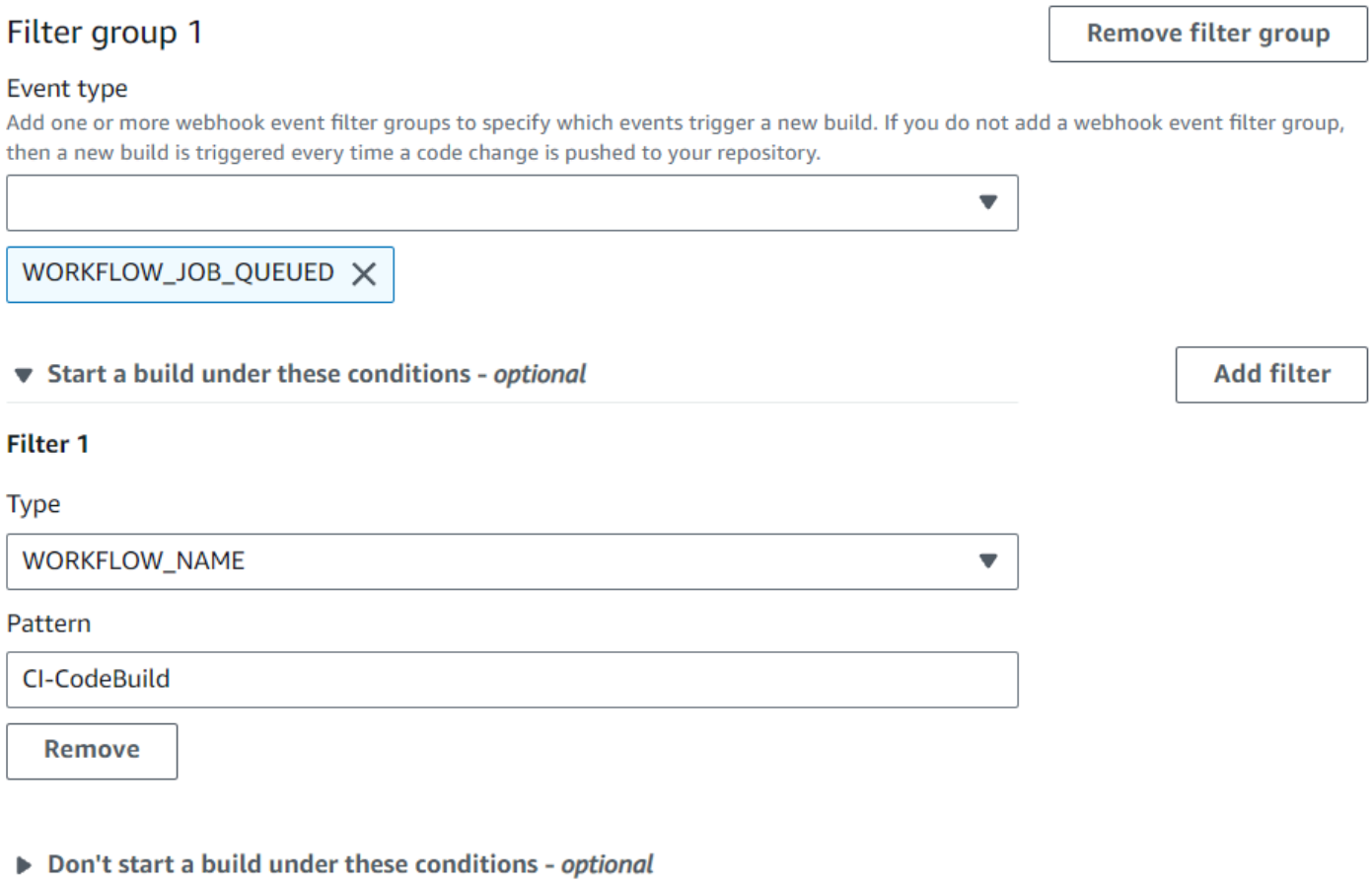


In diesem Beispiel löst eine Webhook-Filtergruppe einen Build nur für GitHub Aktions-Workflow-Auftragsereignisse aus.

> **Note**
>
> CodeBuild verarbeitet GitHub Aktions-Workflow-Jobs nur, wenn ein Webhook Filtergruppen enthält, die den WORKFLOW_JOB_QUEUED-Ereignisfilter enthalten.



In diesem Beispiel löst eine Webhook-Filtergruppe einen Build für einen Workflow-Namen aus, der dem regulären Ausdruck entspricht. `CI-CodeBuild`



GitHub Webhook-Ereignisse filtern (SDK)

Um das AWS CodeBuild SDK zum Filtern von Webhook-Ereignissen zu verwenden, verwenden Sie das `filterGroups` Feld in der Anforderungssyntax der `CreateWebhook` `UpdateWebhook` API-Methoden oder. Weitere Informationen finden Sie WebhookFilterin der CodeBuild API-Referenz.

Um einen Webhook-Filter zu erstellen, der nur für Pull-Anfragen einen Build auslöst, fügen Sie Folgendes in die Anfragesyntax ein:

```
"filterGroups": [
   [
        {
            "type": "EVENT",
            "pattern": "PULL_REQUEST_CREATED, PULL_REQUEST_UPDATED,
 PULL_REQUEST_REOPENED, PULL_REQUEST_MERGED, PULL_REQUEST_CLOSED"
        }
    ]
]
```

Um einen Webhook-Filter zu erstellen, der nur für die angegebenen Verzweigungen einen Build auslöst, verwenden Sie den Parameter `pattern`, um einen regulären Ausdruck zum Filtern von Verzweigungsnamen anzugeben. Bei Verwendung eines Beispiels mit zwei Filtergruppen wird ein Build ausgelöst, wenn mindestens eine Gruppe als wahr ausgewertet wird:

- Die erste Filtergruppe gibt Pull-Anfragen an, die in Verzweigungen mit Git-Referenznamen, die dem regulären Ausdruck `^refs/heads/main$` entsprechen, und mit Kopfreferenzen, die `^refs/heads/myBranch$` entsprechen, erstellt, aktualisiert oder erneut geöffnet werden.
- Die zweite Filtergruppe gibt Push-Anfragen in Verzweigungen mit Git-Referenznamen an, die dem regulären Ausdruck `^refs/heads/myBranch$` entsprechen.

```
"filterGroups": [
    [
        {
            "type": "EVENT",
            "pattern": "PULL_REQUEST_CREATED, PULL_REQUEST_UPDATED,
 PULL_REQUEST_REOPENED"
        },
        {
            "type": "HEAD_REF",
            "pattern": "^refs/heads/myBranch$"
        },
        {
            "type": "BASE_REF",
            "pattern": "^refs/heads/main$"
        }
    ],
    [
        {
            "type": "EVENT",
            "pattern": "PUSH"
        },
        {
            "type": "HEAD_REF",
            "pattern": "^refs/heads/myBranch$"
        }
    ]
]
```

Sie können den Parameter `excludeMatchedPattern` verwenden, um anzugeben, welche Ereignisse keinen Build auslösen. In diesem Beispiel etwa wird für alle Anfragen mit Ausnahme von Tag-Ereignissen ein Build ausgelöst.

```
"filterGroups": [
    [
        {
            "type": "EVENT",
            "pattern": "PUSH, PULL_REQUEST_CREATED, PULL_REQUEST_UPDATED,
 PULL_REQUEST_REOPENED, PULL_REQUEST_MERGED, PULL_REQUEST_CLOSED"
        },
        {
            "type": "HEAD_REF",
            "pattern": "^refs/tags/.*",
            "excludeMatchedPattern": true
        }
    ]
]
```

Sie können einen Filter erstellen, der nur einen Build auslöst, wenn Dateien geändert werden, deren Namen dem regulären Ausdruck in dem Argument `pattern` entsprechen. In diesem Beispiel gibt die Filtergruppe an, dass nur ein Build ausgelöst wird, wenn Dateien geändert werden, deren Name dem regulären Ausdruck `^buildspec.*` entspricht.

```
"filterGroups": [
    [
        {
            "type": "EVENT",
            "pattern": "PUSH"
        },
        {
            "type": "FILE_PATH",
            "pattern": "^buildspec.*"
        }
    ]
]
```

In diesem Beispiel gibt die Filtergruppe an, dass ein Build nur ausgelöst wird, wenn Dateien in `test` Ordnern `src` oder geändert werden.

```
"filterGroups": [
```

```
    [
        {
            "type": "EVENT",
            "pattern": "PUSH"
        },
        {
            "type": "FILE_PATH",
            "pattern": "^src/.+|^test/.+"
        }
    ]
]
```

Sie können einen Filter erstellen, der einen Build nur auslöst, wenn eine Änderung von einem bestimmten Benutzer GitHub oder einem GitHub Enterprise Server-Benutzer mit Konto-ID vorgenommen wird`actor-account-id`.

> **Note**
>
> Informationen dazu, wie Sie Ihre GitHub Konto-ID finden, finden Sie unter *`https://api.github.com/users/ benutzername`*, wobei *`benutzername`* Ihr GitHub Benutzername ist.

```
"filterGroups": [
    [
        {
            "type": "EVENT",
            "pattern": "PUSH, PULL_REQUEST_CREATED, PULL_REQUEST_UPDATED,
 PULL_REQUEST_REOPENED, PULL_REQUEST_MERGED, PULL_REQUEST_CLOSED"
        },
        {
            "type": "ACTOR_ACCOUNT_ID",
            "pattern": "actor-account-id"
        }
    ]
]
```

Sie können einen Filter erstellen, der einen Build nur dann auslöst, wenn die Head-Commit-Nachricht mit dem regulären Ausdruck im Musterargument übereinstimmt. In diesem Beispiel gibt die Filtergruppe an, dass ein Build nur dann ausgelöst wird, wenn die Head-Commit-Nachricht des Push-Ereignisses mit dem regulären Ausdruck `\[CodeBuild\]` übereinstimmt.

```
"filterGroups": [
    [
        {
            "type": "EVENT",
            "pattern": "PUSH"
        },
        {
            "type": "COMMIT_MESSAGE",
            "pattern": "\[CodeBuild\]"
        }
    ]
]
```

Um einen Webhook-Filter zu erstellen, der nur einen Build für Workflow-Jobs für GitHub Aktionen auslöst, fügen Sie Folgendes in die Anforderungssyntax ein:

```
"filterGroups": [
   [
        {
            "type": "EVENT",
            "pattern": "WORKFLOW_JOB_QUEUED"
        }
    ]
]
```

GitHub Webhook-Ereignisse filtern ()AWS CloudFormation

Um eine AWS CloudFormation Vorlage zum Filtern von Webhook-Ereignissen zu verwenden, verwenden Sie die Eigenschaft des AWS CodeBuild `FilterGroups` Projekts. Mit dem folgenden in YAML formatierten Teil einer AWS CloudFormation -Vorlage werden zwei Filtergruppen erstellt. Diese lösen zusammen einen Build aus, wenn mindestens eine Gruppe als wahr ausgewertet wird.

- Die erste Filtergruppe gibt an, dass Pull-Requests für Branches mit Git-Referenznamen, die dem regulären Ausdruck entsprechen, `^refs/heads/main$` von einem GitHub Benutzer ohne Konto-ID erstellt oder aktualisiert `12345` werden.
- Die zweite Filtergruppe gibt an, dass Push-Anfragen für Dateien, deren Namen dem regulären Ausdruck `READ_ME` entsprechen, in Verzweigungen mit Git-Referenznamen, die dem regulären Ausdruck `^refs/heads/.*` entsprechen, erstellt werden.
- Die dritte Filtergruppe gibt eine Push-Anforderung mit einer Head Commit-Nachricht an, die dem regulären Ausdruck `\[CodeBuild\]` entspricht.

- Die vierte Filtergruppe spezifiziert eine Workflow-Auftragsanforderung für GitHub Aktionen mit einem Workflow-Namen, der dem regulären Ausdruck entspricht`\[CI-CodeBuild\]`.

```
CodeBuildProject:
  Type: AWS::CodeBuild::Project
  Properties:
    Name: MyProject
    ServiceRole: service-role
    Artifacts:
      Type: NO_ARTIFACTS
    Environment:
      Type: LINUX_CONTAINER
      ComputeType: BUILD_GENERAL1_SMALL
      Image: aws/codebuild/standard:5.0
    Source:
      Type: GITHUB
      Location: source-location
    Triggers:
      Webhook: true
      FilterGroups:
        - - Type: EVENT
            Pattern: PULL_REQUEST_CREATED,PULL_REQUEST_UPDATED
          - Type: BASE_REF
            Pattern: ^refs/heads/main$
            ExcludeMatchedPattern: false
          - Type: ACTOR_ACCOUNT_ID
            Pattern: 12345
            ExcludeMatchedPattern: true
        - - Type: EVENT
            Pattern: PUSH
          - Type: HEAD_REF
            Pattern: ^refs/heads/.*
          - Type: FILE_PATH
            Pattern: READ_ME
            ExcludeMatchedPattern: true
        - - Type: EVENT
            Pattern: PUSH
          - Type: COMMIT_MESSAGE
            Pattern: \[CodeBuild\]
          - Type: FILE_PATH
            Pattern: ^src/.+|^test/.+
        - - Type: EVENT
```

```
      Pattern: WORKFLOW_JOB_QUEUED
    - Type: WORKFLOW_NAME
      Pattern: \[CI-CodeBuild\]
```

GitLab Webhook-Ereignisse

Sie können Webhook-Filtergruppen verwenden, um anzugeben, welche GitLab Webhook-Ereignisse einen Build auslösen. Sie können beispielsweise angeben, dass ein Build nur bei Änderungen an bestimmten Branches ausgelöst wird.

Sie können eine oder mehrere Webhook-Filtergruppen erstellen, um anzugeben, welche Webhook-Ereignisse einen Build auslösen. Ein Build wird ausgelöst, wenn eine Filtergruppe als wahr ausgewertet wird. Dies ist der Fall, wenn alle Filter in der Gruppe den Wert true ergeben. Beim Erstellen einer Filtergruppe geben Sie Folgendes an:

Ein Ereignis

Für GitLab können Sie eines oder mehrere der folgenden Ereignisse wählen:

- `PUSH`
- `PULL_REQUEST_CREATED`
- `PULL_REQUEST_UPDATED`
- `PULL_REQUEST_MERGED`

Der Webhook-Ereignistyp befindet sich im Header des Feldes `X-Event-Key`. Aus der folgende Tabelle geht die Zuordnung der `X-Event-Key`-Header-Werte zu Ereignistypen hervor.

> **Note**
>
> Sie müssen das `merged` Ereignis in Ihrer GitLab Webhook-Einstellung aktivieren, wenn Sie eine Webhook-Filtergruppe erstellen, die den `PULL_REQUEST_MERGED` Ereignistyp verwendet.

`X-Event-Key` -Header-Wert	Ereignistyp
`repo:push`	`PUSH`
`pullrequest:created`	`PULL_REQUEST_CREATED`

X-Event-Key -Header-Wert	Ereignistyp
`pullrequest:updated`	`PULL_REQUEST_UPDATED`
`pullrequest:fulfilled`	`PULL_REQUEST_MERGED`

Denn `PULL_REQUEST_MERGED` wenn ein Pull-Request mit der Squash-Strategie zusammengeführt und der Pull-Request-Branch geschlossen wird, ist der ursprüngliche Pull-Request-Commit nicht mehr vorhanden. In diesem Fall enthält die `CODEBUILD_WEBHOOK_MERGE_COMMIT` Umgebungsvariable den Bezeichner des gequetschten Merge-Commits.

Ein oder mehrere optionale Filter

Verwenden Sie einen regulären Ausdruck, um einen Filter anzugeben. Damit ein Ereignis einen Build auslöst, muss jeder Filter innerhalb der Gruppe, die diesem Ereignis zugeordnet ist, als wahr ausgewertet werden.

`ACTOR_ACCOUNT_ID`(`ACTOR_ID`in der Konsole)

Ein Webhook-Ereignis löst einen Build aus, wenn eine GitLab Konto-ID dem Muster eines regulären Ausdrucks entspricht. Dieser Wert wird in der Eigenschaft `account_id` des Objekts `actor` in der Webhook-Filternutzlast angezeigt.

`HEAD_REF`

Ein Webhook-Ereignis löst einen Build aus, wenn die Hauptreferenz dem Muster für reguläre Ausdrücke entspricht (z. B. `refs/heads/branch-name` und`refs/tags/tag-name`). Ein `HEAD_REF`-Filter wertet den Git-Referenznamen für den Branch oder Tag aus. Die Branch- oder Tag-Name wird im Feld `name` des Objekts `new` im Objekt `push` der Webhook-Nutzlast angezeigt. Bei Pull-Anforderungsereignissen wird der Branch-Name im Feld `name` im Objekt `branch` des Objekts `source` in der Webhook-Nutzlast angezeigt.

`BASE_REF`

Ein Webhook-Ereignis löst einen Build aus, wenn die Basisreferenz mit dem Muster des regulären Ausdrucks übereinstimmt. Ein `BASE_REF`-Filter kann nur für Pull-Anfrageereignisse verwendet werden (z. B. `refs/heads/branch-name`). Ein `BASE_REF`-Filter wertet den Git-Referenznamen für die Verzweigung aus. Der Branch-Name wird im Feld `name` des Objekts `branch` im Objekt `destination` in der Webhook-Nutzlast angezeigt.

`FILE_PATH`

Ein Webhook löst einen Build aus, wenn der Pfad einer geänderten Datei dem Muster für reguläre Ausdrücke entspricht.

`COMMIT_MESSAGE`

Ein Webhook löst einen Build aus, wenn die Head-Commit-Nachricht dem Muster des regulären Ausdrucks entspricht.

> **Note**
>
> Sie finden die Webhook-Payload in den Webhook-Einstellungen Ihres Repositorys. GitLab

Themen

- GitLab Webhook-Ereignisse filtern (Konsole)
- GitLab Webhook-Ereignisse filtern (SDK)
- GitLab Webhook-Ereignisse filtern ()AWS CloudFormation

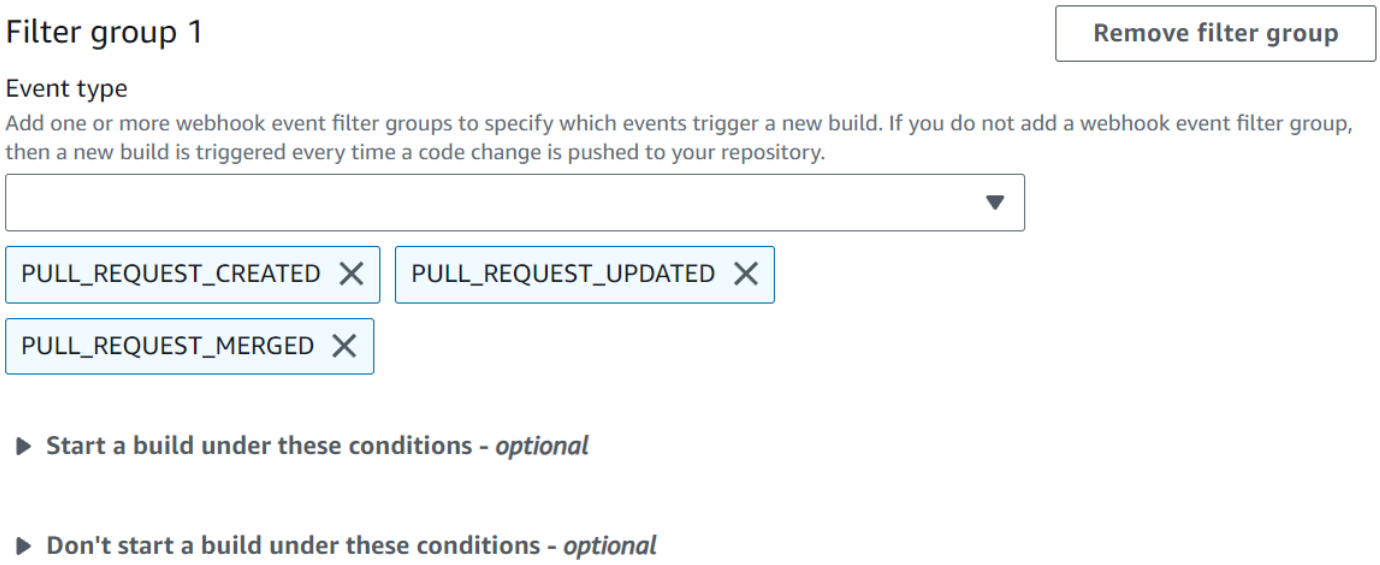
GitLab Webhook-Ereignisse filtern (Konsole)

Um Webhook-Ereignisse AWS Management Console zu filtern, gehen Sie wie folgt vor:

1. Wählen Sie beim Erstellen Ihres Projekts Rebuild every time a code change is pushed to this repository (Erneut erstellen, wenn eine Codeänderung an dieses Repository übergeben wird) aus.
2. Wählen Sie unter Event type (Ereignistyp) eines oder mehrere Ereignisse aus.
3. Wenn Sie Fälle filtern möchten, in denen ein Ereignis einen Build auslöst, fügen Sie unter Start a build under these conditions (Unter diesen Bedingungen Build starten) einen oder mehrere optionale Filter hinzu.
4. Wenn Sie Fälle filtern möchten, in denen kein Ereignis ausgelöst wird, fügen Sie unter Don't start a build under these conditions (Unter diesen Bedingungen keinen Build starten) einen oder mehrere optionale Filter hinzu.
5. Wählen Sie Add filter group (Filtergruppe hinzufügen) aus, um eine weitere Filtergruppe hinzuzufügen.

Weitere Informationen finden Sie unter Erstellen Sie ein Build-Projekt (Konsole) und WebhookFilterin der AWS CodeBuild API-Referenz.

In diesem Beispiel löst eine Webhook-Filtergruppe nur für Pull-Anfragen einen Build aus:



Bei Verwendung eines Beispiels mit zwei Filtergruppen wird ein Build ausgelöst, wenn mindestens eine Gruppe als wahr ausgewertet wird:

- Die erste Filtergruppe gibt Pull-Anfragen an, die in Verzweigungen mit Git-Referenznamen, die dem regulären Ausdruck `^refs/heads/main$` entsprechen, und mit Kopfreferenzen, die `^refs/heads/branch1!` entsprechen, erstellt oder aktualisiert werden.
- Die zweite Filtergruppe gibt Push-Anfragen in Verzweigungen mit Git-Referenznamen an, die dem regulären Ausdruck `^refs/heads/branch1$` entsprechen.

Webhook event filter groups

A build is triggered if any filter group evaluates to true, which occurs when all the filters in the group evaluate to true.

Filter group 1

Remove filter group

Event type

Add one or more webhook event filter groups to specify which events trigger a new build. If you do not add a webhook event filter group, then a new build is triggered every time a code change is pushed to your repository.

PULL_REQUEST_CREATED ✕ PULL_REQUEST_UPDATED ✕

▼ **Start a build under these conditions** - *optional*

Add filter

Filter 1

Type

HEAD_REF

Pattern

^refs/heads/branch1$

Remove

Filter 2

Type

BASE_REF

Pattern

^refs/heads/main

Remove

▶ **Don't start a build under these conditions** - *optional*

Filter group 2

Remove filter group

Event type

Add one or more webhook event filter groups to specify which events trigger a new build. If you do not add a webhook event filter group, then a new build is triggered every time a code change is pushed to your repository.

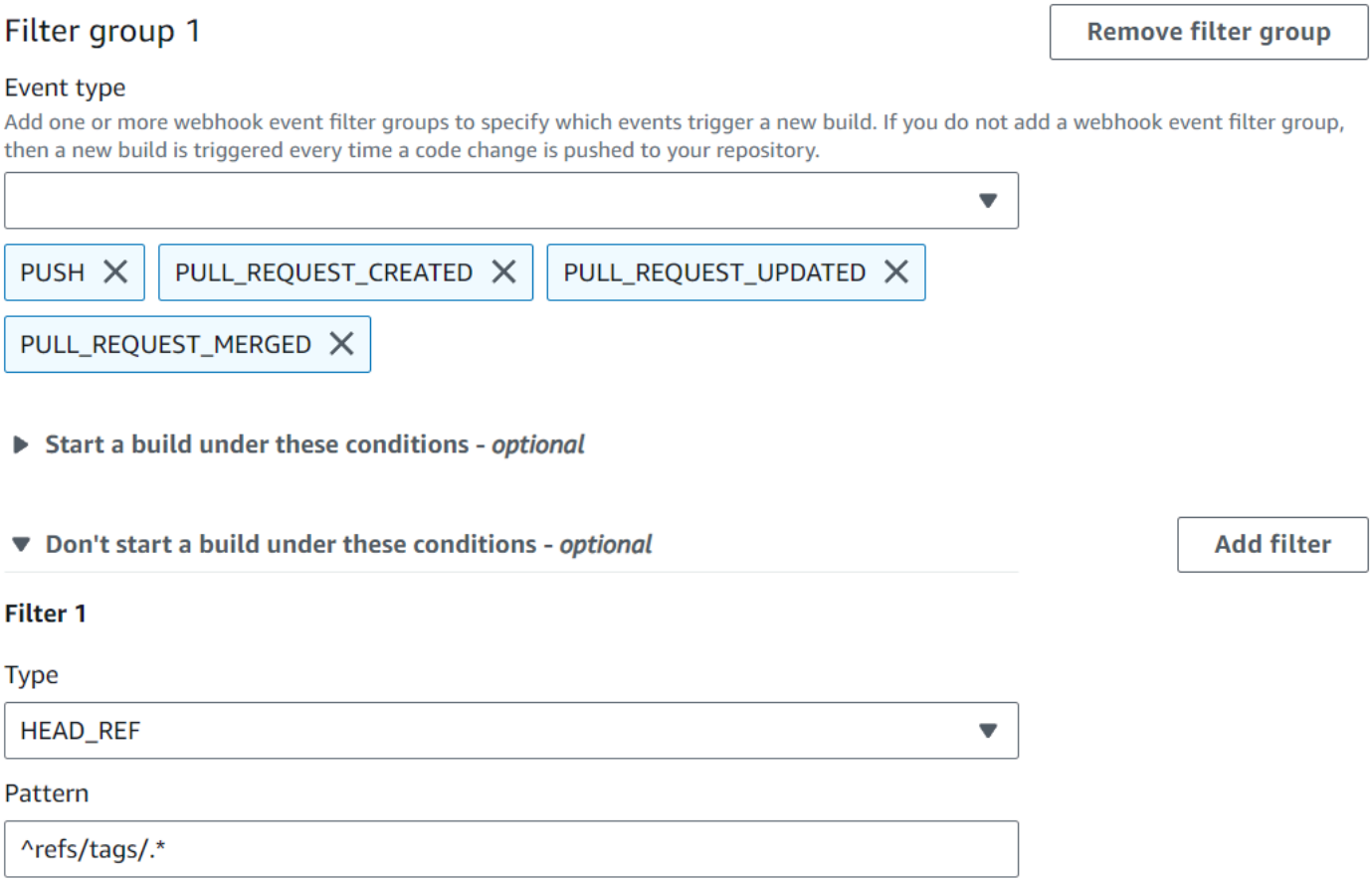
PUSH ✕

▼ **Start a build under these conditions** - *optional*

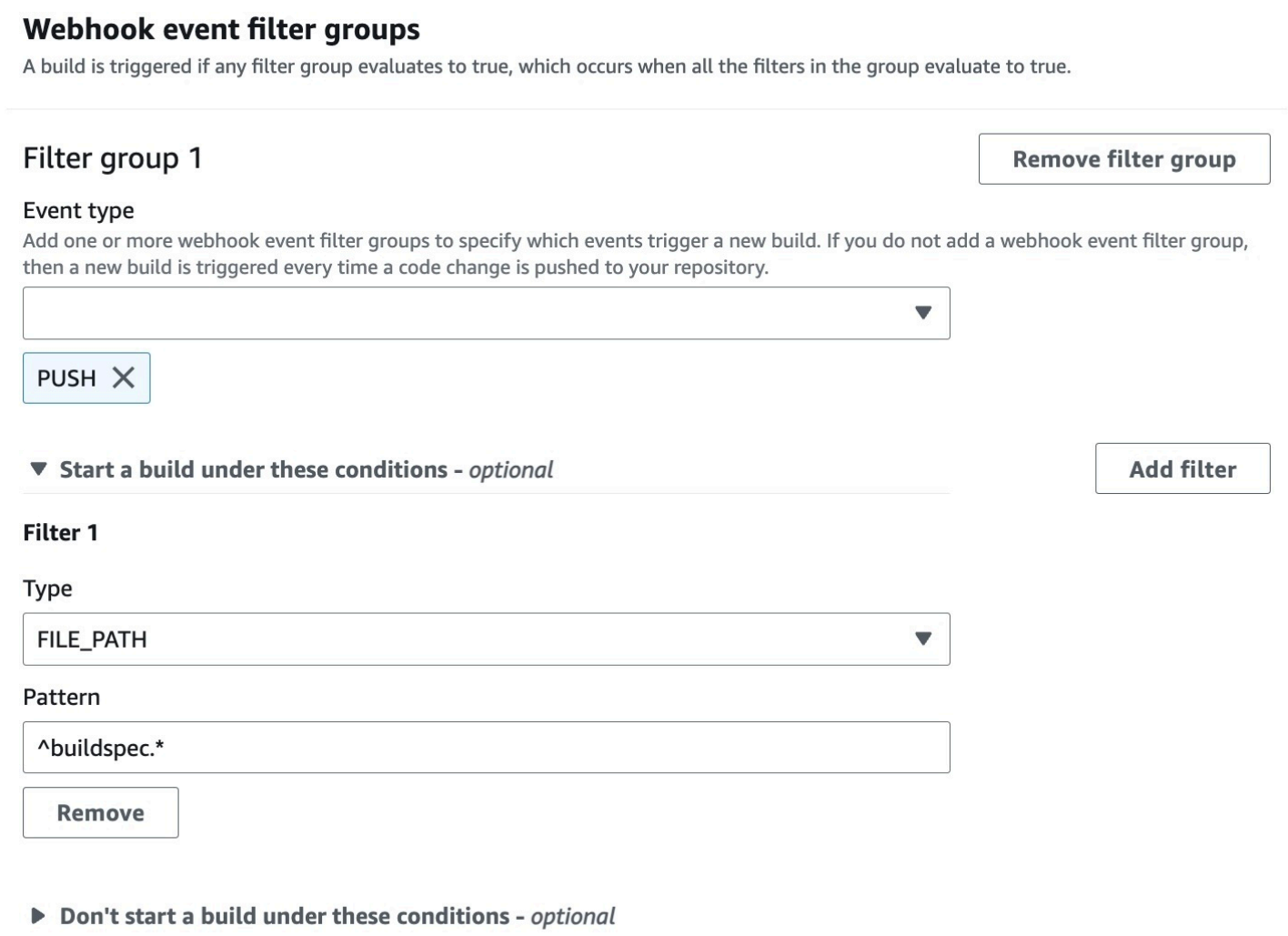
Add filter

Filter 1

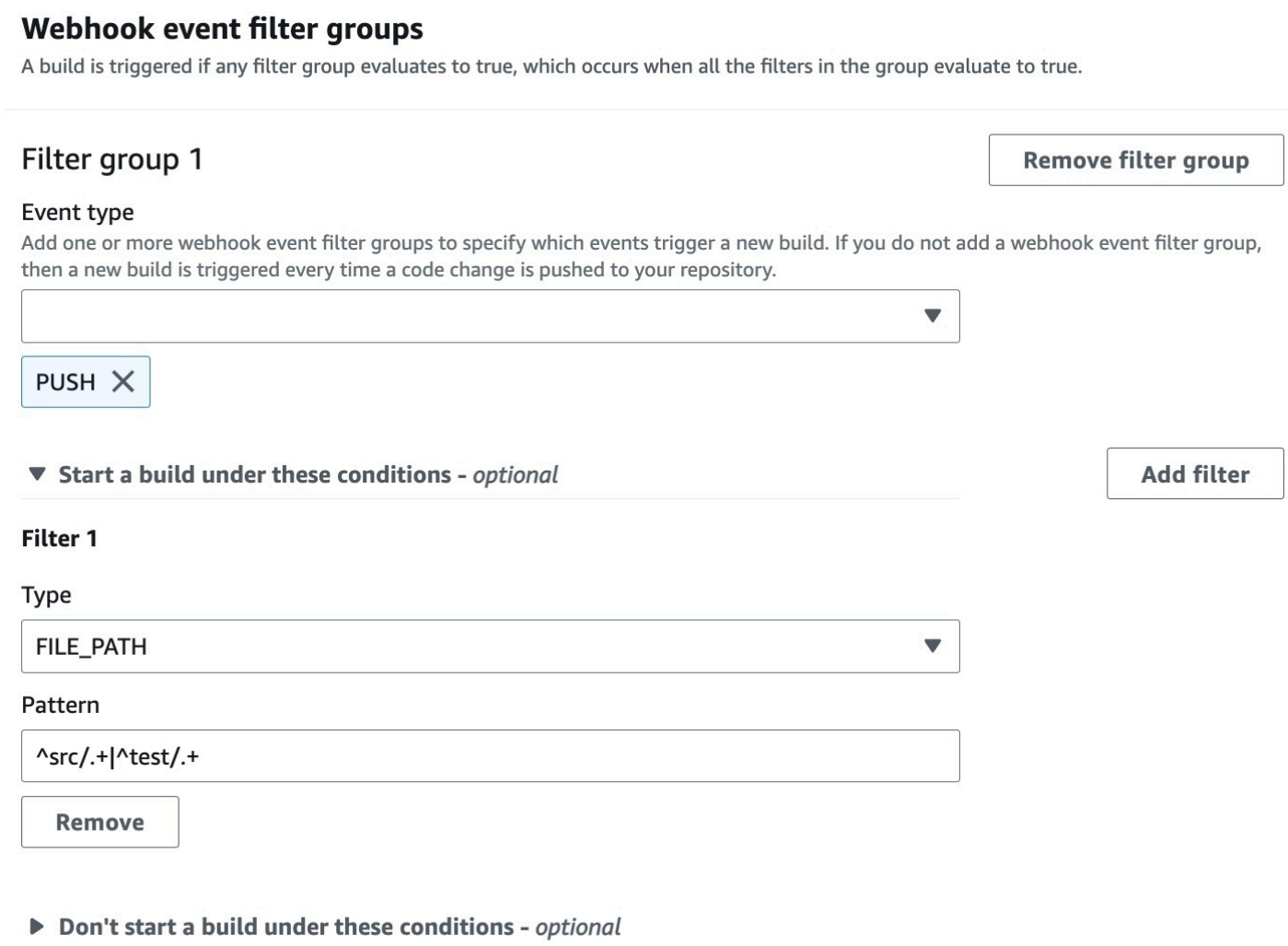
In diesem Beispiel löst eine Webhook-Filtergruppe einen Build für alle Anfragen mit Ausnahme von Tag-Ereignissen aus.



In diesem Beispiel löst eine Webhook-Filtergruppe nur einen Build aus, wenn Dateien geändert werden, deren Namen dem regulären Ausdruck `^buildspec.*` entsprechen.

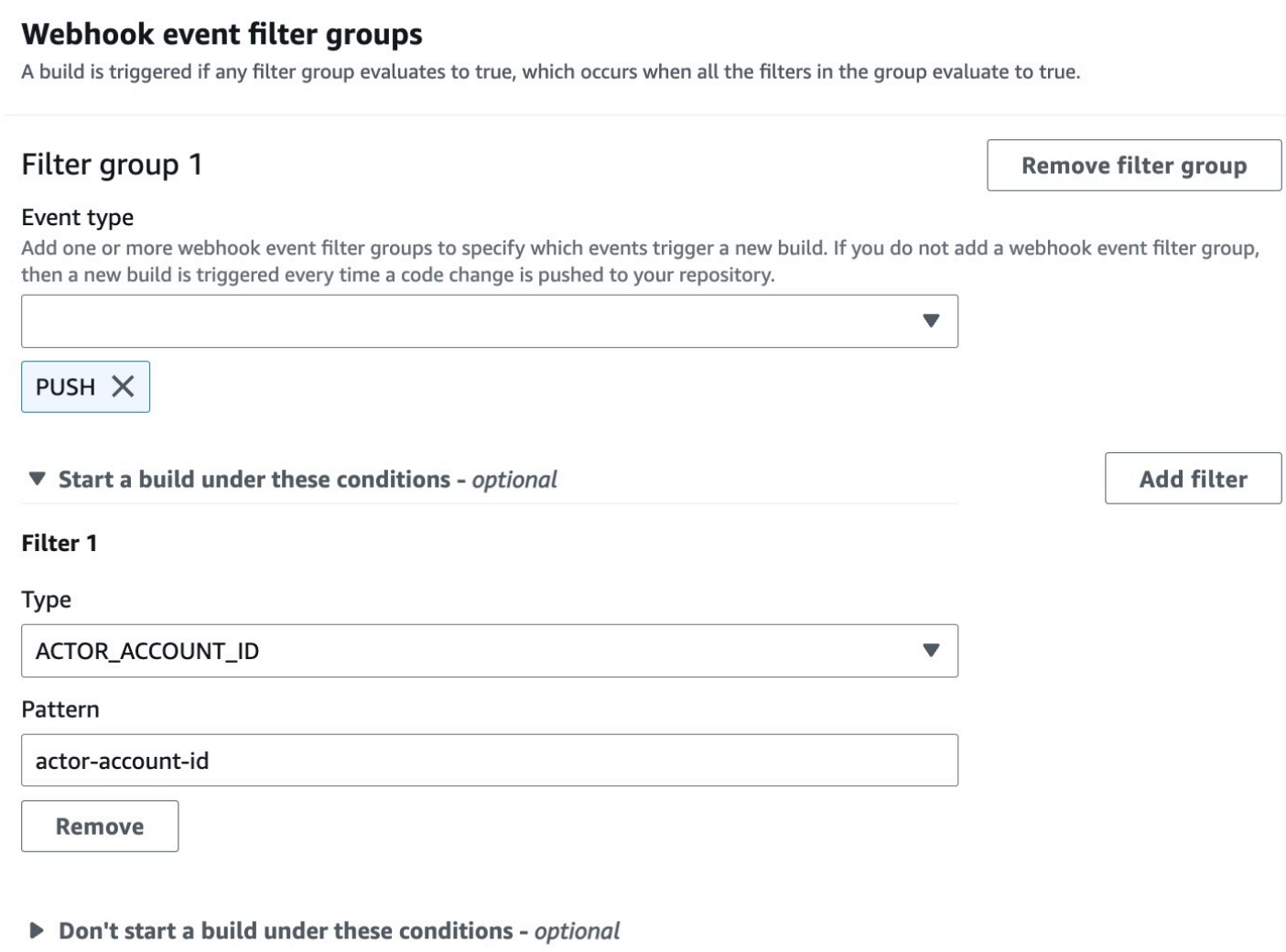


In diesem Beispiel löst eine Webhook-Filtergruppe nur dann einen Build aus, wenn Dateien in `test` Ordnern `src` oder geändert werden.

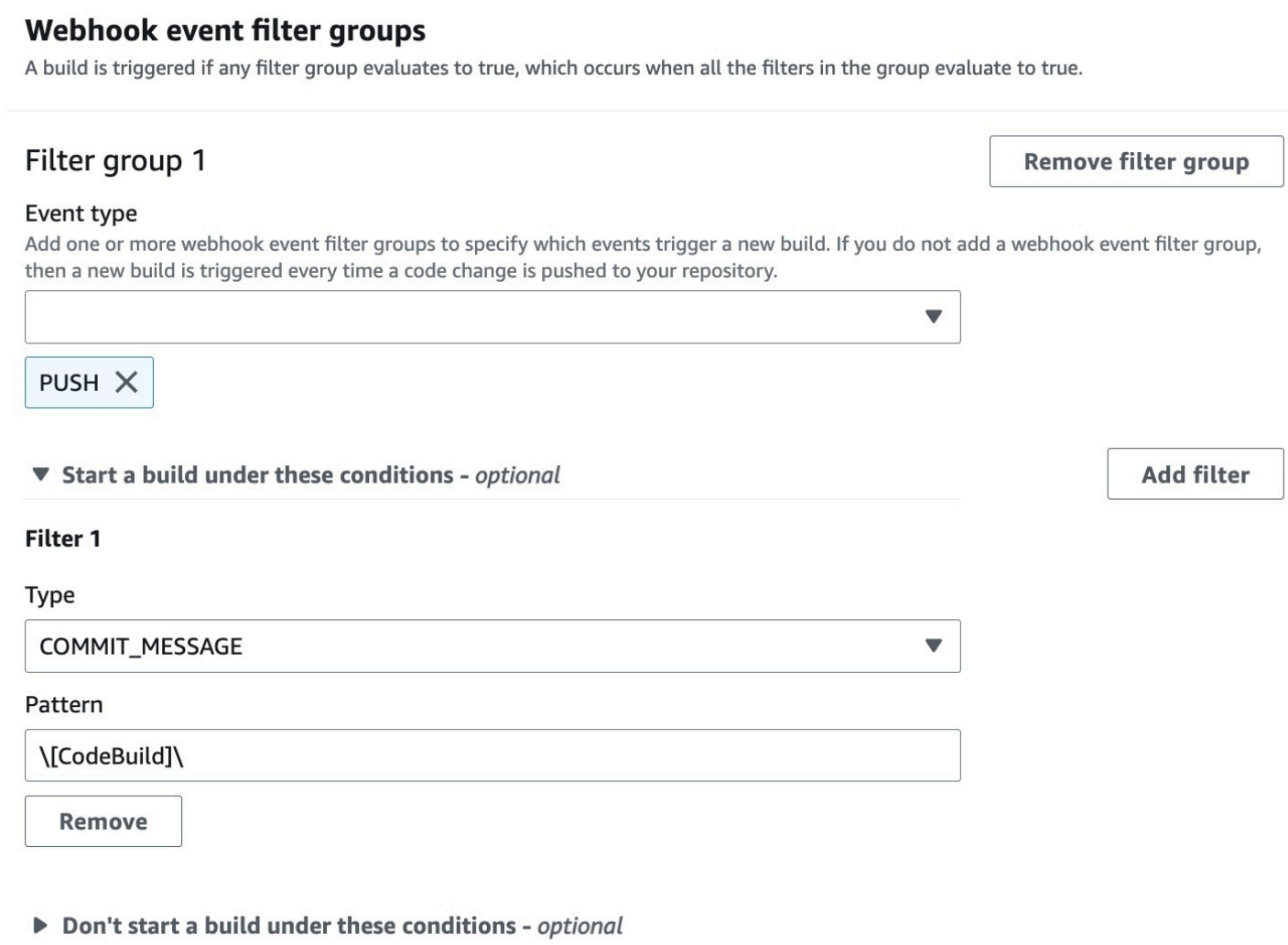


In diesem Beispiel löst eine Webhook-Filtergruppe nur dann einen Build aus, wenn eine Änderung von einem GitLab Benutzer vorgenommen wird, der keine Konto-ID hat, die dem regulären Ausdruck entspricht. `actor-account-id`

> **Note**
>
> Informationen dazu, wie Sie Ihre GitLab Konto-ID finden, finden Sie unter https://api.github.com/users/ *`benutzername`*, wobei *`benutzername Ihr GitLab Benutzername`* ist.



In diesem Beispiel löst eine Webhook-Filtergruppe einen Build für ein Push-Ereignis aus, wenn die Head-Commit-Nachricht mit dem regulären Ausdruck `\[CodeBuild\]` übereinstimmt.



GitLab Webhook-Ereignisse filtern (SDK)

Um das AWS CodeBuild SDK zum Filtern von Webhook-Ereignissen zu verwenden, verwenden Sie das `filterGroups` Feld in der Anforderungssyntax der `CreateWebhook` `UpdateWebhook` API-Methoden oder. Weitere Informationen finden Sie WebhookFilterin der CodeBuild API-Referenz.

Um einen Webhook-Filter zu erstellen, der nur für Pull-Anfragen einen Build auslöst, fügen Sie Folgendes in die Anfragesyntax ein:

```
"filterGroups": [
  [
    {
      "type": "EVENT",
      "pattern": "PULL_REQUEST_CREATED, PULL_REQUEST_UPDATED, PULL_REQUEST_MERGED"
    }
```

```
  ]
]
```

Um einen Webhook-Filter zu erstellen, der nur für die angegebenen Verzweigungen einen Build auslöst, verwenden Sie den Parameter `pattern`, um einen regulären Ausdruck zum Filtern von Verzweigungsnamen anzugeben. Bei Verwendung eines Beispiels mit zwei Filtergruppen wird ein Build ausgelöst, wenn mindestens eine Gruppe als wahr ausgewertet wird:

- Die erste Filtergruppe gibt Pull-Anfragen an, die in Verzweigungen mit Git-Referenznamen, die dem regulären Ausdruck `^refs/heads/main$` entsprechen, und mit Kopfreferenzen, die `^refs/heads/myBranch$` entsprechen, erstellt oder aktualisiert werden.
- Die zweite Filtergruppe gibt Push-Anfragen in Verzweigungen mit Git-Referenznamen an, die dem regulären Ausdruck `^refs/heads/myBranch$` entsprechen.

```
"filterGroups": [
  [
    {
      "type": "EVENT",
      "pattern": "PULL_REQUEST_CREATED, PULL_REQUEST_UPDATED"
    },
    {
      "type": "HEAD_REF",
      "pattern": "^refs/heads/myBranch$"
    },
    {
      "type": "BASE_REF",
      "pattern": "^refs/heads/main$"
    }
  ],
  [
    {
      "type": "EVENT",
      "pattern": "PUSH"
    },
    {
      "type": "HEAD_REF",
      "pattern": "^refs/heads/myBranch$"
    }
  ]
]
```

Sie können den Parameter `excludeMatchedPattern` verwenden, um anzugeben, welche Ereignisse keinen Build auslösen. In diesem Beispiel wird für alle Anfragen mit Ausnahme von Tag-Ereignissen ein Build ausgelöst.

```
"filterGroups": [
  [
    {
      "type": "EVENT",
      "pattern": "PUSH, PULL_REQUEST_CREATED, PULL_REQUEST_UPDATED,
 PULL_REQUEST_MERGED"
    },
    {
      "type": "HEAD_REF",
      "pattern": "^refs/tags/.*",
      "excludeMatchedPattern": true
    }
  ]
]
```

Sie können einen Filter erstellen, der einen Build nur auslöst, wenn eine Änderung von einem GitLab Benutzer mit Konto-ID vorgenommen wird`actor-account-id`.

> **Note**
>
> Informationen dazu, wie Sie Ihre GitLab Konto-ID finden, finden Sie unter https://api.github.com/users/ *`benutzername`*, wobei *`benutzername`* für Ihren GitLab Benutzernamen steht.

```
"filterGroups": [
  [
    {
      "type": "EVENT",
      "pattern": "PUSH, PULL_REQUEST_CREATED, PULL_REQUEST_UPDATED,
 PULL_REQUEST_MERGED"
    },
    {
      "type": "ACTOR_ACCOUNT_ID",
      "pattern": "actor-account-id"
    }
  ]
```

```
]
```

Sie können einen Filter erstellen, der nur einen Build auslöst, wenn Dateien geändert werden, deren Namen dem regulären Ausdruck in dem Argument `pattern` entsprechen. In diesem Beispiel gibt die Filtergruppe an, dass nur ein Build ausgelöst wird, wenn Dateien geändert werden, deren Name dem regulären Ausdruck `^buildspec.*` entspricht.

```
"filterGroups": [
  [
    {
      "type": "EVENT",
      "pattern": "PUSH"
    },
    {
      "type": "FILE_PATH",
      "pattern": "^buildspec.*"
    }
  ]
]
```

In diesem Beispiel gibt die Filtergruppe an, dass ein Build nur ausgelöst wird, wenn Dateien in Ordnern `src` oder `test` geändert werden.

```
"filterGroups": [
    [
        {
            "type": "EVENT",
            "pattern": "PUSH"
        },
        {
            "type": "FILE_PATH",
            "pattern": "^src/.+|^test/.+"
        }
    ]
]
```

Sie können einen Filter erstellen, der einen Build nur dann auslöst, wenn die Head-Commit-Nachricht mit dem regulären Ausdruck im Musterargument übereinstimmt. In diesem Beispiel gibt die Filtergruppe an, dass ein Build nur dann ausgelöst wird, wenn die Head-Commit-Nachricht des Push-Ereignisses mit dem regulären Ausdruck `\[CodeBuild\]` übereinstimmt.

```
  "filterGroups": [
    [
      {
        "type": "EVENT",
        "pattern": "PUSH"
      },
      {
        "type": "COMMIT_MESSAGE",
        "pattern": "\[CodeBuild\]"
      }
    ]
  ]
```

GitLab Webhook-Ereignisse filtern ()AWS CloudFormation

Um eine AWS CloudFormation Vorlage zum Filtern von Webhook-Ereignissen zu verwenden, verwenden Sie die Eigenschaft des AWS CodeBuild `FilterGroups` Projekts. Mit dem folgenden in YAML formatierten Teil einer AWS CloudFormation -Vorlage werden zwei Filtergruppen erstellt. Diese lösen zusammen einen Build aus, wenn mindestens eine Gruppe als wahr ausgewertet wird.

- Die erste Filtergruppe gibt an, dass Pull-Requests für Branches mit Git-Referenznamen, die dem regulären Ausdruck entsprechen, `^refs/heads/main$` von einem GitLab Benutzer, der keine Konto-ID hat, erstellt oder aktualisiert `12345` werden.
- Die zweite Filtergruppe gibt an, dass Push-Anfragen in Verzweigungen mit Git-Referenznamen erstellt werden, die dem regulären Ausdruck `^refs/heads/.*` entsprechen.
- Die dritte Filtergruppe gibt eine Push-Anforderung mit einer Head Commit-Nachricht an, die dem regulären Ausdruck `\[CodeBuild\]` entspricht.

```
CodeBuildProject:
  Type: AWS::CodeBuild::Project
  Properties:
    Name: MyProject
    ServiceRole: service-role
    Artifacts:
      Type: NO_ARTIFACTS
    Environment:
      Type: LINUX_CONTAINER
      ComputeType: BUILD_GENERAL1_SMALL
      Image: aws/codebuild/standard:5.0
    Source:
```

```
      Type: GITLAB
      Location: source-location
    Triggers:
      Webhook: true
      FilterGroups:
        - - Type: EVENT
            Pattern: PULL_REQUEST_CREATED,PULL_REQUEST_UPDATED
          - Type: BASE_REF
            Pattern: ^refs/heads/main$
            ExcludeMatchedPattern: false
          - Type: ACTOR_ACCOUNT_ID
            Pattern: 12345
            ExcludeMatchedPattern: true
        - - Type: EVENT
            Pattern: PUSH
          - Type: HEAD_REF
            Pattern: ^refs/heads/.*
        - - Type: EVENT
            Pattern: PUSH
          - Type: COMMIT_MESSAGE
            Pattern: \[CodeBuild\]
```

Ändern der Einstellungen eines Build-Projekts in AWS CodeBuild

Sie können die AWS CodeBuild-Konsole, die AWS CLI oder AWS-SDKs zum Ändern der Projekteinstellungen verwenden.

Wenn Sie Testberichte zu einem Build-Projekt hinzufügen, stellen Sie sicher, dass Ihre IAM-Rolle über die in beschriebenen Berechtigungen verfügt.Arbeiten mit Testberichtberechtigungenaus.

Themen

- Ändern der Einstellungen eines Build-Projekts (Konsole)
- Ändern der Einstellungen eines Build-Projekts (AWS CLI)
- Ändern der Einstellungen eines Build-Projekts (AWS SDKs)

Ändern der Einstellungen eines Build-Projekts (Konsole)

Gehen Sie wie folgt vor, um die Einstellungen für ein Build-Projekt zu ändern:

1. Öffnen Sie die AWS CodeBuild Konsole unter https://console.aws.amazon.com/codesuite/codebuild/home.

2. Wählen Sie im linken Navigationsbereich Build projects aus.
3. Führen Sie eine der folgenden Aktionen aus:
   - Klicken Sie auf den Link des Build-Projekts, das Sie bearbeiten möchten, und klicken Sie dann auf Build details (Build-Details).
   - Aktivieren Sie das Optionsfeld neben dem Build-Projekt, das Sie ändern möchten, klicken Sie auf View details (Details anzeigen) und klicken Sie dann auf Build details (Build-Details).

Sie können die folgenden Abschnitte ändern:

Sections

- Konfiguration des Projekts
- Quelle
- Umgebung
- Buildspec
- Batch-Konfiguration
- -Artefakte
- Logs (Protokolle)

Konfiguration des Projekts

Wählen Sie im Abschnitt Projektkonfiguration die Option Bearbeiten aus. Wenn Ihre Änderungen abgeschlossen sind, wählen Sie Konfiguration aktualisieren, um die neue Konfiguration zu speichern.

Sie können die folgenden Eigenschaften ändern.

Beschreibung

Geben Sie optional eine Beschreibung des Build-Projekts ein, damit andere Benutzer verstehen, wofür dieses Projekt verwendet wird.

Badge erstellen

Wählen Sie Enable build badge (Build Badge aktivieren) aus, damit der Build-Status Ihres Projekts sichtbar und integrierbar ist. Weitere Informationen finden Sie unter Build Badges-Beispiel.

> **Note**
>
> Das Build-Badge gilt nicht, wenn Ihr Quellanbieter Amazon S3 ist.

Aktivieren Sie das Limit für gleichzeitige Builds

Wenn Sie die Anzahl der gleichzeitigen Builds für dieses Projekt einschränken möchten, führen Sie die folgenden Schritte aus:

1. Wählen Sie „Anzahl gleichzeitiger Builds einschränken“, die mit diesem Projekt gestartet werden können.
2. Geben Sie im Feld Limit für gleichzeitige Builds die maximale Anzahl gleichzeitiger Builds ein, die für dieses Projekt zulässig sind. Dieses Limit darf nicht höher sein als das Limit für gleichzeitige Builds, das für das Konto festgelegt wurde. Wenn Sie versuchen, eine Zahl einzugeben, die über dem Kontolimit liegt, wird eine Fehlermeldung angezeigt.

Neue Builds werden nur gestartet, wenn die aktuelle Anzahl der Builds dieses Limit unterschreitet oder ihm entspricht. Wenn die aktuelle Build-Anzahl dieses Limit erreicht, werden neue Builds gedrosselt und nicht ausgeführt.

Aktivieren Sie den öffentlichen Build-Zugriff

Um die Build-Ergebnisse Ihres Projekts der Öffentlichkeit zugänglich zu machen, einschließlich Benutzern ohne Zugriff auf ein AWS Konto, wählen Sie Öffentlichen Build-Zugriff aktivieren und bestätigen Sie, dass Sie die Build-Ergebnisse veröffentlichen möchten. Die folgenden Eigenschaften werden für öffentliche Build-Projekte verwendet:

Rolle im Public-Build-Dienst

Wählen Sie Neue Servicerolle aus, wenn Sie eine neue Servicerolle für Sie CodeBuild erstellen möchten, oder Bestehende Servicerolle, wenn Sie eine bestehende Servicerolle verwenden möchten.

Die öffentliche Build-Servicerolle ermöglicht es CodeBuild , die CloudWatch Logs zu lesen und die Amazon S3 S3-Artefakte für die Builds des Projekts herunterzuladen. Dies ist erforderlich, um die Build-Logs und Artefakte des Projekts der Öffentlichkeit zugänglich zu machen.

Rolle im Dienst

Geben Sie den Namen der neuen oder einer vorhandenen Servicerolle ein.

Um die Build-Ergebnisse Ihres Projekts privat zu machen, deaktivieren Sie die Option Öffentlichen Build-Zugriff aktivieren.

Weitere Informationen finden Sie unter Öffentliche Build-Projekte inAWS CodeBuild.

> **Warning**
>
> Wenn Sie die Build-Ergebnisse Ihres Projekts veröffentlichen, sollten Sie Folgendes beachten:
>
> - Alle Build-Ergebnisse, Logs und Artefakte eines Projekts, einschließlich Builds, die ausgeführt wurden, als das Projekt privat war, sind öffentlich zugänglich.
> - Alle Build-Logs und Artefakte sind öffentlich zugänglich. Umgebungsvariablen, Quellcode und andere vertrauliche Informationen wurden möglicherweise in die Build-Logs und -Artefakte ausgegeben. Sie müssen vorsichtig sein, welche Informationen in die Build-Logs ausgegeben werden. Einige bewährte Methoden sind:
>   - Speichern Sie keine sensiblen Werte, insbesondere AWS Zugriffsschlüssel-IDs und geheime Zugriffsschlüssel, in Umgebungsvariablen. Wir empfehlen, einen Amazon EC2 Systems Manager Manager-Parameterspeicher zu verwenden oder sensible Werte AWS Secrets Manager zu speichern.
>   - Folgen Sie diesen Regeln, Bewährte Methoden für die Verwendung von Webhooks um einzuschränken, welche Entitäten einen Build auslösen können, und speichern Sie die Buildspec nicht im Projekt selbst, um sicherzustellen, dass Ihre Webhooks so sicher wie möglich sind.
> - Ein böswilliger Benutzer kann öffentliche Builds verwenden, um bösartige Artefakte zu verteilen. Wir empfehlen, dass Projektadministratoren alle Pull-Requests überprüfen, um sicherzustellen, dass es sich bei der Pull-Anfrage um eine legitime Änderung handelt. Wir empfehlen außerdem, dass Sie alle Artefakte anhand ihrer Prüfsummen überprüfen, um sicherzustellen, dass die richtigen Artefakte heruntergeladen werden.

Zusätzliche Informationen

Geben Sie unter Tags den Namen und den Wert aller Tags ein, die von den unterstützenden AWS Diensten verwendet werden sollen. Verwenden Sie Add row, um ein Tag hinzuzufügen. Sie können bis zu 50 Tags hinzufügen.

Quelle

Wählen Sie im Bereich Quelle die Option Bearbeiten aus. Wenn Ihre Änderungen abgeschlossen sind, wählen Sie Konfiguration aktualisieren, um die neue Konfiguration zu speichern.

Sie können die folgenden Eigenschaften ändern:

Quellanbieter

Wählen Sie den Typ des Quellcode-Anbieters. Verwenden Sie die folgenden Listen, um die für Ihren Quellanbieter geeignete Auswahl zu treffen:

> **Note**
>
> CodeBuild unterstützt Bitbucket Server nicht.

Amazon S3

Bucket

Wählen Sie den Namen des Eingabe-Buckets, der den Quellcode enthält.

S3-Objektschlüssel oder S3-Ordner

Geben Sie den Namen der ZIP-Datei oder den Pfad zu dem Ordner ein, der den Quellcode enthält. Geben Sie einen Vorwärtsschrägstrich (/) ein, um den gesamten Inhalt im S3-Bucket herunterzuladen.

Quellversion

Geben Sie die Versions-ID des Objekts ein, das den Build Ihrer Eingabedatei darstellt. Weitere Informationen finden Sie unter Beispiel für eine Quellversion mit AWS CodeBuild.

CodeCommit

Repository

Wählen Sie das Repository aus, das Sie verwenden möchten.

Art der Referenz

Wählen Sie Branch, Git-Tag oder Commit ID, um die Version Ihres Quellcodes anzugeben. Weitere Informationen finden Sie unter Beispiel für eine Quellversion mit AWS CodeBuild.

> **Note**
>
> Wir empfehlen, Git-Branchnamen zu wählen, die nicht wie Commit-IDs aussehen, wie zum Beispiel `811dd1ba1aba14473856cee38308caed7190c0d` oder`5392f7`. Dies hilft dir dabei, Git-Checkout-Kollisionen mit tatsächlichen Commits zu vermeiden.

Tiefe des Git-Klonens

Wählen Sie aus, ob Sie einen flachen Klon erstellen möchten, dessen Verlauf auf die angegebene Anzahl von Commits gekürzt ist. Wenn Sie einen vollständigen Klon erstellen möchten, wählen Sie Full (Vollständig) aus.

Git-Submodule

Wählen Sie Use Git submodules (Git-Untermodule verwenden) aus, wenn Ihr Repository Git-Untermodule enthalten soll.

Bitbucket

Repository

Wähle Mit OAuth Connect oder Mit einem Bitbucket-App-Passwort verbinden und befolge die Anweisungen, um dich mit Bitbucket zu verbinden (oder erneut zu verbinden).

Wähle ein öffentliches Repository oder ein Repository in deinem Konto.

Quellversion

Geben Sie einen Zweig, eine Commit-ID, ein Tag oder eine Referenz und eine Commit-ID ein. Weitere Informationen finden Sie unter Beispiel für eine Quellversion mit AWS CodeBuild.

> **Note**
>
> Wir empfehlen, Git-Branchnamen zu wählen, die nicht wie Commit-IDs aussehen, wie zum Beispiel `811dd1ba1aba14473856cee38308caed7190c0d` oder`5392f7`. Dies hilft dir dabei, Git-Checkout-Kollisionen mit tatsächlichen Commits zu vermeiden.

Tiefe des Git-Klonens

Wählen Sie Git clone depth (Git-Klontiefe) aus, um einen flachen Klon mit einem Verlauf zu erstellen, der auf die angegebene Anzahl von Commits gekürzt ist. Wenn Sie einen vollständigen Klon erstellen möchten, wählen Sie Full (Vollständig) aus.

Git-Submodule

Wählen Sie Use Git submodules (Git-Untermodule verwenden) aus, wenn Ihr Repository Git-Untermodule enthalten soll.

Build-Status

Wählen Sie Build-Status beim Start und Ende Ihrer Builds an den Quellanbieter melden, wenn Sie möchten, dass der Status des Beginns und Abschlusses Ihres Builds Ihrem Quellanbieter gemeldet wird.

Um dem Quellanbieter den Buildstatus melden zu können, muss der mit dem Quellanbieter verknüpfte Benutzer Schreibzugriff auf das Repository haben. Wenn der Benutzer keinen Schreibzugriff hat, kann der Build-Status nicht aktualisiert werden. Weitere Informationen finden Sie unter Zugriff auf den Quellanbieter.

Geben Sie für den Statuskontext den Wert ein, der für den `name` Parameter im Bitbucket-Commit-Status verwendet werden soll. Weitere Informationen finden Sie unter Build in der Bitbucket-API-Dokumentation.

Geben Sie unter Ziel-URL den Wert ein, der für den `url` Parameter im Bitbucket-Commit-Status verwendet werden soll. Weitere Informationen finden Sie unter Build in der Bitbucket-API-Dokumentation.

Der Status eines durch einen Webhook ausgelösten Builds wird immer an den Quellanbieter gemeldet. Damit der Status eines Builds, der von der Konsole aus gestartet wird, oder eines API-Aufrufs an den Quellanbieter gemeldet wird, müssen Sie diese Einstellung auswählen.

Wenn die Builds Ihres Projekts durch einen Webhook ausgelöst werden, müssen Sie einen neuen Commit an das Repo übertragen, damit eine Änderung an dieser Einstellung wirksam wird.

Wählen Sie unter Primäre Quell-Webhook-Ereignisse die Option Jedes Mal neu erstellen, wenn eine Codeänderung in dieses Repository übertragen wird, wenn Sie den Quellcode jedes

Mal erstellen CodeBuild möchten, wenn eine Codeänderung in dieses Repository übertragen wird. Weitere Hinweise zu Webhooks und Filtergruppen finden Sie unter. Bitbucket-Webhook-Ereignisse

GitHub

Repository

Wählen Sie Mit OAuth Connect oder Mit einem GitHub persönlichen Zugriffstoken verbinden und folgen Sie den Anweisungen, um eine Verbindung herzustellen (oder erneut herzustellen) GitHub und den Zugriff zu autorisieren. AWS CodeBuild

Wählen Sie ein öffentliches Repository oder ein Repository in Ihrem Konto.

Quellversion

Geben Sie einen Zweig, eine Commit-ID, ein Tag oder eine Referenz und eine Commit-ID ein. Weitere Informationen finden Sie unter Beispiel für eine Quellversion mit AWS CodeBuild.

> **Note**
>
> Wir empfehlen, Git-Branchnamen zu wählen, die nicht wie Commit-IDs aussehen, wie zum Beispiel `811dd1ba1aba14473856cee38308caed7190c0d` oder`5392f7`. Dies hilft dir dabei, Git-Checkout-Kollisionen mit tatsächlichen Commits zu vermeiden.

Tiefe des Git-Klonens

Wählen Sie Git clone depth (Git-Klontiefe) aus, um einen flachen Klon mit einem Verlauf zu erstellen, der auf die angegebene Anzahl von Commits gekürzt ist. Wenn Sie einen vollständigen Klon erstellen möchten, wählen Sie Full (Vollständig) aus.

Git-Submodule

Wählen Sie Use Git submodules (Git-Untermodule verwenden) aus, wenn Ihr Repository Git-Untermodule enthalten soll.

Build-Status

Wählen Sie Build-Status beim Start und Ende Ihrer Builds an den Quellanbieter melden, wenn Sie möchten, dass der Status des Beginns und Abschlusses Ihres Builds Ihrem Quellanbieter gemeldet wird.

Um dem Quellanbieter den Buildstatus melden zu können, muss der mit dem Quellanbieter verknüpfte Benutzer Schreibzugriff auf das Repository haben. Wenn der Benutzer keinen Schreibzugriff hat, kann der Build-Status nicht aktualisiert werden. Weitere Informationen finden Sie unter Zugriff auf den Quellanbieter.

Geben Sie für den Statuskontext den Wert ein, der für den `context` Parameter im GitHub Commit-Status verwendet werden soll. Weitere Informationen finden Sie im GitHub Entwicklerhandbuch unter Einen Commit-Status erstellen.

Geben Sie für Ziel-URL den Wert ein, der für den `target_url` Parameter im GitHub Commit-Status verwendet werden soll. Weitere Informationen finden Sie im GitHub Entwicklerhandbuch unter Einen Commit-Status erstellen.

Der Status eines durch einen Webhook ausgelösten Builds wird immer an den Quellanbieter gemeldet. Damit der Status eines Builds, der von der Konsole aus gestartet wird, oder eines API-Aufrufs an den Quellanbieter gemeldet wird, müssen Sie diese Einstellung auswählen.

Wenn die Builds Ihres Projekts durch einen Webhook ausgelöst werden, müssen Sie einen neuen Commit an das Repo übertragen, damit eine Änderung an dieser Einstellung wirksam wird.

Wählen Sie unter Primäre Quell-Webhook-Ereignisse die Option Jedes Mal neu erstellen, wenn eine Codeänderung in dieses Repository übertragen wird, wenn Sie den Quellcode jedes Mal erstellen CodeBuild möchten, wenn eine Codeänderung in dieses Repository übertragen wird. Weitere Hinweise zu Webhooks und Filtergruppen finden Sie unter. GitHub Webhook-Ereignisse

GitHub Enterprise Server

GitHub Persönliches Zugriffstoken für Unternehmen

Informationen darüber, wie Sie GitHub Beispiel für einen Enterprise Server ein persönliches Zugriffstoken in Ihre Zwischenablage kopieren, finden Sie unter. Fügen Sie das Token in das Textfeld ein und wählen Sie anschließend Save Token (Token speichern) aus.

> **Note**
>
> Sie müssen das private Zugriffstoken nur einmal eingeben und speichern. CodeBuild verwendet dieses Token in allen future Projekten.

Quellversion

Geben Sie eine Pull-Anfrage, einen Branch, eine Commit-ID, ein Tag oder eine Referenz und eine Commit-ID ein. Weitere Informationen finden Sie unter Beispiel für eine Quellversion mit AWS CodeBuild.

> **Note**
>
> Wir empfehlen, Git-Branchnamen zu wählen, die nicht wie Commit-IDs aussehen, wie zum Beispiel `811dd1ba1aba14473856cee38308caed7190c0d` oder`5392f7`. Dies hilft dir dabei, Git-Checkout-Kollisionen mit tatsächlichen Commits zu vermeiden.

Tiefe des Git-Klonens

Wählen Sie Git clone depth (Git-Klontiefe) aus, um einen flachen Klon mit einem Verlauf zu erstellen, der auf die angegebene Anzahl von Commits gekürzt ist. Wenn Sie einen vollständigen Klon erstellen möchten, wählen Sie Full (Vollständig) aus.

Git-Submodule

Wählen Sie Use Git submodules (Git-Untermodule verwenden) aus, wenn Ihr Repository Git-Untermodule enthalten soll.

Build-Status

Wählen Sie Build-Status beim Start und Ende Ihrer Builds an den Quellanbieter melden, wenn Sie möchten, dass der Status des Beginns und Abschlusses Ihres Builds Ihrem Quellanbieter gemeldet wird.

Um dem Quellanbieter den Buildstatus melden zu können, muss der mit dem Quellanbieter verknüpfte Benutzer Schreibzugriff auf das Repository haben. Wenn der Benutzer keinen Schreibzugriff hat, kann der Build-Status nicht aktualisiert werden. Weitere Informationen finden Sie unter Zugriff auf den Quellanbieter.

Geben Sie für den Statuskontext den Wert ein, der für den `context` Parameter im GitHub Commit-Status verwendet werden soll. Weitere Informationen finden Sie im GitHub Entwicklerhandbuch unter Einen Commit-Status erstellen.

Geben Sie für Ziel-URL den Wert ein, der für den `target_url` Parameter im GitHub Commit-Status verwendet werden soll. Weitere Informationen finden Sie im GitHub Entwicklerhandbuch unter Einen Commit-Status erstellen.

Der Status eines durch einen Webhook ausgelösten Builds wird immer an den Quellanbieter gemeldet. Damit der Status eines Builds, der von der Konsole aus gestartet wird, oder eines API-Aufrufs an den Quellanbieter gemeldet wird, müssen Sie diese Einstellung auswählen.

Wenn die Builds Ihres Projekts durch einen Webhook ausgelöst werden, müssen Sie einen neuen Commit an das Repo übertragen, damit eine Änderung an dieser Einstellung wirksam wird.

Unsicheres SSL

Wählen Sie Unsicheres SSL aktivieren, um SSL-Warnungen zu ignorieren, während Sie eine Verbindung zu Ihrem GitHub Enterprise-Projekt-Repository herstellen.

Wählen Sie unter Primäre Quell-Webhook-Ereignisse die Option Jedes Mal neu erstellen, wenn eine Codeänderung in dieses Repository übertragen wird, wenn Sie den Quellcode jedes Mal neu erstellen CodeBuild möchten, wenn eine Codeänderung in dieses Repository übertragen wird. Weitere Hinweise zu Webhooks und Filtergruppen finden Sie unter. GitHub Webhook-Ereignisse

GitLab

Connection (Verbindung)

Connect dein GitLab Konto über und verwende die Verbindung AWS CodeConnections, um dein Drittanbieter-Repository als Quelle für dein Build-Projekt zu verknüpfen.

Wählen Sie Standardverbindung oder Benutzerdefinierte Verbindung.

Standardverbindung wendet eine GitLab Standardverbindung für alle Projekte an. Benutzerdefinierte Verbindung wendet eine benutzerdefinierte GitLab Verbindung an, die die Standardeinstellungen Ihres Kontos überschreibt.

Standardverbindung

Der Name der Standardverbindung, die mit Ihrem Konto verknüpft ist.

Wenn Sie noch keine Verbindung zu Ihrem Anbieter hergestellt haben, finden Sie Stellen Sie eine Verbindung zu GitLab (Konsole) her eine Anleitung unter.

Benutzerdefinierte Verbindung

Wählen Sie den Namen der benutzerdefinierten Verbindung, die Sie verwenden möchten.

Wenn Sie noch keine Verbindung zu Ihrem Anbieter hergestellt haben, finden Sie Stellen Sie eine Verbindung zu GitLab (Konsole) her Anweisungen unter.

Repository

Wählen Sie das Repository aus, das Sie verwenden möchten.

Quellversion

Geben Sie eine Pull-Request-ID, einen Branch, eine Commit-ID, ein Tag oder eine Referenz und eine Commit-ID ein. Weitere Informationen finden Sie unter Beispiel für eine Quellversion mit AWS CodeBuild.

> **Note**
>
> Wir empfehlen, Git-Branchnamen zu wählen, die nicht wie Commit-IDs aussehen, wie zum Beispiel `811dd1ba1aba14473856cee38308caed7190c0d` oder`5392f7`. Dies hilft dir dabei, Git-Checkout-Kollisionen mit tatsächlichen Commits zu vermeiden.

Tiefe des Git-Klonens

Wählen Sie Git clone depth (Git-Klontiefe) aus, um einen flachen Klon mit einem Verlauf zu erstellen, der auf die angegebene Anzahl von Commits gekürzt ist. Wenn Sie einen vollständigen Klon erstellen möchten, wählen Sie Full (Vollständig) aus.

Build-Status

Wählen Sie Build-Status beim Start und Ende Ihrer Builds an den Quellanbieter melden, wenn Sie möchten, dass der Status des Beginns und Abschlusses Ihres Builds Ihrem Quellanbieter gemeldet wird.

Um dem Quellanbieter den Buildstatus melden zu können, muss der mit dem Quellanbieter verknüpfte Benutzer Schreibzugriff auf das Repository haben. Wenn der Benutzer keinen Schreibzugriff hat, kann der Build-Status nicht aktualisiert werden. Weitere Informationen finden Sie unter Zugriff auf den Quellanbieter.

GitLab Self Managed

Connection (Verbindung)

Connect dein GitLab Konto über und verwende die Verbindung AWS CodeConnections, um dein Drittanbieter-Repository als Quelle für dein Build-Projekt zu verknüpfen.

Wählen Sie Standardverbindung oder Benutzerdefinierte Verbindung.

Standardverbindung wendet eine GitLab selbstverwaltete Standardverbindung für alle Projekte an. Benutzerdefinierte Verbindung wendet eine benutzerdefinierte GitLab selbstverwaltete Verbindung an, die die Standardeinstellungen Ihres Kontos überschreibt.

Standardverbindung

Der Name der Standardverbindung, die mit Ihrem Konto verknüpft ist.

Falls Sie noch keine Verbindung zu Ihrem Anbieter hergestellt haben, finden Sie im Benutzerhandbuch der Developer Tools Console unter Erstellen einer Verbindung zu GitLab Self-Managed eine Anleitung.

Benutzerdefinierte Verbindung

Wählen Sie den Namen der benutzerdefinierten Verbindung, die Sie verwenden möchten.

Wenn Sie noch keine Verbindung zu Ihrem Anbieter hergestellt haben, finden Sie im Benutzerhandbuch der Developer Tools Console unter Erstellen einer Verbindung zu GitLab Self-Managed eine Anleitung.

Repository

Wählen Sie das Repository aus, das Sie verwenden möchten.

Quellversion

Geben Sie eine Pull-Request-ID, einen Branch, eine Commit-ID, ein Tag oder eine Referenz und eine Commit-ID ein. Weitere Informationen finden Sie unter Beispiel für eine Quellversion mit AWS CodeBuild.

> **Note**
>
> Wir empfehlen, Git-Branchnamen zu wählen, die nicht wie Commit-IDs aussehen, wie zum Beispiel `811dd1ba1aba14473856cee38308caed7190c0d` oder`5392f7`. Dies hilft dir dabei, Git-Checkout-Kollisionen mit tatsächlichen Commits zu vermeiden.

Tiefe des Git-Klonens

Wählen Sie Git clone depth (Git-Klontiefe) aus, um einen flachen Klon mit einem Verlauf zu erstellen, der auf die angegebene Anzahl von Commits gekürzt ist. Wenn Sie einen vollständigen Klon erstellen möchten, wählen Sie Full (Vollständig) aus.

Build-Status

Wählen Sie Build-Status beim Start und Ende Ihrer Builds an den Quellanbieter melden, wenn Sie möchten, dass der Status des Beginns und Abschlusses Ihres Builds Ihrem Quellanbieter gemeldet wird.

Um dem Quellanbieter den Buildstatus melden zu können, muss der mit dem Quellanbieter verknüpfte Benutzer Schreibzugriff auf das Repository haben. Wenn der Benutzer keinen Schreibzugriff hat, kann der Build-Status nicht aktualisiert werden. Weitere Informationen finden Sie unter Zugriff auf den Quellanbieter.

Umgebung

Wählen Sie im Bereich Umgebung die Option Bearbeiten aus. Wenn Ihre Änderungen abgeschlossen sind, wählen Sie Konfiguration aktualisieren, um die neue Konfiguration zu speichern.

Sie können die folgenden Eigenschaften ändern:

Bereitstellungsmodell

Um das Bereitstellungsmodell zu ändern, wählen Sie Bereitstellungsmodell ändern aus und führen Sie einen der folgenden Schritte aus:

- Um On-Demand-Flotten zu verwenden, die von verwaltet werden AWS CodeBuild, wählen Sie On-Demand. CodeBuild Bietet mit On-Demand-Flotten Rechenleistung für Ihre Builds. Die Maschinen werden zerstört, wenn der Bau abgeschlossen ist. On-Demand-Flotten werden vollständig verwaltet und verfügen über automatische Skalierungsfunktionen zur Bewältigung von Nachfragespitzen.
- Um Flotten mit reservierter Kapazität zu verwenden, die von verwaltet werden AWS CodeBuild, wählen Sie Reservierte Kapazität und dann einen Flottennamen aus. Bei Flotten mit reservierter Kapazität konfigurieren Sie eine Reihe von dedizierten Instances für Ihre Build-Umgebung. Diese Maschinen bleiben inaktiv und sind bereit, Builds oder Tests sofort zu verarbeiten, wodurch die Build-Dauer reduziert wird. Mit Flotten mit reservierter Kapazität sind Ihre Maschinen immer in Betrieb und es fallen weiterhin Kosten an, solange sie bereitgestellt werden.

Weitere Informationen finden Sie unter Arbeiten mit reservierter Kapazität in AWS CodeBuild.

Bild der Umgebung

Um das Build-Image zu ändern, wählen Sie Override image und führen Sie einen der folgenden Schritte aus:

- Um ein Docker-Image zu verwenden, das von verwaltet wird AWS CodeBuild, wählen Sie Verwaltetes Image aus und treffen Sie dann eine Auswahl unter Betriebssystem, Runtime (s), Image und Image-Version. Treffen Sie eine Auswahl unter Environment type (Umgebungstyp), sofern verfügbar.
- Wenn Sie ein anderes Docker-Image verwenden möchten, wählen Sie Custom image (Benutzerdefiniertes Image) aus. Wählen Sie als Umgebungstyp ARM, Linux, Linux GPU oder Windows aus. Wenn Sie Andere Registrierung wählen, geben Sie für Externe Registrierungs-URL den Namen und das Tag des Docker-Images in Docker Hub ein. Verwenden Sie dabei das Format. *`docker repository`*/*`docker image name`* Wenn Sie sich für Amazon ECR entscheiden, verwenden Sie das Amazon ECR-Repository und das Amazon ECR-Image, um das Docker-Image in Ihrem Konto auszuwählen. AWS
- Um ein privates Docker-Image zu verwenden, wählen Sie Benutzerdefiniertes Image. Wählen Sie als Umgebungstyp ARM, Linux, Linux GPU oder Windows aus. Wählen Sie unter Image registry (Abbildregistrierung) die Option Other registry (Andere Registrierung) aus und geben Sie dann den ARN der Anmeldeinformationen für Ihr privates Docker-Image ein. Die Anmeldeinformationen müssen von Secrets Manager erstellt werden. Weitere Informationen finden Sie unter Was ist AWS Secrets Manager? im AWS Secrets Manager - Benutzerhandbuch.

> **Note**
>
> CodeBuild überschreibt die `ENTRYPOINT` für benutzerdefinierte Docker-Images.

Servicerolle

Führen Sie eine der folgenden Aktionen aus:

- Wenn Sie keine CodeBuild Servicerolle haben, wählen Sie Neue Servicerolle. Geben Sie im Feld Rollenname einen Namen für die neue Rolle ein.
- Wenn Sie eine CodeBuild Servicerolle haben, wählen Sie Bestehende Servicerolle aus. Wählen Sie unter Role ARN die Servicerolle aus.

> **Note**
>
> Wenn Sie die Konsole verwenden, um ein Build-Projekt zu erstellen, können Sie gleichzeitig eine CodeBuild Servicerolle erstellen. In der Standardeinstellung funktioniert diese Rolle ausschließlich mit diesem Projekt. Wenn Sie die Konsole verwenden, um die Servicerolle mit einem anderen Build-Projekt zu verknüpfen, wird die Rolle so aktualisiert, dass sie mit dem anderen Build-Projekt funktioniert. Eine Servicerolle kann in bis zu zehn Build-Projekten verwendet werden.

Zusätzliche Konfiguration

Timeout (Zeitüberschreitung)

Geben Sie einen Wert zwischen 5 Minuten und 36 Stunden an. Danach wird der Build CodeBuild gestoppt, falls er nicht abgeschlossen ist. Wenn Sie die Felder hours und minutes leer lassen, wird der Standardwert von 60 Minuten verwendet.

Privilegiert

Wählen Sie Dieses Flag aktivieren aus, wenn Sie Docker-Images erstellen oder Ihren Builds erweiterte Rechte gewähren möchten. nur, wenn Sie dieses Build-Projekt zum Erstellen von Docker-Images verwenden möchten. Andernfalls schlagen alle zugehörigen Builds fehl, die versuchen, mit dem Docker-Daemon zu interagieren. Sie müssen zudem den Docker-Daemon müssen, damit Ihre Builds interagieren können. Eine Möglichkeit, dies durchzuführen, besteht darin, den Docker-Daemon in der `install`-Phase Ihrer Build-Spezifikation zu initialisieren, indem Sie die folgenden Build-Befehle ausführen. Führen Sie diese Befehle nicht aus, wenn Sie sich für ein Image der Build-Umgebung entschieden haben, das von CodeBuild mit Docker-Unterstützung bereitgestellt wird.

> **Note**
>
> Standardmäßig ist der Docker-Daemon für Nicht-VPC-Builds aktiviert. Wenn Sie Docker-Container für VPC-Builds verwenden möchten, lesen Sie auf der Docker Docs-Website unter Runtime Privilege and Linux Capabilities nach und aktivieren Sie den privilegierten Modus. Außerdem unterstützt Windows den privilegierten Modus nicht.

```
- nohup /usr/local/bin/dockerd --host=unix:///var/run/docker.sock --
host=tcp://127.0.0.1:2375 --storage-driver=overlay2 &
```

```
- timeout 15 sh -c "until docker info; do echo .; sleep 1; done"
```

VPC

Wenn Sie mit Ihrer VPC arbeiten möchten CodeBuild :

- Wählen Sie für VPC die VPC-ID aus, die CodeBuild verwendet wird.
- Wählen Sie für VPC-Subnetze die Subnetze aus, die Ressourcen enthalten, die verwendet werden. CodeBuild
- Wählen Sie für VPC-Sicherheitsgruppen die Sicherheitsgruppen aus, die CodeBuild den Zugriff auf Ressourcen in den VPCs ermöglichen.

Weitere Informationen finden Sie unter Verwendung AWS CodeBuild mit Amazon Virtual Private Cloud.

Datenverarbeitung

Wählen Sie eine der verfügbaren Optionen.

Umgebungsvariablen

Geben Sie den Namen und den Wert ein, und wählen Sie dann den Typ der einzelnen Umgebungsvariablen aus, die von Builds verwendet werden sollen.

> **Note**
>
> CodeBuild legt die Umgebungsvariable für Ihre AWS Region automatisch fest. Sie müssen die folgenden Umgebungsvariablen festlegen, wenn Sie sie nicht zu Ihrer buildspec.yml hinzugefügt haben:
>
> - AWS_ACCOUNT_ID
> - IMAGE_REPO_NAME
> - IMAGE_TAG

Konsole und AWS CLI Benutzer können Umgebungsvariablen sehen. Wenn Sie keine Bedenken hinsichtlich der Sichtbarkeit Ihrer Umgebungsvariablen haben, stellen Sie die Felder Name und Value ein und legen Sie dann den Type auf Plaintext fest.

Wir empfehlen, dass Sie eine Umgebungsvariable mit einem sensiblen Wert wie einer AWS Zugriffsschlüssel-ID, einem AWS geheimen Zugriffsschlüssel oder einem Passwort als

Parameter in Amazon EC2 Systems Manager Parameter Store oder AWS Secrets Manager speichern.

Wenn Sie Amazon EC2 Systems Manager Parameter Store verwenden, wählen Sie als Typ die Option Parameter. Geben Sie unter Name einen Bezeichner ein, CodeBuild auf den verwiesen werden soll. Geben Sie für Value den Namen des Parameters ein, wie er im Amazon EC2 Systems Manager Parameter Store gespeichert ist. Verwenden Sie beispielsweise einen Parameter mit der Bezeichnung `/CodeBuild/dockerLoginPassword` und wählen Sie für Type (Typ) Parameter Store aus. Geben Sie unter Name `LOGIN_PASSWORD` ein. Geben Sie für Wert `/CodeBuild/dockerLoginPassword` ein.

> **Important**
>
> Wenn Sie Amazon EC2 Systems Manager Parameter Store verwenden, empfehlen wir, Parameter mit Parameternamen zu speichern, die mit `/CodeBuild/` (z. B. `/CodeBuild/dockerLoginPassword`) beginnen. Sie können die CodeBuild Konsole verwenden, um einen Parameter in Amazon EC2 Systems Manager zu erstellen. Wählen Sie Create a parameter (Parameter erstellen) aus und befolgen Sie dann die Anweisungen im Dialogfeld. (In diesem Dialogfeld können Sie für KMS-Schlüssel den ARN eines AWS KMS Schlüssels in Ihrem Konto angeben. Amazon EC2 Systems Manager verwendet diesen Schlüssel, um den Wert des Parameters beim Speichern zu verschlüsseln und beim Abrufen zu entschlüsseln.) Wenn Sie die CodeBuild Konsole verwenden, um einen Parameter zu erstellen, beginnt die Konsole den Parameternamen mit dem, `/CodeBuild/` wie er gespeichert wird. Weitere Informationen finden Sie unter Systems Manager Parameter Store und Systems Manager Parameter Store Console Walkthrough im Amazon EC2 Systems Manager Manager-Benutzerhandbuch.
>
> Wenn sich Ihr Build-Projekt auf Parameter Store von Amazon EC2 Systems Manager bezieht, muss die Service-Rolle des Build-Projekts die `ssm:GetParameters` Aktion zulassen. Wenn Sie zuvor Neue Servicerolle ausgewählt haben, wird CodeBuild diese Aktion in die Standard-Servicerolle für Ihr Build-Projekt aufgenommen. Wenn Sie jedoch Existing service role (Vorhandene Servicerolle) ausgewählt haben, müssen Sie diese Aktion separat in Ihre Servicerolle aufnehmen.
>
> Wenn sich Ihr Build-Projekt auf Parameter bezieht, die im Amazon EC2 Systems Manager Parameter Store mit Parameternamen gespeichert sind, die nicht mit beginnen`/CodeBuild/`, und Sie Neue Servicerolle wählen, müssen Sie diese Servicerolle aktualisieren, um Zugriff auf Parameternamen zu gewähren, die nicht

mit `/CodeBuild/` beginnen. Dies liegt daran, dass diese Service-Rolle nur auf Parameternamen zugreift, die mit `/CodeBuild/` beginnen.
Wenn Sie Neue Servicerolle wählen, beinhaltet die Servicerolle die Berechtigung, alle Parameter unter dem `/CodeBuild/` Namespace im Amazon EC2 Systems Manager Parameter Store zu entschlüsseln.
Von Ihnen gesetzte Umgebungsvariablen ersetzen vorhandene Umgebungsvariablen. Wenn das Docker-Image beispielsweise bereits eine Umgebungsvariable mit dem Namen `MY_VAR` und einem Wert von `my_value` enthält und Sie eine Umgebungsvariable mit dem Namen `MY_VAR` und einem Wert von `other_value` festlegen, wird `my_value` durch `other_value` ersetzt. Wenn das Docker-Image demgegenüber bereits eine Umgebungsvariable mit dem Namen `PATH` und einem Wert von `/usr/local/sbin:/usr/local/bin` enthält und Sie eine Umgebungsvariable mit dem Namen `PATH` und einem Wert von `$PATH:/usr/share/ant/bin` festlegen, wird `/usr/local/sbin:/usr/local/bin` durch den Literalwert `$PATH:/usr/share/ant/bin` ersetzt.
Legen Sie keine Umgebungsvariable mit einem Namen fest, der mit `CODEBUILD_` beginnt. Dieses Präfix ist zur -internen Verwendung reserviert.
Wenn eine Umgebungsvariable mit identischem Namen an mehreren Orten definiert ist, wird der Wert folgendermaßen bestimmt:

- Der Wert im Aufruf zum Starten des Build-Vorgangs hat den höchsten Vorrang.
- Der Wert in der Build-Projektdefinition folgt darauf.
- Der Wert in der buildspec-Deklaration hat die niedrigste Priorität.

Wenn Sie Secrets Manager verwenden, wählen Sie für Type Secrets Manager. Geben Sie unter Name einen Bezeichner ein, auf CodeBuild den verwiesen werden soll. Geben Sie unter Wert einen `reference-key` mit dem Muster *`secret-id`*`:`*`json-key`*`:`*`version-stage`*`:`*`version-id`* ein. Weitere Informationen finden Sie unter Secrets Manager reference-key in the buildspec file.

Important

Wenn Sie Secrets Manager verwenden, empfehlen wir, Secrets mit Namen zu speichern, die mit `/CodeBuild/` (z. B.`/CodeBuild/dockerLoginPassword`) beginnen. Weitere Informationen finden Sie unter Was ist AWS Secrets Manager? im AWS Secrets Manager -Benutzerhandbuch.

Wenn sich Ihr Build-Projekt auf Geheimnisse bezieht, die in Secrets Manager gespeichert sind, muss die Service-Rolle des Build-Projekts die `secretsmanager:GetSecretValue` Aktion zulassen. Wenn Sie zuvor Neue Servicerolle ausgewählt haben, wird CodeBuild diese Aktion in die Standard-Servicerolle für Ihr Build-Projekt aufgenommen. Wenn Sie jedoch Existing service role (Vorhandene Servicerolle) ausgewählt haben, müssen Sie diese Aktion separat in Ihre Servicerolle aufnehmen.
Wenn sich Ihr Build-Projekt auf Geheimnisse bezieht, die in Secrets Manager mit geheimen Namen gespeichert sind, die nicht mit beginnen`/CodeBuild/`, und Sie Neue Dienstrolle ausgewählt haben, müssen Sie die Servicerolle aktualisieren, um Zugriff auf geheime Namen zu ermöglichen, die nicht mit beginnen`/CodeBuild/`. Dies liegt daran, dass die Servicerolle nur den Zugriff auf geheime Namen ermöglicht, die mit beginnen`/CodeBuild/`.
Wenn Sie Neue Dienstrolle wählen, beinhaltet die Dienstrolle die Berechtigung, alle Geheimnisse unter dem `/CodeBuild/` Namespace im Secrets Manager zu entschlüsseln.

Buildspec

Wählen Sie im Abschnitt Buildspec die Option Bearbeiten aus. Wenn Ihre Änderungen abgeschlossen sind, wählen Sie Konfiguration aktualisieren, um die neue Konfiguration zu speichern.

Sie können die folgenden Eigenschaften ändern:

Spezifikationen erstellen

Führen Sie eine der folgenden Aktionen aus:

- Wenn Ihr Quellcode eine buildspec-Datei enthält, wählen Sie Use a buildspec file (Eine buildspec-Datei verwenden) aus. Sucht standardmäßig nach einer Datei mit dem Namen `buildspec.yml` im Quellcode-Stammverzeichnis. CodeBuild Wenn Ihre Buildspec-Datei einen anderen Namen oder Speicherort verwendet, geben Sie ihren Pfad vom Quellstammverzeichnis aus im Feld Buildspec-Name ein (zum Beispiel oder. `buildspec-two.yml configuration/buildspec.yml` Wenn sich die Buildspec-Datei in einem S3-Bucket befindet, muss sie sich in derselben Region wie Ihr Build-Projekt befinden. AWS Geben Sie die Buildspec-Datei mit ihrem ARN an (z. B.). `arn:aws:s3:::<my-codebuild-sample2>/buildspec.yml`
- Wenn der Quellcode keine Build-Spezifikationsdatei enthält oder Sie andere Build-Befehle ausführen möchten, als für die `build`-Phase in der `buildspec.yml`-Datei im

Stammverzeichnis des Quellcodes angegeben wurden, wählen Sie Insert build commands (Build-Befehle einfügen) aus. Geben Sie für Build commands (Build-Befehle) die Befehle ein, die in der `build`-Phase ausgeführt werden sollen. Bei mehreren Befehlen unterteilen Sie die einzelnen Befehle mit &&, (wie z. B. `mvn test && mvn package`). Um Befehle in anderen Phasen auszuführen oder wenn Sie eine lange Liste von Befehlen für die `build` Phase haben, fügen Sie dem Quellcode-Stammverzeichnis eine `buildspec.yml` Datei hinzu, fügen Sie die Befehle zur Datei hinzu und wählen Sie dann Use the buildspec.yml im Quellcode-Stammverzeichnis.

Weitere Informationen hierzu finden Sie unter Build-Spezifikationsreferenz.

Batch-Konfiguration

Wählen Sie im Abschnitt Batch-Konfiguration die Option Bearbeiten aus. Wenn Ihre Änderungen abgeschlossen sind, wählen Sie Konfiguration aktualisieren, um die neue Konfiguration zu speichern. Weitere Informationen finden Sie unter Batch integriert AWS CodeBuild.

Sie können die folgenden Eigenschaften ändern:

Batch-Servicerolle

Stellt die Servicerolle für Batch-Builds bereit.

Wählen Sie eine der folgenden Optionen aus:

- Wenn Sie keine Batch-Servicerolle haben, wählen Sie Neue Servicerolle aus. Geben Sie im Feld Servicerolle einen Namen für die neue Rolle ein.
- Wenn Sie eine Batch-Servicerolle haben, wählen Sie Bestehende Servicerolle aus. Wählen Sie unter Servicerolle die Servicerolle aus.

Batch-Builds führen eine neue Sicherheitsrolle in der Batch-Konfiguration ein. Diese neue Rolle ist erforderlich, da sie in der Lage sein CodeBuild muss`StartBuild`, die `RetryBuild` Aktionen`StopBuild`, und in Ihrem Namen aufzurufen, um Builds als Teil eines Batches auszuführen. Kunden sollten aus zwei Gründen eine neue Rolle und nicht dieselbe Rolle verwenden, die sie in ihrem Build verwenden:

- Die Zuweisung der Build-Rolle `StartBuild` und der `RetryBuild` Berechtigungen würde es einem einzelnen Build ermöglichen, mehrere Builds über die Buildspec zu starten. `StopBuild`
- CodeBuild Batch-Builds bieten Einschränkungen, die die Anzahl der Builds und Berechnungstypen einschränken, die für die Builds im Batch verwendet werden können. Wenn

die Build-Rolle über diese Berechtigungen verfügt, ist es möglich, dass die Builds selbst diese Einschränkungen umgehen.

Zulässige Berechnungstypen für Batch

Wählen Sie die für den Stapel zulässigen Berechnungstypen aus. Wählen Sie alle zutreffenden Antworten aus.

Maximal zulässige Anzahl von Builds pro Batch

Geben Sie die maximale Anzahl von Builds ein, die im Stapel zulässig sind. Wenn ein Stapel diesen Grenzwert überschreitet, schlägt der Batch fehl.

Batch-Timeout

Geben Sie die maximale Zeit für den Abschluss des Batch-Builds ein.

Kombinieren Sie Artefakte

Wählen Sie Alle Artefakte aus dem Stapel an einem einzigen Ort kombinieren aus, um alle Artefakte aus dem Stapel an einem einzigen Ort zusammenzufassen.

Batch-Berichtsmodus

Wählen Sie den gewünschten Modus für den Build-Statusbericht für Batch-Builds aus.

> **Note**
>
> Dieses Feld ist nur verfügbar, GitHub wenn die Projektquelle Bitbucket oder GitHub Enterprise ist und unter Quelle die Option Buildstatus an den Quellanbieter melden, wenn deine Builds beginnen und enden, ausgewählt ist.

Aggregierte Builds

Wählen Sie diese Option aus, um die Status aller Builds im Batch in einem einzigen Statusbericht zusammenzufassen.

Einzelne Builds

Wählen Sie diese Option aus, damit der Build-Status für alle Builds im Batch separat gemeldet wird.

-Artefakte

Wählen Sie im Abschnitt Artefakte die Option Bearbeiten aus. Wenn Ihre Änderungen abgeschlossen sind, wählen Sie Konfiguration aktualisieren, um die neue Konfiguration zu speichern.

Sie können die folgenden Eigenschaften ändern:

Typ

Führen Sie eine der folgenden Aktionen aus:

- Wenn keine Build-Ausgabeartefakte erstellt werden sollen, klicken Sie auf die Option No artifacts. Möglicherweise möchten Sie dies tun, wenn Sie nur Build-Tests ausführen oder ein Docker-Image in ein Amazon ECR-Repository übertragen möchten.
- Um die Build-Ausgabe in einem S3-Bucket zu speichern, wählen Sie Amazon S3 und gehen Sie dann wie folgt vor:
  - Lassen Sie Name leer, wenn Sie den Projektnamen für die ZIP-Datei mit der Build-Ausgabe verwenden möchten. Geben Sie andernfalls den Namen ein. (Wenn eine ZIP-Datei mit einer Dateierweiterung ausgegeben werden soll, vergewissern Sie sich, dass Sie die Dateierweiterung an den Namen der ZIP-Datei anfügen.)
  - Wählen Sie Enable semantitic versioning (Semantisches Versioning aktivieren) aus, wenn Sie möchten, dass ein Name in der buildspec-Datei jeden beliebigen in der Konsole angegebenen Namen überschreibt. Der Name in einer buildspec-Datei wird zur Erstellungszeit berechnet und verwendet die Shell-Befehlssprache. Beispielsweise können Sie dem Namen Ihres Artefakts ein Datum und eine Uhrzeit anhängen, damit dieser stets eindeutig ist. Eindeutige Artefakt-Namen verhindern, dass Artefakte überschrieben werden. Weitere Informationen finden Sie unter Syntax der Build-Spezifikation.
  - Wählen Sie für Bucket name den Namen des Ausgabe-Buckets aus.
  - Wenn Sie in diesem Vorgang zuvor die Option Insert build commands (Build-Befehle eingeben) verwendet haben, geben Sie für Output files (Ausgabedateien) die Speicherorte der Build-Dateien ein, die in der ZIP-Datei oder dem Ordner für die Build-Ausgabe enthalten sein sollen. Bei mehreren Speicherorten trennen Sie die einzelnen Speicherorte durch ein Komma, (wie z. B. `appspec.yml, target/my-app.jar`). Weitere Informationen finden Sie in der Beschreibung von `files` in Syntax der Build-Spezifikation.
  - Wenn Sie nicht wollen, dass Ihre Build-Artefakte verschlüsselt werden, wählen Sie Remove artifacts encryption (Verschlüsselung von Artefakten entfernen) aus.

Für jede Gruppe sekundärer Artefakte:

1. Geben Sie für Artifact identifier (Artefakt-ID) einen Wert mit weniger als 128 Zeichen ein, der nur alphanumerische Zeichen und Unterstriche enthält.
2. Wählen Sie Add artifact (Artefakt hinzufügen) aus.
3. Führen Sie die vorherigen Schritte aus, um die sekundären Artefakte zu konfigurieren.
4. Wählen Sie Save artifact (Artefakt speichern) aus.

Zusätzliche Konfiguration

Verschlüsselungsschlüssel

Führen Sie eine der folgenden Aktionen aus:

- Um Von AWS verwalteter Schlüssel Amazon S3 in Ihrem Konto zur Verschlüsselung der Build-Ausgabeartefakte zu verwenden, lassen Sie den Verschlüsselungsschlüssel leer. Dies ist die Standardeinstellung.
- Um einen vom Kunden verwalteten Schlüssel zum Verschlüsseln der Build-Ausgabeartefakte zu verwenden, geben Sie im Feld Verschlüsselungsschlüssel den ARN des vom Kunden verwalteten Schlüssels ein. Verwenden Sie dabei das Format `arn:aws:kms:`*`region-ID`*`:`*`account-ID`*`:key/`*`key-ID`*.

Cache-Typ

Wählen Sie für Cache type (Cache-Typ) eine der folgenden Optionen aus:

- Wenn Sie keinen Cache verwenden möchten, wählen Sie No cache.
- Wenn Sie einen Amazon S3-Cache verwenden möchten, wählen Sie Amazon S3 und gehen Sie dann wie folgt vor:
  - Wählen Sie für Bucket den Namen des S3-Buckets, in dem der Cache gespeichert wird.
  - (Optional) Geben Sie für das Cache-Pfadpräfix ein Amazon S3 S3-Pfadpräfix ein. Der Wert für Cache path prefix (Cache-Pfadpräfix) ist mit einem Verzeichnisnamen vergleichbar. Er ermöglicht Ihnen das Speichern des Cache in demselben Verzeichnis eines Buckets.

    > **Important**
    >
    > Fügen Sie am Ende des Pfadpräfix keinen abschließenden Schrägstrich (/) an.

- Wenn Sie einen lokalen Cache verwenden möchten, wählen Sie Local (Lokal) und dann mindestens einen lokalen Cache-Modus aus.

> **Note**
>
> Der Modus Docker layer cache (Docker-Ebenen-Cache) ist nur für Linux verfügbar. Wenn Sie diesen Modus auswählen, muss Ihr Projekt im privilegierten Modus ausgeführt werden.

Durch die Verwendung eines Caches wird eine erhebliche Ersparnis bei der Erstellungszeit erzielt, da wiederverwendbare Teile der Build-Umgebung im Cache gespeichert und über Builds hinweg verwendet werden. Weitere Informationen über die Angabe eines Cache in der Build-Spezifikationsdatei finden Sie unter Syntax der Build-Spezifikation. Weitere Informationen zum Caching finden Sie unter Build-Caching in AWS CodeBuild.

Logs (Protokolle)

Wählen Sie im Abschnitt Logs die Option Bearbeiten aus. Wenn Ihre Änderungen abgeschlossen sind, wählen Sie Konfiguration aktualisieren, um die neue Konfiguration zu speichern.

Sie können die folgenden Eigenschaften ändern:

Wählen Sie die Protokolle aus, die Sie erstellen möchten. Sie können Amazon CloudWatch Logs, Amazon S3 S3-Protokolle oder beides erstellen.

CloudWatch

Wenn Sie Amazon CloudWatch Logs-Protokolle wünschen:

CloudWatch Logs

Wählen Sie CloudWatch logs (CW-Protokolle).

Group name (Gruppenname)

Geben Sie den Namen Ihrer Amazon CloudWatch Logs-Protokollgruppe ein.

Name des Streams

Geben Sie den Namen Ihres Amazon CloudWatch Logs-Log-Streams ein.

S3

Wenn Sie Amazon S3 S3-Protokolle wünschen:

S3-Protokolle

Wählen Sie S3 logs (S3-Protokolle).

Bucket

Wählen Sie den Namen des S3-Buckets für Ihre Logs.

Pfadpräfix

Geben Sie das Präfix für Ihre Logs ein.

Deaktivieren Sie die S3-Protokollverschlüsselung

Wählen Sie diese Option aus, wenn Sie nicht möchten, dass Ihre S3-Protokolle verschlüsselt werden.

Ändern der Einstellungen eines Build-Projekts (AWS CLI)

Informationen zur Verwendung von AWS CLI mit AWS CodeBuild finden Sie unter Befehlszeilenreferenz.

Um eine zu aktualisierenCodeBuildProjekt mit demAWS CLI, Sie erstellen eine JSON-Datei mit den aktualisierten Eigenschaften und übergeben diese Datei an`update-project`Befehl. Alle Eigenschaften, die nicht in der Aktualisierungsdatei enthalten sind, bleiben unverändert.

In der Aktualisierungs-JSON-Datei nur die`name`Eigenschaft und die geänderten Eigenschaften sind erforderlich. Der`name`Die Eigenschaft identifiziert das zu ändernde Projekt. Für alle modifizierten Strukturen müssen auch die erforderlichen Parameter für diese Strukturen enthalten sein. Um beispielsweise die Umgebung für das Projekt zu ändern,`environment/type`und`environment/computeType`Eigenschaften sind erforderlich. Hier ist ein Beispiel, das das Umgebungsbild aktualisiert:

```
{
  "name": "<project-name>",
  "environment": {
    "type": "LINUX_CONTAINER",
    "computeType": "BUILD_GENERAL1_SMALL",
    "image": "aws/codebuild/amazonlinux2-x86_64-standard:4.0"
  }
}
```

Wenn Sie die aktuellen Immobilienwerte für ein Projekt abrufen möchten, verwenden Sie denbatch-get-projectsBefehl, um die aktuellen Eigenschaften des Projekts abzurufen, das Sie ändern, und die Ausgabe in eine Datei zu schreiben.

```
aws codebuild batch-get-projects --names "<project-name>" > project-info.json
```

Der*projekt-info.json*Die Datei enthält eine Reihe von Projekten und kann daher nicht direkt zum Aktualisieren eines Projekts verwendet werden. Sie können jedoch die Eigenschaften, die Sie ändern möchten, aus dem*projekt-info.json*Datei und fügen Sie sie als Grundlage für die Eigenschaften, die Sie ändern möchten, in Ihre Aktualisierungsdatei ein. Weitere Informationen finden Sie unter Anzeigen der Details eines Build-Projekts (AWS CLI).

Ändern Sie die Aktualisierungs-JSON-Datei wie unter beschriebenErstellen eines Build-Projekts (AWS CLI), und speichern Sie Ihre Ergebnisse. Wenn Sie mit der Änderung der JSON-Datei für das Update fertig sind, führen Sie den`update-project`Befehl, der die Aktualisierungs-JSON-Datei übergibt.

```
aws codebuild update-project --cli-input-json file://<update-project-file>
```

Bei Erfolg wird das aktualisierte Projekt-JSON in der Ausgabe angezeigt. Wenn erforderliche Parameter fehlen, wird in der Ausgabe eine Fehlermeldung angezeigt, die die fehlenden Parameter identifiziert. Dies ist beispielsweise die Fehlermeldung, die angezeigt wird, wenn`environment/type`Parameter fehlt:

```
aws codebuild update-project --cli-input-json file://update-project.json

Parameter validation failed:
Missing required parameter in environment: "type"
```

Ändern der Einstellungen eines Build-Projekts (AWS SDKs)

Informationen zur Verwendung von AWS CodeBuild mit den AWS-SDKs finden Sie unter AWS SDKs- und Tools-Referenz.

Löschen eines Build-Projekts in AWS CodeBuild

Sie können die CodeBuild-Konsole verwenden,AWS CLI, oderAWS-SDKs zum Löschen eines Build-Projekts in CodeBuild. Wenn Sie ein Projekt löschen, werden dessen Builds nicht gelöscht.

> **Warning**
>
> Sie können kein Projekt löschen, das Builds und eine Ressourcenrichtlinie enthält. Um ein Projekt mit einer Ressourcenrichtlinie und Builds zu löschen,müssen Sie zunächst die Ressourcenrichtlinie entfernen und seine Builds löschen.

Themen

- Löschen eines Build-Projekts (Konsole)
- Löschen eines Build-Projekts (AWS CLI)
- Löschen eines Build-Projekts (AWS SDKs)

Löschen eines Build-Projekts (Konsole)

1. Öffnen SieAWS CodeBuild-Konsolehttps://console.aws.amazon.com/codesuite/codebuild/homeaus.
2. Wählen Sie im linken Navigationsbereich Build projects aus.
3. Gehen Sie folgendermaßen vor:
   - Aktivieren Sie das Optionsfeld neben dem Build-Projekt, das Sie löschen möchten, und klicken Sie auf Delete (Löschen).
   - Klicken Sie auf den Link des Build-Projekts, das Sie löschen möchten, und klicken Sie dann auf Delete.

> **Note**
>
> Standardmäßig werden nur die letzten zehn Build-Projekte angezeigt. Zur Anzeige von weiteren Build-Projekten wählen Sie einen anderen Wert für Projects per page (Projekte je Seite) aus oder verwenden die Vorwärts- und Rückwärtspfeile zum Anzeigen von Projekten.

Löschen eines Build-Projekts (AWS CLI)

1. Führen Sie den Befehl `delete-project` aus:

```
aws codebuild delete-project --name name
```

Ersetzen die folgenden Platzhalter:

- *Name*: Erforderliche Zeichenfolge. Zeigt den Namen des zu löschenden Build-Projekts an. Zum Anzeigen einer Liste sämtlicher verfügbarer Build-Projekte führen Sie den Befehl `list-projects` aus. Weitere Informationen finden Sie unter Anzeigen einer Liste mit Build-Projektnamen (AWS CLI).

2. Ist der Befehl erfolgreich, werden in der Ausgabe keine Daten und Fehler angezeigt.

Weitere Informationen zur Verwendung der AWS CLI mit AWS CodeBuild finden Sie unter Befehlszeilenreferenz.

Löschen eines Build-Projekts (AWS SDKs)

Weitere Informationen zur Verwendung von AWS CodeBuild mit den AWS-SDKs finden Sie unter AWS SDKs- und Tools-Referenz.

Arbeiten mit freigegebenen Projekten

Mit der Projektfreigabe können Projekteigentümer ihreAWS CodeBuildprojekte mit anderenAWSKonten oder Benutzer. In diesem Modell gibt das Konto, das Eigentümer des Projekts (Eigentümer) ist, ein Projekt für andere Konten (Verbraucher) frei. Ein Verbraucher kann kein Projekt bearbeiten oder ausführen.

Inhalt

Voraussetzungen für die Freigabe von Projekten

Um ein Projekt freizugeben, muss Ihr AWS-Konto dessen Eigentümer sein. Sie können kein Projekt freigeben, das für Sie freigegeben wurde.

Voraussetzungen für den Zugriff auf freigegebene Projekte

Um auf ein freigegebenes Projekt zuzugreifen, benötigt die IAM-Rolle eines Verbrauchers`BatchGetProjects`-Erlaubnis Sie können die folgende Richtlinie an ihre IAM-Rolle anfügen:

```
{
    "Effect": "Allow",
    "Resource": [
        "*"
    ],
    "Action": [
        "codebuild:BatchGetProjects"
    ]
}
```

Weitere Informationen finden Sie unter Verwendung identitätsbasierter Richtlinien für AWS CodeBuild.

Verwandte Dienstleistungen

Die Projektfreigabe ist mit AWS Resource Access Manager (AWS RAM) integriert, einem Service, der es Ihnen ermöglicht, Ihre AWS- Ressourcen für ein beliebiges AWS-Konto oder über AWS Organizations freizugeben. Mit AWS RAM können Sie Ressourcen gemeinsam nutzen, indem Sie eine Ressourcenfreigabe erstellen, die die Ressourcen und die Verbraucher angibt, für die sie freigegeben werden sollen. Verbraucher können einzelne AWS-Konten oder Organisationseinheiten in AWS Organizations oder gesamte Organisationen in AWS Organizations sein.

Weitere Informationen finden Sie im AWS RAM-Leitfaden.

Freigeben eines Projekts

Der Verbraucher kann sowohl dieAWS CLIundAWS CodeBuild-Konsole, um das Projekt und die Builds anzuzeigen, die Sie freigegeben haben. Der Verbraucher kann das Projekt nicht bearbeiten oder ausführen.

Sie können ein Projekt zu einer vorhandenen Ressourcenfreigabe hinzufügen oder in der AWS RAM-Konsole erstellen.

> **Note**
>
> Sie können kein Projekt mit Builds löschen, das einer Ressourcenfreigabe hinzugefügt wurde.

Um ein Projekt für Organisationseinheiten oder eine ganze Organisation freizugeben, müssen Sie die Freigabe mit AWS Organizations aktivieren. Weitere Informationen finden Sie unter Freigabe für AWS Organizations aktivieren im AWS RAM-Benutzerhandbuch.

Sie können die AWS CodeBuild-Konsole, die AWS RAM-Konsole oder AWS CLI verwenden, um ein Projekt freizugeben, dessen Eigentümer Sie sind.

So geben Sie ein Projekt frei, dessen Eigentümer Sie sind (CodeBuild-Konsole)

1. Öffnen SieAWS CodeBuildconsole beihttps://console.aws.amazon.com/codesuite/codebuild/homeaus.
2. Wählen Sie im linken Navigationsbereich Build projects aus.

   > **Note**
   >
   > Standardmäßig werden nur die letzten 10 Build-Projekte angezeigt. Zur Anzeige von weiteren Build-Projekten wählen Sie das Zahnradsymbol und einen anderen Wert für Projects per page (Projekte je Seite) aus oder verwenden die Vorwärts- und Rückwärtspfeile.

3. Wählen Sie das Projekt aus, das Sie freigeben möchten, und klicken Sie dann auf Share (Freigeben). Weitere Informationen finden Sie unter Erstellen einer Ressourcenfreigabe im AWS RAM-Benutzerhandbuch.

So geben Sie ein Projekt frei, dessen Eigentümer Sie sind (AWS RAM-Konsole):

Siehe .Erstellen einer RessourcenfreigabeimAWS RAM-Benutzerhandbuchaus.

So geben Sie ein Projekt frei, dessen Eigentümer Sie sind (AWS RAM-Befehl):

Verwenden Sie den Befehl create-resource-share.

So geben Sie ein Projekt frei, dessen Eigentümer Sie sind (-Befehl CodeBuild)

Verwenden Sie den Befehl put-resource-policy :

1. Erstellen Sie eine Datei mit dem Namen `policy.json` und kopieren Sie Folgendes in diese Datei.

```
{
  "Version":"2012-10-17",
  "Statement":[{
    "Effect":"Allow",
    "Principal":{
      "AWS":"<consumer-aws-account-id-or-user>"
    },
    "Action":[
      "codebuild:BatchGetProjects",
      "codebuild:BatchGetBuilds",
      "codebuild:ListBuildsForProject"],
    "Resource":"<arn-of-project-to-share>"
  }]
}
```

2. Aktualisieren Sie `policy.json` mit dem Projekt-ARN und Bezeichnern für die Freigabe. Im folgenden Beispiel wird schreibgeschützter Zugriff auf den Root-Benutzer fürAWSKonto identifiziert durch 123456789012.

```
{
  "Version":"2012-10-17",
  "Statement":[{
    "Effect":"Allow",
    "Principal":{
      "AWS": [
        "123456789012"
      ]
    },
    "Action":[
      "codebuild:BatchGetProjects",
      "codebuild:BatchGetBuilds",
      "codebuild:ListBuildsForProject"],
    "Resource":"arn:aws:codebuild:us-west-2:123456789012:project/my-project"
  }]
}
```

3. Ausführen des sput-resource-policybefehl.

```
aws codebuild put-resource-policy --resource-arn <project-arn> --policy file://
policy.json
```

4. Hol dieAWS RAMRessourcenfreigabe-ARN

```
aws ram list-resources --resource-owner SELF --resource-arns <project-arn>
```

Dadurch wird eine Antwort ähnlich wie diese zurückgegeben:

```
{
  "resources": [
    {
      "arn": "<project-arn>",
      "type": "<type>",
      "resourceShareArn": "<resource-share-arn>",
      "creationTime": "<creation-time>",
      "lastUpdatedTime": "<last-update-time>"
    }
  ]
}
```

Kopieren Sie aus der Antwort die*<resource-share-arn>*Wert, der im nächsten Schritt verwendet werden soll.

5. Ausführen des sAWS RAM promote-ressourcen-aktie-erstellt-from-politikbefehl.

```
aws ram promote-resource-share-created-from-policy --resource-share-arn <resource-
share-arn>
```

Aufheben der Freigabe eines freigegebenen Projektes

Auf ein nicht mehr freigegebenes Projekt, einschließlich seiner Builds, kann nur sein Eigentümer zugreifen. Wenn Sie die Freigabe eines Projekts aufheben, kann ein AWS-Konto oder -Benutzer, für das/den Sie es zuvor freigegeben haben, nicht mehr auf das Projekt oder seine Builds zugreifen.

Um die Freigabe eines freigegebenen Projektes, dessen Eigentümer Sie sind, aufzuheben, müssen Sie es aus der Ressourcenfreigabe entfernen. Sie dafür können die AWS CodeBuild-Konsole, die AWS RAM-Konsole oder AWS CLI verwenden.

So heben Sie die Freigabe eines freigegebenen Projekts auf, dessen Eigentümer Sie sind (AWS RAM-Konsole)

Siehe Aktualisieren einer Ressourcenfreigabe im AWS RAM-Benutzerhandbuch.

So heben Sie die Freigabe eines freigegebenen Projekts auf, dessen Eigentümer Sie sind (AWS CLI)

Verwenden Sie den Befehl disassociate-resource-share.

So heben Sie die Freigabe der Freigabe eines Projekts auf, dessen Eigentümer Sie sind (-Befehl

Führen Sie den Befehl delete-resource-policy aus und geben Sie den ARN des Projekts an, dessen Freigabe Sie aufheben möchten:

```
aws codebuild delete-resource-policy --resource-arn project-arn
```

Identifizieren eines freigegebenen Projekts

Eigentümer und Verbraucher können freigegebene Projekte mit AWS CLI identifizieren.

So identifizieren Sie Projekte, die für Ihr AWS-Konto oder Ihren Benutzer freigegeben wurden (AWS CLI)

Verwenden Sie den Befehl list-shared-projects, um die für Sie freigegebenen Projekte anzuzeigen.

Berechtigungen für freigegebene Projekte

Berechtigungen für Besitzer

Ein Projekteigentümer kann das Projekt bearbeiten und es zum Ausführen von Builds verwenden.

Berechtigungen für Konsumenten

Ein Projektverbraucher kann ein Projekt und seine Builds anzeigen, es jedoch nicht bearbeiten oder damit Builds ausführen.

Projekte taggen in AWS CodeBuild

Ein Tag ist eine benutzerdefinierte Attributbezeichnung, die Sie einer AWS Ressource AWS zuweisen oder zuweisen. Jedes AWS Tag besteht aus zwei Teilen:

- einem Tag-Schlüssel (z. B. `CostCenter`, `Environment`, `Project` oder `Secret`). Bei Tag-Schlüsseln wird zwischen Groß- und Kleinschreibung unterschieden.
- einem optionalen Feld, das als Tag-Wert bezeichnet wird (z. B. `111122223333`, `Production` oder ein Team-Name). Ein nicht angegebener Tag-Wert entspricht einer leeren Zeichenfolge. Wie bei Tag-Schlüsseln wird auch bei Tag-Werten zwischen Groß- und Kleinschreibung unterschieden.

Zusammen werden sie als Schlüssel-Wert-Paare bezeichnet. Informationen zur Anzahl der Tags, die ein Projekt besitzen kann, und zu Einschränkungen in Bezug auf Tag-Schlüssel und -Werte finden Sie unter Tags.

Mithilfe von Tags können Sie Ihre AWS Ressourcen identifizieren und organisieren. Viele AWS Dienste unterstützen Tagging, sodass Sie Ressourcen aus verschiedenen Diensten dasselbe Tag zuweisen können, um anzuzeigen, dass die Ressourcen miteinander verknüpft sind. Sie können beispielsweise einem CodeBuild Projekt dasselbe Tag zuweisen, das Sie einem S3-Bucket zuweisen. Weitere Informationen zur Verwendung von Tags finden Sie unter Bewährte Methoden zum Tagging.

In sind CodeBuild die Hauptressourcen das Projekt und die Berichtsgruppe. Sie können die CodeBuild Konsole AWS CLI, die CodeBuild APIs oder AWS SDKs verwenden, um Tags für ein Projekt hinzuzufügen, zu verwalten und zu entfernen. Sie können Ihr Projekt nicht nur anhand von Tags identifizieren, organisieren und verfolgen, sondern auch Tags in IAM-Richtlinien verwenden, um zu kontrollieren, wer Ihr Projekt ansehen und mit ihm interagieren kann. Beispiele für Tag-basierte Zugriffsrichtlinien finden Sie unter Verwenden von Tags zur Steuerung des Zugriffs auf AWS CodeBuild -Ressourcen.

> **Important**
>
> Wenn Sie die Funktion für reservierte Kapazität verwenden, können andere Projekte innerhalb desselben Kontos auf Daten zugreifen, die auf Flotteninstanzen zwischengespeichert sind, einschließlich Quelldateien, Docker-Layern und zwischengespeicherten Verzeichnissen, die in der Buildspec angegeben sind. Dies ist beabsichtigt und ermöglicht es Projekten innerhalb desselben Kontos, Flotteninstanzen gemeinsam zu nutzen.

Themen

- Hinzufügen eines Tags zu einem Projekt
- Anzeigen von Tags für ein Projekt

- Bearbeiten von Tags für ein Projekt
- Entfernen eines Tag aus einem Projekt

Hinzufügen eines Tags zu einem Projekt

Das Hinzufügen von Tags zu einem Projekt kann Ihnen dabei helfen, Ihre AWS Ressourcen zu identifizieren und zu organisieren und den Zugriff darauf zu verwalten. Fügen Sie zunächst ein oder mehrere Tags (Schlüssel-Wert-Paare) zu einem Projekt hinzu. Denken Sie daran, dass es Limits in Bezug auf die Anzahl der Tags für ein Projekt gibt. Es gibt Einschränkungen im Hinblick auf die Zeichen, die Sie in die Felder für Schlüssel und Wert eingeben können. Weitere Informationen finden Sie unter Tags. Sobald Sie über Tags verfügen, können Sie IAM-Richtlinien erstellen, um den Zugriff auf das Projekt auf der Grundlage dieser Tags zu verwalten. Sie können die CodeBuild Konsole oder die verwenden AWS CLI , um einem Projekt Tags hinzuzufügen.

> **Important**
>
> Wenn Sie die Funktion für reservierte Kapazität verwenden, können andere Projekte innerhalb desselben Kontos auf Daten zugreifen, die auf Flotteninstanzen zwischengespeichert sind, einschließlich Quelldateien, Docker-Layern und zwischengespeicherten Verzeichnissen, die in der Buildspec angegeben sind. Dies ist beabsichtigt und ermöglicht es Projekten innerhalb desselben Kontos, Flotteninstanzen gemeinsam zu nutzen.

Weitere Informationen zum Hinzufügen von Tags zu einem Projekt während dessen Erstellung finden Sie unter Hinzufügen eines Tags zu einem Projekt (Konsole).

> **Important**
>
> Bevor Sie einem Projekt ein Tag hinzufügen, sollten Sie alle IAM-Richtlinien überprüfen, die Tags verwenden könnten, um den Zugriff auf Ressourcen wie Build-Projekte zu kontrollieren. Beispiele für Tag-basierte Zugriffsrichtlinien finden Sie unter Verwenden von Tags zur Steuerung des Zugriffs auf AWS CodeBuild -Ressourcen.

Themen

- Hinzufügen eines Tags zu einem Projekt (Konsole)

- Hinzufügen eines Tags zu einem Projekt (AWS CLI)

Hinzufügen eines Tags zu einem Projekt (Konsole)

Sie können die CodeBuild Konsole verwenden, um einem CodeBuild Projekt ein oder mehrere Tags hinzuzufügen.

1. Öffnen Sie die CodeBuild Konsole unter https://console.aws.amazon.com/codebuild/.
2. Wählen Sie in Build projects (Build-Projekte) den Namen des Projekts aus, dem Sie Tags hinzufügen möchten.
3. Wählen Sie im Navigationsbereich Settings (Einstellungen). Wählen Sie Build project tags (Build-Projekt-Tags) aus.
4. Wenn dem Projekt keine Tags hinzugefügt wurden, wählen Sie Add tag (Tag hinzufügen) aus. Wählen Sie andernfalls Edit (Bearbeiten) und Add tag (Tag hinzufügen) aus.
5. Geben Sie für Schlüssel einen Namen für das Tag ein. Sie können einen optionalen Wert für das Tag unter Wert hinzufügen.
6. (Optional) Zum Hinzufügen eines weiteren Tags wählen Sie Tag hinzufügen erneut aus.
7. Wenn Sie mit dem Hinzufügen von Tags fertig sind, klicken Sie auf Submit (Übermitteln).

Hinzufügen eines Tags zu einem Projekt (AWS CLI)

Informationen dazu, wie Sie einem Projekt während dessen Erstellung ein Tag hinzufügen, finden Sie unter Erstellen eines Build-Projekts (AWS CLI). Fügen Sie Ihre Tags in `create-project.json` hinzu.

Bei diesen Schritten wird davon ausgegangen, dass Sie bereits eine aktuelle Version der AWS CLI installiert oder eine Aktualisierung auf die aktuelle Version vorgenommen haben. Weitere Informationen finden Sie unter Installieren der AWS Command Line Interface.

Bei erfolgreicher Ausführung gibt dieser Befehl nichts zurück.

Anzeigen von Tags für ein Projekt

Mithilfe von Tags können Sie Ihre AWS Ressourcen identifizieren und organisieren und den Zugriff darauf verwalten. Weitere Informationen zur Verwendung von Tags finden Sie im Whitepaper Bewährte Methoden für die Markierung. Beispiele für Tag-basierte Zugriffsrichtlinien finden Sie unter Verwenden von Tags zur Steuerung des Zugriffs auf AWS CodeBuild -Ressourcen.

Anzeigen von Tags für ein Projekt (Konsole)

Sie können die CodeBuild Konsole verwenden, um die mit einem CodeBuild Projekt verknüpften Tags anzuzeigen.

1. Öffnen Sie die CodeBuild Konsole unter https://console.aws.amazon.com/codebuild/.
2. Wählen Sie in Build projects (Build-Projekte) den Namen des Projekts aus, in dem Sie Tags anzeigen möchten.
3. Wählen Sie im Navigationsbereich Settings (Einstellungen). Wählen Sie Build project tags (Build-Projekt-Tags) aus.

Anzeigen von Tags für ein Projekt (AWS CLI)

Führen Sie den folgenden Befehl aus, um Tags für ein Build-Projekt anzuzeigen. Verwenden Sie den Namen Ihres Projekts für den Parameter `--names`.

```
aws codebuild batch-get-projects --names your-project-name
```

Wenn der Befehl erfolgreich ist, gibt dieser Befehl JSON-formatierte Informationen zu Ihrem Build-Projekt zurück, das Angaben ähnlich den folgenden enthält:

```
{
    "tags": {
        "Status": "Secret",
        "Team": "JanesProject"
    }
}
```

Wenn das Projekt keine Tags besitzt, ist der `tags`-Abschnitt leer:

```
"tags": []
```

Bearbeiten von Tags für ein Projekt

Sie können den Wert für ein Tag ändern, das mit einem Projekt verknüpft ist. Sie können auch den Namen des Schlüssels ändern. Dies entspricht dem Entfernen des aktuellen Tags und dem Hinzufügen eines anderen Tags mit dem neuen Namen und demselben Wert wie dem des anderen. Denken Sie daran, dass es hinsichtlich der Zeichen, die Sie in die Felder für Schlüssel und Wert eingeben können, Einschränkungen gibt. Weitere Informationen finden Sie unter Tags.

> **Important**
>
> Das Bearbeiten von Tags für ein Projekt kann sich auf den Zugriff auf dieses Projekt auswirken. Bevor Sie den Namen (Schlüssel) oder den Wert eines Tags für ein Projekt bearbeiten, sollten Sie alle IAM-Richtlinien überprüfen, die den Schlüssel oder Wert für ein Tag möglicherweise verwenden, um den Zugriff auf Ressourcen wie Build-Projekte zu steuern. Beispiele für Tag-basierte Zugriffsrichtlinien finden Sie unter Verwenden von Tags zur Steuerung des Zugriffs auf AWS CodeBuild -Ressourcen.

Bearbeiten eines Tags für ein Projekt (Konsole)

Sie können die CodeBuild Konsole verwenden, um die mit einem CodeBuild Projekt verknüpften Tags zu bearbeiten.

1. Öffnen Sie die CodeBuild Konsole unter https://console.aws.amazon.com/codebuild/.
2. Wählen Sie in Build projects (Build-Projekte) den Namen des Projekts aus, in dem Sie Tags bearbeiten möchten.
3. Wählen Sie im Navigationsbereich Settings (Einstellungen). Wählen Sie Build project tags (Build-Projekt-Tags) aus.
4. Wählen Sie Bearbeiten aus.
5. Führen Sie eine der folgenden Aktionen aus:
   - Zum Ändern des Tags geben Sie einen neuen Namen unter Key (Schlüssel) ein. Das Ändern des Namens eines Tags entspricht dem Entfernen eines Tags und Hinzufügen eines neuen Tags mit dem neuen Schlüsselnamen.
   - Geben Sie zum Ändern des Werts eines Tags einen neuen Wert ein. Wenn Sie den Wert in „kein“ ändern möchten, löschen Sie den aktuellen Wert und lassen das Feld leer.
6. Wenn Sie mit dem Bearbeiten der Tags fertig sind, wählen Sie Submit (Übermitteln) aus.

Bearbeiten von Tags für ein Projekt (AWS CLI)

Informationen zum Hinzufügen, Ändern oder Löschen von Tags aus einem Build-Projekt finden Sie unter Ändern der Einstellungen eines Build-Projekts (AWS CLI). Aktualisieren Sie den `tags`-Abschnitt in den JSON-formatierten Daten, die Sie zum Aktualisieren des Projekts verwenden.

Entfernen eines Tag aus einem Projekt

Sie können ein oder mehrere mit einem Projekt verknüpfte Tags entfernen. Durch das Entfernen eines Tags wird das Tag nicht aus anderen AWS Ressourcen gelöscht, die mit diesem Tag verknüpft sind.

> **Important**
>
> Das Entfernen von Tags für ein Projekt kann sich auf den Zugriff auf dieses Projekt auswirken. Bevor Sie ein Tag aus einem Projekt entfernen, sollten Sie alle IAM-Richtlinien überprüfen, die den Schlüssel oder Wert für ein Tag verwenden könnten, um den Zugriff auf Ressourcen wie Build-Projekte zu steuern. Beispiele für Tag-basierte Zugriffsrichtlinien finden Sie unter Verwenden von Tags zur Steuerung des Zugriffs auf AWS CodeBuild -Ressourcen.

Entfernen eines Tag aus einem Projekt (Konsole)

Sie können die CodeBuild Konsole verwenden, um die Zuordnung zwischen einem Tag und einem CodeBuild Projekt zu entfernen.

1. Öffnen Sie die CodeBuild Konsole unter https://console.aws.amazon.com/codebuild/.
2. Wählen Sie in Build projects (Build-Projekte) den Namen des Projekts aus, in dem Sie Tags entfernen möchten.
3. Wählen Sie im Navigationsbereich Settings (Einstellungen). Wählen Sie Build project tags (Build-Projekt-Tags) aus.
4. Wählen Sie Bearbeiten aus.
5. Suchen Sie die Tags, die Sie entfernen möchten, und wählen Sie dann Remove tag (Tag entfernen) aus.
6. Wenn Sie die Tags entfernt haben, klicken Sie auf Submit (Übermitteln).

Entfernen eines Tag aus einem Projekt (AWS CLI)

Informationen zum Löschen eines oder mehrerer Tags aus einem Build-Projekt finden Sie unter Ändern der Einstellungen eines Build-Projekts (AWS CLI). Aktualisieren Sie den `tags`-Abschnitt in den JSON-formatierten Daten mit einer aktualisierten Liste von Tags, die keine der Tags enthält, die Sie löschen möchten. Wenn Sie alle Tags löschen möchten, aktualisieren Sie den `tags`-Abschnitt zu:

```
"tags: []"
```

> **Note**
>
> Wenn Sie ein CodeBuild Build-Projekt löschen, werden alle Tag-Zuordnungen aus dem gelöschten Build-Projekt entfernt. Sie müssen keine Tags entfernen, bevor Sie ein Build-Projekt löschen.

Batch integriert AWS CodeBuild

Sie können AWS CodeBuild es verwenden, um gleichzeitige und koordinierte Builds eines Projekts mit Batch-Builds auszuführen.

Themen

Rolle „Sicherheit“

Batch-Builds führen eine neue Sicherheitsrolle in der Batch-Konfiguration ein. Diese neue Rolle ist erforderlich, da sie in der Lage sein CodeBuild muss`StartBuild`, die `RetryBuild` Aktionen`StopBuild`, und in Ihrem Namen aufzurufen, um Builds als Teil eines Batches auszuführen. Kunden sollten aus zwei Gründen eine neue Rolle und nicht dieselbe Rolle verwenden, die sie in ihrem Build verwenden:

- Die Zuweisung der Build-Rolle `StartBuild` und der `RetryBuild` Berechtigungen würde es einem einzelnen Build ermöglichen, mehrere Builds über die Buildspec zu starten. `StopBuild`
- CodeBuild Batch-Builds bieten Einschränkungen, die die Anzahl der Builds und Berechnungstypen einschränken, die für die Builds im Batch verwendet werden können. Wenn die Build-Rolle über diese Berechtigungen verfügt, ist es möglich, dass die Builds selbst diese Einschränkungen umgehen.

Batch-Build-Typen

CodeBuild unterstützt die folgenden Batch-Build-Typen:

Batch-Build-Typen

- Diagramm erstellen
- Liste erstellen
- Matrix erstellen

Diagramm erstellen

Ein Build-Diagramm definiert eine Reihe von Aufgaben, die von anderen Aufgaben im Stapel abhängig sind.

Das folgende Beispiel definiert ein Build-Diagramm, das eine Abhängigkeitskette erstellt.

```
batch:
  fast-fail: false
  build-graph:
    - identifier: build1
      env:
        variables:
          BUILD_ID: build1
      ignore-failure: false
    - identifier: build2
      buildspec: build2.yml
      env:
        variables:
          BUILD_ID: build2
      depend-on:
        - build1
    - identifier: build3
      env:
        variables:
          BUILD_ID: build3
      depend-on:
        - build2
```

In diesem Beispiel:

- `build1`wird zuerst ausgeführt, weil es keine Abhängigkeiten hat.

- `build2`hat eine Abhängigkeit von`build1`, `build2` wird also nach `build1` Abschluss ausgeführt.
- `build3`ist abhängig von`build2`, `build3` wird also nach `build2` Abschluss ausgeführt.

Weitere Hinweise zur Buildspec-Syntax von Build Graph finden Sie unter. batch/build-graph

Liste erstellen

Eine Build-Liste definiert eine Reihe von Aufgaben, die parallel ausgeführt werden.

Das folgende Beispiel definiert eine Build-Liste. Die `build2` Builds `build1` und werden parallel ausgeführt.

```
batch:
  fast-fail: false
  build-list:
    - identifier: build1
      env:
        variables:
          BUILD_ID: build1
      ignore-failure: false
    - identifier: build2
      buildspec: build2.yml
      env:
        variables:
          BUILD_ID: build2
      ignore-failure: true
```

Weitere Hinweise zur Buildspec-Syntax der Buildliste finden Sie unter. batch/build-list

Matrix erstellen

Eine Build-Matrix definiert Aufgaben mit unterschiedlichen Konfigurationen, die parallel ausgeführt werden. CodeBuild erstellt für jede mögliche Konfigurationskombination einen separaten Build.

Das folgende Beispiel zeigt eine Buildmatrix mit zwei Buildspec-Dateien und drei Werten für eine Umgebungsvariable.

```
batch:
  build-matrix:
    static:
```

```
      ignore-failure: false
    dynamic:
      buildspec:
        - matrix1.yml
        - matrix2.yml
      env:
        variables:
          MY_VAR:
            - VALUE1
            - VALUE2
            - VALUE3
```

In diesem Beispiel werden sechs Builds CodeBuild erstellt:

- `matrix1.yml` mit `$MY_VAR=VALUE1`
- `matrix1.yml` mit `$MY_VAR=VALUE2`
- `matrix1.yml` mit `$MY_VAR=VALUE3`
- `matrix2.yml` mit `$MY_VAR=VALUE1`
- `matrix2.yml` mit `$MY_VAR=VALUE2`
- `matrix2.yml` mit `$MY_VAR=VALUE3`

Jeder Build hat die folgenden Einstellungen:

- `ignore-failure`eingestellt auf `false`
- `env/type`eingestellt auf `LINUX_CONTAINER`
- `env/image`eingestellt auf `aws/codebuild/amazonlinux2-x86_64-standard:4.0`
- `env/privileged-mode`eingestellt auf `true`

Diese Builds laufen parallel.

Weitere Hinweise zur Buildspec-Syntax der Buildmatrix finden Sie unter. batch/build-matrix

Batch-Berichtsmodus

Wenn der Quellanbieter für dein Projekt Bitbucket oder GitHub Enterprise ist und dein Projekt so konfiguriert ist, dass Build-Status an den Quellanbieter gemeldet werden, kannst du auswählen, wie deine Batch-Build-Status an den Quellanbieter gesendet werden sollen. GitHub Du kannst wählen, ob

die Status als ein einziger aggregierter Statusbericht für den Batch gesendet werden sollen oder ob der Status jedes Builds im Batch einzeln gemeldet werden soll.

Weitere Informationen finden Sie unter den folgenden Themen:

- Batch-Konfiguration (erstellen)
- Batch-Konfiguration (Update)

Weitere Informationen

Weitere Informationen finden Sie unter den folgenden Themen:

- Batch-Build-Buildspec Referenz
- Batch-Konfiguration
- Ausführen eines Stapel-Build (AWS CLI)
- Stoppen eines StapelsAWS CodeBuild

GitHub Action-Runner in AWS CodeBuild

Eine GitHub Aktion ist eine Aktion, die speziell für die Verwendung mit GitHub Workflows entwickelt wurde. Einzelheiten zu GitHub Aktionen finden Sie in der Dokumentation zu GitHub Aktionen.

Es gibt zwei Möglichkeiten, GitHub Aktionen mit zu verwenden CodeBuild:

- Sie können Ihr Projekt so konfigurieren, dass selbst gehostete GitHub Actions-Runner in CodeBuild Containern eingerichtet werden, um Ihre GitHub Actions-Workflow-Jobs zu verarbeiten.
- Sie können einen CodeBuild -verwalteten Action-Runner verwenden, um darin GitHub Aktionen auszuführen. CodeBuild

Sie können wählen, ob Sie selbst gehostete GitHub Action-Runner in einrichten möchten. CodeBuild Dazu gehört die Einrichtung eines Webhooks mithilfe Ihres CodeBuild Projekts und die Aktualisierung Ihres GitHub Aktions-Workflow-YAML, sodass selbst gehostete Runner verwendet werden, die auf Computern gehostet werden. CodeBuild Dadurch können Ihre GitHub Actions-Workflow-Jobs nativ integriert werden. AWS

Sie können sich auch dafür entscheiden, einen von CodeBuild -verwalteten Action-Runnern zu verwenden, um darin GitHub CodeBuild Aktionen auszuführen. Dies beinhaltet das Hinzufügen

`steps` zu Ihrer Buildspec mithilfe der GitHub Action-Syntax, die in einer von Befehlen getrennten Phase ausgeführt wird. CodeBuild Dadurch können Ihre GitHub Aktionen in CodeBuild Funktionen wie das Zwischenspeichern von Abhängigkeiten und Batch-Builds integriert werden.

Themen

- Richten Sie selbst gehostete GitHub Action-Runner ein in AWS CodeBuild
- Verwenden Sie die GitHub Aktionssyntax in einer Buildspec in AWS CodeBuild

Richten Sie selbst gehostete GitHub Action-Runner ein in AWS CodeBuild

Sie können Ihr Projekt so konfigurieren, dass selbst gehostete GitHub Actions-Runner in CodeBuild Containern eingerichtet werden, um Ihre GitHub Actions-Workflow-Jobs zu verarbeiten. Dies kann erreicht werden, indem Sie mithilfe Ihres CodeBuild Projekts einen Webhook einrichten und Ihre GitHub Aktions-Workflow-YAML so aktualisieren, dass selbst gehostete Runner verwendet werden, die auf Maschinen gehostet werden. CodeBuild Weitere Informationen finden Sie unter Über selbst gehostete Runner.

Die allgemeinen Schritte zur Konfiguration eines CodeBuild Projekts für die Ausführung von GitHub Actions-Jobs lauten wie folgt:

1. Falls Sie dies noch nicht getan haben, erstellen Sie ein persönliches Zugriffstoken oder stellen Sie eine Verbindung mit einer OAuth-App her, mit der Sie Ihr Projekt verbinden möchten. GitHub
2. Navigieren Sie zur CodeBuild Konsole, erstellen Sie ein CodeBuild Projekt mit einem Webhook und richten Sie Ihre Webhook-Filter ein.
3. Aktualisieren Sie Ihren GitHub Aktions-Workflow YAML, um Ihre GitHub Build-Umgebung zu konfigurieren.

Ein detaillierteres Verfahren finden Sie unterTutorial: Einen CodeBuild selbst GitHub gehosteten Actions-Runner konfigurieren.

Mit dieser Funktion können Ihre GitHub Actions-Workflow-Jobs nativ integriert werden AWS, was durch Funktionen wie IAM AWS CloudTrail, AWS Secrets Manager Integration und Amazon VPC für Sicherheit und Komfort sorgt. Sie können auf die neuesten Instance-Typen zugreifen, einschließlich ARM-basierter Instances.

Themen

- Tutorial: Einen CodeBuild selbst GitHub gehosteten Actions-Runner konfigurieren

- Über CodeBuild -hosted GitHub Actions Runner

Tutorial: Einen CodeBuild selbst GitHub gehosteten Actions-Runner konfigurieren

Dieses Tutorial zeigt Ihnen, wie Sie Ihre CodeBuild Projekte für die Ausführung GitHub von Actions-Jobs konfigurieren.

Voraussetzungen

Um dieses Tutorial abzuschließen, müssen Sie zunächst:

- Stellen Sie eine Connect mit einer OAuth-App her oder erstellen Sie ein persönliches Zugriffstoken. Wenn Sie eine Verbindung mit einer OAuth-App herstellen möchten, müssen Sie dazu die CodeBuild Konsole verwenden. Wenn Sie ein persönliches Zugriffstoken erstellen möchten, können Sie entweder die CodeBuild Konsole oder die API verwenden. ImportSourceCredentials Weitere Anweisungen finden Sie unterGitHub und GitHub Enterprise Server-Zugriffstoken.
- Connect CodeBuild zu Ihrem GitHub Konto her. Dazu können Sie einen der folgenden Schritte ausführen:
  - Sie können in GitHub der Konsole einen Quellanbieter hinzufügen. Sie können eine Verbindung entweder mit einer OAuth-App oder einem persönlichen Zugriffstoken herstellen. Anweisungen finden Sie unter GitHub Mit einem Zugriffstoken (Konsole) Connect .
  - Sie können Ihre GitHub Anmeldeinformationen über die ImportSourceCredentials API importieren. Dies ist nur mit einem persönlichen Zugriffstoken möglich. Wenn Sie eine Verbindung mit einer OAuth-App herstellen, müssen Sie die Verbindung stattdessen über die Konsole herstellen. Anweisungen finden Sie unter GitHub Mit einem Zugriffstoken (CLI) Connect .

> **Note**
>
> Dies ist nur erforderlich, wenn Sie noch keine Verbindung zu GitHub Ihrem Konto hergestellt haben.

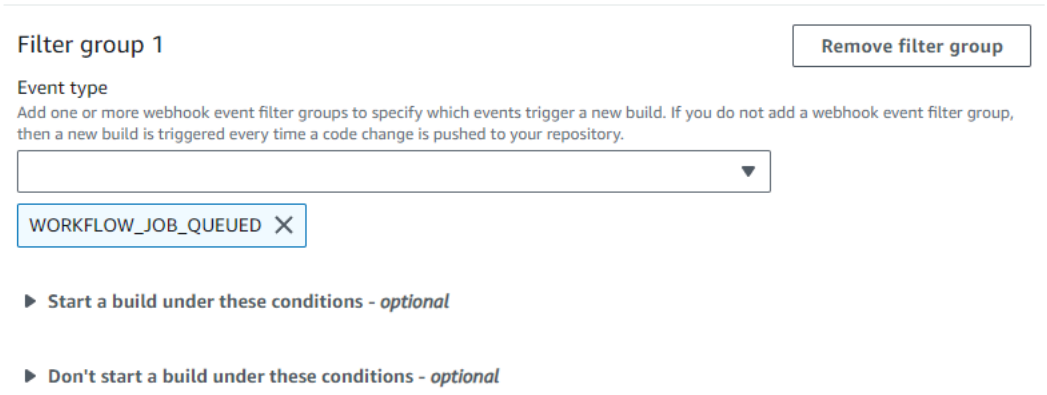
Schritt 1: Erstellen Sie ein CodeBuild Projekt mit einem Webhook

In diesem Schritt erstellen Sie ein CodeBuild Projekt mit einem Webhook und überprüfen es in der GitHub Konsole. Sie können GitHub Enterprise auch als Quellanbieter wählen. Weitere Informationen zum Erstellen eines Webhooks in GitHub Enterprise finden Sie unterGitHub manuelle Webhooks.

So erstellen Sie ein CodeBuild Projekt mit einem Webhook

1. Öffnen Sie die AWS CodeBuild Konsole unter https://console.aws.amazon.com/codesuite/codebuild/home.
2. Erstellen Sie ein Build-Projekt. Weitere Informationen finden Sie unter Erstellen Sie ein Build-Projekt (Konsole) und Ausführen eines Build (Konsole).
   - In Source (Quelle):
     - Wählen Sie als Quellanbieter die Option GitHub.
     - Wählen Sie unter Repository die Option Repository in meinem GitHub Konto aus.
     - Geben Sie als Repository URL (Repository-URL) die URL **`https://github.com/`*`user-name`*`/`*`repository-name`*** ein.
   - Unter Webhook-Ereignisse der Primärquelle:
     - Wählen Sie für Webhook — optional die Option Jedes Mal neu erstellen, wenn eine Codeänderung in dieses Repository übertragen wird.
     - Wählen Sie als Ereignistyp WORKFLOW_JOB_QUEUED aus. Sobald diese Option aktiviert ist, werden Builds nur noch durch Workflow-Auftragsereignisse für Aktionen ausgelöst. GitHub

     > **Note**
     >
     > CodeBuild verarbeitet GitHub Aktions-Workflow-Job-Ereignisse nur, wenn ein Webhook Filtergruppen enthält, die den WORKFLOW_JOB_QUEUED-Ereignisfilter enthalten.



   - In Environment (Umgebung):

- Wählen Sie ein unterstütztes Umgebungs-Image und Compute aus. Beachten Sie, dass Sie die Möglichkeit haben, die Image- und Instanzeinstellungen zu überschreiben, indem Sie ein Label in Ihrem GitHub Aktions-Workflow YAML verwenden. Weitere Informationen finden Sie unter Schritt 2: Aktualisieren Sie Ihren GitHub Aktions-Workflow YAML.
- In Buildspec (Build-Spezifikation):
  - Beachten Sie, dass Ihre Buildspec ignoriert wird. Stattdessen CodeBuild wird es überschrieben, um Befehle zu verwenden, die den selbst gehosteten Runner einrichten. Die Hauptaufgabe dieses Projekts besteht darin, einen selbst gehosteten Runner für die Ausführung von CodeBuild GitHub Actions-Workflow-Jobs einzurichten.

3. Fahren Sie mit den Standardwerten fort und wählen Sie dann Build-Projekt erstellen.
4. Öffnen Sie die GitHub Konsole unter, `https://github.com/`*`user-name`*`/`*`repository-name`*`/settings/hooks` um zu überprüfen, ob ein Webhook erstellt wurde und für die Übermittlung von Workflow-Auftragsereignissen aktiviert ist.

Schritt 2: Aktualisieren Sie Ihren GitHub Aktions-Workflow YAML

In diesem Schritt aktualisieren Sie Ihre GitHub Aktions-Workflow-YAML-Datei, um Ihre Build-Umgebung `GitHub`zu konfigurieren und selbst gehostete GitHub Actions-Runner zu verwenden. CodeBuild Weitere Informationen finden Sie unter Verwenden von Labels mit selbst gehosteten Runnern.

Aktualisieren Sie Ihren GitHub Aktions-Workflow (YAML)

Navigieren Sie zu Ihrem GitHub Aktions-Workflow-YAML `GitHub`und aktualisieren Sie die `runs-on`Einstellung, um Ihre Build-Umgebung zu konfigurieren. Dazu können Sie einen der folgenden Schritte ausführen:

- Sie können den Projektnamen und die Run-ID angeben. In diesem Fall verwendet der Build Ihre bestehende Projektkonfiguration für die Berechnung, das Image, die Image-Version und die Instanzgröße. Der Projektname wird benötigt, um die zugehörigen AWS Einstellungen Ihres GitHub Actions-Jobs mit einem bestimmten CodeBuild Projekt zu verknüpfen. Durch die Aufnahme des Projektnamens in die YAML CodeBuild ist es möglich, Jobs mit den richtigen Projekteinstellungen aufzurufen. Durch Angabe der Run-ID CodeBuild wird Ihr Build bestimmten Workflow-Läufen zugeordnet und der Build gestoppt, wenn der Workflow-Lauf abgebrochen wird. Weitere Informationen finden Sie unter `github`Kontext.

```
runs-on: codebuild-<project-name>-${{ github.run_id }}-${{ github.run_attempt }}
```

> **Note**
>
> Stellen Sie sicher, dass Ihr <project-name>Name mit dem Namen des Projekts übereinstimmt, das Sie im vorherigen Schritt erstellt haben. Wenn es nicht übereinstimmt, CodeBuild wird der Webhook nicht verarbeitet und der GitHub Aktionsworkflow kann hängen bleiben.

Im Folgenden finden Sie ein Beispiel für einen GitHub Aktions-Workflow-YAML:

```
name: Hello World
on: [push]
jobs:
  Hello-World-Job:
    runs-on: codebuild-myProject-${{ github.run_id }}-${{ github.run_attempt }}
    steps:
      - run: echo "Hello World!"
```

- Sie können auch Ihr Bild und Ihren Berechnungstyp im Label überschreiben. Dadurch werden die Umgebungseinstellungen Ihres Projekts außer Kraft gesetzt. Verwenden Sie die folgende Syntax, um Ihre Umgebungseinstellungen für einen Amazon EC2 EC2-Compute-Build zu überschreiben:

```
runs-on: codebuild-<project-name>-${{ github.run_id }}-
${{ github.run_attempt }}-<image>-<image-version>-<instance-size>
```

Verwenden Sie die folgende Syntax, um Ihre Umgebungseinstellungen für einen Lambda-Compute-Build zu überschreiben:

```
runs-on: codebuild-<project-name>-${{ github.run_id }}-
${{ github.run_attempt }}-<environment-type>-<runtime-version>-<instance-size>
```

Im Folgenden finden Sie ein Beispiel für einen GitHub Aktions-Workflow-YAML:

```
name: Hello World
on: [push]
jobs:
  Hello-World-Job:
    runs-on: codebuild-myProject-${{ github.run_id }}-${{ github.run_attempt }}-
arm-3.0-small
```

```
    steps:
      - run: echo "Hello World!"
```

> **Note**
>
> Wenn eine von GitHub -hosted runners bereitgestellte Abhängigkeit in der CodeBuild Umgebung nicht verfügbar ist, können Sie die Abhängigkeit mithilfe von GitHub Actions in Ihrer Workflow-Ausführung installieren. Sie können die `setup-python`Aktion beispielsweise verwenden, um Python für Ihre Build-Umgebung zu installieren.

Unterstützte Compute-Images

In Ihrem Label können Sie Ihre Amazon EC2 EC2-Umgebungseinstellungen überschreiben, indem Sie die Werte in den ersten drei Spalten verwenden. CodeBuild stellt die folgenden Amazon EC2 EC2-Compute-Images bereit:

Image	Image-Version	Instance-Größe	Plattform	Image-Kennung	Definition

`linux`	`4.0`	`small`
		`medium`
		`large`
		`xlarge`
			Amazon Linux 2	`aws/codebuild/amazonlinux2-x86_64-standard:4.0`	al2/standard/4.0

`linux`	`5.0`	`2xlarge`
		`gpu_small`
		`gpu_large`
			Amazon Linux 2023	`aws/codebuild/amazonlinux2-x86_64-standard:5.0`	al2/standard/5,0

`arm`	`2.0`	`small` `large`	Amazon Linux 2	`aws/codebuild/amazonlinux2-`	al2/aarch64/standard/2.0

Image	Image-Version	Instance-Größe	Plattform	Image-Kennung	Definition
				`aarch64-standard:2.0`	
`arm`	`3.0`		Amazon Linux 2023	`aws/codebuild/amazonlinux2-aarch64-standard:3.0`	al2/aarch64/standard/3.0
`ubuntu`	`5.0`	`small` `medium` `large` `xlarge` `2xlarge` `gpu_small` `gpu_large`	Ubuntu 20.04	`aws/codebuild/standard:5.0`	ubuntu/standard/5.0
`ubuntu`	`6.0`		Ubuntu 22.04	`aws/codebuild/standard:6.0`	Ubuntu/Standard/6.0
`ubuntu`	`7.0`		Ubuntu 22.04	`aws/codebuild/standard:7.0`	Ubuntu/Standard/7.0
`windows`	`1.0`	`medium` `large`	Windows Server Core 2019	`aws/codebuild/windows-base:2019-1.0`	N/A
`windows`	`2.0`		Windows Server Core 2019	`aws/codebuild/windows-base:2019-2.0`	N/A

Image	Image-Version	Instance-Größe	Plattform	Image-Kennung	Definition
`windows`	`3.0`		Windows Server Core 2019	`aws/codebuild/windows-base:2019-3.0`	N/A

Darüber hinaus können Sie Ihre Lambda-Umgebungseinstellungen überschreiben, indem Sie die folgenden Werte verwenden. Weitere Informationen zu CodeBuild Lambda Compute finden Sie unter Working withArbeiten mit AWS Lambda Compute in AWS CodeBuild. CodeBuild unterstützt die folgenden Lambda-Compute-Images:

Umgebungstyp	Laufzeitversion	Instance-Größe			

`linux-lambda`
`arm-lambda`
	`dotnet6`
	`go1.21`
	`corretto11`
	`corretto17`
	`corretto21`
	`nodejs18`
	`nodejs20`
	`python3.11`
	`python3.12`
		`1GB`
		`2GB`
		`4GB`
		`8GB`
		`10GB`			

Umgebungstyp	Laufzeitversion	Instance-Größe			
	`ruby3.2`				

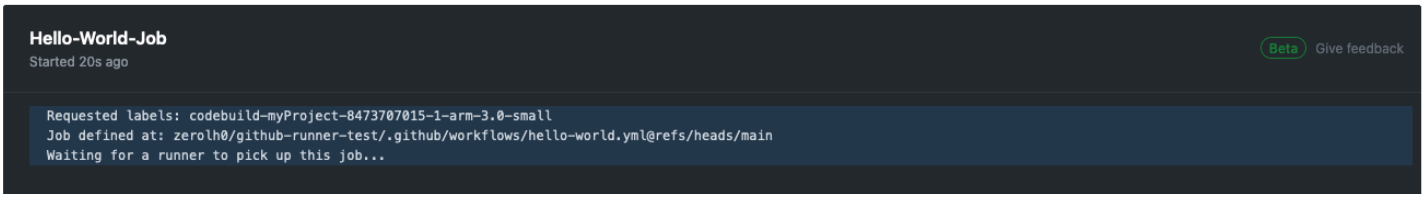
Weitere Informationen finden Sie unter Berechnungsmodi und Typen der Build-Umgebung und Docker-Images bereitgestellt von CodeBuild.

Schritt 3: Überprüfen Sie Ihre Ergebnisse

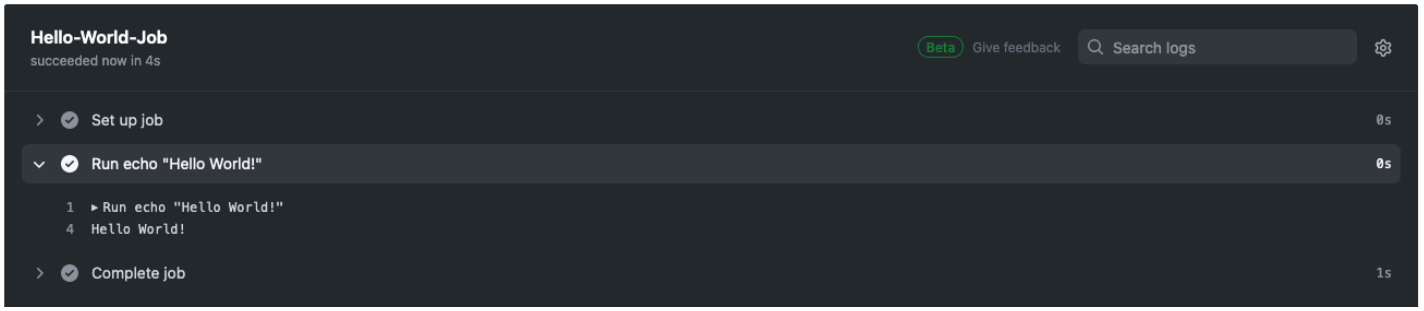
Immer wenn ein GitHub Aktionsworkflow ausgeführt wird, CodeBuild werden die Workflow-Auftragsereignisse über den Webhook empfangen. CodeBuild Startet für jeden Job im Workflow einen Build, um einen kurzlebigen GitHub Actions-Runner auszuführen. Der Runner ist für die Ausführung eines einzelnen Workflow-Jobs verantwortlich. Sobald der Job abgeschlossen ist, werden der Runner und der zugehörige Build-Prozess sofort beendet.

Um Ihre Workflow-Job-Logs einzusehen, navigieren Sie zu Ihrem Repository in GitHub, wählen Sie Aktionen, wählen Sie den gewünschten Workflow aus und wählen Sie dann den spezifischen Job aus, für den Sie die Logs überprüfen möchten.

Sie können die angeforderten Labels im Protokoll überprüfen, während der Job darauf wartet, von einem selbst gehosteten Runner übernommen zu werden. CodeBuild



Sobald der Job abgeschlossen ist, können Sie das Protokoll des Jobs einsehen.



Über CodeBuild -hosted GitHub Actions Runner

Wann sollte ich die Bild- und Instanzüberschreibungen in das Label aufnehmen?

Sie können die Image- und Instanzüberschreibungen in das Label aufnehmen, um für jeden Ihrer GitHub Actions-Workflow-Jobs eine andere Build-Umgebung anzugeben. Dies ist möglich, ohne dass mehrere CodeBuild Projekte oder Webhooks erstellt werden müssen. Dies ist beispielsweise nützlich, wenn Sie eine Matrix für Ihre Workflow-Jobs verwenden müssen.

```
name: Hello World
on: [push]
jobs:
  Hello-World-Job:
    runs-on: codebuild-myProject-${{ github.run_id }}-${{ github.run_attempt }}-
${{ matrix.os }}
    strategy:
      matrix:
        os: [arm-3.0-small, al2-5.0-large]
    steps:
      - run: echo "Hello World!"
```

> **Note**
>
> Anführungszeichen können erforderlich sein, wenn `runs-on` mehrere Beschriftungen den GitHub Aktionskontext enthalten.

Kann ich diese Funktion verwenden AWS CloudFormation ?

Ja, Sie können eine Filtergruppe in Ihre AWS CloudFormation Vorlage aufnehmen, die in Ihrem Projekt-Webhook einen Jobereignisfilter für den GitHub Aktionsworkflow festlegt.

```
Triggers:
  Webhook: true
  FilterGroups:
    - - Type: EVENT
        Pattern: WORKFLOW_JOB_QUEUED
```

Weitere Informationen finden Sie unter GitHub Webhook-Ereignisse filtern ()AWS CloudFormation.

Wenn Sie Hilfe beim Einrichten von Projektanmeldedaten in Ihrer AWS CloudFormation Vorlage benötigen, finden Sie weitere Informationen AWS::CodeBuild::SourceCredentialim AWS CloudFormation Benutzerhandbuch.

Wie kann ich Geheimnisse maskieren, wenn ich diese Funktion verwende?

Standardmäßig werden Geheimnisse, die im Protokoll gedruckt werden, nicht maskiert. Wenn Sie Ihre Geheimnisse maskieren möchten, können Sie die folgende Syntax verwenden:`::add-mask::`*`value`*. Das Folgende ist ein Beispiel dafür, wie Sie diese Syntax in Ihrer YAML verwenden können:

```
name: Secret Job
on: [push]
jobs:
  Secret-Job:
    runs-on: codebuild-myProject-${{ github.run_id }}-${{ github.run_attempt }}
    env:
      SECRET_NAME: "secret-name"
    steps:
      - run: echo "::add-mask::$SECRET_NAME"
```

Weitere Informationen finden Sie unter Maskieren eines Werts bei einer Anmeldung. GitHub

In welchen Regionen wird die Verwendung eines CodeBuild -gehosteten GitHub Actions-Runners unterstützt?

CodeBuild-gehostete GitHub Actions-Runner werden in allen CodeBuild Regionen unterstützt. Weitere Informationen darüber, AWS-Regionen wo CodeBuild es verfügbar ist, findest du unter AWS Dienste nach Regionen.

Welche Plattformen unterstützen die Verwendung eines CodeBuild -gehosteten GitHub Actions-Runners?

CodeBuild-gehostete GitHub Actions-Runner werden sowohl auf Amazon EC2 als auch auf Compute unterstützt. AWS Lambda Sie können die folgenden Plattformen verwenden: Amazon Linux 2, Amazon Linux 2023, Ubuntu und Windows Server Core 2019. Weitere Informationen finden Sie unter EC2-Compute-Images und Lambda-Computing-Bilder.

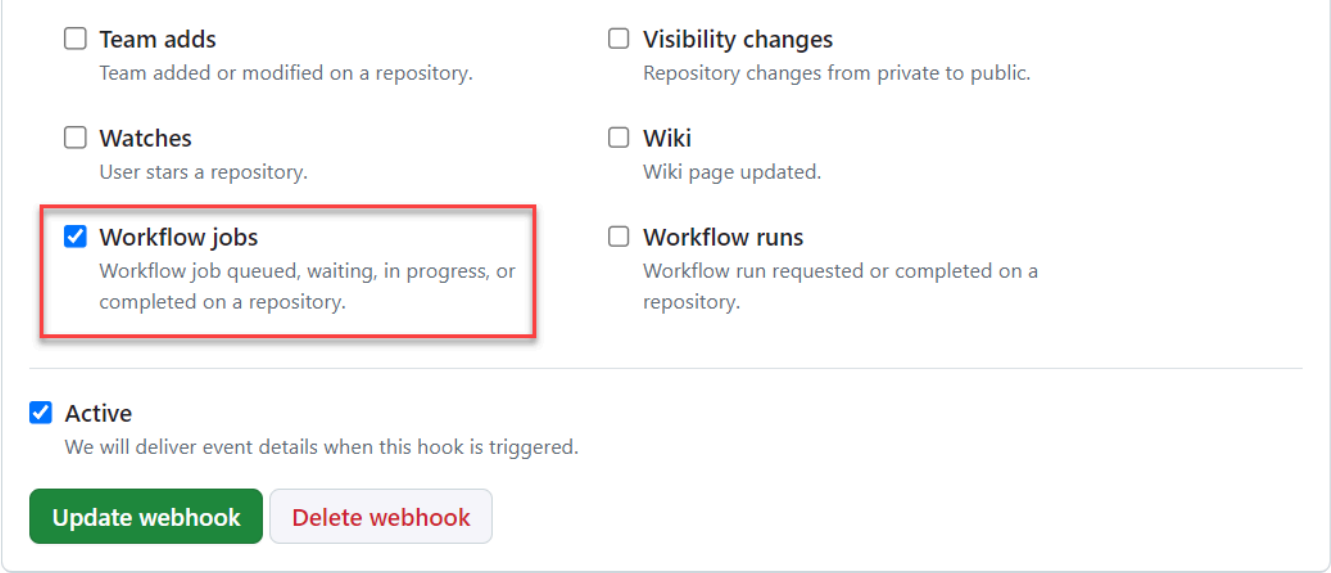
Fehlerbehebung: Wie behebe ich Fehler, wenn der Webhook nicht funktioniert?

Problem: Ihr Webhook funktioniert nicht oder Ihr Workflow-Job hängt. GitHub

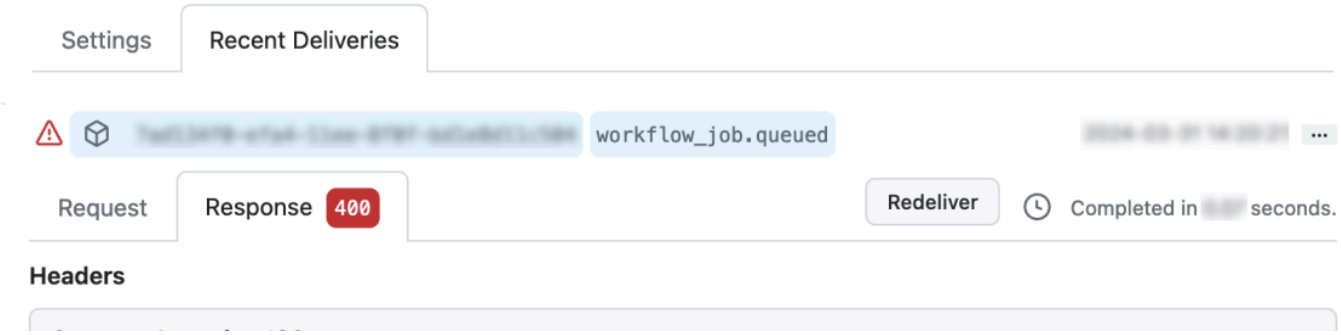
Mögliche Ursache: Ihr Webhook-Workflow-Job-Ereignis löst möglicherweise keinen Build aus. Überprüfen Sie die Antwortprotokolle, um die Antwort oder Fehlermeldung einzusehen.

Empfohlene Lösung: Verwenden Sie die folgenden Anweisungen, um diesen Fehler zu debuggen.

1. Öffnen Sie die GitHub Konsole unter`https://github.com/user-name/repository-name/settings/hooks`, um die Webhook-Einstellungen Ihres Repositorys einzusehen. Auf dieser Seite siehst du einen Webhook, der für dein Repository erstellt wurde.
2. Wähle Bearbeiten und bestätige, dass der Webhook für die Übermittlung von Workflow-Job-Ereignissen aktiviert ist.



3. Navigieren Sie zur Registerkarte Letzte Lieferungen, suchen Sie das entsprechende `workflow_job.queued` Ereignis und erweitern Sie das Ereignis.
4. Überprüfe das Feld „Labels“ in der Payload und vergewissere dich, dass es den Erwartungen entspricht.
5. Überprüfen Sie abschließend die Registerkarte „Antwort“, da sie die Antwort oder Fehlermeldung enthält, von CodeBuild der zurückgegeben wurde.



Verwenden Sie die GitHub Aktionssyntax in einer Buildspec in AWS CodeBuild

Sie können einen von CodeBuild -managed Action Runner verwenden, um darin Aktionen auszuführen. GitHub CodeBuild Dies kann durch Hinzufügen `steps` zu einer beliebigen Phase in Ihrer Buildspec-Datei geschehen.

CodeBuild Buildspecs unterstützen eine Liste sequentieller GitHub Aktionsschritte, die in einer von Befehlen getrennten Phase ausgeführt werden. CodeBuild Diese GitHub Aktionen lassen sich in die vorhandenen Funktionen integrieren, zu denen das Zwischenspeichern von Abhängigkeiten, Batch-Builds, Zugriff AWS Secrets Manager auf und mehr gehören. CodeBuild

Themen

Wie fange ich an, eine GitHub Aktion in meiner Buildspec zu verwenden?

Die wichtigsten Schritte zur Verwendung einer GitHub Aktion in Ihrer Buildspec lauten wie folgt:

1. Falls Sie dies noch nicht getan haben, verbinden Sie Ihr Projekt mit. GitHub

   Dazu können Sie einen der folgenden Schritte ausführen:

   - Sie können in GitHub der Konsole einen Quellanbieter hinzufügen. Weitere Informationen finden Sie unter GitHub Mit einem Zugriffstoken (Konsole) Connect .
   - Sie können Ihre GitHub Anmeldeinformationen über die CodeBuild API importieren. Weitere Informationen finden Sie unter GitHub Mit einem Zugriffstoken (CLI) Connect .

> **Note**
>
> Dies muss nur geschehen, wenn Sie noch keine Verbindung zu GitHub einem anderen Projekt hergestellt haben.

2. In der Buildspec Ihres Projekts können Sie hinzufügen`steps`, von denen jede auf eine Aktion verweist. GitHub Dies kann in der CodeBuild Konsole oder in Ihrem Quell-Repository bearbeitet werden. Jede Buildphase unterstützt entweder eine Liste von Befehlen oder eine Liste von Schritten, aber beide können nicht in derselben Phase verwendet werden. Weitere Informationen finden Sie unter Verwenden Sie die GitHub Aktionssyntax in einer Buildspec in AWS CodeBuild.

Welche GitHub Aktionen kann ich in meiner Buildspec verwenden?

Sie können jede auf dem GitHub Marketplace verfügbare Aktion verwenden, die nicht mit diesen Einschränkungen kollidiert.

Kann ich andere Quellanbieter verwenden als GitHub wenn ich GitHub Actions in meiner Buildspec verwende?

Ja, aber es GitHub ist immer noch eine Verbindung zu erforderlich, um sich mit GitHub Aktionen zu authentifizieren und darauf zuzugreifen. GitHub Weitere Informationen finden Sie unter GitHub und GitHub Enterprise Server-Zugriffstoken.

Warum muss ich eine Verbindung GitHub als Quellanbieter herstellen, um GitHub Actions in meiner Buildspec verwenden zu können?

Um GitHub Actions in Ihrer Buildspec verwenden zu können, muss die Quelle auf einen Build-Computer heruntergeladen werden. Anonyme Downloads sind ratenbegrenzt. Wenn Sie also eine Verbindung herstellen, kann dies dazu beitragen GitHub, einen konsistenten Zugriff zu gewährleisten.

Wie viel kostet es, GitHub Aktionen in meiner Buildspec zu verwenden?

Die Verwendung von GitHub Aktionen in Ihrer Buildspec wird ohne zusätzliche Kosten unterstützt.

Welche Regionen unterstützen die Verwendung von GitHub Aktionen in meiner Buildspec?

Die Verwendung von GitHub Aktionen in Ihrer Buildspec wird in allen Regionen unterstützt. CodeBuild Weitere Informationen darüber, AWS-Regionen wo verfügbar CodeBuild ist, finden Sie unter AWS Dienste nach Regionen.

Bewährte Methoden für die Verwendung von GitHub Aktionen in Ihrer Buildspec

GitHub Aktionen sind Open Source und werden von der Community erstellt und verwaltet. Wir folgen dem Modell der gemeinsamen Verantwortung und betrachten den Quellcode von GitHub Actions als Kundendaten, für die Sie verantwortlich sind. GitHub Aktionen können Zugriff auf Geheimnisse, Repository-Token, Quellcode und Konto-Links gewährt werden. Stellen Sie sicher, dass Sie von der Vertrauenswürdigkeit und Sicherheit der GitHub Aktionen, die Sie ausführen möchten, überzeugt sind.

Spezifischere Leitlinien und bewährte Sicherheitsverfahren für GitHub Aktionen:

- Verschärfung der Sicherheit
- Verhinderung eigener Anfragen
- Nicht vertrauenswürdige Eingaben
- Wie können Sie Ihren Bausteinen vertrauen

Einschränkungen bei der Verwendung von GitHub Aktionen in Ihrer Buildspec in CodeBuild

- GitHub Aktionen in Ihrer Buildspec, die intern auf den `github`Kontext angewiesen sind oder auf GitHub spezifische Ressourcen verweisen, wie Pull-Requests und Issues, werden in nicht unterstützt. CodeBuild Die folgenden Aktionen funktionieren beispielsweise nicht in: CodeBuild
  - GitHub Aktionen, mit denen versucht wird, GitHub Ressourcen hinzuzufügen, zu ändern oder zu aktualisieren, z. B. Aktionen, die Pull-Requests aktualisieren oder Probleme in verursachen GitHub.

> **Note**
>
> Die meisten offiziellen GitHub Aktionen, die unter https://github.com/actions aufgeführt sind, hängen vom `github` Kontext ab. Verwenden Sie stattdessen die im GitHub Marketplace verfügbaren Aktionen.

- GitHub Aktionen in Ihrer Buildspec, bei denen es sich um Docker-Container-Aktionen handelt, funktionieren, aber für Ihr Build-Projekt muss der privilegierte Modus aktiviert sein und vom Standard-Docker-Benutzer (root) ausgeführt werden.
  - Aktionen müssen als Root-Benutzer ausgeführt werden. Weitere Informationen finden Sie im Thema USER unter Dockerfile-Unterstützung für GitHub Aktionen.

- GitHub Aktionen in Ihrer Buildspec werden in CodeBuild Projekten, die für die Ausführung unter Windows konfiguriert sind, nicht unterstützt.
- GitHub Aktionsaufträge (Gruppen von Schritten) und Eigenschaften von GitHub Aktionsaufträgen in Ihrer Buildspezifikation werden nicht unterstützt.
- GitHub Aktionen in Ihrer Buildspec werden in CodeBuild Projekten nicht unterstützt, die so konfiguriert sind, dass sie durch einen Webhook für ein öffentliches Git-Repository ausgelöst werden. Weitere Informationen finden Sie unter. git-credential-helper
- VPC-Builds ohne öffentlichen Internetzugang können keine GitHub Aktionen in Ihrer Buildspec ausführen.
- Jede Buildphase unterstützt entweder eine Liste von Befehlen oder eine Liste von Schritten, aber beide können nicht in derselben Phase verwendet werden. Im folgenden Beispiel werden Schritte in der Phase vor der Erstellung zum Auflisten von GitHub Aktionen verwendet, während Befehle in der Erstellungsphase zum Auflisten von CodeBuild Befehlen verwendet werden.

```
version: 0.2
phases:
  pre-build:
    steps:
      - name: Lint Code Base
        uses: github/super-linter@v4
        env:
          VALIDATE_ALL_CODEBASE: 'true'
          DEFAULT_BRANCH: main
  build:
    commands:
      - echo "Building..."
      - npm run build
```

GitHub Action-Runner-Buildspec-Referenz

Dieses Thema enthält die Buildspec-Referenz für Action-Runner-Eigenschaften. GitHub

steps

Optionale Sequenz. Schritte werden verwendet, um Befehle und Aktionen in auszuführen. CodeBuild Weitere Informationen finden Sie unter Verwenden Sie die GitHub Aktionssyntax in einer Buildspec in AWS CodeBuild.

> **Note**
>
> Jede Buildphase unterstützt entweder eine Liste von `commands` oder eine Liste von`steps`, aber beide können nicht in derselben Phase verwendet werden.

Jeder Build-Schritt enthält die folgenden Eigenschaften.

id

Optional. Der Bezeichner für den Schritt, der verwendet werden kann, um in anderen Kontexten auf den Schritt zu verweisen.

if

Optional. Eine bedingte Anweisung, die verwendet werden kann, um zu verhindern, dass ein Schritt ausgeführt wird, sofern keine Bedingung erfüllt ist. Diese Anweisung kann jeden unterstützten Kontext verwenden, z. B. den Verweis auf Umgebungsvariablen von CodeBuild oder einen Ausdruck.

Name

Optional. Der Name für den Schritt. Wenn der Name nicht angegeben ist, wird als Name standardmäßig der im `run` Befehl angegebene Text verwendet.

verwendet

Die Aktion, die für den Schritt ausgeführt wird. Bei einigen Aktionen müssen Sie Eingaben mithilfe von festlegen`with`. In der README-Datei der Aktion finden Sie Informationen darüber, welche Eingaben erforderlich sind. Weitere Informationen finden Sie unter Welche GitHub Aktionen kann ich in meiner Buildspec verwenden?.

Wenn in Ihrer Erstellungsphase angegeben `uses` ist, kann es nicht mit `run` verwendet werden.

> **Note**
>
> Es wird empfohlen, dass Sie die Version der Aktion angeben, die Sie verwenden. Dies kann durch Angabe eines Git-Ref-, SHA- oder Docker-Tags erfolgen. Weitere Informationen finden Sie unter steps.uses Syntax.

ausführen

Ein Befehl, der Befehlszeilenprogramme ausführt. Dies können einzeilige Befehle oder mehrzeilige Befehle sein. Standardmäßig werden diese Befehle mit Shells ausgeführt, bei denen keine Anmeldung erforderlich ist. Um eine andere Shell auszuwählen, verwenden Sie`shell`.

Wenn in Ihrer Buildphase angegeben `run` ist, kann es nicht mit verwendet werden`uses`.

Schale

Optional. Die für diese Sequenz angegebene Shell. Informationen zu unterstützten Shell-Parametern finden Sie unter steps.shell. Falls nicht spezifiziert, wird bash als Shell verwendet. Wenn bash nicht verfügbar ist, wird sh verwendet.

mit

Optional. Eine Karte von Eingabeparametern, die durch die Aktion definiert wurden. Jeder Parameter ist ein Schlüssel/Wert-Paar.

mit.args

Optional. Eine Zeichenfolge, die Eingaben für einen Docker-Container definiert.

mit.entrypoint

Optional. Der für das Dockerfile angegebene Docker-Einstiegspunkt.

env

Optional. Die Variablen, die für Schritte angegeben sind, die in der Umgebung verwendet werden sollen.

continue-on-error

Optional. Ein boolescher Wert, der angibt, ob ein Fehler dieser Schrittsequenz ignoriert werden kann.

`false`

Der Standardwert. Wenn diese Schrittsequenz fehlschlägt, schlägt der Build fehl.

`true`

Wenn diese Schrittsequenz fehlschlägt, kann der Build trotzdem erfolgreich sein.

Timeout in Minuten

Optional. Die maximale Anzahl von Minuten, für die der Schritt ausgeführt werden kann, bevor er beendet wird. Standardmäßig gibt es kein Timeout. Wenn das Schritttimeout das Build-Timeout überschreitet, wird der Schritt beendet, wenn das Build-Timeout erreicht ist.

Im Folgenden finden Sie ein Beispiel für die Verwendung der Super-Linter-Aktion: GitHub

```
version: 0.2
phases:
  build:
    steps:
      - name: Lint Code Base
        uses: github/super-linter@v5
        env:
          VALIDATE_ALL_CODEBASE: true
          USE_FIND_ALGORITHM: true
          FILTER_REGEX_INCLUDE: '/github/workspace/buildspec.yml'
```

GitHub Syntaxbeispiele für Aktionen mit AWS CodeBuild

Diese Gruppen von Beispielen können verwendet werden, um mit GitHub Aktionen in Ihrer Buildspec in zu experimentieren. CodeBuild

Themen

- Beispiel Super-Linter Action GitHub
- Beispiel für eine Batch-Erstellung eines Diagramms
- Beispiel für Amazon CodeGuru Reviewer
- AWS Secrets Manager Beispiel
- Beispiel für eine Umgebungsvariable
- Beispiel für eine exportierte Umgebungsvariable

Beispiel Super-Linter Action GitHub

Dieses Beispiel zeigt, wie die GitHub Super-Linter-Aktion zu einem Projekt hinzugefügt wird. CodeBuild Die Super-Linter-Aktion untersucht den Code, findet Bereiche, in denen der Code Fehler, Formatierungsprobleme und verdächtige Konstrukte enthält, und gibt die Ergebnisse dann an der Konsole aus. CodeBuild

Sie können die GitHub Super-Linter-Aktion zu Ihrem CodeBuild Projekt hinzufügen, indem Sie den Phasenabschnitt Ihrer Buildspec-Datei aktualisieren.

```
version: 0.2
phases:
  build:
    steps:
      - name: Lint Code Base
        uses: github/super-linter@v5
        env:
          VALIDATE_ALL_CODEBASE: true
```

Die Super-Linter-Protokolle werden etwa wie folgt aussehen:

```
/github/workspace/hello-world/app.js:3:13: Extra semicolon.
/github/workspace/hello-world/app.js:9:92: Trailing spaces not allowed.
/github/workspace/hello-world/app.js:21:7: Unnecessarily quoted property 'body' found.
/github/workspace/hello-world/app.js:31:1: Expected indentation of 2 spaces but found
 4.
/github/workspace/hello-world/app.js:32:2: Newline required at end of file but not
 found.
```

Beispiel für eine Batch-Erstellung eines Diagramms

Das folgende Beispiel definiert ein Build-Diagramm, das eine Abhängigkeitskette erstellt und Befehle mithilfe von ausführt`steps`. In diesem Beispiel wird es zuerst `build1` ausgeführt, da es keine Abhängigkeiten hat. Da von abhängig `build2` ist, `build2` wird also ausgeführt`build1`, nachdem Build1 abgeschlossen ist. Weitere Informationen finden Sie unter Diagramm erstellen.

```
version: 0.2
batch:
  fast-fail: false
  build-graph:
    - identifier: build1
      env:
        variables:
          BUILD_ID: build1
      ignore-failure: false
    - identifier: build2
      env:
        variables:
          BUILD_ID: build2
```

```
      depend-on:
        - build1

phases:
  build:
    steps:
      - run: echo $BUILD_ID
```

Beispiel für Amazon CodeGuru Reviewer

Amazon CodeGuru Reviewer findet Probleme in Ihrem Java- und Python-Code und empfiehlt, wie Sie diese beheben können. Im folgenden Beispiel wird CodeGuru Reviewer verwendet, um vollständige Repository-Analyse-Code-Reviews bereitzustellen. Bei diesen Codeüberprüfungen wird der gesamte Code in einem bestimmten Zweig gescannt. Weitere Informationen finden Sie unter Code-Rezensionen mit GitHub Aktionen erstellen im Amazon CodeGuru Reviewer-Benutzerhandbuch.

```
version: 0.2
phases:
  build:
    steps:
        - name: Amazon CodeGuru Reviewer Scanner
          if: ${{ always() }}
          uses: aws-actions/codeguru-reviewer@v1.1
          with:
            s3_bucket: codeguru-reviewer-user

artifacts:
  files:
    - codeguru-results.sarif.json
```

> **Note**
>
> Ihr Amazon S3 S3-Bucket muss mit dem `codeguru-reviewer-` Präfix beginnen.

Die Protokolle werden in etwa wie folgt aussehen:

```
INFO CodeReview created with arn=arn:aws:codeguru-reviewer:region:account-
id:association:id:code-review:RepositoryAnalysis-job for job=job
INFO SARIF persisted to /github/workspace/codeguru-results.sarif.json
INFO Amazon CodeGuru Reviewer job execution completed
```

Nachdem der Amazon CodeGuru Reviewer-Job abgeschlossen ist, wird ein Sarif-Bericht als CodeBuild Artefakt generiert. Weitere Informationen finden Sie unter Vollständige Repository-Analyse im Amazon CodeGuru Reviewer-Benutzerhandbuch.

AWS Secrets Manager Beispiel

AWS Secrets Manager hilft Ihnen dabei, Datenbankanmeldedaten, Anwendungsanmeldedaten, OAuth-Token, API-Schlüssel und andere Geheimnisse während ihrer gesamten Lebensdauer zu verwalten, abzurufen und zu rotieren. Das folgende Beispiel definiert ein Geheimnis mit Secrets Manager und führt Befehle mit aus`steps`. Weitere Informationen finden Sie unter Was ist AWS Secrets Manager? im AWS Secrets Manager Benutzerhandbuch.

```
version: 0.2
env:
  secrets-manager:
    SECRET_VALUE: "arn:aws:secretsmanager:us-east-1:xxxx:secret:/secret-
l3IJg9:my_super_secret_key"
phases:
  build:
    steps:
      - run: echo $SECRET_VALUE
```

Die Protokolle werden in etwa wie folgt aussehen:

```
echo $SECRET_VALUE
env:
  SECRET_VALUE: ***
***
```

Beispiel für eine Umgebungsvariable

Das folgende Beispiel definiert Umgebungsvariablen in der `env` Sequenz. <bucket-name>Eine *`S3_BUCKET-Variable`* ist in der Buildspec definiert und ihr als Wert zugewiesen. Auf diese Variable wird in der if-Bedingung wie auf eine normale Umgebungsvariable verwiesen, indem das Dollarzeichen ($) für den Zugriff auf den Action env-Kontext verwendet wird. GitHub Weitere Informationen finden Sie unter envSequenz.

```
version: 0.2
env:
  variables:
    S3_BUCKET: "<bucket-name>"
```

```
phases:
  build:
    steps:
      - if: ${{ env.S3_BUCKET == '<bucket-name>' }}
        run: echo "S3 bucket is $S3_BUCKET"
```

Die Protokolle werden in etwa wie folgt aussehen:

```
echo "S3 bucket is $S3_BUCKET"
env:
  S3_BUCKET: my-s3-bucket
S3 bucket is my-s3-bucket
```

Beispiel für eine exportierte Umgebungsvariable

Exportierte Umgebungsvariablen werden in Verbindung mit verwendet CodePipeline , um Umgebungsvariablen aus der aktuellen Erstellungsphase in nachfolgende Phasen der Pipeline zu exportieren. Das folgende Beispiel definiert eine exportierte Umgebungsvariable unter der `env` Sequenz *`MY_VARIABLE und schreibt in die GITHUB_ENV-Umgebungsdatei`*.

```
version: 0.2
env:
  exported-variables:
    - MY_VARIABLE
phases:
  build:
    steps:
      - run: echo "MY_VARIABLE=my-value" >> $GITHUB_ENV
```

Weitere Informationen finden Sie in der API-Referenz. ExportedEnvironmentVariableAWS CodeBuild

Öffentliche Build-Projekte inAWS CodeBuild

AWS CodeBuildermöglicht es Ihnen, die Build-Ergebnisse, Protokolle und Artefakte für Ihre Build-Projekte der Öffentlichkeit zugänglich zu machen. Auf diese Weise können Mitwirkende an Ihren Quell-Repositorys die Ergebnisse anzeigen und die Artefakte eines Builds herunterladen, ohne dass sie Zugriff auf eineAWSKonto.

Wenn Sie die Builds Ihres Projekts der Öffentlichkeit zugänglich machen, werden alle Build-Ergebnisse, Protokolle und Artefakte eines Projekts, einschließlich Builds, die ausgeführt wurden, als das Projekt privat war, der Öffentlichkeit zugänglich gemacht. Wenn Sie ein öffentliches Build-

Projekt privat machen, stehen die Build-Ergebnisse für dieses Projekt ebenfalls nicht mehr für die Öffentlichkeit zur Verfügung.

Weitere Informationen zum Ändern der öffentlichen Sichtbarkeit der Build-Ergebnisse Ihres Projekts finden Sie unterAktivieren Sie den öffentlichen Build-Zugriffaus.

CodeBuild-URL für die öffentlichen Builds Ihres Projekts, die für Ihr Projekt eindeutig ist. Gehen Sie wie folgt vor, um die öffentliche URL für Ihr Build-Projekt zu erhalten:

1. Öffnen SieAWS CodeBuildconsole beihttps://console.aws.amazon.com/codesuite/codebuild/homeaus.
2. Wählen Sie im linken Navigationsbereich Build projects aus.
3. Klicken Sie auf den Link des Build-Projekts, für das Sie die öffentliche URL abrufen möchten.
4. Die öffentliche URL wird imURL des öffentlichen Projektsfield imKonfigurationAbschnitts erstellt. Sie können den Link zum Öffnen der URL auswählen oder die URL mit der Schaltfläche „Kopieren“ kopieren.

⚠ Warning

Beachten Sie beim Publizieren der Build-Ergebnisse Ihres Projekts:

- Alle Build-Ergebnisse, Protokolle und Artefakte eines Projekts, einschließlich Builds, die ausgeführt wurden, als das Projekt privat war, stehen der Öffentlichkeit zur Verfügung.
- Alle Build-Protokolle und Artefakte sind der Öffentlichkeit zugänglich. Umgebungsvariablen, Quellcode und andere sensible Informationen wurden möglicherweise an die Build-Protokolle und Artefakte ausgegeben. Sie müssen vorsichtig sein, welche Informationen an die Build-Protokolle ausgegeben werden. Einige Best Practices sind:
  - Speichern Sie keine sensiblen Werte, dies gilt insbesondere fürAWSZugriff auf Schlüssel-IDs und geheime Zugriffsschlüssel in Umgebungsvariablen. Wir empfehlen Ihnen, einen Amazon EC2 Systems Manager Parameter Store zu verwenden oderAWS Secrets Managerum sensible Werte zu speichern.
  - FollowBewährte Methoden für die Verwendung von Webhooksum zu beschränken, welche Entitäten einen Build auslösen können und die Buildspec nicht im Projekt selbst speichern, um sicherzustellen, dass Ihre Webhooks so sicher wie möglich sind.
- Ein böswilliger Benutzer kann öffentliche Builds verwenden, um bösartige Artefakte zu verteilen. Wir empfehlen Projektadministratoren, alle Pull-Anfragen zu überprüfen,

um zu überprüfen, ob es sich bei der Pull-Anfrage um eine legitime Änderung handelt. Wir empfehlen Ihnen auch, alle Artefakte mit ihren Prüfsummen zu validieren, um sicherzustellen, dass die richtigen Artefakte heruntergeladen werden.

Arbeiten mit Builds in AWS CodeBuild

Bei einem Build handelt es sich um eine Reihe von Aktionen, die von AWS CodeBuild ausgeführt werden, um basierend auf einem Satz von Eingabeartefakten (beispielsweise einer Sammlung von Java-Klassendateien) Ausgabeartefakte (wie z. B. eine JAR-Datei) zu erstellen.

Die folgenden Regeln gelten, wenn Sie mehrere Builds ausführen:

- Wenn möglich, werden Builds gleichzeitig ausgeführt. Die maximale Anzahl gleichzeitig ausgeführter Builds ist variabel. Weitere Informationen finden Sie unter Kontingente für AWS CodeBuild.
- Wenn für das Build-Projekt ein gleichzeitiges Build-Limit festgelegt wurde, geben Builds einen Fehler zurück, wenn die Anzahl der ausgeführten Builds das gleichzeitige Build-Limit für das Projekt erreicht. Weitere Informationen finden Sie unterGleichzeitiges Build-Limit aktivierenaus.
- Wenn für das Build-Projekt kein gleichzeitiges Build-Limit festgelegt ist, werden Builds in die Warteschlange gestellt, wenn die Anzahl der ausgeführten Builds das gleichzeitige Build-Limit für die Plattform und den Berechnungstyp erreicht. Die maximale Anzahl von Builds in einer Warteschlange ist das Fünffache des Grenzwerts für gleichzeitig ausgeführte Builds. Weitere Informationen finden Sie unter Kontingente für AWS CodeBuild.

  Ein Build in einer Warteschlange, der nach der für die Zeitüberschreitung angegebenen Anzahl von Minuten noch nicht gestartet wurde, wird aus der Warteschlange entfernt. Die Vorgabe für die Zeitüberschreitung beträgt 8 Stunden. Sie können die Zeitüberschreitung der Build-Warteschlange beim Ausführen des Builds auf einen Wert zwischen fünf Minuten und acht Stunden einstellen. Weitere Informationen finden Sie unter Ausführen eines Build in AWS CodeBuild.

  Es ist nicht möglich, vorherzusagen, in welcher Reihenfolge die Builds in der Warteschlange gestartet werden.

> **Note**
>
> Sie können auf den Verlauf eines Builds für ein Jahr zugreifen.

Sie können diese Aufgaben bei der Arbeit mit Builds durchführen:

Themen

Ausführen eines Build in AWS CodeBuild

Sie können dasAWS CodeBuild-KonsoleAWS CLI, oderAWSUm für die Ausführung eines Build in CodeBuild zu verwenden.

Themen

Ausführen eines Build (Konsole)

Um zu verwendenAWS CodePipelineUm für die Ausführung eines Build mit CodeBuild zu verwenden, überspringen Sie diese Schritte und folgen Sie den Anweisungen im BereichVerwenden von CodePipeline mit CodeBuildaus.

1. Öffnen SieAWS CodeBuild-Konsolehttps://console.aws.amazon.com/codesuite/codebuild/homeaus.

2. Wählen Sie im linken Navigationsbereich Build projects aus.
3. Wählen Sie in der Liste der Build-Projekte das Build-Projekt aus.
4. Sie können den Build mit den Standardeinstellungen für Build-Projekte ausführen oder nur Build-Einstellungen für diesen Build überschreiben.

   a. Wenn Sie möchten, dass Build mit den Standard-Build-Projekteinstellungen ausgeführt werden soll, wählen SieBuild startenaus. Der Build beginnt sofort.

   b. Wenn Sie möchten, dass die Standardeinstellungen für Build-Projekte überschreiben möchten, wählen SieBuild mit Overrides startenaus. In derBuild startenkönnen Sie Folgendes überschreiben:

      - Build-Konfiguration
      - Source (Quelle)
      - Überschreiben von Umgebungsvariablen

      Wenn Sie erweiterte Überschreibungen auswählen müssen, wählen SieErweiterte Build-Überschreibungenaus. Auf dieser Seite können Sie Folgendes festlegen:

      - Build-Konfiguration
      - Source (Quelle)
      - Umgebung
      - BuildSpec
      - -Artefakte
      - Protokolle

      Wenn Sie Ihre Auswahl für die Überschreibung getroffen haben, wählen SieBuild startenaus.

Weitere Informationen zu diesem Build finden Sie unter Anzeigen von Build-Details (Konsole).

Ausführen eines Build (AWS CLI)

> **Note**
>
> Um CodePipeline für die Ausführung eines Build mit AWS CodeBuild zu verwenden, überspringen Sie diese Schritte und folgen Sie den Anweisungen im Abschnitt Erstellen einer Pipeline unter Verwendung von CodeBuild (AWS CLI).
> Weitere Informationen zur Verwendung der AWS CLI mit CodeBuild finden Sie unter Befehlszeilenreferenz.

1. Führen Sie den Befehl `start-build` auf eine der folgenden Weisen aus:

   ```
   aws codebuild start-build --project-name <project-name>
   ```

   Sie können diesen Befehl verwenden, wenn Sie ein Build ausführen möchten, dass die neueste Version des Build-Eingabeartifakts und die vorhandenen Einstellungen des Build-Projekts einsetzt.

   ```
   aws codebuild start-build --generate-cli-skeleton
   ```

   Verwenden Sie diesen Befehl, wenn Sie einen Build mit einer früheren Version des Build-Eingabeartefakts verwenden oder die Einstellungen für Build-Ausgabeartefakte, Umgebungsvariablen, Build-Spezifikationen oder Standard-Build-Zeitbeschränkungen überschreiben möchten.

2. Wenn Sie den ausführenstart-buildBefehl mit dem`--project-name`Option, ersetzen`<project-name>`mit dem Namen des Build-Projekts, und fahren Sie dann mit Schritt 6 dieses Verfahrens fort. Informationen zum Abrufen einer Liste von Build-Projekten finden Sie unter Anzeigen einer Liste mit Build-Projektnamen.
3. Wenn Sie das ausführenstart-buildBefehl mit dem`--idempotency-token`Option, ein eindeutiger Bezeichner oder Token, bei dem Groß- und Kleinschreibung beachtet wird, ist im Lieferumfang von`start-build`Anfrage. Das Token ist nach der Anforderung 5 Minuten gültig. Wenn Sie die `start-build`-Anforderung mit dem Token wiederholen, jedoch einen Parameter ändern, gibt CodeBuild einen Fehler wegen des abweichenden Parameters zurück.
4. Wenn Sie den start-build Befehl mit der Option `--generate-cli-skeleton` ausführen, werden Daten im JSON-Format in der Ausgabe angezeigt. Kopieren Sie die Daten in eine Datei

(z. B. *start-build.json*) auf dem lokalen Computer oder auf einer Instance, auf der AWS CLI installiert ist. Ändern Sie die kopierten Daten, damit diese mit dem nachfolgenden Format übereinstimmen und speichern Sie die Ergebnisse:

```
{
  "projectName": "projectName",
  "sourceVersion": "sourceVersion",
  "artifactsOverride": {
    "type": "type",
    "location": "location",
    "path": "path",
    "namespaceType": "namespaceType",
    "name": "artifactsOverride-name",
    "packaging": "packaging"
  },
  "buildspecOverride": "buildspecOverride",
  "cacheOverride": {
    "location": "cacheOverride-location",
    "type": "cacheOverride-type"
  },
  "certificateOverride": "certificateOverride",
  "computeTypeOverride": "computeTypeOverride",
  "environmentTypeOverride": "environmentTypeOverride",
  "environmentVariablesOverride": {
    "name": "environmentVariablesOverride-name",
    "value": "environmentVariablesValue",
    "type": "environmentVariablesOverride-type"
  },
  "gitCloneDepthOverride": "gitCloneDepthOverride",
  "imageOverride": "imageOverride",
  "idempotencyToken": "idempotencyToken",
  "insecureSslOverride": "insecureSslOverride",
  "privilegedModeOverride": "privilegedModeOverride",
  "queuedTimeoutInMinutesOverride": "queuedTimeoutInMinutesOverride",
  "reportBuildStatusOverride": "reportBuildStatusOverride",
  "timeoutInMinutesOverride": "timeoutInMinutesOverride",
  "sourceAuthOverride": "sourceAuthOverride",
  "sourceLocationOverride": "sourceLocationOverride",
  "serviceRoleOverride": "serviceRoleOverride",
  "sourceTypeOverride": "sourceTypeOverride"
}
```

Ersetzen die folgenden Platzhalter:

- *projectName*: Erforderliche Zeichenfolge. Der Name des Build-Projekts, der für diesen Build zu verwenden ist.
- *sourceVersion*: Optionale Zeichenfolge. Eine Version des Quellcodes, der wie folgt zu erstellen ist:
  - Für Amazon S3 die Versions-ID, die der Version der Eingabe-ZIP-Datei entspricht, die Sie erstellen möchten. Wenn die *SourceVersion* nicht angegeben ist, wird die neueste Version verwendet.
  - Die Commit-ID für CodeCommit, die der Version des Quellcodes entspricht, die Sie erstellen möchten. Wenn die *SourceVersion* nicht angegeben ist, wird die Commit-ID von HEAD für die Standard-Branch verwendet. (Sie können zwar keinen Tag-Namen für *SourceVersion* eingeben, aber Sie können die Commit-ID des Tags eingeben.)
  - FürGitHub, die Commit-ID, die Pull-Request-ID, der Branch-Name oder der Tag-Name, der der Version des Quellcodes entspricht, den Sie erstellen möchten. Wenn eine Pull-Anforderungs-ID angegeben ist, muss diese das Format `pr/`*pull-request-ID* verwenden (Beispiel: `pr/25`). Wenn ein Branch-Name angegeben wird, wird die Commit-ID von HEAD verwendet. Wenn die *SourceVersion* nicht angegeben ist, wird die Commit-ID von HEAD für die Standard-Branch verwendet.
  - Für Bitbucket, Commit-ID, Branch-Name oder Tag-Name, die/der der Version des Quellcodes entspricht, die Sie erstellen möchten. Wenn ein Branch-Name angegeben wird, wird die Commit-ID von HEAD verwendet. Wenn die *SourceVersion* nicht angegeben ist, wird die Commit-ID von HEAD für die Standard-Branch verwendet.
- Die folgenden Platzhalter gelten für `artifacts0verride`.
  - *type*: Optional. Die Art des Build-Ausgabeartefakts, der für diesen Build den Build überschreibt, der im Build-Projekt festgelegt ist.
  - *location*: Optional. Der Speicherort des Build-Ausgabeartefakt, der für diesen Build den Build überschreibt, der im Build-Projekt festgelegt ist.
  - *path*: Optional. Der Pfad des Build-Ausgabeartefakts, der für diesen Build den Build überschreibt, der im Build-Projekt festgelegt ist.
  - *namespaceType*: Optional. Der Pfadtyp des Build-Ausgabeartefakts, der für diesen Build den Build überschreibt, der im Build-Projekt festgelegt ist.
  - *name*: Optional. Der Name des Build-Ausgabeartefakts, der für diesen Build den Build überschreibt, der im Build-Projekt festgelegt ist.

- *packaging*: Optional. Die Verpackungsart des Build-Ausgabeartefakts, der für diesen Build den Build überschreibt, der im Build-Projekt festgelegt ist.

- *buildspecOverride*: Optional. Eine Build-Spezifikationsdeklaration, die für diesen Build den Build überschreibt, der im Build-Projekt festgelegt ist. Wenn dieser Wert festgelegt ist, kann es sich entweder um eine Inline-Definition einer Build-Spezifikation, den Pfad zu einer alternativen buildspec-Datei relativ zum Wert der integrierten Umgebungsvariablen `CODEBUILD_SRC_DIR` oder den Pfad zu einem S3-Bucket handeln. Der S3-Bucket muss sich in derselben AWS-Region wie das Build-Projekt befinden. Geben Sie die buildspec-Datei mit ihrem ARN an (z. B. `arn:aws:s3:::`*<my-codebuild-sample2>*`/buildspec.yml`). Wenn der Wert nicht angegeben oder eine leere Zeichenfolge ist, muss der Quellcode eine `buildspec.yml`-Datei im Stammverzeichnis enthalten. Weitere Informationen finden Sie unter Dateiname der Build-Spezifikation und Speicherort.

- Die folgenden Platzhalter gelten für `cacheOverride`.

  - *cacheOverride-location*: Optional. Der Speicherort eines `ProjectCache`-Objekts für diesen Build, der das im Build-Projekt angegebene `ProjectCache`-Objekt überschreibt. `cacheOverride` ist optional und akzeptiert ein `ProjectCache`-Objekt. `location` ist in einem `ProjectCache`-Objekt erforderlich.

  - *cacheOverride-type*: Optional. Der Typ eines `ProjectCache`-Objekts für diesen Build, der das im Build-Projekt angegebene `ProjectCache`-Objekt überschreibt. `cacheOverride` ist optional und akzeptiert ein `ProjectCache`-Objekt. `type` ist in einem `ProjectCache`-Objekt erforderlich.

- *certificateOverride*: Optional. Der Name eines Zertifikats für diesen Build, das das im Build-Projekt angegebene überschreibt.

- *environmentTypeOverride*: Fakultativ. Ein Containertyp für diesen Build, der den im Build-Projekt angegebenen überschreibt. Die aktuell gültige Zeichenfolge ist `LINUX_CONTAINER`.

- Die folgenden Platzhalter gelten für `environmentVariablesOverride`.

  - *environmentVariablesOverride-name*: Fakultativ. Der Name einer Umgebungsvariable in dem Build_Projekt, deren Wert Sie für diesen Build überschreiben möchten.

  - *environmentVariablesOverride-Typ*: Fakultativ. Der Typ der Umgebungsvariablen im Build-Projekt, deren Wert für diesen Build überschrieben werden soll.

  - *environmentVariablesValue*: Fakultativ. Der Wert der Umgebungsvariablen, der im Build-Projekt festgelegt wurde und für diesen Build überschrieben werden soll.

- *gitCloneDepthÜberschreiben*: Fakultativ. Der Wert für Git clone depth, der in dem Build-Projekt festgelegt ist, dessen Wert Sie für diesen Build überschreiben möchten. Wenn Ihr Quelltyp Amazon S3 ist, wird dieser Wert nicht unterstützt.
- *imageOverride*: Optional. Der Name eines Abbilds für diesen Build, das das im Build-Projekt angegebene überschreibt.
- *idempotencyToken*: Optional. Eine Zeichenfolge, die als Token dient und angibt, dass die Build-Anforderung idempotent ist. Sie können eine beliebige Zeichenfolge mit maximal 64 Zeichen verwenden. Das Token ist nach der Start-Build-Anfrage 5 Minuten lang gültig. Wenn Sie die Start-Build-Anfrage mit demselben Token wiederholen, aber einen Parameter ändern,CodeBuildgibt einen Fehler zurück, bei dem die Parameter nicht übereinstimmen.
- *insecureSslOverride*: Optionaler boolescher Wert, der angibt, ob die im Build-Projekt angegebene unsichere TLS-Einstellung überschrieben werden soll. Die unsichere TLS-Einstellung bestimmt, ob TLS-Warnungen ignoriert werden sollen, während die Verbindung zum Projektquellcode hergestellt wird. Diese Überschreibung gilt nur, wenn die Quelle des BuildsGitHubUnternehmensserver.
- *privilegedModeOverride*: Optionaler boolescher Wert. Ist der Wert "true" festgelegt, überschreibt der Build den privilegierten Modus im Build-Projekt.
- *queuedTimeoutInMinutesOverride*: Optionale Ganzzahl, die angibt, wie viele Minuten ein Build in die Warteschlange gestellt werden darf, bevor das Timeout eintritt. Der kleinste Wert beträgt fünf Minuten, der größte Wert beträgt 480 Minuten (8 Stunden).
- *reportBuildStatusÜberschreiben*: Optionaler boolescher Wert, der angibt, ob Ihrem Quellanbieter der Status des Beginns und des Abschlusses eines Builds gesendet werden soll. Wenn Sie dies mit einem anderen Quellanbieter festlegen alsGitHub,GitHubEnterprise Server oder Bitbucket, eininvalidInputExceptionwird geworfen.
- *sourceAuthOverride*: Optionale Zeichenfolge. Ein Autorisierungstyp für diesen Build, der den im Build-Projekt angegebenen überschreibt. Diese Überschreibung gilt nur, wenn die Quelle des Build-Projekts Bitbucket ist oderGitHub.
- *sourceLocationOverride*: Optionale Zeichenfolge. Ein Speicherort, der für diesen Build den Quellspeicherort überschreibt, der im Build-Projekt definiert ist.
- *serviceRoleOverride*: Optionale Zeichenfolge. Der Name einer Servicerolle für diesen Build, die die im Build-Projekt angegebene Servicerolle überschreibt.
- *sourceTypeOverride*: Optionale Zeichenfolge. Ein Quelleingabetyp für diesen Build, der die im Build-Projekt definierte Quelleingabe überschreibt. Gültige Zeichenfolgen sind `NO_SOURCE`, `CODECOMMIT`, `CODEPIPELINE`, `GITHUB`, `S3`, `BITBUCKET` und `GITHUB_ENTERPRISE`.

- *timeoutInMinutesÜberschreiben*: Optionale Zahl. Die Anzahl der Minuten der Build-Zeitbeschränkung, die für diesen Build den Build überschreiben, der im Build-Projekt festgelegt ist.

Wir empfehlen, eine Umgebungsvariable mit einem sensiblen Wert zu speichern, z. B.AWSZugriffsschlüssel-ID, einAWSgeheimer Zugriffsschlüssel oder ein Passwort als Parameter im Amazon EC2 Systems Manager Parameter Store. CodeBuildkann einen im Amazon EC2 Systems Manager Parameter Store gespeicherten Parameter nur verwenden, wenn der Name dieses Parameters mit beginnt`/CodeBuild/`(zum Beispiel`/CodeBuild/dockerLoginPassword`). Sie können die verwendenCodeBuildKonsole, um einen Parameter in Amazon EC2 Systems Manager zu erstellen. Wählen Sie Create a parameter (Parameter erstellen) aus und befolgen Sie dann die Anweisungen. (In diesem Dialogfeld fürKMS-Schlüssel, Sie können optional den ARN eines angebenAWS KMSgeben Sie Ihr Konto ein. Amazon EC2 Systems Manager verwendet diesen Schlüssel, um den Wert des Parameters beim Speichern zu verschlüsseln und beim Abrufen zu entschlüsseln.) Wenn Sie die CodeBuild-Konsole verwenden, um einen Parameter zu erstellen, startet die Konsole den Parameter beim Speichern mit `/CodeBuild/`. Wenn Sie jedoch die Amazon EC2 Systems Manager Parameter Store-Konsole verwenden, um einen Parameter zu erstellen, müssen Sie den Namen des Parameters mit beginnen`/CodeBuild/`, und Sie müssen Folgendes festlegenTypzuSichere Zeichenfolge. Weitere Informationen finden Sie unterAWS Systems ManagerParameter speichernundExemplarische Vorgehensweise: Erstellen und Testen eines String-Parameters (Konsole)in derAmazon EC2 Systems Manager-Benutzerhandbuch.

Wenn Ihr Build-Projekt auf Parameter Store von Amazon EC2 Systems Manager gespeicherte Parameter verweist, muss die Service-Rolle des Build-Projekts Folgendes zulassen`ssm:GetParameters`Aktion. Wenn Sie zuvor Create a new service role in your account (Neue Servicerolle in Ihrem Konto erstellen) ausgewählt haben, dann nimmt CodeBuild diese Aktion automatisch in die Standard-Servicerolle für Ihr Build-Projekt auf. Wenn Sie jedoch Choose an existing service role from your account ausgewählt haben, müssen Sie diese Aktion separat in Ihre Service-Rolle aufnehmen.

Von Ihnen gesetzte Umgebungsvariablen ersetzen vorhandene Umgebungsvariablen. Wenn das Docker-Image beispielsweise bereits eine Umgebungsvariable mit dem Namen `MY_VAR` und einem Wert von `my_value` enthält und Sie eine Umgebungsvariable mit dem Namen `MY_VAR` und einem Wert von `other_value` festlegen, wird `my_value` durch `other_value` ersetzt. Wenn das Docker-Image demgegenüber bereits eine Umgebungsvariable mit dem Namen `PATH` und einem Wert von `/usr/local/sbin:/usr/local/bin` enthält und Sie eine

Umgebungsvariable mit dem Namen `PATH` und einem Wert von `$PATH:/usr/share/ant/bin` festlegen, wird `/usr/local/sbin:/usr/local/bin` durch den Literalwert `$PATH:/usr/share/ant/bin` ersetzt.

Legen Sie keine Umgebungsvariable mit einem Namen fest, der mit `CODEBUILD_` beginnt. Dieses Präfix ist zur -internen Verwendung reserviert.

Wenn eine Umgebungsvariable mit identischem Namen an mehreren Orten definiert ist, wird der Wert der Umgebungsvariable folgendermaßen bestimmt:

- Der Wert im Aufruf zum Starten des Build-Vorgangs hat den höchsten Vorrang.
- Der Wert in der Build-Projektdefinition folgt darauf.
- Der Wert in der Deklaration in der buildspec-Datei hat die niedrigste Priorität.

Weitere Informationen zu gültigen Werten für diese Platzhalter finden Sie unter Erstellen eines Build-Projekts (AWS CLI). Eine Liste der neuesten Einstellungen für ein Build-Projekt finden Sie unter Anzeigen der Details eines Build-Projekts.

5. Wechseln Sie in das Verzeichnis, das die soeben gespeicherte Datei enthält, und führen Sie den Befehl `start-build` erneut aus.

```
aws codebuild start-build --cli-input-json file://start-build.json
```

6. Bei Erfolg enthält die Ausgabe Daten wie in der So führen Sie den Build aus-Anleitung beschrieben.

Um mit den detaillierten Informationen über diesen Build zu arbeiten, notieren Sie sich den Wert `id` im der Ausgabe und zeigen Sie sich dann Anzeigen von Build-Details (AWS CLI) an.

Ausführen eines Stapel-Build (AWS CLI)

1. Führen Sie den Befehl `start-build-batch` auf eine der folgenden Weisen aus:

```
aws codebuild start-build-batch --project-name <project-name>
```

Sie können diesen Befehl verwenden, wenn Sie ein Build ausführen möchten, dass die neueste Version des Build-Eingabeartifakts und die vorhandenen Einstellungen des Build-Projekts einsetzt.

```
aws codebuild start-build-batch --generate-cli-skeleton > <json-file>
```

Verwenden Sie diesen Befehl, wenn Sie einen Build mit einer früheren Version des Build-Eingabeartefakts verwenden oder die Einstellungen für Build-Ausgabeartefakte, Umgebungsvariablen, Build-Spezifikationen oder Standard-Build-Zeitbeschränkungen überschreiben möchten.

2. Wenn Sie diestart-build-batchBefehl mit dem`--project-name`Option, ersetzen*<project-name>*mit dem Namen des Build-Projekts, und dann springen Sie weiter zu Schritt 6 dieses Verfahrens. Informationen zum Abrufen einer Liste von Build-Projekten finden Sie unter Anzeigen einer Liste mit Build-Projektnamen.
3. Wenn Sie diestart-build-batchBefehl mit dem`--idempotency-token`-Option ist ein eindeutiger Bezeichner, in dem die Groß- und Kleinschreibung berücksichtigt wird (auch als Token`start-build-batch`request. Das Token ist nach der -Anforderung 5 Minuten gültig. Wenn du das wiederholst`start-build-batch`-Anforderung mit dem Token, jedoch einen Parameter ändern, gibt CodeBuild einen Fehler wegen des abweichenden Parameters zurück.
4. Wenn Sie diestart-build-batchBefehl mit dem`--generate-cli-skeleton`Option werden JSON-formatierte Daten an die*<json-file>*file. Diese Datei ist mit dem Skelton vergleichbar, das von derstart-buildBefehl, mit dem Hinzufügen des folgenden Objekts. Weitere Informationen zu den gemeinsamen Objekten finden Sie unterAusführen eines Build (AWS CLI)aus.

   Ändern Sie diese Datei, um Build-Overrides hinzuzufügen, und speichern Sie die Ergebnisse.

   ```
   "buildBatchConfigOverride": {
     "combineArtifacts": combineArtifacts,
     "restrictions": {
       "computeTypesAllowed": [
         allowedComputeTypes
       ],
       "maximumBuildsAllowed": maximumBuildsAllowed
     },
     "serviceRole": "batchServiceRole",
     "timeoutInMins": batchTimeout
   }
   ```

   Die`buildBatchConfigOverride`-Objekt ist einProjectBuildBatchConfig-Struktur, die die Batch-Build-Konfigurations-Overides für diesen Build enthält.

CombineArtifacts

Ein boolescher Wert, der angibt, ob die Build-Artefakte für den Stapel-Build zu einem einzigen Artefakt-Speicherort kombiniert werden sollen.

AllowedComputeTypes

Ein Array von Zeichenfolgen, die die Datenverarbeitungstypen angeben, die für den Stapel-Build zulässig sind. Siehe .Arten der Datenverarbeitung bei der Build-Umgebungfür diese Werte.

MaximumBuildsAllowed

Gibt die maximal zulässige Anzahl von Builds an.

BatchServiceRole

Gibt die Servicerollen-ARN für das Stapel-Build-Projekt an.

BatchTimeout

Gibt die maximale Zeitspanne in Minuten an, in der die Stapelerstellung abgeschlossen werden muss.

5. Wechseln Sie in das Verzeichnis, das die soeben gespeicherte Datei enthält, und führen Sie den Befehl `start-build-batch` erneut aus.

```
aws codebuild start-build-batch --cli-input-json file://start-build.json
```

6. Bei Erfolg wird die JSON-Darstellung einesBuildBatch-Objekt wird in der Konsolenausgabe angezeigt. Sehen Sie dieStartBuildBatch-Antwortsyntaxfür ein Beispiel für diese Daten.

Ausführung von Builds automatisch starten (AWS CLI)

Wenn Ihr Quellcode in einem GitHub- oder GitHub Enterprise Server-Repository gespeichert ist, können Sie GitHub-Webhooks verwenden, damit AWS CodeBuild Ihren Quellcode neu erstellt, sobald eine Codeänderung an das Repository übergeben wird.

Führen Sie den Befehl create-webhook wie folgt aus:

```
aws codebuild create-webhook --project-name <project-name>
```

*<project-name>*ist der Name des Build-Projekts, das den neu zu erstellenden Quellcode enthält.

Für GitHub werden Informationen wie die Folgenden in der Ausgabe angezeigt:

```
{
  "webhook": {
    "url": "<url>"
  }
}
```

*<url>*ist die URL zum GitHub-Webhook.

Für GitHub Enterprise Server werden Informationen wie die Folgenden in der Ausgabe angezeigt:

1. Kopieren Sie den geheimen Schlüssel und Nutzlast-URL aus der Ausgabe. Sie brauchen sie, um einen Webhook in GitHub Enterprise Server hinzuzufügen.
2. Wählen Sie in GitHub Enterprise Server das Repository aus, in dem Ihr CodeBuild--Projekt gespeichert ist. Wählen Sie Settings (Einstellungen), Hooks & services (Hooks und Services) und anschließend Add webhook (Webhook hinzufügen) aus.
3. Geben Sie die Nutzlast-URL und den geheimen Schlüssels ein, übernehmen Sie die Standardeinstellungen für die anderen Felder, und wählen Sie dann Add webhook.

Ausführung von Builds automatisch beenden (AWS CLI)

Wenn Ihr Quellcode in einem GitHub- oder GitHub Enterprise Server-Repository gespeichert ist, können Sie GitHub-Webhooks einrichten, damit AWS CodeBuild Ihren Quellcode neu erstellt, sobald eine Codeänderung an das Repository übergeben wird. Weitere Informationen finden Sie unter Ausführung von Builds automatisch starten (AWS CLI).

Wenn Sie dieses Verhalten aktiviert haben, können Sie es ausschalten, indem Sie den `delete-webhook`-Befehl wie folgt ausführen:

```
aws codebuild delete-webhook --project-name <project-name>
```

- woher*<project-name>*ist der Name des Build-Projekts, das den neu zu erstellenden Quellcode enthält.

Ist dieser Befehl erfolgreich, werden in der Ausgabe keine Informationen und Fehler angezeigt.

> **Note**
>
> Dies löscht den Webhook nur aus Ihrem CodeBuild-Projekt. Sie sollten den Webhook auch aus Ihrem GitHub- oder GitHub Enterprise Server-Repository löschen.

Ausführen eines Build (AWS-SDKs)

So verwenden Sie CodePipeline zum Ausführen eines Builds mitAWS CodeBuild, überspringen Sie diese Schritte und folgen Sie den Anweisungen im AbschnittVerwenden von AWS CodePipeline mit AWS CodeBuild zum Testen von Codes und zum Ausführen von BuildsStattdessen geschieht dies.

Weitere Informationen zur Verwendung von CodeBuild mit demAWSDie SDKs finden Sie im AbschnittAWS SDKs- und Tools-Referenzaus.

Anzeigen von Build-Details in AWS CodeBuild

Sie können dasAWS CodeBuildconsole,AWS CLI, oderAWS-SDKs, um Details über Builds, die von CodeBuild verwaltet werden, anzuzeigen.

Themen

- Anzeigen von Build-Details (Konsole)
- Anzeigen von Build-Details (AWS CLI)
- Anzeigen von Build-Details (AWS-SDKs)
- Übergang von Build-Phasen

Anzeigen von Build-Details (Konsole)

1. Öffnen SieAWS CodeBuildconsolehttps://console.aws.amazon.com/codesuite/codebuild/homeaus.

2. Gehen Sie folgendermaßen vor:

   - Wählen Sie im Navigationsbereich Build history aus. Wählen Sie in der Liste der Builds in der Spalte Build run (Build-Ausführung) den Link für den gewünschten Build aus.
   - Wählen Sie im linken Navigationsbereich Build projects aus. Wählen Sie in der Liste der Build-Projekte in der Spalte Name den Link mit dem Namen für das gewünschte Build-Projekt aus. Wählen Sie dann in der Liste der Builds in der Spalte Build run (Build-Ausführung) den Link für den Build aus.

> **Note**
>
> In der Standardeinstellung werden nur die letzten 10 Builds oder Build-Projekte angezeigt. Zur Anzeige von weiteren Builds oder Build-Projekten wählen Sie das Zahnradsymbol und einen anderen Wert für Builds per page (Builds je Seite) oder Projects per page (Projekte je Seite) aus oder verwenden Sie die Vorwärts- und Rückwärtspfeile.

Anzeigen von Build-Details (AWS CLI)

Weitere Informationen zur Verwendung der AWS CLI mit AWS CodeBuild finden Sie unter Befehlszeilenreferenz.

Führen Sie den Befehl batch-get-builds aus:

```
aws codebuild batch-get-builds --ids ids
```

Ersetzen die folgenden Platzhalter:

- *IDs*: Erforderliche Zeichenfolge. Ein oder mehrere IDs, deren Details angezeigt werden können. Um mehr als ein Build-ID anzugeben, fügen Sie zwischen den einzelnen Build-IDs ein Leerzeichen ein. Sie können bis zu 100 Build-IDs angeben. Weitere Informationen zum Abrufen einer Liste mit Build-IDs finden Sie in den folgenden Themen:
  - Anzeigen einer Liste mit Build-IDs (AWS CLI)
  - Anzeigen einer Liste mit Build-IDs für ein Build-Projekt (AWS CLI)

Wenn Sie z. B. den folgenden Befehl ausführen:

```
aws codebuild batch-get-builds --ids codebuild-demo-project:e9c4f4df-3f43-41d2-
ab3a-60fe2EXAMPLE codebuild-demo-project:815e755f-bade-4a7e-80f0-efe51EXAMPLE my-other-
project:813bb6c6-891b-426a-9dd7-6d8a3EXAMPLE
```

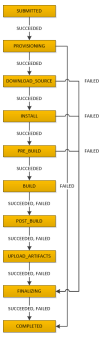
Kann der Befehl erfolgreich ausgeführt werden, werden Daten wie die in So zeigen Sie eine Zusammenfassung der Build-Informationen an beschriebenen in der Ausgabe angezeigt.

Anzeigen von Build-Details (AWS-SDKs)

Weitere Informationen zur Verwendung von AWS CodeBuild mit den AWS-SDKs finden Sie unter AWS SDKs- und Tools-Referenz.

Übergang von Build-Phasen

Builds in AWS CodeBuild setzen sich in Phasen fort:



> **Important**
>
> Die `UPLOAD_ARTIFACTS`-Phase wird immer angestrebt, auch wenn die `BUILD`-Phase fehlschlägt.

Anzeigen einer Liste mit Build-IDs in AWS CodeBuild

Sie können dasAWS CodeBuild-KonsoleAWS CLI, oderAWS-SDKs zum Anzeigen einer Liste mit Build-IDs für Builds, die von CodeBuild verwaltet werden.

Themen

- Anzeigen einer Liste von Build-IDs (Konsole)

- Anzeigen einer Liste mit Build-IDs (AWS CLI)
- Anzeigen einer Liste mit Batch-Build-IDs (AWS CLI)
- Anzeigen einer Liste mit Build-IDs (AWS SDKs)

Anzeigen einer Liste von Build-IDs (Konsole)

1. Öffnen SieAWS CodeBuildconsole beihttps://console.aws.amazon.com/codesuite/codebuild/homeaus.
2. Wählen Sie im Navigationsbereich Build history aus.

> **Note**
>
> In der Standardeinstellung werden nur die letzten 10 Builds angezeigt. Zur Anzeige von weiteren Builds wählen Sie das Zahnradsymbol und einen anderen Wert für Builds per page (Builds je Seite) aus oder verwenden die Vorwärts- und Rückwärtspfeile.

Anzeigen einer Liste mit Build-IDs (AWS CLI)

Weitere Informationen zur Verwendung der finden Sie unter.AWS CLImit CodeBuild finden Sie imBefehlszeilenreferenzaus.

- Führen Sie den Befehl list-builds aus:

```
aws codebuild list-builds --sort-order sort-order --next-token next-token
```

Ersetzen Sie im Befehl oben die folgenden Platzhalter:

- *`sort-order`*: Optionale Zeichenfolge, die angibt, wie die Build-IDs aufgelistet werden. Gültige Werte sind `ASCENDING` und `DESCENDING`.
- *`nächstes Token`*: Optionale Zeichenfolge. Falls während einer vorherigen Ausführung mehr als 100 Elemente in der Liste enthalten waren, werden nur die ersten 100 Elemente zurückgegeben, zusammen mit einer eindeutigen Zeichenfolge namens next token. Führen Sie diesen Befehl erneut aus, um das nächste Token hinzuzufügen und den nächsten Stapel von Listenelementen abzurufen. Um alle Elemente in der Liste abzurufen, führen Sie diesen Befehl mit jedem nachfolgenden "next token" aus, bis keine weiteren nächsten Token zurückgegeben werden.

Wenn Sie z. B. den folgenden Befehl ausführen:

```
aws codebuild list-builds --sort-order ASCENDING
```

Sollte die Ausgabe folgendermaßen oder ähnlich aussehen:

```
{
  "nextToken": "4AEA6u7J...The full token has been omitted for brevity...MzY2OA==",
  "ids": [
    "codebuild-demo-project:815e755f-bade-4a7e-80f0-efe51EXAMPLE"
    "codebuild-demo-project:84a7f3d1-d40e-4956-b4cf-7a9d4EXAMPLE"
    ... The full list of build IDs has been omitted for brevity ...
    "codebuild-demo-project:931d0b72-bf6f-4040-a472-5c707EXAMPLE"
  ]
}
```

Wenn Sie diesen Befehl erneut ausführen:

```
aws codebuild list-builds --sort-order ASCENDING --next-token 4AEA6u7J...The full
 token has been omitted for brevity...MzY2OA==
```

Sollte die Ausgabe folgendermaßen oder ähnlich aussehen:

```
{
  "ids": [
    "codebuild-demo-project:49015049-21cf-4b50-9708-df115EXAMPLE",
    "codebuild-demo-project:543e7206-68a3-46d6-a4da-759abEXAMPLE",
    ... The full list of build IDs has been omitted for brevity ...
    "codebuild-demo-project:c282f198-4582-4b38-bdc0-26f96EXAMPLE"
  ]
}
```

Anzeigen einer Liste mit Batch-Build-IDs (AWS CLI)

Weitere Informationen zur Verwendung der finden Sie unter.AWS CLImit CodeBuild finden Sie imBefehlszeilenreferenzaus.

- Führen Sie den Befehl list-build-batches aus:

```
aws codebuild list-build-batches --sort-order sort-order --next-token next-token
```

Ersetzen Sie im Befehl oben die folgenden Platzhalter:

- *sort-order*: Optionale Zeichenfolge, die angibt, wie die Batch-Build-IDs aufgelistet werden. Gültige Werte sind `ASCENDING` und `DESCENDING`.
- *nächstes Token*: Optionale Zeichenfolge. Falls während einer vorherigen Ausführung mehr als 100 Elemente in der Liste enthalten waren, werden nur die ersten 100 Elemente zurückgegeben, zusammen mit einer eindeutigen Zeichenfolge namens next token. Führen Sie diesen Befehl erneut aus, um das nächste Token hinzuzufügen und den nächsten Stapel von Listenelementen abzurufen. Um alle Elemente in der Liste abzurufen, führen Sie diesen Befehl mit jedem nachfolgenden "next token" aus, bis keine weiteren nächsten Token zurückgegeben werden.

Wenn Sie z. B. den folgenden Befehl ausführen:

```
aws codebuild list-build-batches --sort-order ASCENDING
```

Sollte die Ausgabe folgendermaßen oder ähnlich aussehen:

```
{
  "nextToken": "4AEA6u7J...The full token has been omitted for brevity...MzY2OA==",
  "ids": [
    "codebuild-demo-project:815e755f-bade-4a7e-80f0-efe51EXAMPLE"
    "codebuild-demo-project:84a7f3d1-d40e-4956-b4cf-7a9d4EXAMPLE"
    ... The full list of build IDs has been omitted for brevity ...
    "codebuild-demo-project:931d0b72-bf6f-4040-a472-5c707EXAMPLE"
  ]
}
```

Wenn Sie diesen Befehl erneut ausführen:

```
aws codebuild list-build-batches --sort-order ASCENDING --next-token 4AEA6u7J...The
 full token has been omitted for brevity...MzY2OA==
```

Sollte die Ausgabe folgendermaßen oder ähnlich aussehen:

```
{
  "ids": [
    "codebuild-demo-project:49015049-21cf-4b50-9708-df115EXAMPLE",
    "codebuild-demo-project:543e7206-68a3-46d6-a4da-759abEXAMPLE",
    ... The full list of build IDs has been omitted for brevity ...
    "codebuild-demo-project:c282f198-4582-4b38-bdc0-26f96EXAMPLE"
  ]
}
```

Anzeigen einer Liste mit Build-IDs (AWS SDKs)

Weitere Informationen zur Verwendung von CodeBuild mit der finden Sie unter.AWS-SDKs finden Sie unter.AWS SDKs- und Tools-Referenzaus.

Anzeigen einer Liste mit Build-IDs für ein Build-Projekt in AWS CodeBuild

Sie können dasAWS CodeBuild-KonsoleAWS CLI, oderAWS-SDKs zum Anzeigen einer Liste mit Build-IDs für ein Build-Projekt in CodeBuild.

Themen

- Anzeigen einer Liste mit Build-IDs für ein Build-Projekt (Konsole)
- Anzeigen einer Liste mit Build-IDs für ein Build-Projekt (AWS CLI)
- Anzeigen einer Liste mit Batch-Build-IDs für ein Build-Projekt (AWS CLI)
- Anzeigen einer Liste mit Build-IDs für ein Build-Projekt (AWS-SDKs)

Anzeigen einer Liste mit Build-IDs für ein Build-Projekt (Konsole)

1. Öffnen Sie die CodeBuild-Konsole unter https://console.aws.amazon.com/codebuild/.
2. Wählen Sie im linken Navigationsbereich Build projects aus. Wählen Sie in der Liste der Build-Projekte in der Spalte Name das gewünschte Build-Projekt aus.

> **Note**
>
> In der Standardeinstellung werden nur die letzten 100 Builds oder Build-Projekte angezeigt. Zur Anzeige von weiteren Builds oder Build-Projekten wählen Sie das Zahnradsymbol und

einen anderen Wert für Builds per page (Builds je Seite) oder Projects per page (Projekte je Seite) aus oder verwenden Sie die Vorwärts- und Rückwärtspfeile.

Anzeigen einer Liste mit Build-IDs für ein Build-Projekt (AWS CLI)

Weitere Informationen zur Verwendung der AWS CLI mit AWS CodeBuild finden Sie unter Befehlszeilenreferenz.

Führen Sie den Befehl list-builds-for-project aus, wie folgt:

```
aws codebuild list-builds-for-project --project-name project-name --sort-order sort-order --next-token next-token
```

Ersetzen Sie im Befehl oben die folgenden Platzhalter:

- *Projektname (Projektname)*: Erforderliche Zeichenfolge zur Angabe des Namens des Build-Projekts, für das Build-IDs aufgelistet werden. Informationen zum Abrufen einer Liste von Build-Projekten finden Sie unter Anzeigen einer Liste mit Build-Projektnamen (AWS CLI).
- *Sort-Reihenfolge*: Optionale Zeichenfolge, die angibt, wie die Build-IDs aufgelistet werden. Gültige Werte sind `ASCENDING` und `DESCENDING`.
- *nächstes Token*: Optionale Zeichenfolge. Falls während einer vorherigen Ausführung mehr als 100 Elemente in der Liste enthalten waren, werden nur die ersten 100 Elemente zurückgegeben, zusammen mit einer eindeutigen Zeichenfolge namens next token. Führen Sie diesen Befehl erneut aus, um das nächste Token hinzuzufügen und den nächsten Stapel von Listenelementen abzurufen. Um alle Elemente in der Liste abzurufen, führen Sie diesen Befehl mit jedem nachfolgend zurückgegebenen "next token" aus, bis keine weiteren nächsten Token zurückgegeben werden.

Wenn Sie beispielsweise folgenden Befehl ausführen:

```
aws codebuild list-builds-for-project --project-name codebuild-demo-project --sort-order ASCENDING
```

Sollte die Ausgabe etwa wie folgt aussehen:

```
{
```

```
  "nextToken": "4AEA6u7J...The full token has been omitted for brevity...MzY2OA==",
  "ids": [
    "codebuild-demo-project:9b175d16-66fd-4e71-93a0-50a08EXAMPLE"
    "codebuild-demo-project:a9d1bd09-18a2-456b-8a36-7d65aEXAMPLE"
    ... The full list of build IDs has been omitted for brevity ...
    "codebuild-demo-project:fe70d102-c04f-421a-9cfa-2dc15EXAMPLE"
  ]
}
```

Wenn Sie diesen Befehl erneut ausführen:

```
aws codebuild list-builds-for-project --project-name codebuild-demo-project --
sort-order ASCENDING --next-token 4AEA6u7J...The full token has been omitted for
 brevity...MzY2OA==
```

Die Ausgabe sollte etwa wie folgt aussehen:

```
{
  "ids": [
    "codebuild-demo-project:98253670-7a8a-4546-b908-dc890EXAMPLE"
    "codebuild-demo-project:ad5405b2-1ab3-44df-ae2d-fba84EXAMPLE"
    ... The full list of build IDs has been omitted for brevity ...
    "codebuild-demo-project:f721a282-380f-4b08-850a-e0ac1EXAMPLE"
  ]
}
```

Anzeigen einer Liste mit Batch-Build-IDs für ein Build-Projekt (AWS CLI)

Weitere Informationen zur Verwendung der AWS CLI mit AWS CodeBuild finden Sie unter Befehlszeilenreferenz.

Führen Sie den Befehl list-build-batches-for-project aus, wie folgt:

```
aws codebuild list-build-batches-for-project --project-name project-name --sort-
order sort-order --next-token next-token
```

Ersetzen Sie im Befehl oben die folgenden Platzhalter:

- *Projektname (Projektname)*: Erforderliche Zeichenfolge zur Angabe des Namens des Build-Projekts, für das Build-IDs aufgelistet werden. Informationen zum Abrufen einer Liste von Build-Projekten finden Sie unter Anzeigen einer Liste mit Build-Projektnamen (AWS CLI).

- *Sort-Reihenfolge*: Optionale Zeichenfolge, die angibt, wie die Build-IDs aufgelistet werden. Gültige Werte sind ASCENDING und DESCENDING.
- *nächstes Token*: Optionale Zeichenfolge. Falls während einer vorherigen Ausführung mehr als 100 Elemente in der Liste enthalten waren, werden nur die ersten 100 Elemente zurückgegeben, zusammen mit einer eindeutigen Zeichenfolge namens next token. Führen Sie diesen Befehl erneut aus, um das nächste Token hinzuzufügen und den nächsten Stapel von Listenelementen abzurufen. Um alle Elemente in der Liste abzurufen, führen Sie diesen Befehl mit jedem nachfolgend zurückgegebenen "next token" aus, bis keine weiteren nächsten Token zurückgegeben werden.

Wenn Sie beispielsweise folgenden Befehl ausführen:

```
aws codebuild list-build-batches-for-project --project-name codebuild-demo-project --
sort-order ASCENDING
```

Sollte die Ausgabe etwa wie folgt aussehen:

```
{
  "nextToken": "4AEA6u7J...The full token has been omitted for brevity...MzY2OA==",
  "ids": [
    "codebuild-demo-project:9b175d16-66fd-4e71-93a0-50a08EXAMPLE"
    "codebuild-demo-project:a9d1bd09-18a2-456b-8a36-7d65aEXAMPLE"
    ... The full list of build IDs has been omitted for brevity ...
    "codebuild-demo-project:fe70d102-c04f-421a-9cfa-2dc15EXAMPLE"
  ]
}
```

Wenn Sie diesen Befehl erneut ausführen:

```
aws codebuild list-build-batches-for-project --project-name codebuild-demo-project
 --sort-order ASCENDING --next-token 4AEA6u7J...The full token has been omitted for
 brevity...MzY2OA==
```

Die Ausgabe sollte etwa wie folgt aussehen:

```
{
  "ids": [
    "codebuild-demo-project:98253670-7a8a-4546-b908-dc890EXAMPLE"
```

```
    "codebuild-demo-project:ad5405b2-1ab3-44df-ae2d-fba84EXAMPLE"
    ... The full list of build IDs has been omitted for brevity ...
    "codebuild-demo-project:f721a282-380f-4b08-850a-e0ac1EXAMPLE"
  ]
}
```

Anzeigen einer Liste mit Build-IDs für ein Build-Projekt (AWS-SDKs)

Weitere Informationen zur Verwendung von AWS CodeBuild mit den AWS-SDKs finden Sie unter AWS SDKs- und Tools-Referenz.

Stoppen eines Builds in AWS CodeBuild

Zum Stoppen eines Builds in AWS CodeBuild können Sie die AWS CodeBuild-Konsole, die AWS CLI oder AWS-SDKs verwenden.

Themen

- Stoppen eines Builds (Konsole)
- Stoppen eines Builds (AWS CLI)
- Stoppen eines Builds (AWS-SDKs)

Stoppen eines Builds (Konsole)

1. Öffnen SieAWS CodeBuild-Konsolehttps://console.aws.amazon.com/codesuite/codebuild/homeaus.
2. Gehen Sie folgendermaßen vor:
   - Wenn die Seite ***build-project-name*:*build-ID*** angezeigt wird, wählen Sie Stop build (Build stoppen).
   - Wählen Sie im Navigationsbereich Build history aus. Wählen Sie in der Liste der Builds das Kontrollkästchen für den Build aus und klicken Sie dann auf Stop build (Build stoppen).
   - Wählen Sie im linken Navigationsbereich Build projects aus. Wählen Sie in der Liste der Build-Projekte in der Spalte Name den Link mit dem Namen für das gewünschte Build-Projekt aus. Wählen Sie in der Liste der Builds das Kontrollkästchen für den Build aus und klicken Sie dann auf Stop build (Build stoppen).

> **Note**
>
> In der Standardeinstellung werden nur die letzten 100 Builds oder Build-Projekte angezeigt. Zur Anzeige von weiteren Builds oder Build-Projekten wählen Sie das Zahnradsymbol und einen anderen Wert für Builds per page (Builds je Seite) oder Projects per page (Projekte je Seite) aus oder verwenden Sie die Vorwärts- und Rückwärtspfeile.
> Wenn AWS CodeBuild einen Build nicht stoppen kann (wenn der Build-Prozess beispielsweise bereits abgeschlossen ist), ist die Schaltfläche Stop (Stopp) deaktiviert oder wird nicht angezeigt.

Stoppen eines Builds (AWS CLI)

- Führen Sie den Befehl stop-build aus:

  ```
  aws codebuild stop-build --id id
  ```

  Ersetzen Sie im Befehl oben den folgenden Platzhalter:

  - *id*: Erforderliche Zeichenfolge. Die ID des zu stoppenden Builds. Weitere Informationen zum Abrufen einer Liste mit Build-IDs finden Sie in den folgenden Themen:
    - Anzeigen einer Liste mit Build-IDs (AWS CLI)
    - Anzeigen einer Liste mit Build-IDs für ein Build-Projekt (AWS CLI)

  Falls AWS CodeBuild den Build erfolgreich stoppt, lautet der Wert für `buildStatus` im Objekt `build` in der Ausgabe `STOPPED`.

  Wenn CodeBuild den Build nicht stoppen kann (wenn der Build-Vorgang beispielsweise bereits abgeschlossen ist),`buildStatus`-Wert im`build`Objekt in der Ausgabe ist der endgültige Build-Status (z. B.`SUCCEEDED`) enthalten.

Stoppen eines Builds (AWS-SDKs)

Weitere Informationen zur Verwendung von AWS CodeBuild mit den AWS-SDKs finden Sie unter AWS SDKs- und Tools-Referenz.

Stoppen eines StapelsAWS CodeBuild

Sie können dasAWS CodeBuildconsole,AWS CLIoderAWS-SDKs zum Stoppen eines StapelaufbausAWS CodeBuildaus.

Themen

- Stoppen eines Stapelbuilds (Konsole)
- Stoppen eines Stapelbuilds (AWS CLI)
- Stoppen eines Stapelbuilds (AWSSDKs)

Stoppen eines Stapelbuilds (Konsole)

1. Öffnen SieAWS CodeBuildconsole beihttps://console.aws.amazon.com/codesuite/codebuild/homeaus.
2. Gehen Sie folgendermaßen vor:

   - Wenn die Seite ***build-project-name*:*build-ID*** angezeigt wird, wählen Sie Stop build (Build stoppen).
   - Wählen Sie im Navigationsbereich Build history aus. Wählen Sie in der Liste der Builds das Kontrollkästchen für den Build aus und klicken Sie dann auf Stop build (Build stoppen).
   - Wählen Sie im linken Navigationsbereich Build projects aus. Wählen Sie in der Liste der Build-Projekte in der Spalte Name den Link mit dem Namen für das gewünschte Build-Projekt aus. Wählen Sie in der Liste der Builds das Kontrollkästchen für den Build aus und klicken Sie dann auf Stop build (Build stoppen).

> **Note**
>
> In der Standardeinstellung werden nur die letzten 100 Builds oder Build-Projekte angezeigt. Zur Anzeige von weiteren Builds oder Build-Projekten wählen Sie das Zahnradsymbol und einen anderen Wert für Builds per page (Builds je Seite) oder Projects per page (Projekte je Seite) aus oder verwenden Sie die Vorwärts- und Rückwärtspfeile.
> WennAWS CodeBuildWenn einen Stapel-Build nicht stoppen kann (wenn der Build-Prozess beispielsweise bereits abgeschlossen ist),Stoppen-Schaltfläche ist deaktiviert.

Stoppen eines Stapelbuilds (AWS CLI)

- Führen Sie den Befehl `stop-build-batch` aus:

  ```
  aws codebuild stop-build-batch --id <batch-build-id>
  ```

  Ersetzen Sie im Befehl oben den folgenden Platzhalter:

  - *<batch-build-id>*: Erforderliche Zeichenfolge. Die Kennung des zu stoppenden Stapel-Builds. Weitere Informationen zum Abrufen einer Liste mit Stapel-Build-IDs finden Sie in den folgenden Themen:
    - Anzeigen einer Liste mit Batch-Build-IDs (AWS CLI)
    - Anzeigen einer Liste mit Batch-Build-IDs für ein Build-Projekt (AWS CLI)

  WennAWS CodeBuildstoppt erfolgreich den Batch-Build, der`buildBatchStatus`wert im`buildBatch`Objekt in der Ausgabe ist`STOPPED`aus.

  Wenn CodeBuild den Stapel-Build nicht stoppen kann (wenn der Stapel-Build beispielsweise bereits abgeschlossen ist), wird die`buildBatchStatus`wert im`buildBatch`Objekt in der Ausgabe ist der endgültige Build-Status (z. B.`SUCCEEDED`) enthalten.

Stoppen eines Stapelbuilds (AWSSDKs)

Weitere Informationen zur Verwendung von AWS CodeBuild mit den AWS-SDKs finden Sie unter AWS SDKs- und Tools-Referenz.

Wiederholen eines Builds inAWS CodeBuild

Sie können dasAWS CodeBuild-KonsoleAWS CLI, oderAWSSDKs zum Wiederholen eines einzelnen Builds oder eines Batch-Build-InsAWS CodeBuildaus.

Themen

- Wiederholen eines Build (Konsole)
- Wiederholen eines Build (AWS CLI)
- Wiederholen eines Build (AWSSDKs)

Wiederholen eines Build (Konsole)

1. Öffnen SieAWS CodeBuildconsole beihttps://console.aws.amazon.com/codesuite/codebuild/homeaus.
2. Gehen Sie folgendermaßen vor:
   - Wenn das Symbol***build-projekt-name*:*Build-ID***Seite wird angezeigt, wählen Sie ausWiederholen Buildaus.
   - Wählen Sie im Navigationsbereich Build history aus. Wählen Sie in der Liste der Builds das Kontrollkästchen für den Build aus und klicken Sie dann aufWiederholen Buildaus.
   - Wählen Sie im linken Navigationsbereich Build projects aus. Wählen Sie in der Liste der Build-Projekte in der Spalte Name den Link mit dem Namen für das gewünschte Build-Projekt aus. Wählen Sie in der Liste der Builds das Kontrollkästchen für den Build aus und klicken Sie dann aufWiederholen Buildaus.

> **Note**
>
> In der Standardeinstellung werden nur die letzten 100 Builds oder Build-Projekte angezeigt. Zur Anzeige von weiteren Builds oder Build-Projekten wählen Sie das Zahnradsymbol und einen anderen Wert für Builds per page (Builds je Seite) oder Projects per page (Projekte je Seite) aus oder verwenden Sie die Vorwärts- und Rückwärtspfeile.

Wiederholen eines Build (AWS CLI)

- Führen Sie den Befehl retry-build aus:

```
aws codebuild retry-build --id <build-id> --idempotency-token <idempotencyToken>
```

Ersetzen Sie im Befehl oben den folgenden Platzhalter:

- *<build-id>*: Erforderliche Zeichenfolge. Die ID des zu wiederholenden Build- oder Batch-Builds. Weitere Informationen zum Abrufen einer Liste mit Build-IDs finden Sie in den folgenden Themen:
  - Anzeigen einer Liste mit Build-IDs (AWS CLI)
  - Anzeigen einer Liste mit Batch-Build-IDs (AWS CLI)

- Anzeigen einer Liste mit Build-IDs für ein Build-Projekt (AWS CLI)
- Anzeigen einer Liste mit Batch-Build-IDs für ein Build-Projekt (AWS CLI)

- `--idempotency-token`: Optional. Wenn du dieretry-build-Befehl mit der Option, einem eindeutigen Bezeichner, in dem die Groß- und Kleinschreibung berücksichtigt wird, wird im`retry-build`request. Das Token ist nach der -Anforderung 5 Minuten gültig. Wenn du das wiederholst`retry-build`-Anforderung mit dem Token, jedoch einen Parameter ändern, gibt CodeBuild einen Fehler wegen des abweichenden Parameters zurück.

Wiederholen eines Build (AWSSDKs)

Weitere Informationen zur Verwendung von AWS CodeBuild mit den AWS-SDKs finden Sie unter AWS SDKs- und Tools-Referenz.

Anzeigen eines laufenden Builds in Session Manager

In können Sie einen laufenden Build anhalten und dann AWS Systems Manager Session Manager verwenden AWS CodeBuild, um eine Verbindung zum Build-Container herzustellen und den Status des Containers anzuzeigen.

> **Note**
>
> Diese Funktion ist in Windows-Umgebungen nicht verfügbar.

Themen

- Voraussetzungen
- Pausieren des Builds
- Starten des Builds
- Herstellen einer Verbindung mit dem Build-Container
- Fortsetzen des Builds

Voraussetzungen

Damit Session Manager mit der Build-Sitzung verwendet werden kann, müssen Sie die Sitzungsverbindung für den Build aktivieren. Es gibt zwei Voraussetzungen:

- CodeBuild Bei kuratierten Linux-Standard-Images ist der SSM-Agent bereits installiert und der SSM-Agent ContainerMode aktiviert.

  Wenn Sie ein benutzerdefiniertes Image für Ihren Build verwenden, gehen Sie wie folgt vor:

  1. Installieren Sie den SSM Agent. Weitere Informationen finden Sie unter Manuelles Installieren von SSM Agent auf EC2-Instances für Linux im AWS Systems Manager -Benutzerhandbuch. Die SSM-Agent-Version muss 3.0.1295.0 oder höher sein.
  2. Kopieren Sie die Datei https://github.com/aws/aws-codebuild-docker-images/blob/master/ubuntu/standard/5.0/amazon-ssm-agent.json in das `/etc/amazon/ssm/` Verzeichnis in Ihrem Image. Dadurch wird der Containermodus im SSM-Agenten aktiviert.

  > **Note**
  >
  > Benutzerdefinierte Images erfordern den aktuellsten SSM-Agenten, damit diese Funktion wie erwartet funktioniert.

- Die CodeBuild Servicerolle muss über die folgende SSM-Richtlinie verfügen:

```
{
  "Effect": "Allow",
  "Action": [
    "ssmmessages:CreateControlChannel",
    "ssmmessages:CreateDataChannel",
    "ssmmessages:OpenControlChannel",
    "ssmmessages:OpenDataChannel"
  ],
  "Resource": "*"
}
```

  Sie können die CodeBuild Konsole diese Richtlinie automatisch an Ihre Servicerolle anfügen lassen, wenn Sie den Build starten. Alternativ können Sie diese Richtlinie manuell an Ihre Servicerolle anfügen.

- Wenn Sie die Prüfung und Protokollierung von Sitzungsaktivitäten in den Systems Manager-Einstellungen aktiviert haben, muss die CodeBuild Servicerolle auch über zusätzliche Berechtigungen verfügen. Die Berechtigungen sind unterschiedlich, je nachdem, wo die Protokolle gespeichert sind.

CloudWatch Protokolle

Wenn Sie CloudWatch Protokolle zum Speichern Ihrer Protokolle verwenden, fügen Sie der CodeBuild Servicerolle die folgende Berechtigung hinzu:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "logs:DescribeLogGroups",
      "Resource": "arn:aws:logs:<region-id>:<account-id>:log-group:*:*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "logs:CreateLogStream",
        "logs:PutLogEvents"
      ],
      "Resource": "arn:aws:logs:<region-id>:<account-id>:log-group:<log-group-
name>:*"
    }
  ]
}
```

Amazon S3

Wenn Sie Amazon S3 zum Speichern Ihrer Protokolle verwenden, fügen Sie der CodeBuild Servicerolle die folgende Berechtigung hinzu:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetEncryptionConfiguration",
        "s3:PutObject"
      ],
      "Resource": [
        "arn:aws:s3:::<bucket-name>",
        "arn:aws:s3:::<bucket-name>/*"
```

```
      ]
    }
  ]
}
```

Weitere Informationen finden Sie unter Prüfen und Protokollieren von Sitzungsaktivitäten im AWS Systems Manager -Benutzerhandbuch.

Pausieren des Builds

Um den Build anzuhalten, fügen Sie den codebuild-breakpoint Befehl in eine der Build-Phasen in Ihre buildspec-Datei ein. Der Build wird an dieser Stelle angehalten, sodass Sie eine Verbindung zum Build-Container herstellen und den Container in seinem aktuellen Zustand anzeigen können.

Fügen Sie beispielsweise Folgendes zu den Build-Phasen in Ihrer buildspec-Datei hinzu.

```
phases:
  pre_build:
    commands:
      - echo Entered the pre_build phase...
      - echo "Hello World" > /tmp/hello-world
      - codebuild-breakpoint
```

Dieser Code erstellt die `/tmp/hello-world` Datei und pausiert dann den Build an dieser Stelle.

Starten des Builds

Damit Session Manager mit der Build-Sitzung verwendet werden kann, müssen Sie Sitzungsverbindungen für den Build aktivieren. Führen Sie dazu beim Starten des Builds die folgenden Schritte aus:

1. Öffnen Sie die - AWS CodeBuild Konsole unter https://console.aws.amazon.com/codesuite/codebuild/home.
2. Wählen Sie im linken Navigationsbereich Build projects aus. Wählen Sie das Build-Projekt und dann Build mit Überschreibungen starten aus.
3. Wählen Sie Advanced build overrides (Erweiterter Build überschreibt).
4. Wählen Sie im Abschnitt Umgebung die Option Sitzungsverbindung aktivieren aus. Wenn diese Option nicht ausgewählt ist, werden alle codebuild-resume Befehle codebuild-breakpoint und ignoriert.

5. Nehmen Sie alle anderen gewünschten Änderungen vor und wählen Sie Build starten aus.
6. Überwachen Sie den Build-Status in der -Konsole. Wenn die Sitzung verfügbar ist, wird der AWS Session Manager-Link im Abschnitt Build-Status angezeigt.

Herstellen einer Verbindung mit dem Build-Container

Sie können sich auf zwei Arten mit dem Build-Container verbinden:

CodeBuild -Konsole

Öffnen Sie in einem Webbrowser den AWS Session Manager-Link, um eine Verbindung zum Build-Container herzustellen. Eine Terminalsitzung wird geöffnet, mit der Sie den Build-Container durchsuchen und steuern können.

AWS CLI

> **Note**
>
> Auf Ihrem lokalen Computer muss das Session Manager-Plugin für dieses Verfahren installiert sein. Weitere Informationen finden Sie unter Installieren des Session Manager-Plugins für die AWS CLI im AWS Systems Manager -Benutzerhandbuch.

1. Rufen Sie die batch-get-builds API mit der Build-ID auf, um Informationen über den Build zu erhalten, einschließlich der Sitzungsziel-ID. Der Eigenschaftsname der Sitzungsziel-ID variiert je nach Ausgabetyp des `aws` Befehls. Aus diesem Grund `--output json` wird dem Befehl hinzugefügt.

   ```
   aws codebuild batch-get-builds --ids <buildID> --region <region> --output json
   ```

2. Kopieren Sie den `sessionTarget` Eigenschaftswert. Der `sessionTarget` Eigenschaftsname kann je nach Ausgabetyp des `aws` Befehls variieren. Aus diesem Grund `--output json` wird dem Befehl im vorherigen Schritt hinzugefügt.
3. Verwenden Sie den folgenden Befehl, um eine Verbindung mit dem Build-Container herzustellen.

   ```
   aws ssm start-session --target <sessionTarget> --region <region>
   ```

Stellen Sie in diesem Beispiel sicher, dass die `/tmp/hello-world` Datei vorhanden ist und den Text enthält`Hello World`.

Fortsetzen des Builds

Nachdem Sie den Build-Container untersucht haben, geben Sie den codebuild-resume Befehl aus der Container-Shell aus.

```
$ codebuild-resume
```

Löschen von Builds in AWS CodeBuild

Sie können die AWS CLI oder die AWS SDKs verwenden, um Builds in AWS CodeBuild zu löschen.

Löschen von Builds (AWS CLI)

Führen Sie den Befehl `batch-delete-builds` aus:

```
aws codebuild batch-delete-builds --ids ids
```

Ersetzen Sie im Befehl oben den folgenden Platzhalter:

- *`IDs`*: Erforderliche Zeichenfolge. Die ID des zu löschenden Builds. Um mehrere Builds anzugeben, fügen Sie zwischen den einzelnen Build-IDs ein Leerzeichen ein. Weitere Informationen zum Abrufen einer Liste mit Build-IDs finden Sie in den folgenden Themen:
  - Anzeigen einer Liste mit Build-IDs (AWS CLI)
  - Anzeigen einer Liste mit Build-IDs für ein Build-Projekt (AWS CLI)

Wenn erfolgreich, erscheint ein `buildsDeleted`-Array in der Ausgabe, der den Amazon-Ressourcennamen (ARN) der einzelnen erfolgreich gelöschten Builds enthält. Informationen zu Builds, die nicht erfolgreich gelöscht werden konnten, werden in der Ausgabe innerhalb eines `buildsNotDeleted`-Arrays angezeigt.

Wenn Sie z. B. den folgenden Befehl ausführen:

```
aws codebuild batch-delete-builds --ids my-demo-build-project:f8b888d2-5e1e-4032-8645-
b115195648EX my-other-demo-build-project:a18bc6ee-e499-4887-b36a-8c90349c7eEX
```

Die Informationen in der Ausgabe ähneln den folgenden:

```
{
  "buildsNotDeleted": [
    {
      "id": "arn:aws:codebuild:us-west-2:123456789012:build/my-demo-build-
project:f8b888d2-5e1e-4032-8645-b115195648EX",
      "statusCode": "BUILD_IN_PROGRESS"
    }
  ],
  "buildsDeleted": [
    "arn:aws:codebuild:us-west-2:123456789012:build/my-other-demo-build-
project:a18bc6ee-e499-4887-b36a-8c90349c7eEX"
  ]
}
```

Löschen von Builds (AWS SDKs)

Informationen zur Verwendung von AWS CodeBuild mit den AWS-SDKs finden Sie unter AWS SDKs- und Tools-Referenz.

Arbeiten mit AWS Lambda Compute in AWS CodeBuild

AWS Lambda Compute bietet optimierte Startgeschwindigkeiten für Ihre Builds. AWS Lambda unterstützt schnellere Builds aufgrund einer geringeren Startlatenz. AWS Lambda skaliert außerdem automatisch, sodass Builds nicht in der Warteschlange warten, bis sie ausgeführt werden. Es gibt jedoch einige Anwendungsfälle, die AWS Lambda nicht unterstützt werden, und wenn sie Sie betreffen, verwenden Sie die EC2-Rechenleistung. Weitere Informationen finden Sie unter Einschränkungen der AWS Lambda Datenverarbeitung.

Themen

Welche Tools und Laufzeiten werden in der kuratierten Laufzeitumgebung enthalten sein, auf der Docker-Images ausgeführt werden? AWS Lambda

AWS Lambda unterstützt die folgenden Tools: AWS CLI v2, AWS SAM CLI, Git, Go, Java, Node.js, Python, Pip, Ruby und.NET.

Was ist, wenn das kuratierte Bild nicht die Tools enthält, die ich benötige?

Wenn das kuratierte Image nicht die Tools enthält, die Sie benötigen, können Sie ein Docker-Image für die benutzerdefinierte Umgebung bereitstellen, das die erforderlichen Tools enthält.

Beachten Sie, dass Sie die folgenden Amazon ECR-Berechtigungen benötigen, um benutzerdefinierte Images für Lambda-Berechnungen zu verwenden:

```
{
```

```
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ecr:GetAuthorizationToken"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "ecr:BatchCheckLayerAvailability",
        "ecr:GetDownloadUrlForLayer",
        "ecr:BatchGetImage"
      ],
      "Resource": "arn:aws:ecr:image-region:image-account-id:repository/image-repo"
    }
  ]
}
```

Beachten Sie auch, dass `curl` or installiert sein `wget` muss, um benutzerdefinierte Images verwenden zu können.

In welchen Regionen wird AWS Lambda Rechenleistung unterstützt CodeBuild?

In CodeBuild wird AWS Lambda Compute in den folgenden Ländern unterstützt AWS-Regionen: USA Ost (Nord-Virginia), USA Ost (Ohio), USA West (Oregon), Asien-Pazifik (Mumbai), Asien-Pazifik (Singapur), Asien-Pazifik (Sydney), Asien-Pazifik (Tokio), Europa (Frankfurt), Europa (Irland) und Südamerika (São Paulo). Weitere Informationen darüber, AWS-Regionen wo verfügbar CodeBuild ist, finden Sie unter AWS Services nach Regionen.

Einschränkungen der AWS Lambda Datenverarbeitung

Es gibt einige Anwendungsfälle, die AWS Lambda nicht unterstützt werden, und wenn sie sich auf Sie auswirken, verwenden Sie die EC2-Berechnung:

- AWS Lambda unterstützt keine Tools, für die Root-Rechte erforderlich sind. Verwenden Sie für Tools wie `yum` oder `rpm` den EC2-Compute-Typ oder andere Tools, für die keine Root-Rechte erforderlich sind.
- AWS Lambda unterstützt keine Docker-Builds oder -Runs.
- AWS Lambda unterstützt das Schreiben in Dateien außerhalb `/tmp` nicht. Die mitgelieferten Paketmanager sind so konfiguriert, dass sie das `/tmp` Verzeichnis standardmäßig zum Herunterladen und Referenzieren von Paketen verwenden.
- AWS Lambda unterstützt den Umgebungstyp nicht `LINUX_GPU_CONTAINER` und wird unter Windows Server Core 2019 nicht unterstützt.
- AWS Lambda unterstützt Caching, Batch-Builds, Timeouts für benutzerdefinierte Builds, Warteschlangen-Timeout, Build-Badges, privilegierten Modus, benutzerdefinierte Laufzeitumgebungen oder Laufzeiten von mehr als 15 Minuten nicht.
- AWS Lambda unterstützt keine VPC-Konnektivität, einen festen Bereich von CodeBuild Quell-IP-Adressen, EFS, semantische Versionierung, Installation von Zertifikaten oder SSH-Zugriff mit Session Manager.

AWS Lambda berechne Beispiele mit AWS CodeBuild

Diese Gruppen von Stichproben können verwendet werden, um mit AWS Lambda Compute In zu experimentieren CodeBuild.

Themen

- Bereitstellen einer Lambda-Funktion mit AWS SAM mit CodeBuild Lambda Java
- Erstellen einer einseitigen React-App mit CodeBuild Lambda Node.js
- Aktualisieren einer Lambda-Funktionskonfiguration mit CodeBuild Lambda Python

Bereitstellen einer Lambda-Funktion mit AWS SAM mit CodeBuild Lambda Java

Die AWS Serverless Application Model (AWS SAM) ist ein Open-Source-Framework für die Erstellung von Serverless-Anwendungen. Weitere Informationen finden Sie im AWS Serverless Application ModelRepository auf GitHub. Das folgende Java-Beispiel verwendet Gradle, um eine AWS Lambda Funktion zu erstellen und zu testen. Danach wird die AWS SAM CLI verwendet, um die AWS CloudFormation Vorlage und das Bereitstellungspaket bereitzustellen. Durch die Verwendung

von CodeBuild Lambda werden die Build-, Test- und Bereitstellungsschritte alle automatisch ausgeführt, sodass die Infrastruktur schnell und ohne manuellen Eingriff in einem einzigen Build aktualisiert werden kann.

Einrichten Ihres AWS SAMRepositorys

Erstellen Sie ein -AWS SAM`Hello World`Projekt mit der -AWS SAMCLI.

So erstellen Sie Ihr AWS SAM Projekt

1. Folgen Sie den Anweisungen im -AWS Serverless Application ModelEntwicklerhandbuch zum Installieren der AWS SAM -CLI auf Ihrem lokalen Computer.
2. Führen Sie aus `sam init` und wählen Sie die folgende Projektkonfiguration aus.

```
Which template source would you like to use?: 1 - AWS Quick Start Templates
Choose an AWS Quick Start application template: 1 - Hello World Example
Use the most popular runtime and package type? (Python and zip) [y/N]: N
Which runtime would you like to use?: 8 - java21
What package type would you like to use?: 1 - Zip
Which dependency manager would you like to use?: 1 - gradle
Would you like to enable X-Ray tracing on the function(s) in your application? [y/
N]: N
Would you like to enable monitoring using CloudWatch Application Insights? [y/N]: N
Would you like to set Structured Logging in JSON format on your Lambda functions?
 [y/N]:  N
Project name [sam-app]: <insert project name>
```

3. Laden Sie den AWS SAM Projektordner in ein unterstütztes Quell-Repository hoch. Eine Liste der unterstützten Quelltypen finden Sie unter ProjectSource.

Erstellen eines CodeBuild Lambda-Java-Projekts

Erstellen Sie ein AWS CodeBuild Lambda-Java-Projekt und richten Sie die für den Build erforderlichen IAM-Berechtigungen ein.

So erstellen Sie Ihr CodeBuild Lambda-Java-Projekt

1. Öffnen Sie die -AWS CodeBuildKonsole unter https://console.aws.amazon.com/codesuite/codebuild/home.

2. Wenn eine CodeBuild Informationsseite angezeigt wird, wählen Sie Build-Projekt erstellen aus. Erweitern Sie andernfalls im Navigationsbereich Build , wählen Sie Build-Projekte und dann Build-Projekt erstellen aus.
3. Geben Sie unter Project name (Projektname) einen Namen für dieses Build-Projekt ein. Build-Projektnamen müssen in allen AWS-Konten eindeutig sein. Sie können auch eine optionale Beschreibung des Build-Projekts einfügen, um anderen Benutzern zu helfen zu verstehen, wofür dieses Projekt verwendet wird.
4. Wählen Sie unter Quelle das Quell-Repository aus, in dem sich Ihr AWS SAM Projekt befindet.
5. In Environment (Umgebung):
   - Wählen Sie für Datenverarbeitung die Option Lambda aus.
   - Wählen Sie für Laufzeit(en) Java aus.
   - Wählen Sie für Image aws/codebuild/amazonlinux-x86_64-lambda-standard :corretto21 aus.
   - Lassen Sie für Servicerolle die Option Neue Servicerolle ausgewählt. Notieren Sie sich den Rollennamen . Dies ist erforderlich, wenn Sie die IAM-Berechtigungen des Projekts später in diesem Beispiel aktualisieren.
6. Wählen Sie Create build project (Build-Projekt erstellen) aus.
7. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
8. Wählen Sie im Navigationsbereich Rollen und dann die Servicerolle aus, die Ihrem Projekt zugeordnet ist. Sie finden Ihre Projektrolle in , CodeBuild indem Sie Ihr Build-Projekt auswählen, Bearbeiten, Umgebung und dann Servicerolle auswählen.
9. Wählen Sie die Registerkarte Trust Relationships (Vertrauensstellungen) und dann Edit trust policy (Vertrauensrichtlinie bearbeiten) aus.
10. Fügen Sie Ihrer IAM-Rolle die folgende Inline-Richtlinie hinzu. Dies wird verwendet, um Ihre AWS SAM Infrastruktur später bereitzustellen. Informationen finden Sie im Abschnitt Hinzufügen und Entfernen von IAM-Identitätsberechtigungen im -IAM-Benutzerhandbuch.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Action": [
                "cloudformation:*",
                "lambda:*",
```

```
                "iam:*",
                "apigateway:*",
                "s3:*"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
```

Einrichten der Projekt-Build-Spezifikation

Um Ihre Lambda-Funktion zu erstellen, zu testen und bereitzustellen, CodeBuild liest und führt Build-Befehle aus einer Build-Spezifikation aus.

So richten Sie Ihre Projekt-Build-Spezifikation ein

1. Wählen Sie in der - CodeBuild Konsole Ihr Build-Projekt und dann Bearbeiten und Buildspec aus.
2. Wählen Sie unter Buildspec die Option Build-Befehle einfügen und dann Zum Editor wechseln aus.
3. Löschen Sie die vorausgefüllten Build-Befehle und fügen Sie die folgende Build-Spezifikation ein.

```
version: 0.2
env:
  variables:
    GRADLE_DIR: "HelloWorldFunction"
phases:
  build:
    commands:
      - echo "Running unit tests..."
      - cd $GRADLE_DIR; gradle test; cd ..
      - echo "Running build..."
      - sam build --template-file template.yaml
      - echo "Running deploy..."
      - sam package --output-template-file packaged.yaml --resolve-s3 --template-
file template.yaml
      - yes | sam deploy
```

4. Wählen Sie Update buildspec (Buildspec aktualisieren).

Bereitstellen Ihrer AWS SAM Lambda-Infrastruktur

Verwenden Sie CodeBuild Lambda, um Ihre Lambda-Infrastruktur automatisch bereitzustellen

So stellen Sie Ihre Lambda-Infrastruktur bereit

1. Wählen Sie Start build (Build starten). Dadurch wird Ihre AWS SAM Anwendung automatisch AWS Lambda mithilfe von erstellt, getestet und in bereitgestelltAWS CloudFormation.
2. Sobald der Build abgeschlossen ist, navigieren Sie zur AWS Lambda Konsole und suchen Sie nach Ihrer neuen Lambda-Funktion unter dem AWS SAM Projektnamen.
3. Testen Sie Ihre Lambda-Funktion, indem Sie API Gateway in der Funktionsübersicht auswählen und dann auf die API-Endpunkt-URL klicken. Sie sollten eine Seite mit der Meldung geöffnet sehen`"message": "hello world"`.

Bereinigen Ihrer Infrastruktur

Um weitere Gebühren für Ressourcen zu vermeiden, die Sie in diesem Tutorial verwendet haben, löschen Sie die Ressourcen, die von Ihrer AWS SAM Vorlage und erstellt wurden CodeBuild.

So bereinigen Sie Ihre Infrastruktur

1. Navigieren Sie zur -AWS CloudFormationKonsole und wählen Sie aus`aws-sam-cli-managed-default`.
2. Leeren Sie unter Ressourcen den Bereitstellungs-Bucket `SamCliSourceBucket`.
3. Löschen Sie den `aws-sam-cli-managed-default`Stack.
4. Löschen Sie den AWS CloudFormationStack, der Ihrem AWS SAM Projekt zugeordnet ist. Dieser Stack sollte denselben Namen wie Ihr AWS SAM Projekt haben.
5. Navigieren Sie zur - CloudWatch Konsole und löschen Sie die CloudWatch Protokollgruppen, die Ihrem CodeBuild Projekt zugeordnet sind.
6. Navigieren Sie zur - CodeBuild Konsole und löschen Sie Ihr CodeBuild Projekt, indem Sie Build-Projekt löschen auswählen.

Erstellen einer einseitigen React-App mit CodeBuild Lambda Node.js

Create React App ist eine Möglichkeit, einseitige React-Anwendungen zu erstellen. Das folgende Node.js-Beispiel verwendet Node.js, um die Quellartefakte aus Create React App zu erstellen, und gibt die Build-Artefakte zurück.

Einrichten Ihres Quell-Repositorys und Artefakt-Buckets

Erstellen Sie ein Quell-Repository für Ihr Projekt mithilfe von Yarn und Create React App.

So richten Sie das Quell-Repository und den Artefakt-Bucket ein

1. Führen Sie auf Ihrem lokalen Computer aus, `yarn create react-app <app-name>` um eine einfache React-App zu erstellen.
2. Laden Sie den Projektordner der React-App in ein unterstütztes Quell-Repository hoch. Eine Liste der unterstützten Quelltypen finden Sie unter ProjectSource.

Erstellen eines CodeBuild Lambda-Node.js-Projekts

Erstellen Sie ein AWS CodeBuild Lambda-Node.js-Projekt.

So erstellen Sie Ihr CodeBuild Lambda-Node.js-Projekt

1. Öffnen Sie die -AWS CodeBuildKonsole unter https://console.aws.amazon.com/codesuite/codebuild/home.
2. Wenn eine CodeBuild Informationsseite angezeigt wird, wählen Sie Build-Projekt erstellen aus. Erweitern Sie andernfalls im Navigationsbereich Build , wählen Sie Build-Projekte und dann Build-Projekt erstellen aus.
3. Geben Sie unter Project name (Projektname) einen Namen für dieses Build-Projekt ein. Build-Projektnamen müssen in allen AWS-Konten eindeutig sein. Sie können auch eine optionale Beschreibung des Build-Projekts einfügen, um anderen Benutzern zu helfen, zu verstehen, wofür dieses Projekt verwendet wird.
4. Wählen Sie unter Quelle das Quell-Repository aus, in dem sich Ihr AWS SAM Projekt befindet.
5. In Environment (Umgebung):
   - Wählen Sie für Datenverarbeitung die Option Lambda aus.
   - Wählen Sie für Laufzeit(en) Node.js aus.

- Wählen Sie für Image aws/codebuild/amazonlinux-x86_64-lambda-standard :nodejs20 aus.

6. In Artifacts (Artefakte):

   - Wählen Sie für Typ Amazon S3 aus.
   - Wählen Sie für Bucket-Name den Projektartefakt-Bucket aus, den Sie zuvor erstellt haben.
   - Wählen Sie für Artifacts packaging die Option Zip aus.

7. Wählen Sie Create build project (Build-Projekt erstellen) aus.

Einrichten der Projekt-Build-Spezifikation

Um Ihre React-App zu erstellen, CodeBuild liest und führt Build-Befehle aus einer buildspec-Datei aus.

So richten Sie Ihre Projekt-Build-Spezifikation ein

1. Wählen Sie in der - CodeBuild Konsole Ihr Build-Projekt und dann Bearbeiten und Buildspec aus.
2. Wählen Sie unter Buildspec die Option Build-Befehle einfügen und dann Zum Editor wechseln aus.
3. Löschen Sie die vorausgefüllten Build-Befehle und fügen Sie die folgende buildspec ein.

```
version: 0.2
phases:
  build:
    commands:
      - yarn
      - yarn add --dev jest-junit @babel/plugin-proposal-private-property-in-object
      - yarn run build
      - yarn run test -- --coverage --watchAll=false --testResultsProcessor="jest-
junit" --detectOpenHandles
artifacts:
  name: "build-output"
  files:
    - "**/*"
reports:
  test-report:
    files:
      - 'junit.xml'
    file-format: 'JUNITXML'
  coverage-report:
```

```
    files:
      - 'coverage/clover.xml'
    file-format: 'CLOVERXML'
```

4. Wählen Sie Update buildspec (Buildspec aktualisieren).

Erstellen und Ausführen Ihrer React-App

Erstellen Sie die React-App auf CodeBuild Lambda, laden Sie die Build-Artefakte herunter und führen Sie die React-App lokal aus.

So erstellen und führen Sie Ihre React-App aus

1. Wählen Sie Start build (Build starten).
2. Sobald der Build abgeschlossen ist, navigieren Sie zu Ihrem Amazon S3-Projektartefakt-Bucket und laden Sie das React-App-Artefakt herunter.
3. Entpacken Sie das React-Build-Artefakt und `run npm install -g serve && serve -s build` im Projektordner.
4. Der `serve` Befehl stellt die statische Site auf einem lokalen Port bereit und gibt die Ausgabe auf Ihrem Terminal aus. Sie können die localhost-URL unter `Local:` in der Terminalausgabe aufrufen, um Ihre React-App anzuzeigen.

Weitere Informationen zur Handhabung der Bereitstellung für einen React-basierten Server finden Sie unter Erstellen einer React-App-Bereitstellung.

Bereinigen Ihrer Infrastruktur

Um weitere Gebühren für Ressourcen zu vermeiden, die Sie in diesem Tutorial verwendet haben, löschen Sie die für Ihr CodeBuild Projekt erstellten Ressourcen.

So bereinigen Sie Ihre Infrastruktur

1. Löschen Ihres Amazon S3-Buckets für Projektartefakte
2. Navigieren Sie zur CloudWatch -Konsole und löschen Sie die CloudWatch Protokollgruppen, die Ihrem CodeBuild Projekt zugeordnet sind.
3. Navigieren Sie zur CodeBuild -Konsole und löschen Sie Ihr CodeBuild Projekt, indem Sie Build-Projekt löschen auswählen.

Aktualisieren einer Lambda-Funktionskonfiguration mit CodeBuild Lambda Python

Das folgende Python-Beispiel verwendet Boto3 und CodeBuild Lambda Python, um die Konfiguration einer Lambda-Funktion zu aktualisieren. Dieses Beispiel kann erweitert werden, um andere AWS Ressourcen programmgesteuert zu verwalten. Weitere Informationen finden Sie in der Boto3-Dokumentation.

Voraussetzungen

Erstellen oder finden Sie eine Lambda-Funktion in Ihrem Konto.

In diesem Beispiel wird davon ausgegangen, dass Sie bereits eine Lambda-Funktion in Ihrem Konto erstellt haben und verwenden, CodeBuild um die Umgebungsvariablen der Lambda-Funktion zu aktualisieren. Weitere Informationen zum Einrichten einer Lambda-Funktion über CodeBuildfinden Sie im Bereitstellen einer Lambda-Funktion mit AWS SAM mit CodeBuild Lambda Java Beispiel oder unter AWS Lambda.

Einrichten Ihres Quell-Repositorys

Erstellen Sie ein Quell-Repository, um Ihr Boto3-Python-Skript zu speichern.

So richten Sie das Quell-Repository ein

1. Kopieren Sie das folgende Python-Skript in eine neue Datei namens `update_lambda_environment_variables.py`.

```
import boto3
from os import environ


def update_lambda_env_variable(lambda_client):
    lambda_function_name = environ['LAMBDA_FUNC_NAME']
    lambda_env_variable = environ['LAMBDA_ENV_VARIABLE']
    lambda_env_variable_value = environ['LAMBDA_ENV_VARIABLE_VALUE']
    print("Updating lambda function " + lambda_function_name + " environment
 variable "
          + lambda_env_variable + " to " + lambda_env_variable_value)
    lambda_client.update_function_configuration(
        FunctionName=lambda_function_name,
```

```
        Environment={
            'Variables': {
                lambda_env_variable: lambda_env_variable_value
            }
        },
    )


if __name__ == "__main__":
    region = environ['AWS_REGION']
    client = boto3.client('lambda', region)
    update_lambda_env_variable(client)
```

2. Laden Sie die Python-Datei in ein unterstütztes Quell-Repository hoch. Eine Liste der unterstützten Quelltypen finden Sie unter ProjectSource.

Erstellen eines CodeBuild Lambda-Python-Projekts

Erstellen Sie ein CodeBuild Lambda-Python-Projekt.

So erstellen Sie Ihr CodeBuild Lambda-Java-Projekt

1. Öffnen Sie die -AWS CodeBuildKonsole unter https://console.aws.amazon.com/codesuite/codebuild/home.
2. Wenn eine CodeBuild Informationsseite angezeigt wird, wählen Sie Build-Projekt erstellen aus. Erweitern Sie andernfalls im Navigationsbereich Build, wählen Sie Build-Projekte und dann Build-Projekt erstellen aus.
3. Geben Sie unter Project name (Projektname) einen Namen für dieses Build-Projekt ein. Build-Projektnamen müssen in allen AWS-Konten eindeutig sein. Sie können auch eine optionale Beschreibung des Build-Projekts einfügen, um anderen Benutzern zu helfen zu verstehen, wofür dieses Projekt verwendet wird.
4. Wählen Sie unter Quelle das Quell-Repository aus, in dem sich Ihr AWS SAM Projekt befindet.
5. In Environment (Umgebung):
   - Wählen Sie für Datenverarbeitung die Option Lambda aus.
   - Wählen Sie für Laufzeit(en) Python aus.
   - Wählen Sie für Image aws/codebuild/amazonlinux-x86_64-lambda-standard :python3.12 aus.

- Lassen Sie für Servicerolle die Option Neue Servicerolle ausgewählt. Notieren Sie sich den Rollennamen . Dies ist erforderlich, wenn Sie die IAM-Berechtigungen des Projekts später in diesem Beispiel aktualisieren.

6. Wählen Sie Create build project (Build-Projekt erstellen) aus.
7. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
8. Wählen Sie im Navigationsbereich Rollen und dann die Servicerolle aus, die Ihrem Projekt zugeordnet ist. Sie finden Ihre Projektrolle in , CodeBuild indem Sie Ihr Build-Projekt auswählen, Bearbeiten, Umgebung und dann Servicerolle auswählen.
9. Wählen Sie die Registerkarte Trust Relationships (Vertrauensstellungen) und dann Edit trust policy (Vertrauensrichtlinie bearbeiten) aus.
10. Fügen Sie Ihrer IAM-Rolle die folgende Inline-Richtlinie hinzu. Dies wird verwendet, um Ihre AWS SAM Infrastruktur später bereitzustellen. Informationen finden Sie im Abschnitt Hinzufügen und Entfernen von IAM-Identitätsberechtigungen im -IAM-Benutzerhandbuch.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "UpdateLambdaPermissions",
            "Effect": "Allow",
            "Action": [
                "lambda:UpdateFunctionConfiguration"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
```

Einrichten der Projekt-Build-Spezifikation

Um die Lambda-Funktion zu aktualisieren, liest das Skript Umgebungsvariablen aus der buildspec, um den Namen der Lambda-Funktion, den Namen der Umgebungsvariablen und den Wert der Umgebungsvariablen zu finden.

So richten Sie Ihre Projekt-Build-Spezifikation ein

1. Wählen Sie in der - CodeBuild Konsole Ihr Build-Projekt und dann Bearbeiten und Buildspec aus.
2. Wählen Sie unter Buildspec die Option Build-Befehle einfügen und dann Zum Editor wechseln aus.
3. Löschen Sie die vorausgefüllten Build-Befehle und fügen Sie die folgende Build-Spezifikation ein.

```
version: 0.2
env:
  variables:
    LAMBDA_FUNC_NAME: "<lambda-function-name>"
    LAMBDA_ENV_VARIABLE: "FEATURE_ENABLED"
    LAMBDA_ENV_VARIABLE_VALUE: "true"
phases:
  install:
    commands:
       - pip3 install boto3
  build:
    commands:
       - python3 update_lambda_environment_variables.py
```

4. Wählen Sie Update buildspec (Buildspec aktualisieren).

Aktualisieren Ihrer Lambda-Konfiguration

Verwenden Sie CodeBuild Lambda Python, um die Konfiguration Ihrer Lambda-Funktion automatisch zu aktualisieren.

So aktualisieren Sie die Konfiguration Ihrer Lambda-Funktion

1. Wählen Sie Start build (Build starten).
2. Sobald der Build abgeschlossen ist, navigieren Sie zu Ihrer Lambda-Funktion.
3. Wählen Sie Konfiguration und dann Umgebungsvariablen aus. Sie sollten eine neue Umgebungsvariable mit Schlüssel `FEATURE_ENABLED` und Wert sehen`true`.

Bereinigen Ihrer Infrastruktur

Um weitere Gebühren für Ressourcen zu vermeiden, die Sie in diesem Tutorial verwendet haben, löschen Sie die für Ihr CodeBuild Projekt erstellten Ressourcen.

So bereinigen Sie Ihre Infrastruktur

1. Navigieren Sie zur - CloudWatch Konsole und löschen Sie die CloudWatch Protokollgruppen, die Ihrem CodeBuild Projekt zugeordnet sind.
2. Navigieren Sie zur - CodeBuild Konsole und löschen Sie Ihr CodeBuild Projekt, indem Sie Build-Projekt löschen auswählen.
3. Wenn Sie für dieses Beispiel eine Lambda-Funktion erstellt haben, wählen Sie Aktionen und Funktion löschen, um Ihre Lambda-Funktion zu bereinigen.

Erweiterungen

Wenn Sie dieses Beispiel erweitern möchten, um andere AWS Ressourcen mit AWS CodeBuild Lambda Python zu verwalten:

- Aktualisieren Sie das Python-Skript, um die neuen Ressourcen mit Boto3 zu ändern.
- Aktualisieren Sie die Ihrem CodeBuild Projekt zugeordnete IAM-Rolle, um über Berechtigungen für die neuen Ressourcen zu verfügen.
- Fügen Sie Ihrer Build-Spezifikation alle neuen Umgebungsvariablen hinzu, die den neuen Ressourcen zugeordnet sind.

Arbeiten mit reservierter Kapazität in AWS CodeBuild

CodeBuild bietet die folgenden Rechenflotten:

- Flotten auf Abruf
- Flotten mit reservierter Kapazität

Mit On-Demand-Flotten CodeBuild bietet es Rechenleistung für Ihre Builds. Die Maschinen werden zerstört, wenn der Bau abgeschlossen ist. On-Demand-Flotten werden vollständig verwaltet und verfügen über automatische Skalierungsfunktionen zur Bewältigung von Nachfragespitzen.

> **Note**
>
> On-Demand-Flotten unterstützen Windows Server 2022 nicht.

CodeBuild bietet auch Flotten mit reservierter Kapazität, die von Amazon EC2 betriebene Instances enthalten, die von verwaltet werden. CodeBuild Mit Flotten mit reservierter Kapazität konfigurieren Sie eine Reihe von dedizierten Instances für Ihre Build-Umgebung. Diese Maschinen bleiben inaktiv und sind bereit, Builds oder Tests sofort zu verarbeiten, wodurch die Build-Dauer reduziert wird. Mit Flotten mit reservierter Kapazität sind Ihre Maschinen immer in Betrieb und es fallen weiterhin Kosten an, solange sie bereitgestellt werden.

> **Important**
>
> Unabhängig davon, wie lange Sie eine Instance ausführen, fällt für Flotten mit reservierter Kapazität eine anfängliche Gebühr pro Instanz an. Danach können zusätzliche Kosten anfallen. Weitere Informationen finden Sie unter https://aws.amazon.com/codebuild/pricing/.

Themen

- Flotteneigenschaften mit reservierter Kapazität
- Beispiele für reservierte Kapazität mit AWS CodeBuild
- Beschränkungen von Flotten mit reservierter Kapazität

Wie fange ich mit Flotten mit reservierter Kapazität an?

Um eine Flotte mit reservierter Kapazität zu erstellen

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die AWS CodeBuild Konsole unter https://console.aws.amazon.com/codesuite/codebuild/home.
2. Wählen Sie im Navigationsbereich Compute Fleets und dann Create Compute Fleets aus.
3. Geben Sie im Textfeld „Flottenname berechnen“ einen Namen für Ihre Flotte ein.
4. Wählen Sie im Drop-down-Menü Betriebssystem das Betriebssystem aus.
5. Wählen Sie im Dropdownmenü Architektur die Architektur aus.
6. Wählen Sie im Dropdownmenü Compute den Computermaschinentyp für Ihren Computer aus.
7. Geben Sie im Textfeld Kapazität die Mindestanzahl von Instanzen in der Flotte ein.
8. Wählen Sie im Feld Überlaufverhalten das Verhalten aus, wenn der Bedarf die Flottenkapazität übersteigt. Weitere Informationen zu diesen Optionen finden Sie unter Flotteneigenschaften mit reservierter Kapazität.
9. (Optional) Wenn Sie das Amazon Linux-Betriebssystem ausgewählt haben, gehen Sie unter Zusätzliche Konfiguration wie folgt vor:
   - Wählen Sie im Drop-down-Menü VPC — optional eine VPC aus, auf die Ihre CodeBuild Flotte zugreifen wird.
   - Wählen Sie im Dropdownmenü Subnetze die Subnetze aus, die Sie für die Einrichtung Ihrer VPC-Konfiguration verwenden CodeBuild sollten.
   - Wählen Sie im Dropdownmenü Sicherheitsgruppen die Sicherheitsgruppen aus, die für die Zusammenarbeit mit Ihrer VPC verwendet werden CodeBuild sollen.
   - Wählen Sie im Feld Flotten-Servicerolle entweder eine Neue Servicerolle oder eine Bestehende Servicerolle
     - Fahren Sie für eine bestehende Servicerolle im Drop-down-Menü Servicerolle fort und wählen Sie eine Servicerolle aus.
10. Wählen Sie Rechenflotte erstellen aus.

11. Nachdem die Rechenflotte erstellt wurde, erstellen Sie ein neues CodeBuild Projekt oder bearbeiten Sie ein vorhandenes. Wählen Sie unter Umgebung unter Bereitstellungsmodell die Option Reservierte Kapazität und dann unter Flottenname die angegebene Flotte aus.

Bewährte Methoden

Wenn Sie Flotten mit reservierter Kapazität verwenden, empfehlen wir Ihnen, diese bewährten Methoden zu befolgen.

- Wir empfehlen, den Quell-Cache-Modus zu verwenden, um die Build-Performance durch Zwischenspeichern der Quelle zu verbessern.
- Wir empfehlen die Verwendung von Docker-Layer-Caching, um die Build-Performance zu verbessern, indem vorhandene Docker-Ebenen zwischengespeichert werden.

Kann ich eine Flotte mit reservierter Kapazität für mehrere Projekte gemeinsam nutzen? CodeBuild

Ja, Sie können die Auslastung der Kapazität einer Flotte maximieren, indem Sie sie projektübergreifend einsetzen.

> **Important**
>
> Wenn Sie die Funktion für reservierte Kapazität verwenden, können Daten, die auf Flotteninstanzen zwischengespeichert wurden, einschließlich Quelldateien, Docker-Ebenen und zwischengespeicherten Verzeichnissen, die in der Buildspec angegeben sind, für andere Projekte innerhalb desselben Kontos zugänglich sein. Dies ist beabsichtigt und ermöglicht es Projekten innerhalb desselben Kontos, Flotteninstanzen gemeinsam zu nutzen.

Welche Regionen unterstützen Flotten mit reservierter Kapazität?

Flotten mit reservierter Kapazität werden in den folgenden Ländern unterstützt AWS-Regionen: USA Ost (Nord-Virginia), USA Ost (Ohio), USA West (Oregon), Asien-Pazifik (Mumbai), Asien-Pazifik (Singapur), Asien-Pazifik (Sydney), Asien-Pazifik (Tokio), Europa (Frankfurt), Europa (Irland) und Südamerika (São Paulo). Weitere Informationen darüber, AWS-Regionen wo sie verfügbar CodeBuild sind, finden Sie unter AWS Services nach Regionen.

Flotteneigenschaften mit reservierter Kapazität

Eine Flotte mit reservierter Kapazität umfasst die folgenden Eigenschaften:

Betriebssystem

Das Betriebssystem. Die folgenden Betriebssysteme sind verfügbar:

- Amazon Linux
- Windows Server 2019
- Windows Server 2022

Architektur

Die Prozessorarchitektur. Die folgenden Architekturen sind verfügbar:

- x86_64
- Arm64

Datenverarbeitung

Der Computertyp für jede Instanz. Die folgenden Maschinentypen sind verfügbar:

Datenverarbeitung	ComputeType-Wert für die Umgebung	Wert für den Umgebungstyp	Arbeitsspeicher	vCPUs	Festplattenkapazität
ARM Klein	`BUILD_GENERAL1_SMALL`	`ARM_CONTAINER`	4 GB	2	50 GB
ARM Groß	`BUILD_GENERAL1_LARGE`	`ARM_CONTAINER`	16 GB	8	50 GB
Linux Klein [1]	`BUILD_GENERAL1_SMALL`	`LINUX_CONTAINER`	3 GB	2	64 GB

Datenverarbeitung	ComputeType-Wert für die Umgebung	Wert für den Umgebungstyp	Arbeitsspeicher	vCPUs	Festplattenkapazität
Linux Medium [1]	BUILD_GENERAL1_MEDIUM	LINUX_CONTAINER	7 GB	4	128 GB
Linux Large [1]	BUILD_GENERAL1_LARGE	LINUX_CONTAINER	15 GB	8	128 GB
Linux XLarge	BUILD_GENERAL1_XLARGE	LINUX_CONTAINER	70 GB	36	256 GB
Linux 2 x groß	BUILD_GENERAL1_2XLARGE	LINUX_CONTAINER	145 GB	72	824 GB (SSD)
Kleine Linux-GPU	BUILD_GENERAL1_SMALL	LINUX_GPU_CONTAINER	16 GB	4	220 GB
Große Linux-GPU	BUILD_GENERAL1_LARGE	LINUX_GPU_CONTAINER	255 GB	32	50 GB
Windows Medium	BUILD_GENERAL1_MEDIUM	WINDOWS_SERVER_2019_CONTAINER	7 GB	4	128 GB
Windows Medium	BUILD_GENERAL1_MEDIUM	WINDOWS_SERVER_2022_CONTAINER	7 GB	4	128 GB

Datenverarbeitung	ComputeType-Wert für die Umgebung	Wert für den Umgebungstyp	Arbeitsspeicher	vCPUs	Festplattenkapazität
Windows groß	`BUILD_GENERAL1_LARGE`	`WINDOWS_SERVER_2019_CONTAINER`	15 GB	8	128 GB
Große Fenster	`BUILD_GENERAL1_LARGE`	`WINDOWS_SERVER_2022_CONTAINER`	15 GB	8	128 GB

Capacity (Kapazität)

Die anfängliche Anzahl der Maschinen, die der Flotte zugewiesen wurden. Sie definiert die Anzahl der Builds, die parallel ausgeführt werden können.

Verhalten bei Überlauf

Definiert das Verhalten, wenn die Anzahl der Builds die Flottenkapazität überschreitet.

Auf Abruf

Overflow-Builds werden auf CodeBuild Abruf ausgeführt.

> **Note**
>
> Wenn Sie Ihr Überlaufverhalten bei der Erstellung einer VPC-verbundenen Flotte auf On-Demand-Einstellung festlegen möchten, stellen Sie sicher, dass Sie Ihrer Projektservice-Rolle die erforderlichen VPC-Berechtigungen hinzufügen. Weitere Informationen finden Sie unter Beispiel einer Richtlinienanweisung für den CodeBuild Zugriff auf AWS Dienste, die für die Erstellung einer VPC-Netzwerkschnittstelle erforderlich sind.

> **Important**
>
> Wenn Sie Ihr Overflow-Verhalten auf On-Demand-Modus setzen möchten, beachten Sie, dass Overflow-Builds separat in Rechnung gestellt werden, ähnlich wie bei Amazon EC2 auf Abruf. Weitere Informationen finden Sie unter https://aws.amazon.com/codebuild/pricing/.

Warteschlange

Build-Läufe werden in eine Warteschlange gestellt, bis ein Computer verfügbar ist. Dadurch werden zusätzliche Kosten begrenzt, da keine zusätzlichen Maschinen zugewiesen werden.

Zusätzliche Konfiguration

VPC — optional

Die VPC, auf die Ihre CodeBuild Flotte zugreifen wird. Beachten Sie, dass Konnektivität von CodeBuild nur für Flotten mit reservierter Kapazität auf Amazon Linux unterstützt wird. Weitere Informationen finden Sie unter Verwendung AWS CodeBuild mit Amazon Virtual Private Cloud.

Subnets

Die VPC-Subnetze, die zum Einrichten Ihrer VPC-Konfiguration CodeBuild verwendet werden. Beachten Sie, dass Flotten mit reservierter Kapazität nur ein Subnetz in einer einzigen Availablity Zone unterstützen. Stellen Sie außerdem sicher, dass Ihre Subnetze ein NAT-Gateway enthalten.

Sicherheitsgruppen

Die VPC-Sicherheitsgruppen, die mit Ihrer VPC CodeBuild verwendet werden. Stellen Sie sicher, dass Ihre Sicherheitsgruppen ausgehende Verbindungen zulassen.

Rolle im Flottenservice

Definiert die Servicerolle für Ihre Flotte.

Neue Servicerolle

Erstellt eine neue Servicerolle in Ihrem Konto.

Bestehende Servicerolle

Wählt eine bestehende Servicerolle aus Ihrem Konto aus.

Beispiele für reservierte Kapazität mit AWS CodeBuild

Diese Beispiele können verwendet werden, um mit Flotten mit reservierter Kapazität zu experimentieren. CodeBuild

Themen

Zwischenspeichern einer Stichprobe mit reservierter Kapazität

In einem Cache können wiederverwendbare Teile Ihrer Build-Umgebung gespeichert werden, die dann für mehrere Builds verwendet werden können. In diesem Beispiel wurde gezeigt, wie Sie das Caching innerhalb Ihres Build-Projekts mithilfe reservierter Kapazität aktivieren können. Weitere Informationen finden Sie unter Build-Caching in AWS CodeBuild.

Sie können damit beginnen, einen oder mehrere Cache-Modi in Ihren Projekteinstellungen anzugeben:

```
Cache:
        Type: LOCAL
        Modes:
          - LOCAL_CUSTOM_CACHE
          - LOCAL_DOCKER_LAYER_CACHE
          - LOCAL_SOURCE_CACHE
```

> **Note**
>
> Stellen Sie sicher, dass der privilegierte Modus aktiviert ist, um den Docker-Layer-Cache verwenden zu können.

Die Buildspec-Einstellungen Ihres Projekts sollten wie folgt aussehen:

```
version: 0.2
      phases:
        build:
          commands:
            - echo testing local source cache
            - touch /codebuild/cache/workspace/foobar.txt
```

```
            - git checkout -b cached_branch
            - echo testing local docker layer cache
            - docker run alpine:3.14 2>&1 | grep 'Pulling from' || exit 1
            - echo testing local custom cache
            - touch foo
            - mkdir bar && ln -s foo bar/foo2
            - mkdir bar/bar && touch bar/bar/foo3 && touch bar/bar/foo4
            - "[ -f foo ] || exit 1"
            - "[ -L bar/foo2 ] || exit 1"
            - "[ -f bar/bar/foo3 ] || exit 1"
            - "[ -f bar/bar/foo4 ] || exit 1"
      cache:
        paths:
           - './foo'
           - './bar/**/*'
           - './bar/bar/foo3'
```

Sie können damit beginnen, einen Build mit dem neuen Projekt auszuführen, um den Cache zu laden. Sobald dies abgeschlossen ist, sollten Sie einen weiteren Build mit einer übergeordneten Buildspezifikation starten, ähnlich der folgenden:

```
version: 0.2
      phases:
        build:
          commands:
            - echo testing local source cache
            - git branch | if grep 'cached_branch'; then (exit 0); else (exit 1); fi
            - ls /codebuild/cache/workspace | if grep 'foobar.txt'; then (exit 0); else
 (exit 1); fi
            - echo testing local docker layer cache
            - docker run alpine:3.14 2>&1 | if grep 'Pulling from'; then (exit 1); else
 (exit 0); fi
            - echo testing local custom cache
            - "[ -f foo ] || exit 1"
            - "[ -L bar/foo2 ] || exit 1"
            - "[ -f bar/bar/foo3 ] || exit 1"
            - "[ -f bar/bar/foo4 ] || exit 1"
      cache:
        paths:
           - './foo'
           - './bar/**/*'
           - './bar/bar/foo3'
```

Beschränkungen von Flotten mit reservierter Kapazität

Es gibt einige Anwendungsfälle, die von Flotten mit reservierter Kapazität nicht unterstützt werden. Wenn sie Sie betreffen, sollten Sie stattdessen Flotten auf Abruf verwenden:

- Flotten mit reservierter Kapazität unterstützen keine Batch-Builds, Build-Nutzungsmetriken oder semantische Versionierung.
- Flotten mit reservierter Kapazität unterstützen keine VPC-Konnektivität für Windows Server 2019 oder Windows Server 2022.

Weitere Informationen zu Grenzwerten und Kontingenten finden Sie unter. Computerflotten

Arbeiten mit Testberichten in AWS CodeBuild

Sie können darin Berichte erstellen CodeBuild , die Details zu Tests enthalten, die während der Builds ausgeführt werden. Sie können Tests wie Komponententests, Konfigurationstests und Funktionstests erstellen.

Die folgenden Dateiformate für Testberichte werden unterstützt:

- Gurke JSON (.json)
- Einheits-XML (.xml)
- Einheit XML (.xml)
- NUnit3-XML (.xml)
- TestNG XML (.xml)
- Visual Studio TRX (.trx)
- Visual Studio TRX XML (.xml)

> **Note**
>
> Die neueste unterstützte Version von `cucumber-js` ist 7.3.2.

Erstellen Sie Ihre Testfälle mit einem beliebigen Test-Framework, das Testberichtdateien in einem dieser Formate erstellen kann (zum Beispiel Surefire JUnit-Plugin, TestNG oder Cucumber).

Um einen Testbericht zu erstellen, fügen Sie der buildspec-Datei eines Build-Projekts einen Berichtsgruppennamen mit Informationen zu Ihren Testfällen hinzu. Wenn Sie das Build-Projekt ausführen, werden die Testfälle ausgeführt und ein Testbericht erstellt. Sie müssen keine Berichtsgruppe erstellen, bevor Sie die Tests ausführen. Wenn Sie einen Berichtsgruppennamen angeben, CodeBuild wird beim Ausführen Ihrer Berichte eine Berichtsgruppe für Sie erstellt. Wenn Sie eine bereits vorhandene Berichtsgruppe verwenden möchten, geben Sie deren ARN in der buildspec-Datei an.

Sie können einen Testbericht verwenden, um ein Problem während einer Build-Ausführung zu beheben. Wenn Sie viele Testberichte aus mehreren Builds eines Build-Projekts haben, können Sie Ihre Testberichte verwenden, um Trends und Test- und Ausfallraten anzuzeigen, um Builds zu optimieren.

Ein Bericht läuft 30 Tage nach seiner Erstellung ab. Sie können einen abgelaufenen Testbericht nicht anzeigen. Wenn Sie Testberichte länger als 30 Tage aufbewahren möchten, können Sie die Rohdatendateien Ihrer Testergebnisse in einen Amazon S3 S3-Bucket exportieren. Exportierte Testdateien laufen nicht ab. Informationen zum S3-Bucket werden beim Erstellen der Berichtsgruppe angegeben.

> **Note**
>
> Die im Projekt angegebene CodeBuild Servicerolle wird für Berechtigungen zum Hochladen in den S3-Bucket verwendet.

Themen

Erstellen eines Testberichts

Um einen Testbericht zu erstellen, führen Sie ein Build-Projekt aus, das mit einer bis fünf Berichtsgruppen in der buildspec-Datei konfiguriert ist. Während des Laufs wird ein Testbericht erstellt. Dieser enthält die Ergebnisse der Testfälle, die für die Berichtsgruppen angegeben sind. Für jeden nachfolgenden Build, der dieselbe buildspec-Datei verwendet, wird ein neuer Testbericht generiert.

So erstellen Sie einen Testbericht:

1. Erstellen Sie ein Build-Projekt. Weitere Informationen finden Sie unter Erstellen eines Build-Projekts in AWS CodeBuild.
2. Konfigurieren Sie die buildspec-Datei Ihres Projekts mit Testberichtinformationen:

a. Fügen Sie einen `reports:` Abschnitt hinzu und geben Sie entweder den ARN einer vorhandenen Berichtsgruppe oder den Namen einer Berichtsgruppe an.

   Wenn Sie einen ARN angeben, CodeBuild verwendet diese Berichtsgruppe.

   Wenn Sie einen Namen angeben, CodeBuild wird eine Berichtsgruppe für Sie erstellt, die Ihren Projektnamen und den von Ihnen angegebenen Namen im Format <project-name>- *< report-group-name >* verwendet. Wenn die benannte Berichtsgruppe bereits vorhanden ist, CodeBuild wird diese Berichtsgruppe verwendet.

b. Geben Sie unter der Berichtsgruppe den Speicherort der Dateien an, die die Testergebnisse enthalten. Wenn Sie mehr als eine Berichtsgruppe verwenden, geben Sie die Speicherorte der Testergebnisdatei für jede Berichtsgruppe an. Jedes Mal, wenn Ihr Build-Projekt ausgeführt wird, wird ein neuer Testbericht erstellt. Weitere Informationen finden Sie unter Angeben der Testdateien.

c. Geben Sie im `commands`-Abschnitt der Sequenz `build` oder `post_build` die Befehle an, mit denen die Testfälle ausgeführt werden, die Sie für Ihre Berichtsgruppen angegeben haben. Weitere Informationen finden Sie unter Angeben der Testbefehle .

Im Folgenden finden Sie ein Beispiel für einen Buildspec-Abschnitt`reports`:

```
reports:
  php-reports:
    files:
      - "reports/php/*.xml"
    file-format: "JUNITXML"
  nunit-reports:
    files:
      - "reports/nunit/*.xml"
    file-format: "NUNITXML"
```

3. Führen Sie einen Build des Build-Projekts aus. Weitere Informationen finden Sie unter Ausführen eines Build in AWS CodeBuild.

4. Wenn der Build abgeschlossen ist, wählen Sie den neuen Build Run unter Build history (Build-Verlauf) auf Ihrer Projektseite aus. Wählen Sie Reports (Berichte), um den Testbericht anzuzeigen. Weitere Informationen finden Sie unter Anzeigen von Testberichten für einen Build.

Arbeiten mit Berichtsgruppen

Eine Berichtsgruppe enthält Testberichte und gibt gemeinsame Einstellungen an. Mit der buildspec-Datei geben Sie die Testfälle an, die ausgeführt werden sollen, und die Befehle, die beim Erstellen auszuführen sind. Für jede Berichtsgruppe, die in einem Build-Projekt konfiguriert ist, erstellt ein Lauf des Build-Projekts einen Testbericht. Mehrere Läufe eines Build-Projekts, das mit einer Berichtsgruppe konfiguriert wurde, erstellen mehrere Testberichte in dieser Berichtsgruppe, wobei jeweils Ergebnisse der gleichen Testfälle enthalten sind, die für diese Berichtsgruppe angegeben wurden.

Die Testfälle werden für eine Berichtsgruppe in der buildspec-Datei eines Build-Projekts angegeben. Sie können bis zu fünf Berichtsgruppen in einem Build-Projekt angeben. Wenn Sie einen Build ausführen, werden alle Testfälle ausgeführt. Ein neuer Testbericht wird mit den Ergebnissen jedes Testfalls erstellt, der für eine Berichtsgruppe angegeben ist. Jedes Mal, wenn Sie einen neuen Build ausführen, werden die Testfälle ausgeführt und ein neuer Testbericht mit den neuen Testergebnissen erstellt.

Berichtsgruppen können in mehr als einem Build-Projekt verwendet werden. Alle Testberichte, die mit einer Berichtsgruppe erstellt wurden, haben dieselbe Konfiguration, wie die Exportoption und die Berechtigungen, auch wenn die Testberichte mit unterschiedlichen Build-Projekten erstellt werden. Testberichte, die mit einer Berichtsgruppe in mehreren Build-Projekten erstellt wurden, können die aus Ausführungen verschiedener Sätze von Testfällen enthalten (ein Satz von Testfällen für jedes Build-Projekt). Dies liegt daran, dass Sie verschiedene Testfalldateien für die Berichtsgruppe in der buildspec-Datei jedes Projekts angeben können. Sie können die Testfalldateien für eine Berichtsgruppe in einem Build-Projekt auch ändern, indem Sie die buildspec-Datei bearbeiten. Nachfolgende Build-Durchläufe erstellen neue Testberichte, die die Ergebnisse der Testfalldateien in der aktualisierten buildspec-Datei enthalten.

Themen

Erstellen einer Berichtsgruppe

Sie können die CodeBuild Konsole, die oder eine Buildspec-Datei verwenden AWS CLI, um eine Berichtsgruppe zu erstellen. Ihre IAM-Rolle muss über die Berechtigungen verfügen, die zum Erstellen einer Berichtsgruppe erforderlich sind. Weitere Informationen finden Sie unter Arbeiten mit Testberichtberechtigungen.

Themen

Erstellen einer Berichtsgruppe (Buildspec)

Eine mit der buildspec-Datei erstellte Berichtsgruppe exportiert keine rohen Testergebnisdateien. Sie können die Berichtsgruppe anzeigen und Exporteinstellungen festlegen. Weitere Informationen finden Sie unter Aktualisieren einer Berichtsgruppe.

So erstellen Sie eine Berichtsgruppe mit einer buildspec-Datei:

1. Wählen Sie einen Berichtsgruppennamen, der keiner Berichtsgruppe in Ihrem Konto zugeordnet ist. AWS
2. Konfigurieren Sie den `reports`-Abschnitt der buildspec-Datei mit diesem Namen. In diesem Beispiel ist der Name der Berichtsgruppe `new-report-group` und die Verwendungstestfälle werden mit dem JUnit-Framework erstellt:

   ```
   reports:
    new-report-group: #surefire junit reports
      files:
        - '**/*'
      base-directory: 'surefire/target/surefire-reports'
   ```

   Der Name der Berichtsgruppe kann auch mithilfe von Umgebungsvariablen in der Buildspec angegeben werden:

   ```
   version: 0.2
   ```

```
env:
  variables:
    REPORT_GROUP_NAME: "new-report-group"
phases:
  build:
    commands:
      - ...
...
reports:
 $REPORT_GROUP_NAME:
   files:
     - '**/*'
   base-directory: 'surefire/target/surefire-reports'
```

Weitere Informationen finden Sie unter Angeben der Testdateien und Reports syntax in the buildspec file.

3. Geben Sie im `commands`-Abschnitt den Befehl zum Ausführen der Tests an. Weitere Informationen finden Sie unter Angeben der Testbefehle .
4. Führen Sie den Build aus. Wenn der Build abgeschlossen ist, wird eine neue Berichtsgruppe mit einem Namen erstellt, der das Format `project-name-report-group-name` verwendet. Weitere Informationen finden Sie unter Benennung von Berichtsgruppen.

Erstellen einer Berichtsgruppe (Konsole)

So erstellen Sie einen Testbericht:

1. Öffnen SieAWS CodeBuildconsole (console)https://console.aws.amazon.com/codesuite/codebuild/homeaus.
2. Wählen Sie im Navigationsbereich Report Groups (Berichtsgruppen) aus.
3. Wählen Sie Create report group (Berichtsgruppe erstellen) aus.
4. Geben Sie unter Report group name (Berichtsgruppenname) einen Namen für die Berichtsgruppe ein.
5. (Optional) Geben Sie unter Tags Namen und Werte für Tags ein, die in unterstützten AWS-Services verwendet werden sollen. Verwenden Sie Add row, um ein Tag hinzuzufügen. Sie können bis zu 50 Tags hinzufügen.
6. Wenn Sie die Rohdaten Ihrer Testberichtsergebnisse in einen Amazon S3 S3-Bucket hochladen möchten:

a. SelectExportieren nach Amazon S3aus.

b. Geben Sie für S3 bucket name (S3-Bucket-Name) den Namen des S3-Buckets ein.

c. (Optional) FürS3-Bucket-Eigentümerden Wert ein.AWS-Kontokennung des Kontos, dem der S3-Bucket gehört. Auf diese Weise können Berichtsdaten in einen Amazon S3-Bucket exportiert werden, der einem anderen Konto als dem Konto gehört, das den Build ausführt.

d. Geben Sie unter Path prefix (Pfad-Präfix) den Pfad zu dem S3-Bucket ein, in den Sie die Testergebnisse hochladen möchten.

e. Wählen Sie Compress test result data in a zip file (Testergebnisdaten in einer ZIP-Datei komprimieren) aus, um die Testergebnisdatendateien zu komprimieren.

f. Erweitern Sie Additional configuration (Zusätzliche Konfiguration), um Verschlüsselungsoptionen anzuzeigen. Wählen Sie eine der folgenden Optionen aus:

   - StandardAWS-verwalteter Schlüsselum eine zu benutzenVon AWS verwalteter Schlüsselfür Amazon S3. Weitere Informationen finden Sie unter Kundenverwaltete CMKs im AWS Key Management Service-Benutzerhandbuch. Dies ist die Standardverschlüsselungsoption.
   - Wählen Sie einen benutzerdefinierten Schlüssel ausum einen vom Kunden verwalteten Schlüssel zu verwenden, den Sie erstellen und konfigurieren. Geben Sie für AWS KMS Encryption key (Verschlüsselungsschlüssel) den ARN Ihres Verschlüsselungsschlüssels ein. Das Format ist `arn:aws:kms:<region-id>: <aws-account-id>:key/<key-id>` . Weitere Informationen finden Sie unter Erstellen von KMS-Schlüsseln im AWS Key Management Service-Benutzerhandbuch.
   - Deaktivieren Sie die Artefaktverschlüsselung, um die Verschlüsselung zu deaktivieren. Sie können diese Option wählen, wenn Sie Ihre Testergebnisse freigeben oder auf einer statischen Website veröffentlichen möchten. (Eine dynamische Website kann Code ausführen, um Testergebnisse zu entschlüsseln.)

   Weitere Informationen zur Verschlüsselung von Daten im Ruhezustand finden Sie unter Datenverschlüsselung.

> **Note**
>
> Die im Projekt angegebene CodeBuild-Dienstrolle wird für Berechtigungen zum Hochladen in den S3-Bucket verwendet.

7. Wählen Sie Create report group (Berichtsgruppe erstellen) aus.

Erstellen einer Berichtsgruppe (CLI)

So erstellen Sie eine Berichtsgruppe

1. Erstellen Sie eine Datei namens `CreateReportGroup.json`.
2. Kopieren Sie je nach Ihren Anforderungen eines der folgenden JSON-Code-Snippets in `CreateReportGroup.json`:

   - Verwenden Sie den folgenden JSON, um anzugeben, dass Ihre Testberichtsgruppe Roh-Testergebnisdateien in einen Amazon S3 S3-Bucket exportiert.

```
{
  "name": "<report-name>",
  "type": "TEST",
  "exportConfig": {
    "exportConfigType": "S3",
    "s3Destination": {
      "bucket": "<bucket-name>",
      "bucketOwner": "<bucket-owner>",
      "path": "<path>",
      "packaging": "NONE | ZIP",
      "encryptionDisabled": "false",
      "encryptionKey": "<your-key>"
    },
    "tags": [
      {
        "key": "tag-key",
        "value": "tag-value"
      }
    ]
  }
}
```

- Ersetzen *<bucket-name>* durch Ihren Amazon S3 S3-Bucket-Namen und *<path>* durch den Pfad in Ihrem Bucket, zu dem Sie die Dateien exportieren möchten.
- Wenn Sie die exportierten Dateien komprimieren möchten, geben Sie für `packaging` `ZIP` an. Andernfalls geben Sie `NONE` an.
- `bucketOwner`ist optional und ist nur erforderlich, wenn der Amazon S3 S3-Bucket einem anderen Konto als dem Konto gehört, das den Build ausführt.
- Geben Sie `encryptionDisabled` an, ob die exportierten Dateien verschlüsselt werden sollen. Wenn Sie die exportierten Dateien verschlüsseln, geben Sie Ihren vom Kunden verwalteten Schlüssel ein. Weitere Informationen finden Sie unter Aktualisieren einer Berichtsgruppe.

- Verwenden Sie den folgenden JSON, um anzugeben, dass der Testbericht keine rohen Testdateien exportiert:

```
{
  "name": "<report-name>",
  "type": "TEST",
  "exportConfig": {
    "exportConfigType": "NO_EXPORT"
  }
}
```

> **Note**
>
> Die im Projekt angegebene CodeBuild--Servicerolle wird für Berechtigungen zum Hochladen zum S3-Bucket verwendet.

3. Führen Sie den Befehl aus:

```
aws codebuild create-report-group --cli-input-json file://
CreateReportGroupInput.json
```

Erstellen einer Berichtsgruppe (AWS CloudFormation)

So erstellen Sie einen Testbericht mit der AWS CloudFormation -Vorlage:

Sie können eine AWS CloudFormation Vorlagendatei verwenden, um eine Berichtsgruppe zu erstellen und bereitzustellen. Weitere Informationen finden Sie im AWS CloudFormation - Benutzerhandbuch.

Die folgende AWS CloudFormation YAML-Vorlage erstellt eine Berichtsgruppe, die keine rohen Testergebnisdateien exportiert.

```
Resources:
  CodeBuildReportGroup:
    Type: AWS::CodeBuild::ReportGroup
    Properties:
      Name: my-report-group-name
      Type: TEST
      ExportConfig:
        ExportConfigType: NO_EXPORT
```

Die folgende AWS CloudFormation YAML-Vorlage erstellt eine Berichtsgruppe, die rohe Testergebnisdateien in einen Amazon S3 S3-Bucket exportiert.

```
Resources:
  CodeBuildReportGroup:
    Type: AWS::CodeBuild::ReportGroup
    Properties:
      Name: my-report-group-name
      Type: TEST
      ExportConfig:
        ExportConfigType: S3
        S3Destination:
          Bucket: my-s3-bucket-name
          Path: path-to-folder-for-exported-files
          Packaging: ZIP
          EncryptionKey: my-KMS-encryption-key
          EncryptionDisabled: false
```

> **Note**
>
> Die im Projekt angegebene CodeBuild Servicerolle wird für Berechtigungen zum Hochladen in den S3-Bucket verwendet.

Aktualisieren einer Berichtsgruppe

Wenn Sie eine Berichtsgruppe aktualisieren, können Sie Informationen darüber angeben, ob die rohen Testergebnisdaten in Dateien in einem Amazon S3 S3-Bucket exportiert werden sollen. Wenn Sie in einen S3-Bucket exportieren möchten, können Sie beim Erstellen der Berichtsgruppe Folgendes angeben:

- Ob die rohen Testergebnisdateien in einer ZIP-Datei komprimiert werden.
- Ob die rohen Testergebnisdateien verschlüsselt sind. Sie können die Verschlüsselung mit einer der folgenden Optionen angeben:
  - Und Von AWS verwalteter Schlüssel für Amazon S3.
  - Ein vom Kunden verwalteter Schlüssel, den Sie erstellen und konfigurieren.

Weitere Informationen finden Sie unter Datenverschlüsselung.

Wenn Sie die verwenden AWS CLI , um eine Berichtsgruppe zu aktualisieren, können Sie auch Tags aktualisieren oder hinzufügen. Weitere Informationen finden Sie unter Markieren von Berichtsgruppen in AWS CodeBuild.

> **Note**
>
> Die im Projekt angegebene CodeBuild Servicerolle wird für Berechtigungen zum Hochladen in den S3-Bucket verwendet.

Themen

- Aktualisieren einer Berichtsgruppe (Konsole)
- Aktualisieren einer Berichtsgruppe (CLI)

Aktualisieren einer Berichtsgruppe (Konsole)

So aktualisieren Sie eine Berichtsgruppe:

1. Öffnen Sie die AWS CodeBuild Konsole unter https://console.aws.amazon.com/codesuite/codebuild/home.
2. Wählen Sie im Navigationsbereich Report Groups (Berichtsgruppen) aus.

3. Wählen Sie die Berichtsgruppe aus, die Sie aktualisieren möchten.
4. Wählen Sie Bearbeiten aus.
5. Wählen oder deaktivieren Sie Backup to Amazon S3. Wenn Sie diese Option ausgewählt haben, geben Sie die Exporteinstellungen an:

   a. Geben Sie für S3 bucket name (S3-Bucket-Name) den Namen des S3-Buckets ein.

   b. Geben Sie unter Path prefix (Pfad-Präfix) den Pfad zu dem S3-Bucket ein, in den Sie die Testergebnisse hochladen möchten.

   c. Wählen Sie Compress test result data in a zip file (Testergebnisdaten in einer ZIP-Datei komprimieren) aus, um die Testergebnisdatendateien zu komprimieren.

   d. Erweitern Sie Additional configuration (Zusätzliche Konfiguration), um Verschlüsselungsoptionen anzuzeigen. Wählen Sie eine der folgenden Optionen aus:

      - AWS Verwalteter Standardschlüssel zur Verwendung von a Von AWS verwalteter Schlüssel für Amazon S3. Weitere Informationen finden Sie unter Kundenverwaltete CMKs im AWS Key Management Service -Benutzerhandbuch. Dies ist die Standardverschlüsselungsoption.
      - Wählen Sie einen benutzerdefinierten Schlüssel, um einen vom Kunden verwalteten Schlüssel zu verwenden, den Sie erstellen und konfigurieren. Geben Sie für AWS KMS Encryption key (Verschlüsselungsschlüssel) den ARN Ihres Verschlüsselungsschlüssels ein. Das Format ist `arn:aws:kms:<region-id>: <aws-account-id>:key/<key-id>` . Weitere Informationen finden Sie unter Erstellen von KMS-Schlüsseln im AWS Key Management Service -Benutzerhandbuch.
      - Deaktivieren Sie die Artefaktverschlüsselung, um die Verschlüsselung zu deaktivieren. Sie können diese Option wählen, wenn Sie Ihre Testergebnisse freigeben oder auf einer statischen Website veröffentlichen möchten. (Eine dynamische Website kann Code ausführen, um Testergebnisse zu entschlüsseln.)

Aktualisieren einer Berichtsgruppe (CLI)

So aktualisieren Sie eine Berichtsgruppe:

1. Erstellen Sie eine Datei namens `UpdateReportGroupInput.json`.
2. Kopieren Sie Folgendes in `UpdateReportGroupInput.json`:

```
{
```

```
    "arn": "",
    "exportConfig": {
        "exportConfigType": "S3",
        "s3Destination": {
            "bucket": "bucket-name",
            "path": "path",
            "packaging": "NONE | ZIP",
            "encryptionDisabled": "false",
            "encryptionKey": "your-key"
         }
     },
     "tags": [
        {
            "key": "tag-key",
            "value": "tag-value"
        }
     ]
}
```

3. Geben Sie den ARN Ihrer Berichtsgruppe in die `arn`-Zeile ein (z. B. `"arn":"arn:aws:codebuild:region:123456789012:report-group/report-group-1"`)
4. Aktualisieren Sie `UpdateReportGroupInput.json` mit den Aktualisierungen, die Sie auf Ihre Berichtsgruppe anwenden möchten.

   - Wenn Sie Ihre Berichtsgruppe aktualisieren möchten, um rohe Testergebnisdateien in einen S3-Bucket zu exportieren, aktualisieren Sie den `exportConfig`-Abschnitt. Ersetzen Sie `bucket-name` durch Ihren S3-Bucket-Namen und `path` durch den Pfad in Ihrem S3-Bucket, zu dem Sie die Dateien exportieren möchten. Wenn Sie die exportierten Dateien komprimieren möchten, geben Sie für `packaging` `ZIP` an. Andernfalls geben Sie `NONE` an. Geben Sie `encryptionDisabled` an, ob die exportierten Dateien verschlüsselt werden sollen. Wenn Sie die exportierten Dateien verschlüsseln, geben Sie Ihren vom Kunden verwalteten Schlüssel ein.
   - Wenn Sie Ihre Berichtsgruppe so aktualisieren möchten, dass keine rohen Testergebnisdateien in einen S3-Bucket exportiert werden, aktualisieren Sie den `exportConfig`-Abschnitt mit folgendem JSON:

     ```
     {
       "exportConfig": {
           "exportConfigType": "NO_EXPORT"
     ```

```
  }
}
```

- Wenn Sie die Tags der Berichtsgruppe aktualisieren möchten, aktualisieren Sie den `tags`-Abschnitt. Sie können Tags ändern, hinzufügen oder entfernen. Wenn Sie alle Tags entfernen möchten, aktualisieren Sie sie mit dem folgenden JSON:

```
"tags": []
```

5. Führen Sie den folgenden Befehl aus:

```
aws codebuild update-report-group \
--cli-input-json file://UpdateReportGroupInput.json
```

Angeben der Testdateien

Sie geben die Testergebnisdateien und deren Speicherort für jede Berichtsgruppe im `reports`-Abschnitt der buildspec-Datei Ihres Build-Projekts an. Weitere Informationen finden Sie unter Reports syntax in the buildspec file.

Im Folgenden finden Sie einen `reports`-Beispielabschnitt, der zwei Berichtsgruppen für ein Build-Projekt angibt. Einer wird mit seinem ARN angegeben, der andere mit einem Namen. Der `files`-Abschnitt gibt die Dateien an, die die Testfallergebnisse enthalten. Der optionale `base-directory`-Abschnitt gibt das Verzeichnis an, in dem sich die Testfalldateien befinden. Der optionale `discard-paths` Abschnitt gibt an, ob Pfade zu Testergebnisdateien, die in einen Amazon S3 S3-Bucket hochgeladen wurden, verworfen werden.

```
reports:
  arn:aws:codebuild:your-region:your-aws-account-id:report-group/report-group-name-1:
 #surefire junit reports
    files:
      - '**/*'
    base-directory: 'surefire/target/surefire-reports'
    discard-paths: false

  sampleReportGroup: #Cucumber reports from json plugin
    files:
      - 'cucumber-json/target/cucumber-json-report.json'
    file-format: CUCUMBERJSON #Type of the report, defaults to JUNITXML
```

Angeben der Testbefehle

Die Befehle, die Ihre Testfälle ausführen, geben Sie im `commands`-Abschnitt Ihrer buildspec-Datei an. Mit diesen Befehlen werden die Testfälle ausgeführt, die für Ihre Berichtsgruppen im `reports`-Abschnitt Ihrer buildspec-Datei angegeben sind. Im Folgenden finden Sie einen `commands`-Beispielabschnitt, der Befehle zum Ausführen der Tests in Testdateien enthält:

```
commands:
    - echo Running tests for surefire junit
    - mvn test -f surefire/pom.xml -fn
    - echo
    - echo Running tests for cucumber with json plugin
    - mvn test -Dcucumber.options="--plugin json:target/cucumber-json-report.json" -f
 cucumber-json/pom.xml -fn
```

Weitere Informationen finden Sie unter Syntax der Build-Spezifikation.

Benennung von Berichtsgruppen

Wenn Sie die AWS CLI oder die AWS CodeBuild Konsole verwenden, um eine Berichtsgruppe zu erstellen, geben Sie einen Namen für die Berichtsgruppe an. Wenn Sie die Build-Spezifikationsdatei verwenden, um eine neue Berichtsgruppe zu erstellen, wird sie im Format *`project-name`*`-`*`report-group-name-specified-in-buildspec`* benannt. Alle Berichte, die durch Ausführen von Builds dieses Build-Projekts erstellt wurden, gehören zur neuen Berichtsgruppe mit dem neuen Namen.

Wenn Sie keine neue Berichtsgruppe erstellen CodeBuild möchten, geben Sie den ARN der Berichtsgruppe in der Buildspec-Datei eines Buildprojekts an. Sie können den ARN einer Berichtsgruppe in mehreren Build-Projekten angeben. Nach der Ausführung jedes Build-Projekts enthält die Berichtsgruppe Testberichte, die von jedem Build-Projekt erstellt wurden.

Wenn Sie beispielsweise eine Berichtsgruppe mit dem Namen `my-report-group` erstellen und dann ihren Namen in zwei verschiedenen Build-Projekten verwenden, die als `my-project-1` und `my-project-2` benannt sind, und einen Build beider Projekte erstellen, werden zwei neue Berichtsgruppen erstellt. Das Ergebnis sind drei Berichtsgruppen mit folgenden Namen:

- `my-report-group`: Hat keine Testberichte.
- `my-project-1-my-report-group`: Enthält Berichte mit Ergebnissen von Tests, die vom Build-Projekt mit dem Namen `my-project-1` ausgeführt werden.

- `my-project-2-my-report-group`: Enthält Berichte mit Ergebnissen von Tests, die vom Build-Projekt mit dem Namen `my-project-2` ausgeführt werden.

Wenn Sie den ARN der Berichtsgruppe mit dem Namen `my-report-group` in beiden Projekten verwenden und dann Builds für jedes Projekt ausführen, haben Sie immer noch eine Berichtsgruppe (`my-report-group`). Diese Berichtsgruppe enthält Testberichte mit Ergebnissen von Tests, die von beiden Build-Projekten ausgeführt werden.

Wenn Sie einen Berichtsgruppennamen auswählen, der nicht zu einer Berichtsgruppe in Ihrem AWS -Konto gehört, und diesen Namen dann für eine Berichtsgruppe in einer buildspec-Datei verwenden und einen Build des Build-Projekts ausführen, wird eine neue Berichtsgruppe erstellt. Das Format des Namens der neuen Berichtsgruppe lautet *`project-name-new-group-name`*. Wenn es in Ihrem AWS Konto beispielsweise keine Berichtsgruppe mit dem Namen gibt und geben Sie ihn in einem Build-Projekt mit dem Namen `new-report-group` an`test-project`, wird bei einem Buildlauf eine neue Berichtsgruppe mit dem Namen erstellt. `test-project-new-report-group`

Markieren von Berichtsgruppen in AWS CodeBuild

Ein Tag ist eine benutzerdefinierte Attributbezeichnung, die Sie oder AWS einer AWS-Ressource zuweisen. Jedes AWS-Tag besteht aus zwei Teilen:

- einem Tag-Schlüssel (z. B. `CostCenter`, `Environment`, `Project` oder `Secret`). Bei Tag-Schlüsseln wird zwischen Groß- und Kleinschreibung unterschieden.
- einem optionalen Feld, das als Tag-Wert bezeichnet wird (z. B. `111122223333`, `Production` oder ein Team-Name). Ein nicht angegebener Tag-Wert entspricht einer leeren Zeichenfolge. Wie bei Tag-Schlüsseln wird auch bei Tag-Werten zwischen Groß-/Kleinschreibung unterschieden.

Dies wird als Schlüssel-Wert-Paar bezeichnet. Informationen zu den Limits in Bezug auf die Anzahl der Tags für eine Berichtsgruppe und zu den Einschränkungen in Bezug auf Tag-Schlüssel und -Werte finden Sie unter Tags.

Tags helfen Ihnen, Ihre AWS-Ressourcen zu identifizieren und zu organisieren. Viele AWS-Services unterstützen das Markieren mit Tags (kurz: Tagging). So können Ressourcen aus verschiedenen Services dasselbe Tag zuweisen, um anzugeben, dass die Ressourcen verbunden sind. Sie können beispielsweise dasselbe Tag einer CodeBuild-Berichtsgruppe zuweisen, das Sie auch einem Amazon S3 S3-Bucket zuweisen. Weitere Informationen zur Verwendung von Tags finden Sie im Whitepaper Bewährte Methoden für die Markierung.

In CodeBuild sind die primären Ressourcen die Berichtsgruppe und das Projekt. Sie können die CodeBuild-Konsole verwenden, dieAWS CLI, CodeBuild-APIs oderAWSSDKs zum Hinzufügen, Verwalten und Entfernen von Tags für eine Berichtsgruppe. Zusätzlich zum Identifizieren, Organisieren und Nachverfolgen Ihrer Berichtsgruppe mit Tags können Sie Tags in IAM-Richtlinien verwenden, um zu steuern, wer Ihre Berichtsgruppe anzeigen und mit ihr interagieren kann. Beispiele für Tag-basierte Zugriffsrichtlinien finden Sie unter Verwenden von Tags zur Steuerung des Zugriffs auf AWS CodeBuild -Ressourcen.

Themen

Hinzufügen eines Tags zu einer Berichtsgruppe

Das Hinzufügen von Tags zu einer Berichtsgruppe kann Ihnen helfen, Ihre AWS-Ressourcen zu identifizieren und zu organisieren und den Zugriff auf sie zu verwalten. Fügen Sie zunächst ein oder mehrere Tags (Schlüssel-Wert-Paare) zu einer Berichtsgruppe hinzu. Denken Sie daran, dass es Limits in Bezug auf die Anzahl der Tags für eine Berichtsgruppe gibt. Es gibt Einschränkungen im Hinblick auf die Zeichen, die Sie in die Felder für Schlüssel und Wert eingeben können. Weitere Informationen finden Sie unter Tags. Nach der Erstellung der Tags können Sie IAM-Richtlinien für die Verwaltung des Zugriffs auf die Berichtsgruppe basierend auf diesen Tags erstellen. Sie können die CodeBuild-Konsole oder dasAWS CLIum einer Berichtsgruppe Tags hinzuzufügen.

> **Important**
>
> Das Hinzufügen von Tags zu einer Berichtsgruppe kann sich auf den Zugriff auf diese Berichtsgruppe auswirken. Bevor Sie ein Tag zu einer Berichtsgruppe hinzufügen, müssen Sie alle IAM-Richtlinien überprüfen, die möglicherweise Tags für die Steuerung des Zugriffs auf Ressourcen wie z. B. Berichtsgruppen verwenden. Beispiele für Tag-basierte Zugriffsrichtlinien finden Sie unter Verwenden von Tags zur Steuerung des Zugriffs auf AWS CodeBuild -Ressourcen.

Weitere Informationen zum Hinzufügen von Tags zu einer Berichtsgruppe während ihrer Erstellung finden Sie unter Erstellen einer Berichtsgruppe (Konsole).

Themen

Hinzufügen eines Tags zu einer Berichtsgruppe (Konsole)

Sie können die CodeBuild-Konsole verwenden, um einer CodeBuild-Berichtsgruppe ein oder mehrere Tags hinzuzufügen.

1. Öffnen Sie die CodeBuild-Konsole unter https://console.aws.amazon.com/codebuild/.
2. Wählen Sie in Report groups (Berichtsgruppen) den Namen der Berichtsgruppe aus, der Sie Tags hinzufügen möchten.
3. Klicken Sie im Navigationsbereich auf Settings.
4. Wenn der Berichtsgruppe noch keine Tags hinzugefügt wurden, wählen Sie Add tag (Tag hinzufügen) aus. Sie können auch Edit (Bearbeiten) und anschließend Add tag (Tag hinzufügen) auswählen.
5. Geben Sie für Key (Schlüssel) einen Namen für das Tag ein. Sie können einen optionalen Wert für das Tag unter Value (Wert) hinzufügen.
6. (Optional) Zum Hinzufügen eines weiteren Tags wählen Sie Add tag (Tag hinzufügen) erneut aus.
7. Wenn Sie mit dem Hinzufügen von Tags fertig sind, klicken Sie auf Submit (Übermitteln).

Hinzufügen eines Tags zu einer Berichtsgruppe (AWS CLI)

Informationen zum Hinzufügen eines Tags zu einer Berichtsgruppe bei ihrer Erstellung finden Sie unter Erstellen einer Berichtsgruppe (CLI). Fügen Sie Ihre Tags in `CreateReportGroup.json` hinzu.

Informationen zum Hinzufügen von Tags zu einer vorhandenen Berichtsgruppe finden Sie unter Aktualisieren einer Berichtsgruppe (CLI). Fügen Sie Ihre Tags in `UpdateReportGroupInput.json` hinzu.

Bei diesen Schritten wird davon ausgegangen, dass Sie bereits eine aktuelle Version der AWS CLI installiert oder eine Aktualisierung auf die aktuelle Version vorgenommen haben. Weitere Informationen finden Sie unter Installieren der AWS Command Line Interface.

Anzeigen von Tags für eine Berichtsgruppe

Tags helfen Ihnen, Ihre AWS-Ressourcen zu identifizieren und den Zugriff auf diese zu verwalten. Weitere Informationen zur Verwendung von Tags finden Sie im Whitepaper Bewährte Methoden für die Markierung. Beispiele für Tag-basierte Zugriffsrichtlinien finden Sie unter Deny or allow actions on report groups based on resource tags.

Anzeigen von Tags für eine Berichtsgruppe (Konsole)

Sie können die CodeBuild-Konsole verwenden, um die Tags anzuzeigen, die mit einer CodeBuild-Berichtsgruppe verknüpft sind.

1. Öffnen Sie die CodeBuild-Konsole unter https://console.aws.amazon.com/codebuild/.
2. Wählen Sie in Report groups (Berichtsgruppen) den Namen der Berichtsgruppe aus, in der Sie Tags anzeigen möchten.
3. Klicken Sie im Navigationsbereich auf Settings.

Anzeigen von Tags für eine Berichtsgruppe (AWS CLI)

Führen Sie die folgenden Schritte aus, um über die AWS CLI die AWS-Tags für eine Berichtsgruppe anzuzeigen. Wenn keine Tags hinzugefügt wurden, ist die Liste der zurückgegebenen Tags leer.

1. Verwenden Sie die Konsole oder die AWS CLI, um den ARN Ihrer Berichtsgruppe zu suchen. Notieren Sie diesen.

   AWS CLI

   Führen Sie den folgenden Befehl aus.

   ```
   aws list-report-groups
   ```

   Dieser Befehl gibt JSON-formatierte Informationen ähnlich den folgenden zurück:

   ```
   {
       "reportGroups": [
           "arn:aws:codebuild:region:123456789012:report-group/report-group-1",
           "arn:aws:codebuild:region:123456789012:report-group/report-group-2",
           "arn:aws:codebuild:region:123456789012:report-group/report-group-3"
       ]
   ```

```
}
```

Ein Berichtsgruppen-ARN endet mit dem Namen. Diesen können Sie verwenden, um den ARN für Ihre Berichtsgruppe zu identifizieren.

Console

1. Öffnen Sie die CodeBuild-Konsole unter https://console.aws.amazon.com/codebuild/.
2. Wählen Sie in Report groups (Berichtsgruppen) den Namen der Berichtsgruppe mit den Tags aus, die Sie anzeigen möchten.
3. Suchen Sie in Configuration (Konfiguration) den ARN Ihrer Berichtsgruppe.

2. Führen Sie den folgenden Befehl aus. Verwenden Sie den ARN, den Sie notiert haben, für den Parameter `--report-group-arns`.

```
aws codebuild batch-get-report-groups --report-group-arns
 arn:aws:codebuild:region:123456789012:report-group/report-group-name
```

Bei erfolgreicher Ausführung gibt dieser Befehl JSON-formatierte Informationen mit einem `tags`-Abschnitt ähnlich dem folgenden zurück:

```
{
    ...
    "tags": {
        "Status": "Secret",
        "Project": "TestBuild"
    }
    ...
}
```

Bearbeiten von Tags für eine Berichtsgruppe

Sie können den Wert für ein Tag ändern, das mit einer Berichtsgruppe verknüpft ist. Sie können auch den Namen des Schlüssels ändern. Dies entspricht dem Entfernen des aktuellen Tags und dem Hinzufügen eines anderen Tags mit dem neuen Namen und demselben Wert wie dem des anderen. Denken Sie an die Einschränkungen in Bezug auf die Zeichen, die Sie in den Schlüssel- und Wertfeldern verwenden können. Weitere Informationen finden Sie unter Tags.

> **Important**
>
> Das Bearbeiten von Tags für eine Berichtsgruppe kann sich auf den Zugriff auf diese Berichtsgruppe auswirken. Bevor Sie den Namen (Schlüssel) oder den Wert eines Tags für eine Berichtsgruppe bearbeiten, müssen Sie alle IAM-Richtlinien überprüfen, die den Schlüssel oder Wert eines Tags möglicherweise für die Steuerung des Zugriffs auf Ressourcen wie z. B. Berichtsgruppen verwenden. Beispiele für Tag-basierte Zugriffsrichtlinien finden Sie unter Deny or allow actions on report groups based on resource tags.

Bearbeiten eines Tags für eine Berichtsgruppe (Konsole)

Sie können die CodeBuild-Konsole verwenden, um die Tags zu bearbeiten, die mit einer CodeBuild-Berichtsgruppe verknüpft sind.

1. Öffnen Sie die CodeBuild-Konsole unter https://console.aws.amazon.com/codebuild/.
2. Wählen Sie in Report groups (Berichtsgruppen) den Namen der Berichtsgruppe aus, in der Sie Tags bearbeiten möchten.
3. Klicken Sie im Navigationsbereich auf Settings.
4. Wählen Sie Edit aus.
5. Gehen Sie folgendermaßen vor:
   - Zum Ändern des Tags geben Sie einen neuen Namen unter Key (Schlüssel) ein. Das Ändern des Namens eines Tags entspricht dem Entfernen eines Tags und Hinzufügen eines neuen Tags mit dem neuen Schlüsselnamen.
   - Geben Sie zum Ändern des Werts eines Tags einen neuen Wert ein. Wenn Sie den Wert in „kein“ ändern möchten, löschen Sie den aktuellen Wert und lassen das Feld leer.
6. Wenn Sie mit dem Bearbeiten der Tags fertig sind, wählen Sie Submit (Übermitteln) aus.

Bearbeiten von Tags für eine Berichtsgruppe (AWS CLI)

Informationen zum Hinzufügen, Ändern oder Löschen von Tags für eine Berichtsgruppe finden Sie unter Aktualisieren einer Berichtsgruppe (CLI). Aktualisieren Sie die Tags in `UpdateReportGroupInput.json`.

Entfernen eines Tags aus einer Berichtsgruppe

Sie können ein oder mehrere Tags entfernen, die mit einer Berichtsgruppe verknüpft sind. Das Entfernen eines Tags löscht nicht das Tag anderer AWS-Ressourcen, die mit diesem Tag verknüpft sind.

> **Important**
>
> Das Entfernen von Tags für eine Berichtsgruppe kann sich auf den Zugriff auf diese Berichtsgruppe auswirken. Bevor Sie ein Tag aus einer Berichtsgruppe entfernen, müssen Sie alle IAM-Richtlinien überprüfen, die den Schlüssel oder Wert eines Tags möglicherweise für die Steuerung des Zugriffs auf Ressourcen wie z. B. Berichtsgruppen verwenden. Beispiele für Tag-basierte Zugriffsrichtlinien finden Sie unter Verwenden von Tags zur Steuerung des Zugriffs auf AWS CodeBuild -Ressourcen.

Entfernen eines Tags aus einer Berichtsgruppe (Konsole)

Sie können die CodeBuild-Konsole verwenden, um die Verknüpfung zwischen einem Tag und einer CodeBuild-Berichtsgruppe zu entfernen.

1. Öffnen Sie die CodeBuild-Konsole unter https://console.aws.amazon.com/codebuild/.
2. Wählen Sie in Report groups (Berichtsgruppen) den Namen der Berichtgruppe aus, in der Sie Tags entfernen möchten.
3. Klicken Sie im Navigationsbereich auf Settings.
4. Wählen Sie Edit aus.
5. Suchen Sie die Tags, die Sie entfernen möchten, und wählen Sie dann Remove tag (Tag entfernen) aus.
6. Wenn Sie die Tags entfernt haben, klicken Sie auf Submit (Übermitteln).

Entfernen eines Tags aus einer Berichtsgruppe (AWS CLI)

Gehen Sie folgendermaßen vor, um dieAWS CLIum ein Tag aus einer CodeBuild-Berichtsgruppe zu entfernen. Durch das Entfernen wird ein Tag nicht gelöscht, sondern lediglich die Verknüpfung zwischen dem Tag und der Berichtsgruppe entfernt.

> **Note**
>
> Wenn Sie eine CodeBuild-Berichtsgruppe löschen, werden alle Tag-Zuordnungen aus der gelöschten Berichtsgruppe entfernt. Sie müssen vor dem Löschen einer Berichtsgruppe keine Tags entfernen.

Informationen zum Löschen eines oder mehrerer Tags aus einer Berichtsgruppe finden Sie unter Bearbeiten von Tags für eine Berichtsgruppe (AWS CLI). Aktualisieren Sie den `tags`-Abschnitt in den JSON-formatierten Daten mit einer aktualisierten Liste von Tags, die keine der Tags enthält, die Sie löschen möchten. Wenn Sie alle Tags löschen möchten, aktualisieren Sie den `tags`-Abschnitt zu:

```
"tags: []"
```

Arbeiten mit freigegebenen Berichtsgruppen

Mithilfe der Berichtsgruppenfreigabe können mehrere AWS-Konten oder Benutzer eine Berichtsgruppe, ihre nicht abgelaufenen Berichte und die Testergebnisse ihrer Berichte anzeigen. In diesem Modell verwendet das Konto, das die Berichtsgruppe besitzt (Eigentümer), eine Berichtsgruppe mit anderen Konten (Verbraucher). Ein Verbraucher kann keine Berichtsgruppe bearbeiten. Ein Bericht läuft 30 Tage nach seiner Erstellung ab.

Inhalt

Voraussetzungen für die Freigabe von Berichtsgruppen

Um eine Berichtsgruppe freizugeben, muss Ihr AWS-Konto Eigentümer dieser Gruppe sein. Sie können keine Berichtsgruppe freigeben, die für Sie freigegeben wurde.

Voraussetzungen für den Zugriff auf für Sie freigegebene Berichtsgruppen

Um auf eine freigegebene Berichtsgruppe zuzugreifen, benötigt die IAM-Rolle eines Verbrauchers die `BatchGetReportGroups`-Berechtigung. Sie können die folgende Richtlinie an ihre IAM-Rolle anfügen:

```
{
    "Effect": "Allow",
    "Resource": [
        "*"
    ],
    "Action": [
        "codebuild:BatchGetReportGroups"
    ]
}
```

Weitere Informationen finden Sie unter Verwendung identitätsbasierter Richtlinien für AWS CodeBuild.

Verwandte Dienstleistungen

Die Berichtsgruppenfreigabe wird mit AWS Resource Access Manager (AWS RAM) integriert, einem Service, der es Ihnen ermöglicht, Ihre AWS-Ressourcen für jedes AWS-Konto oder über AWS Organizations freizugeben. Mit AWS RAM können Sie Ressourcen, deren Eigentümer Sie sind, freigeben, indem Sie eine Ressourcenfreigabe erstellen, die die Ressourcen und die Verbraucher angibt, für die sie freigegeben werden sollen. Verbraucher können einzelne AWS-Konten oder Organisationseinheiten in AWS Organizations oder gesamte Organisationen in AWS Organizations sein.

Weitere Informationen finden Sie im AWS RAM-Leitfaden.

Freigeben einer Berichtsgruppe

Wenn Sie eine Berichtsgruppe freigeben, erhält der Verbraucher schreibgeschützten Zugriff auf die Berichtsgruppe und ihre Berichte. Der Verbraucher kann die AWS CLI verwenden, um die Berichtsgruppe, ihre Berichte und die Testfallergebnisse für jeden Bericht anzuzeigen. Der Verbraucher kann nicht:

- Eine freigegebene Berichtsgruppe oder ihre Berichte in der CodeBuild-Konsole anzeigen.
- Eine freigegebene Berichtsgruppe bearbeiten.

- Den ARN der freigegebenen Berichtsgruppe in einem Projekt verwenden, um einen Bericht auszuführen. Ein Projekt-Build, der eine freigegebene Berichtsgruppe angibt, schlägt fehl.

Sie können die CodeBuild-Konsole verwenden, um einer vorhandenen Ressourcenfreigabe eine Berichtsgruppe hinzuzufügen. Wenn Sie die Berichtsgruppe einer neuen Ressourcenfreigabe hinzufügen möchten, müssen Sie sie zuerst in der AWS RAM-Konsole erstellen.

Um eine Berichtsgruppe für Organisationseinheiten oder eine ganze Organisation freizugeben, müssen Sie die Freigabe für AWS Organizations aktivieren. Weitere Informationen finden Sie unter Freigabe für AWS Organizations aktivieren im AWS RAM-Benutzerhandbuch.

Sie können die CodeBuild-Konsole verwenden,AWS RAM-Konsole oderAWS CLISo geben Sie Berichtsgruppen frei, deren Eigentümer Sie sind.

So geben Sie eine Berichtsgruppe frei, deren Eigentümer Sie sind (CodeBuild-Konsole):

1. Öffnen SieAWS CodeBuildconsole beihttps://console.aws.amazon.com/codesuite/codebuild/homeaus.
2. Wählen Sie im Navigationsbereich Report Groups (Berichtsgruppen) aus.
3. Wählen Sie das Projekt aus, das Sie freigeben möchten, und klicken Sie dann auf Share (Freigeben). Weitere Informationen finden Sie unter Erstellen einer Ressourcenfreigabe im AWS RAM-Benutzerhandbuch.

So geben Sie Berichtsgruppen frei, deren Eigentümer Sie sind (AWS RAM-Konsole):

Siehe .Erstellen einer RessourcenfreigabeimAWS RAM-Benutzerhandbuchaus.

So geben Sie Berichtsgruppen frei, deren Eigentümer Sie sind (AWS RAM-Befehl):

Verwenden Sie den Befehl create-resource-share.

So geben Sie eine Berichtsgruppe frei, deren Eigentümer Sie sind (-Befehl CodeBuild)

Verwenden Sie den Befehl put-resource-policy :

1. Erstellen Sie eine Datei mit dem Namen `policy.json` und kopieren Sie Folgendes in diese Datei.

```
{
```

```
   "Version":"2012-10-17",
   "Statement":[{
     "Effect":"Allow",
     "Principal":{
       "AWS":"consumer-aws-account-id-or-user"
     },
     "Action":[
       "codebuild:BatchGetReportGroups",
       "codebuild:BatchGetReports",
       "codebuild:ListReportsForReportGroup",
       "codebuild:DescribeTestCases"],
     "Resource":"arn-of-report-group-to-share"
   }]
 }
```

2. Aktualisieren Sie `policy.json` mit dem ARN der Berichtsgruppe und Bezeichnern für die Freigabe. Im folgenden Beispiel wird schreibgeschützter Zugriff auf die Berichtsgruppe mit dem ARN gewährt`arn:aws:codebuild:us-west-2:123456789012:report-group/my-report-group`an Alice und den Root-Benutzer für dieAWSKonto identifiziert durch 123456789012.

```
{
   "Version":"2012-10-17",
   "Statement":[{
     "Effect":"Allow",
     "Principal":{
       "AWS": [
          "arn:aws:iam::123456789012:user/Alice",
          "123456789012"
        ]
     },
     "Action":[
       "codebuild:BatchGetReportGroups",
       "codebuild:BatchGetReports",
       "codebuild:ListReportsForReportGroup",
       "codebuild:DescribeTestCases"],
     "Resource":"arn:aws:codebuild:us-west-2:123456789012:report-group/my-report-group"
   }]
 }
```

3. Führen Sie den folgenden Befehl aus.

```
aws codebuild put-resource-policy --resource-arn report-group-arn --policy file://
policy.json
```

Aufheben der Freigabe einer freigegebenen Berichtsgruppe

Auf eine nicht mehr freigegebene Berichtsgruppe, einschließlich ihrer Berichte und ihrer Testergebnisse, kann nur der Besitzer zugreifen. Wenn Sie die Freigabe einer Berichtsgruppe aufheben, kann jedes AWS-Konto oder jeder Benutzer, für das/den die Freigabe zuvor galt, nicht mehr auf die Berichtsgruppe, ihre Berichte oder die Ergebnisse von Testfällen in den Berichten zugreifen.

Um die Freigabe eines freigegebenen Berichtsgruppe, deren Eigentümer Sie sind, aufzuheben, müssen Sie sie aus der Ressourcenfreigabe entfernen. Sie können dafür die AWS RAM-Konsole oder AWS CLI verwenden.

So heben Sie die Freigabe einer freigegebenen Berichtsgruppe auf, deren Eigentümer Sie sind (AWS RAM-Konsole):

Siehe Aktualisieren einer Ressourcenfreigabe im AWS RAM-Benutzerhandbuch.

So heben Sie die Freigabe einer freigegebenen Berichtsgruppe auf, deren Eigentümer Sie sind (AWS RAM-Befehl)

Verwenden Sie den Befehl disassociate-resource-share.

So heben Sie die Freigabe der Berichtsgruppe auf, deren Eigentümer CodeBuild -Befehl:

Führen Sie den Befehl delete-resource-policy aus und geben Sie den ARN der Berichtsgruppe an, deren Freigabe Sie aufheben möchten:

```
aws codebuild delete-resource-policy --resource-arn report-group-arn
```

Identifizieren einer freigegebenen Berichtsgruppe

Eigentümer und Verbraucher können freigegebene Berichtsgruppen mit AWS CLI identifizieren.

Verwenden Sie die folgenden Befehle, um eine freigegebene Berichtsgruppe zu identifizieren und Informationen dazu und zu ihren Berichten abzurufen:

- Um die ARNs der für Sie freigegebenen Berichtsgruppen anzuzeigen, führen Sie `list-shared-report-groups` aus:

```
aws codebuild list-shared-report-groups
```

- Um die ARNs der Berichte in einer Berichtsgruppe anzuzeigen, führen Sie `list-reports-for-report-group` mit dem ARN der Berichtsgruppe aus:

```
aws codebuild list-reports-for-report-group --report-group-arn report-group-arn
```

- Um Informationen zu Testfällen in einem Bericht anzuzeigen, führen Sie `describe-test-cases` mithilfe des ARN des Berichts aus:

```
aws codebuild describe-test-cases --report-arn report-arn
```

Die Ausgabe sollte wie folgt aussehen:

```
{
    "testCases": [
        {
            "status": "FAILED",
            "name": "Test case 1",
            "expired": 1575916770.0,
            "reportArn": "report-arn",
            "prefix": "Cucumber tests for agent",
            "message": "A test message",
            "durationInNanoSeconds": 1540540,
            "testRawDataPath": "path-to-output-report-files"
        },
        {
            "status": "SUCCEEDED",
            "name": "Test case 2",
            "expired": 1575916770.0,
            "reportArn": "report-arn",
            "prefix": "Cucumber tests for agent",
            "message": "A test message",
            "durationInNanoSeconds": 1540540,
            "testRawDataPath": "path-to-output-report-files"
        }
    ]
}
```

Berechtigungen für freigegebene Berichtsgruppen

Berechtigungen für Besitzer

Ein Besitzer der Berichtsgruppe kann die Berichtsgruppe bearbeiten und in einem Projekt angeben, um Berichte auszuführen.

Berechtigungen für Konsumenten

Ein Benutzer der Berichtsgruppe kann eine Berichtsgruppe, ihre Berichte und die Testfallergebnisse für die Berichte anzeigen. Ein Verbraucher kann eine Berichtsgruppe oder ihre Berichte nicht bearbeiten und sie nicht zum Erstellen von Berichten verwenden.

Arbeiten mit Berichten

Ein Bericht enthält die Ergebnisse von Testfällen, die für eine Berichtsgruppe angegeben sind. Ein Testbericht wird während der Ausführung eines Build-Projekts erstellt. Sie geben eine Berichtsgruppe, Testfalldateien und Befehle an, um die Testfälle in der buildspec-Datei auszuführen. Bei jeder Ausführung der Testfälle wird in der Berichtsgruppe ein neuer Testbericht erstellt.

Ein Testbericht läuft 30 Tage nach der Erstellung ab. Sie können einen abgelaufenen Testbericht nicht anzeigen, aber Sie können die Testergebnisse in rohe Testergebnisdateien in einem S3-Bucket exportieren. Exportierte Roh-Testdateien laufen nicht ab. Weitere Informationen finden Sie unter Aktualisieren einer Berichtsgruppe.

Der Status eines Testberichts kann einer der folgenden sein:

- `GENERATING`: Die Ausführung der Testfälle ist noch im Gange.
- `DELETING`: Der Testbericht wird gelöscht. Wenn ein Testbericht gelöscht wird, werden auch seine Testfälle gelöscht. Roh-Testergebnis-Datendateien, die in einen S3-Bucket exportiert werden, werden nicht gelöscht.
- `INCOMPLETE`: Der Testbericht wurde nicht abgeschlossen. Dieser Status kann aus einem der folgenden Gründe angezeigt werden:
  - Ein Problem mit der Konfiguration der Berichtsgruppe, die die Testfälle dieses Berichts angibt. Beispielsweise könnte der Pfad zu den Testfällen unter der Berichtsgruppe in der buildspec-Datei falsch sein.
  - Der IAM-Benutzer, der den Build ausgeführt hat, hat keine Berechtigungen zum Ausführen von Tests. Weitere Informationen finden Sie unter Arbeiten mit Testberichtberechtigungen.

  - Der Build wurde aufgrund eines Fehlers nicht abgeschlossen, der nicht mit den Tests zusammenhängt.
- `SUCCEEDED`: Alle Testfälle waren erfolgreich.
- `FAILED`: Einige der Testfälle waren nicht erfolgreich.

Jeder Testfall gibt einen Status zurück. Der Status für einen Testfall kann einer der folgenden sein:

- `SUCCEEDED`: Der Testfall wurde bestanden.
- `FAILED`: Der Testfall ist fehlgeschlagen.
- `ERROR`: Der Testfall führte zu einem unerwarteten Fehler.
- `SKIPPED`: Der Testfall wurde nicht ausgeführt.
- `UNKNOWN`: Der Testfall hat einen anderen Status als `SUCCEEDED`, `FAILED`, `ERROR` oder `SKIPPED` zurückgegeben.

Ein Testbericht kann maximal 500 Testfallergebnisse haben. Wenn mehr als 500 Testfälle ausgeführt werden, CodeBuild priorisiert die Tests anhand des Status `FAILED` und kürzt die Testfallergebnisse.

Arbeiten mit Testberichtberechtigungen

In diesem Thema werden wichtige Informationen zu Berechtigungen für Testberichte beschrieben.

Themen

Erstellen einer Rolle für Testberichte

Um einen Testbericht auszuführen und ein Projekt zu aktualisieren, um Testberichte einzuschließen, benötigt Ihre IAM-Rolle die folgenden Berechtigungen. Diese Berechtigungen sind in den vordefinierten verwalteten Richtlinien enthalten. AWS Wenn Sie Testberichte zu einem vorhandenen Build-Projekt hinzufügen möchten, müssen Sie diese Berechtigungen selbst hinzufügen.

- `CreateReportGroup`

- `CreateReport`
- `UpdateReport`
- `BatchPutTestCases`

Um einen Bericht über die Codeabdeckung zu erstellen, muss Ihre IAM-Rolle auch die `BatchPutCodeCoverages` entsprechende Berechtigung enthalten.

> **Note**
>
> `BatchPutTestCases`, `CreateReportUpdateReport`, und `BatchPutCodeCoverages` sind keine öffentlichen Berechtigungen. Für diese Berechtigungen können Sie keinen entsprechenden AWS CLI Befehl oder keine SDK-Methode aufrufen.

Um sicherzustellen, dass Sie über diese Berechtigungen verfügen, können Sie Ihrer IAM-Rolle die folgende Richtlinie hinzufügen:

```
{
    "Effect": "Allow",
    "Resource": [
        "*"
    ],
    "Action": [
        "codebuild:CreateReportGroup",
        "codebuild:CreateReport",
        "codebuild:UpdateReport",
        "codebuild:BatchPutTestCases",
        "codebuild:BatchPutCodeCoverages"
    ]
}
```

Es wird empfohlen, diese Richtlinie nur auf die Berichtsgruppen zu beschränken, die Sie verwenden müssen. Folgendes beschränkt die Berechtigungen nur auf die Berichtsgruppen mit den beiden ARNs in der Richtlinie:

```
{
    "Effect": "Allow",
    "Resource": [
```

```
        "arn:aws:codebuild:your-region:your-aws-account-id:report-group/report-group-
name-1",
        "arn:aws:codebuild:your-region:your-aws-account-id:report-group/report-group-
name-2"
    ],
    "Action": [
        "codebuild:CreateReportGroup",
        "codebuild:CreateReport",
        "codebuild:UpdateReport",
        "codebuild:BatchPutTestCases",
        "codebuild:BatchPutCodeCoverages"
    ]
}
```

Folgendes beschränkt die Berechtigungen nur auf Berichtsgruppen, die durch Ausführen von Builds eines Projekts mit dem Namen `my-project` erstellt wurden:

```
{
    "Effect": "Allow",
    "Resource": [
        "arn:aws:codebuild:your-region:your-aws-account-id:report-group/my-project-*"
    ],
    "Action": [
        "codebuild:CreateReportGroup",
        "codebuild:CreateReport",
        "codebuild:UpdateReport",
        "codebuild:BatchPutTestCases",
        "codebuild:BatchPutCodeCoverages"
    ]
}
```

> **Note**
>
> Die im Projekt angegebene CodeBuild Servicerolle wird für Berechtigungen zum Hochladen in den S3-Bucket verwendet.

Berechtigungen für Testberichtoperationen

Sie können Berechtigungen für die folgenden CodeBuild API-Operationen für Testberichte angeben:

- `BatchGetReportGroups`
- `BatchGetReports`
- `CreateReportGroup`
- `DeleteReportGroup`
- `DeleteReport`
- `DescribeTestCases`
- `ListReportGroups`
- `ListReports`
- `ListReportsForReportGroup`
- `UpdateReportGroup`

Weitere Informationen finden Sie unter AWS CodeBuild Referenz zu Berechtigungen.

Beispiele für Testberichtberechtigungen

Informationen zu Beispielrichtlinien für Testberichte finden Sie in:

- Benutzern das Ändern einer Berichtsgruppe ermöglichen
- Benutzern das Erstellen einer Berichtsgruppe ermöglichen
- Benutzern das Löschen eines Berichts ermöglichen
- Benutzern das Löschen einer Berichtsgruppe ermöglichen
- Benutzern das Abrufen von Informationen über Berichtsgruppen ermöglichen
- Benutzern das Abrufen von Informationen über Berichte ermöglichen
- Benutzern das Abrufen einer Liste von Berichtsgruppen ermöglichen
- Benutzern das Abrufen einer Liste von Berichten ermöglichen
- Benutzern das Abrufen einer Liste von Berichten für eine Berichtsgruppe ermöglichen
- Benutzern das Abrufen einer Liste von Testfällen für einen Bericht ermöglichen

Anzeigen von Testberichten

Sie können Details zu einem Testbericht anzeigen, z. B. Informationen zu seinen Testfällen, zum Bestehen und Fehlschlagen sowie zur Dauer der Ausführung. Sie können Testberichte nach Build-

Run, Berichtsgruppe oder Ihrem AWS Konto gruppiert anzeigen. Wählen Sie einen Testbericht in der Konsole aus, um die Details und die Ergebnisse der Testfälle anzuzeigen.

Sie können Testberichte anzeigen, die nicht abgelaufen sind. Testberichte laufen 30 Tage nach ihrer Erstellung ab. Sie können einen abgelaufenen Bericht nicht in anzeigen CodeBuild.

Themen

Anzeigen von Testberichten für einen Build

So zeigen Sie Testberichte für einen Build an:

1. Öffnen Sie die AWS CodeBuild Konsole unter https://console.aws.amazon.com/codesuite/codebuild/home.
2. Suchen Sie den Build, den Sie anzeigen möchten. Wenn Sie das Projekt kennen, das den Build ausgeführt hat, der den Testbericht erstellt hat:

   1. Wählen Sie im Navigationsbereich Build projects (Build-Projekte) und dann das Projekt mit dem Build aus, der den Testbericht ausgeführt hat, den Sie anzeigen möchten.
   2. Wählen Sie Build history (Build-Verlauf) und wählen Sie dann den Build aus, der die Berichte erstellt hat, die Sie anzeigen möchten.

   Sie können den Build auch im Build-Verlauf für Ihr AWS -Konto finden:

   1. Wählen Sie im Navigationsbereich die Option Build history (Build-Verlauf) und dann den Build aus, der die Testberichte erstellt hat, die Sie anzeigen möchten.
3. Wählen Sie auf der Build-Seite Reports (Berichte) aus, und wählen Sie dann einen Testbericht aus, um die Details anzuzeigen.

Anzeigen der Testberichte für eine Berichtsgruppe

So zeigen Sie Testberichte in einer Berichtsgruppe an:

1. Öffnen Sie die AWS CodeBuild Konsole unter https://console.aws.amazon.com/codesuite/codebuild/home.
2. Wählen Sie im Navigationsbereich Report Groups (Berichtsgruppen) aus.
3. Wählen Sie die Berichtsgruppe aus, die die Testberichte enthält, die Sie anzeigen möchten.
4. Wählen Sie einen Testbericht, um die Details anzuzeigen.

Anzeigen von Testberichten in Ihrem AWS -Konto

Um Testberichte in Ihrem AWS Konto einzusehen

1. Öffnen Sie die AWS CodeBuild Konsole unter https://console.aws.amazon.com/codesuite/codebuild/home.
2. Wählen Sie im Navigationsbereich Report history (Berichtsverlauf) aus.
3. Wählen Sie einen Testbericht, um die Details anzuzeigen.

Testberichte mit Test-Frameworks

Die Themen in diesem Abschnitt zeigen, wie Sie die Testberichterstattung für verschiedene Testframeworks einrichten. AWS CodeBuild

Themen

- Einrichten von Testberichten mit Jasmine
- Einrichten von Testberichten mit Jest
- Einrichten von Testberichten mit Pytest
- Einrichten von Testberichten mit RSpec

Einrichten von Testberichten mit Jasmine

Das folgende Verfahren veranschaulicht, wie Testberichte in AWS CodeBuild mit dem JasmineBDD-Test-Framework eingerichtet werden.

Das Verfahren erfordert die folgenden Voraussetzungen:

- Sie haben ein bestehendes CodeBuild--Projekt.
- Ihr Projekt ist ein Node.js-Projekt, das so eingerichtet ist, dass das Jasmine-Test-Framework verwendet werden kann.

Fügen Sie das `jasmine-reporters`-Paket dem Abschnitt `devDependencies` der `package.json`-Projektdatei hinzu. Dieses Paket enthält eine Sammlung von JavaScript-Reporter-Klassen, die mit Jasmine verwendet werden können.

```
npm install --save-dev jasmine-reporters
```

Wenn es noch nicht vorhanden ist, fügen Sie das `test`-Skript der `package.json`-Projektdatei hinzu. Die`test`-Skript stellt sicher, dass Jasmine aufgerufen wirdnpm testist geführt.

```
{
  "scripts": {
    "test": "npx jasmine"
  }
}
```

CodeBuild unterstützt die folgenden Jasmine-Test-Reporter:

JUnitXmlReporter

Wird verwendet, um Berichte im `JunitXml`-Format zu generieren.

NUnitXmlReporter

Wird verwendet, um Berichte im `NunitXml`-Format zu generieren.

Ein Node.js-Projekt mit Jasmine hat standardmäßig ein `spec`-Unterverzeichnis, das die Jasmine-Konfiguration und Testskripte enthält.

Um Jasmine so zu konfigurieren, dass Berichte im `JunitXML`-Format generiert werden, instanziieren Sie den `JUnitXmlReporter`-Reporter, indem Sie den folgenden Code zu Ihren Tests hinzufügen.

```
var reporters = require('jasmine-reporters');

var junitReporter = new reporters.JUnitXmlReporter({
```

```
  savePath: <test report directory>,
  filePrefix: <report filename>,
  consolidateAll: true
});

jasmine.getEnv().addReporter(junitReporter);
```

Um Jasmine so zu konfigurieren, dass Berichte im `NunitXML`-Format generiert werden, instanziieren Sie den `NUnitXmlReporter`-Reporter, indem Sie den folgenden Code zu Ihren Tests hinzufügen.

```
var reporters = require('jasmine-reporters');

var nunitReporter = new reporters.NUnitXmlReporter({
  savePath: <test report directory>,
  filePrefix: <report filename>,
  consolidateAll: true
});

jasmine.getEnv().addReporter(nunitReporter)
```

Die Testberichte werden in die durch *<test report directory>*/*<report filename>* angegebene Datei exportiert.

Aktualisieren Sie die folgenden Abschnitte in Ihrer `buildspec.yml`-Datei oder fügen Sie sie hinzu.

```
version: 0.2

phases:
  pre_build:
    commands:
      - npm install
  build:
    commands:
      - npm build
      - npm test

reports:
  jasmine_reports:
    files:
      - <report filename>
    file-format: JUNITXML
    base-directory: <test report directory>
```

Wenn Sie das `NunitXml`-Berichtsformat verwenden, ändern Sie den `file-format`-Wert in den folgenden Wert.

```
    file-format: NUNITXML
```

Einrichten von Testberichten mit Jest

Das folgende Verfahren veranschaulicht, wie Testberichte in AWS CodeBuild mit dem Jest-Test-Framework eingerichtet werden.

Das Verfahren erfordert die folgenden Voraussetzungen:

- Sie haben ein bestehendes CodeBuild--Projekt.
- Ihr Projekt ist ein Node.js-Projekt, das so eingerichtet ist, dass das Jest-Test-Framework verwendet werden kann.

Fügen Sie das `jest-junit`-Paket dem Abschnitt `devDependencies` der `package.json`-Projektdatei hinzu. CodeBuild verwendet dieses Paket, um Berichte in`JunitXml`.

```
npm install --save-dev jest-junit
```

Wenn es noch nicht vorhanden ist, fügen Sie das `test`-Skript der `package.json`-Projektdatei hinzu. Die`test`-Skript stellt sicher, dass Jest aufgerufen wirdnpm testSo führen Sie aus.

```
{
  "scripts": {
    "test": "jest"
  }
}
```

Konfigurieren Sie Jest, um den `JunitXml`-Reporter zu verwenden, indem Sie der Jest-Konfigurationsdatei Folgendes hinzufügen. Wenn Ihr Projekt keine Jest-Konfigurationsdatei enthält, erstellen Sie eine Datei mit dem Namen `jest.config.js` im Stammverzeichnis Ihres Projekts und fügen Sie Folgendes hinzu. Die Testberichte werden in die durch *<test report directory>*/*<report filename>* angegebene Datei exportiert.

```
module.exports = {
  reporters: [
```

```
    'default',
    [ 'jest-junit', {
      outputDirectory: <test report directory>,
      outputName: <report filename>,
    } ]
  ]
};
```

Aktualisieren Sie die folgenden Abschnitte in Ihrer `buildspec.yml`-Datei oder fügen Sie sie hinzu.

```
version: 0.2

phases:
  pre_build:
    commands:
      - npm install
  build:
    commands:
      - npm build
      - npm test

reports:
  jest_reports:
    files:
      - <report filename>
    file-format: JUNITXML
    base-directory: <test report directory>
```

Einrichten von Testberichten mit Pytest

Das folgende Verfahren veranschaulicht, wie Testberichte AWS CodeBuild mit dem Pytest-Test-Framework eingerichtet werden.

Das Verfahren erfordert die folgenden Voraussetzungen:

- Sie haben ein bestehendes -Projekt.
- Ihr Projekt ist ein Python-Projekt, das so eingerichtet ist, dass das Pytest-Test-Framework verwendet werden kann.

Fügen Sie den folgenden Eintrag entweder der `build` oder `post_build`-Phase Ihrer `buildspec.yml`-Datei hinzu. Dieser Code erkennt automatisch Tests im aktuellen Verzeichnis

und exportiert die Testberichte in die durch *<test report directory>*/*<report filename>* angegebene Datei. Der Bericht verwendet das `JunitXml`-Format.

```
      - python -m pytest --junitxml=<test report directory>/<report filename>
```

Aktualisieren Sie die folgenden Abschnitte in Ihrer `buildspec.yml`-Datei oder fügen Sie sie hinzu.

```
version: 0.2

phases:
  install:
    runtime-versions:
      python: 3.7
    commands:
      - pip3 install pytest
  build:
    commands:
      - python -m pytest --junitxml=<test report directory>/<report filename>

reports:
  pytest_reports:
    files:
      - <report filename>
    base-directory: <test report directory>
    file-format: JUNITXML
```

Einrichten von Testberichten mit RSpec

Das folgende Verfahren veranschaulicht, wie Testberichte in AWS CodeBuild mit dem RSpec-Test-Framework eingerichtet werden.

Das Verfahren erfordert die folgenden Voraussetzungen:

- Sie haben ein bestehendes -Projekt.
- Ihr Projekt ist ein Ruby-Projekt, das so eingerichtet ist, dass das RSpec-Testframework zu verwendet werden kann.

Aktualisieren Sie die folgenden Abschnitte in Ihrer `buildspec.yml`-Datei oder fügen Sie sie hinzu. Dieser Code führt die Tests im Verzeichnis *<test source directory>* durch und exportiert die

Testberichte in die durch *<test report directory>*/*<report filename>* angegebene Datei. Der Bericht verwendet das `JunitXml`-Format.

```
version: 0.2

phases:
  install:
    runtime-versions:
      ruby: 2.6
  pre_build:
    commands:
      - gem install rspec
      - gem install rspec_junit_formatter
  build:
    commands:
      - rspec <test source directory>/* --format RspecJunitFormatter --out <test report
 directory>/<report filename>
reports:
    rspec_reports:
        files:
            - <report filename>
        base-directory: <test report directory>
        file-format: JUNITXML
```

Berichte zur Codeabdeckung

CodeBuild ermöglicht es Ihnen, Berichte über die Codeabdeckung für Ihre Tests zu erstellen. Die folgenden Berichte zur Codeabdeckung werden bereitgestellt:

Netzabdeckung

Die Leitungsabdeckung gibt an, wie viele Aussagen Ihre Tests abdecken. Eine Aussage ist eine einzelne Anweisung, ohne Kommentare oder Bedingungen.

```
line coverage = (total lines covered)/(total number of lines)
```

Abdeckung der Filialen

Die Filialabdeckung gibt an, wie viele Zweige Ihre Tests von allen möglichen Zweigen einer Kontrollstruktur, wie z. B. einer `if case` Oder-Anweisung, abdecken.

```
branch coverage = (total branches covered)/(total number of branches)
```

Die folgenden Dateiformate für Codeabdeckungsberichte werden unterstützt:

- JaCoCo XML
- SimpleCov JSON[1]
- Kleeblatt XML
- XML-Coverage
- LCOV-INFORMATIONEN

[1] CodeBuild akzeptiert Berichte zur JSON-Codeabdeckung, die von simplecov generiert wurden, nicht von simplecov-json.

Erstellen Sie einen Bericht zur Codeabdeckung

Um einen Bericht über die Codeabdeckung zu erstellen, führen Sie ein Buildprojekt aus, das mit mindestens einer Berichtsgruppe zur Codeabdeckung in der Buildspec-Datei konfiguriert ist. CodeBuild interpretiert die Ergebnisse der Codeabdeckung und erstellt einen Bericht über die Codeabdeckung für den Testlauf. Für jeden nachfolgenden Build, der dieselbe buildspec-Datei verwendet, wird ein neuer Testbericht generiert.

So erstellen Sie einen Testbericht:

1. Erstellen Sie ein Build-Projekt. Weitere Informationen finden Sie unter Erstellen eines Build-Projekts in AWS CodeBuild.
2. Konfigurieren Sie die Buildspec-Datei Ihres Projekts mit Testberichtsinformationen:

    a. Fügen Sie einen `reports:` Abschnitt hinzu und geben Sie den Namen für Ihre Berichtsgruppe an. CodeBuild erstellt eine Berichtsgruppe für Sie unter Verwendung Ihres Projektnamens und des Namens, den Sie im Format `project-name` - angegeben haben`report-group-name-in-buildspec`. Wenn Sie bereits über eine Berichtsgruppe verfügen, die Sie verwenden möchten, geben Sie deren ARN an. Wenn Sie den Namen anstelle des ARN verwenden, CodeBuild wird eine neue Berichtsgruppe erstellt. Weitere Informationen finden Sie unter Reports syntax in the buildspec file.

    b. Geben Sie unter der Berichtsgruppe den Speicherort der Dateien an, die die Ergebnisse der Codeabdeckung enthalten. Wenn Sie mehr als eine Berichtsgruppe verwenden, geben Sie die Speicherorte der Ergebnisdateien für jede Berichtsgruppe an. Jedes Mal, wenn Ihr Build-Projekt ausgeführt wird, wird ein neuer Bericht zur Codeabdeckung erstellt. Weitere Informationen finden Sie unter Angeben der Testdateien.

Dies ist ein Beispiel, das einen Bericht über die Codeabdeckung für eine JaCoCo XML-Ergebnisdatei generiert, die sich im Verzeichnis test- befindet`results/jacoco-coverage-report.xml`.

```
reports:
  jacoco-report:
    files:
      - 'test-results/jacoco-coverage-report.xml'
    file-format: 'JACOCOXML'
```

c. Geben Sie im `commands` Abschnitt der `post_build` Sequenz `build` oder die Befehle an, mit denen die Codeabdeckungsanalyse ausgeführt wird. Weitere Informationen finden Sie unter Angeben der Testbefehle .

3. Führen Sie einen Build des Build-Projekts aus. Weitere Informationen finden Sie unter Ausführen eines Build in AWS CodeBuild.
4. Wenn der Build abgeschlossen ist, wählen Sie den neuen Build Run unter Build history (Build-Verlauf) auf Ihrer Projektseite aus. Wählen Sie Berichte aus, um den Bericht zur Codeabdeckung anzuzeigen. Weitere Informationen finden Sie unter Anzeigen von Testberichten für einen Build.

Automatische Erkennung melden

Mit der automatischen Erkennung werden nach Abschluss der Buildphase alle Ihre Build-Dateien durchsucht, nach allen unterstützten Berichtsdateitypen gesucht und automatisch neue Test- und Codeabdeckungsberichtsgruppen und Berichte erstellt. CodeBuild CodeBuild Erstellt für alle erkannten Berichtstypen neue Berichtsgruppen mit dem folgenden Muster:

```
<project-name>-<report-file-format>-AutoDiscovered
```

> **Note**
>
> Wenn die erkannten Berichtsdateien denselben Formattyp haben, werden sie derselben Berichtsgruppe oder demselben Bericht zugeordnet.

Die automatische Erkennung von Berichten wird durch Ihre Projektumgebungsvariablen konfiguriert:

`CODEBUILD_CONFIG_AUTO_DISCOVER`

Diese Variable bestimmt, ob die automatische Erkennung von Berichten während des Builds deaktiviert ist. Standardmäßig ist die automatische Erkennung von Berichten für alle Builds aktiviert. Um diese Funktion zu deaktivieren, stellen Sie `CODEBUILD_CONFIG_AUTO_DISCOVER` auf `false` ein.

`CODEBUILD_CONFIG_AUTO_DISCOVER_DIR`

(Optional) Diese Variable bestimmt, wo CodeBuild nach potenziellen Berichtsdateien gesucht wird. Beachten Sie, dass `**/*` standardmäßig in CodeBuild gesucht wird.

Diese Umgebungsvariablen können während der Erstellungsphase geändert werden. Wenn Sie beispielsweise die automatische Berichtssuche nur für Builds auf dem Git-Branch aktivieren möchten, können Sie den `main` Git-Branch während des Build-Prozesses überprüfen und auf false setzen, wenn `CODEBUILD_CONFIG_AUTO_DISCOVER` sich der Build nicht im `main` Branch befindet. Die automatische Erkennung von Berichten kann über die Konsole oder mithilfe von Projektumgebungsvariablen deaktiviert werden.

Themen

- Automatische Erkennung von Berichten mithilfe der Konsole konfigurieren
- Automatische Erkennung von Berichten mithilfe von Projektumgebungsvariablen konfigurieren

Automatische Erkennung von Berichten mithilfe der Konsole konfigurieren

So konfigurieren Sie die automatische Erkennung von Berichten mit der Konsole

1. Erstellen Sie ein Build-Projekt oder wählen Sie ein Build-Projekt zur Bearbeitung aus. Weitere Informationen finden Sie unter Erstellen eines Build-Projekts in AWS CodeBuild oder Ändern der Einstellungen eines Build-Projekts in AWS CodeBuild.
2. Wählen Sie unter Umgebung die Option Zusätzliche Konfiguration aus.
3. Um die automatische Erkennung von Berichten zu deaktivieren, wählen Sie unter Automatische Erkennung von Berichten die Option Automatische Erkennung von Berichten deaktivieren aus.
4. (Optional) Geben Sie unter Verzeichnis automatisch ermitteln — optional ein Verzeichnismuster ein, um nach Dateien im unterstützten CodeBuild Berichtsformat zu suchen. Beachten Sie, dass `**/*` standardmäßig in CodeBuild gesucht wird.

Automatische Erkennung von Berichten mithilfe von Projektumgebungsvariablen konfigurieren

So konfigurieren Sie die automatische Erkennung von Berichten mithilfe von Projektumgebungsvariablen

1. Erstellen Sie ein Build-Projekt oder wählen Sie ein Build-Projekt zur Bearbeitung aus. Weitere Informationen finden Sie unter Erstellen eines Build-Projekts in AWS CodeBuild oder Ändern der Einstellungen eines Build-Projekts in AWS CodeBuild.
2. Gehen Sie unter Umgebungsvariablen wie folgt vor:

   a. Um die automatische Erkennung von Berichten zu deaktivieren, geben Sie für Name **`CODEBUILD_CONFIG_AUTO_DISCOVER`** und für Wert Folgendes ein**`false`**. Dadurch wird die automatische Erkennung von Berichten deaktiviert.

   b. (Optional) Geben Sie als Name **`CODEBUILD_CONFIG_AUTO_DISCOVER_DIR`** und als Wert das Verzeichnis ein, in dem nach Dateien im unterstützten Berichtsformat gesucht werden CodeBuild soll. `output/*xml`Sucht beispielsweise nach `.xml` Dateien im `output` Verzeichnis

Protokollieren und Überwachen in AWS CodeBuild

Die Überwachung ist ein wichtiger Teil der Aufrechterhaltung von Zuverlässigkeit, Verfügbarkeit und Performance von AWS CodeBuild und Ihren AWS-Lösungen. Sie sollten Überwachungsdaten aller Bestandteile IhrerAWS-Lösung, damit Sie auftretende Multipunkt-Fehler leichter debuggen können.AWSstellt die folgenden Tools für die Überwachung Ihrer CodeBuild-Ressourcen und Builds und zur Reaktion auf potenzielle Vorfälle bereit:

Themen

Protokollierung von AWS CodeBuild-API-Aufrufen mit AWS CloudTrail

AWS CodeBuildist in integriertAWS CloudTrail, einem Service, der eine Aufzeichnung der von einem Benutzer, einer Rolle oder einemAWS -Service durchgeführten Aktionen in CodeBuild. CloudTrail erfasst alle CodeBuild CodeBuild CodeBuild Wenn Sie einen Trail erstellen, können Sie die kontinuierliche Bereitstellung von CloudTrail Ereignissen in einem S3-Bucket aktivieren CodeBuild. Auch wenn Sie keinen Trail konfigurieren, können Sie die neuesten Ereignisse in der CloudTrail-Konsole in Event history (Ereignisverlauf) anzeigen. Mit den von CloudTrail gesammelten Informationen können Sie die an gestellte Anfrage CodeBuild, die IP-Adresse, von der die Anfrage gestellt wurde, den Initiator der Anfrage, den Zeitpunkt der Anfrage und weitere Angaben bestimmen.

Weitere Informationen CloudTrail finden Sie im AWS CloudTrailBenutzerhandbuch.

AWS CodeBuildInformationen in CloudTrail

CloudTrail wird beim Erstellen IhresAWS -Kontos für Sie aktiviert. Die in CodeBuild auftretenden Aktivitäten werden als CloudTrail Ereignis zusammen mit anderenAWS -Serviceereignissen in Event history (Ereignisverlauf) aufgezeichnet. Sie können die neusten Ereignisse in Ihr AWS-Konto herunterladen und dort suchen und anzeigen. Weitere Informationen finden Sie im AWS CloudTrailBenutzerhandbuch unter Ereignisse mit CloudTrail Ereignisverlauf anzeigen.

Um die Ereignisse in IhremAWS Konto einschließlich Ereignissen für kontinuierlich aufzuzeichnen CodeBuild, erstellen Sie einen Trail. Ein Trail ermöglicht CloudTrail die Bereitstellung von

Protokolldateien in einem S3-Bucket. Wenn Sie einen Trail in der Konsole anlegen, gilt dieser standardmäßig für alle Regionen. Der Trail protokolliert Ereignisse aus allen Regionen in der AWS-Partition und stellt die Protokolldateien in dem S3-Bucket bereit, den Sie angeben. Sie können andereAWS -Services konfigurieren CloudTrail von Weitere Informationen finden Sie unter:

- Übersicht zum Erstellen eines Trails
- CloudTrail unterstützte Dienste und Integrationen
- Amazon SNS von CloudTrail
- CloudTrail CloudTrail Empfangen von

Alle CodeBuild Aktionen werden von der CodeBuild API-Referenz protokolliert CloudTrail und sind in dieser dokumentiert. Beispielsweise generieren Aufrufe der Aktionen`CreateProject` (in derAWS CLI,`create-project`),`StartBuild` (in derAWS CLI,`start-project`) und`UpdateProject` (in derAWS CLI,`update-project`) Einträge in den CloudTrail Protokolldateien.

Jeder Ereignis- oder Protokolleintrag enthält Informationen zu dem Benutzer, der die Anforderung generiert hat. Die Identitätsinformationen unterstützen Sie bei der Ermittlung der folgenden Punkte:

- Ob die Anforderung mit Root- oder -Benutzeranmeldeinformationen ausgeführt wurde.
- Gibt an, ob die Anforderung mit temporären Sicherheitsanmeldeinformationen für eine Rolle oder einen verbundenen Benutzer gesendet wurde.
- Gibt an, ob die Anforderung aus einem anderen AWS-Service gesendet wurde

Weitere Informationen finden Sie unter dem ElementCloudTrail userIdentity im AWS CloudTrailBenutzerhandbuch.

Grundlagen zu AWS CodeBuild-Protokolldateieinträgen

Ein Trail ist eine Konfiguration, durch die Ereignisse als Protokolldateien an den von Ihnen angegebenen S3-Bucket übermittelt werden. CloudTrail Protokolldateien können einen oder mehrere Einträge enthalten. Ein Ereignis stellt eine einzelne Anfrage aus einer beliebigen Quelle dar und enthält unter anderem Informationen über die angeforderte Aktion, das Datum und die Uhrzeit der Aktion sowie über die Anfrageparameter. CloudTrail Protokolleinträge sind kein geordnetes Stacktrace der öffentlichen API-Aufrufe und erscheinen daher nicht in einer bestimmten Reihenfolge.

> **Note**
>
> Um vertrauliche Informationen zu schützen, ist Folgendes in den CodeBuild Protokollen versteckt:
>
> - AWS-Zugriffsschlüssel-IDs. Weitere Informationen finden Sie unter Verwalten vonAWS Identity and Access Management
> - Mit dem Parameter Store angegebene Zeichenfolgen. Weitere Informationen finden Sie unter Systems Manager Parameter Store und Systems Manager Parameter Store Console Walkthrough im Amazon EC2 Systems Manager Manager-Benutzerhandbuch.
> - Zeichenfolgen, die mit angegeben wurdenAWS Secrets Manager. Weitere Informationen finden Sie unter Schlüsselverwaltung.

Das folgende Beispiel zeigt einen CloudTrail Protokolleintrag, der die Erstellen von Build-Projekten veranschaulicht CodeBuild.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "FederatedUser",
    "principalId": "account-ID:user-name",
    "arn": "arn:aws:sts::account-ID:federated-user/user-name",
    "accountId": "account-ID",
    "accessKeyId": "access-key-ID",
    "sessionContext": {
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2016-09-06T17:59:10Z"
      },
      "sessionIssuer": {
        "type": "IAMUser",
        "principalId": "access-key-ID",
        "arn": "arn:aws:iam::account-ID:user/user-name",
        "accountId": "account-ID",
        "userName": "user-name"
      }
    }
  },
  "eventTime": "2016-09-06T17:59:11Z",
  "eventSource": "codebuild.amazonaws.com",
```

```
  "eventName": "CreateProject",
  "awsRegion": "region-ID",
  "sourceIPAddress": "127.0.0.1",
  "userAgent": "user-agent",
  "requestParameters": {
    "awsActId": "account-ID"
  },
  "responseElements": {
    "project": {
      "environment": {
        "image": "image-ID",
        "computeType": "BUILD_GENERAL1_SMALL",
        "type": "LINUX_CONTAINER",
        "environmentVariables": []
      },
      "name": "codebuild-demo-project",
      "description": "This is my demo project",
      "arn": "arn:aws:codebuild:region-ID:account-ID:project/codebuild-demo-
project:project-ID",
      "encryptionKey": "arn:aws:kms:region-ID:key-ID",
      "timeoutInMinutes": 10,
      "artifacts": {
        "location": "arn:aws:s3:::codebuild-region-ID-account-ID-output-bucket",
        "type": "S3",
        "packaging": "ZIP",
        "outputName": "MyOutputArtifact.zip"
      },
      "serviceRole": "arn:aws:iam::account-ID:role/CodeBuildServiceRole",
      "lastModified": "Sep 6, 2016 10:59:11 AM",
      "source": {
        "type": "GITHUB",
        "location": "https://github.com/my-repo.git"
      },
      "created": "Sep 6, 2016 10:59:11 AM"
    }
  },
  "requestID": "9d32b228-745b-11e6-98bb-23b67EXAMPLE",
  "eventID": "581f7dd1-8d2e-40b0-aeee-0dbf7EXAMPLE",
  "eventType": "AwsApiCall",
  "recipientAccountId": "account-ID"
}
```

Überwachung von AWS CodeBuild

Sie können Amazon CloudWatch verwenden, um Ihre Builds zu überwachen, Fehler zu melden und gegebenenfalls automatisch Maßnahmen zu ergreifen. Sie können Ihre Builds auf zwei Ebenen überwachen:

Ebene des Projekts

Diese Metriken gelten für alle Builds im angegebenen Projekt. Um Metriken für ein Projekt anzuzeigen, geben Sie`ProjectName`für die Dimension in CloudWatch.

AWSKonto-Level

Diese Metriken gelten für alle Builds in einem -Konto. Um Metriken auf derAWS-Kontoebene, geben Sie keine Dimension in CloudWatch ein. Metriken für die Build-Ressourcennutzung sind imAWSKonto-Level.

CloudWatch-Metriken zeigen das Verhalten Ihrer Builds im Laufe der Zeit. Beispielsweise können Sie Folgendes überwachen:

- Wie viele Builds in einem Build-Projekt oder einem AWS-Konto im Laufe der Zeit versucht wurden.
- Wie viele Builds in einem Build-Projekt oder einem AWS-Konto im Laufe der Zeit erfolgreich erstellt wurden.
- Wie viele Builds in einem Build-Projekt oder einem AWS-Konto im Laufe der Zeit fehlgeschlagen haben.
- Wie viel Zeit CodeBuild zur Ausführung von Builds in einem Build-Projekt oder einemAWS-Konto im Laufe der Zeit.
- Erstellen Sie die Ressourcenauslastung für einen Build oder ein ganzes Build-Projekt. Metriken zur Build-Ressourcenauslastung umfassen Metriken wie z B. CPU, Arbeitsspeicher und Speicherauslastung.

Weitere Informationen finden Sie unter Überwachung von CodeBuild-Metriken.

CodeBuild-CloudWatch-Metriken

Die folgenden Metriken können proAWS-Konto oder Build-Projekt.

buildDuration

Misst die Dauer der `BUILD`-Phase des Builds.

Einheiten: Sekunden

Gültige CloudWatch-Statistiken Durchschnitt (empfohlen), Maximum, Minimum

Builds

Misst die Anzahl der ausgelösten Builds.

Einheiten: Anzahl

Gültige CloudWatch-Statistiken Summe

DownloadSourceDuration

Misst die Dauer der `DOWNLOAD_SOURCE`-Phase des Builds.

Einheiten: Sekunden

Gültige CloudWatch-Statistiken Durchschnitt (empfohlen), Maximum, Minimum

Dauer

Misst die Dauer aller Builds im Laufe der Zeit.

Einheiten: Sekunden

Gültige CloudWatch-Statistiken Durchschnitt (empfohlen), Maximum, Minimum

FailedBuilds

Misst die Anzahl von Builds, die fehlgeschlagen haben, weil ein Client-Fehler oder ein Timeout auftritt.

Einheiten: Anzahl

Gültige CloudWatch-Statistiken Summe

FinalizingDuration

Misst die Dauer der `FINALIZING`-Phase des Builds.

Einheiten: Sekunden

Gültige CloudWatch-Statistiken Durchschnitt (empfohlen), Maximum, Minimum

installDuration

Misst die Dauer der `INSTALL`-Phase des Builds.

Einheiten: Sekunden

Gültige CloudWatch-Statistiken Durchschnitt (empfohlen), Maximum, Minimum

PostBuildDuration

Misst die Dauer der `POST_BUILD`-Phase des Builds.

Einheiten: Sekunden

Gültige CloudWatch-Statistiken Durchschnitt (empfohlen), Maximum, Minimum

PrebuildDuration

Misst die Dauer der `PRE_BUILD`-Phase des Builds.

Einheiten: Sekunden

Gültige CloudWatch-Statistiken Durchschnitt (empfohlen), Maximum, Minimum

ProvisioningDuration

Misst die Dauer der `PROVISIONING`-Phase des Builds.

Einheiten: Sekunden

Gültige CloudWatch-Statistiken Durchschnitt (empfohlen), Maximum, Minimum

queuedDuration

Misst die Dauer der `QUEUED`-Phase des Builds.

Einheiten: Sekunden

Gültige CloudWatch-Statistiken Durchschnitt (empfohlen), Maximum, Minimum

SubmittedDuration

Misst die Dauer der `SUBMITTED`-Phase des Builds.

Einheiten: Sekunden

Gültige CloudWatch-Statistiken Durchschnitt (empfohlen), Maximum, Minimum

SucceededBuilds

Misst die Anzahl der erfolgreichen Builds.

Einheiten: Anzahl

Gültige CloudWatch-Statistiken Summe

uploadArtifactsDuration

Misst die Dauer der `UPLOAD_ARTIFACTS`-Phase des Builds.

Einheiten: Sekunden

Gültige CloudWatch-Statistiken Durchschnitt (empfohlen), Maximum, Minimum

CodeBuild CloudWatch-Metriken zur Ressourcennutzung

> **Note**
>
> CodeBuild-Metriken zur Ressourcennutzung sind nur in den folgenden Regionen verfügbar:
>
> - Asia Pacific (Tokyo) Region
> - Asia Pacific (Seoul) Region
> - Region Asien-Pazifik (Mumbai)
> - Asia Pacific (Singapore) Region
> - Asia Pacific (Sydney) Region
> - Region Kanada (Zentral)
> - Region Europa (Frankfurt)
> - Europe (Ireland) Region
> - Europe (London) Region
> - Region Europa (Paris)
> - South America (São Paulo) Region
> - US East (N. Virginia) Region
> - US East (Ohio) Region
> - US West (N. California) Region
> - US West (Oregon) Region

Die folgenden Metriken zur Ressourcennutzung können verfolgt werden.

CPUUtilized

Die Anzahl von CPU-Einheiten der zugewiesenen Verarbeitung, die vom Build-Container verwendet werden.

Einheiten: CPU-Einheiten

Gültige CloudWatch-Statistiken Durchschnitt (empfohlen), Maximum, Minimum

CpuUused Percent

Der Prozentsatz der zugewiesenen Verarbeitung, die vom Build-Container verwendet wird.

Einheiten: Prozent

Gültige CloudWatch-Statistiken Durchschnitt (empfohlen), Maximum, Minimum

MemoryUtilized

Die Anzahl von Megabyte an Speicherplatz, der vom Build-Container verwendet wird.

Einheiten: Megabytes

Gültige CloudWatch-Statistiken Durchschnitt (empfohlen), Maximum, Minimum

MemoryUtilizedPercent

Der Prozentsatz des zugewiesenen Speichers, der vom Build-Container verwendet wird.

Einheiten: Prozent

Gültige CloudWatch-Statistiken Durchschnitt (empfohlen), Maximum, Minimum

StorageReadBytes

Die vom Build-Container verwendete Lesegeschwindigkeit des Speichers.

Einheiten: Byte/Sekunde

Gültige CloudWatch-Statistiken Durchschnitt (empfohlen), Maximum, Minimum

StorageWriteBytes

Die vom Build-Container verwendete Schreibgeschwindigkeit des Speichers.

Einheiten: Byte/Sekunde

Gültige CloudWatch-Statistiken Durchschnitt (empfohlen), Maximum, Minimum

CodeBuild-CloudWatch-Abmessungen

CodeBuild bietet die folgenden CloudWatch-Metrik-Dimensionen. Wenn keine davon angegeben ist, gelten die Metriken für die aktuelleAWSKonto.

BuildId, buildNumber, ProjectName

Metriken werden für einen Build-Bezeichner, eine Build-Nummer und einen Projektnamen bereitgestellt.

ProjectName

Für einen Projektnamen werden Metriken bereitgestellt.

CodeBuild-CloudWatch-Alarme

Sie können die CloudWatch-Konsole verwenden, um Alarme basierend auf CodeBuild-Metriken zu erstellen. Auf diese Weise können Sie reagieren, wenn ein Problem mit Ihren Builds auftritt. Die beiden praktischsten Metriken mit Alarmen sind:

- `FailedBuild`daus. Sie können einen Alarm einrichten, der ausgelöst wird, wenn eine bestimmte Anzahl von fehlgeschlagenen Builds innerhalb einer festgelegten Anzahl von Sekunden erkannt wird. In CloudWatch geben Sie die Anzahl der Sekunden an und wie viele fehlgeschlagene Builds einen Alarm auslösen.
- `Duration`aus. Sie können einen Alarm einrichten, der ausgelöst wird, wenn ein Build länger dauert als erwartet. Sie geben an, wie viele Sekunden vergehen müssen, nachdem ein Build gestartet wurde und bevor ein Build abgeschlossen wurde, bevor der Alarm ausgelöst wird.

Weitere Informationen zum Erstellen von Alarmen für CodeBuild-Metriken finden Sie unterÜberwachen von Builds mit CloudWatch-Alarmenaus. Weitere Informationen über Alarme finden Sie unterErstellen von Amazon CloudWatch CloudWatch-AlarmenimAmazon CloudWatch-Benutzerhandbuchaus.

Überwachung von CodeBuild-Metriken

AWS CodeBuildüberwacht Funktionen für Sie und meldet Metriken über Amazon CloudWatch. Diese Metriken umfassen die Anzahl der gesamten Builds, fehlgeschlagene Builds, erfolgreiche Builds und die Dauer des Builds.

Sie können Metriken für CodeBuild mithilfe der CodeBuild-Konsole oder der CloudWatch-Konsole überwachen. Die folgenden Verfahren zeigt, wie Sie auf Metriken zugreifen.

Themen

Zugriff auf Build-Metriken (CodeBuild-Konsole)

> **Note**
>
> Sie können die Metriken oder die Diagramme für ihre Anzeige in der CodeBuild-Konsole nicht anpassen. Wenn Sie die Anzeige anpassen möchten, können Sie die Build-Metriken mithilfe der Amazon CloudWatch CloudWatch-Konsole anzeigen.

Metriken auf Konto-Ebene

Zugriff aufAWSMetriken auf Konto-Ebene

1. Melden Sie sich bei der anAWS Management ConsoleSo öffnen Sie denAWS CodeBuildconsole unterhttps://console.aws.amazon.com/codesuite/codebuild/homeaus.
2. Wählen Sie im Navigationsbereich Account metrics (Konto-Metriken) aus.

Metriken auf Projektebene

Zugriff auf Metriken auf Projektebene

1. Melden Sie sich bei der anAWS Management ConsoleSo öffnen Sie denAWS CodeBuildconsole unterhttps://console.aws.amazon.com/codesuite/codebuild/homeaus.
2. Wählen Sie im linken Navigationsbereich Build projects aus.

3. Wählen Sie in der Liste der Build-Projekte in der Spalte Name das Projekt aus, in dem Sie die Metriken anzeigen möchten.
4. Wählen Sie den Tab Metrics.

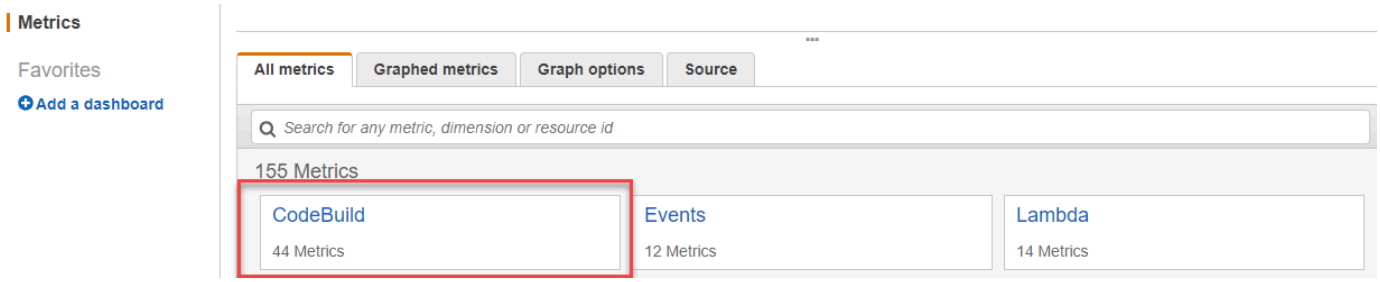
Zugriff auf Build-Metriken (Amazon CloudWatch CloudWatch-Konsole)

Sie können die Metriken und die Diagramme für ihre Anzeige mit der CloudWatch-Konsole anpassen.

Metriken auf Konto-Ebene

So greifen Sie auf Metriken auf Konto-Ebene zu

1. Melden Sie sich bei AWS Management Console an und öffnen Sie die CloudWatch-Konsole unter https://console.aws.amazon.com/cloudwatch/.
2. Wählen Sie im Navigationsbereich Metrics (Metriken) aus.
3. Wählen Sie auf der Registerkarte All metrics (Alle Metriken) auf CodeBuild.

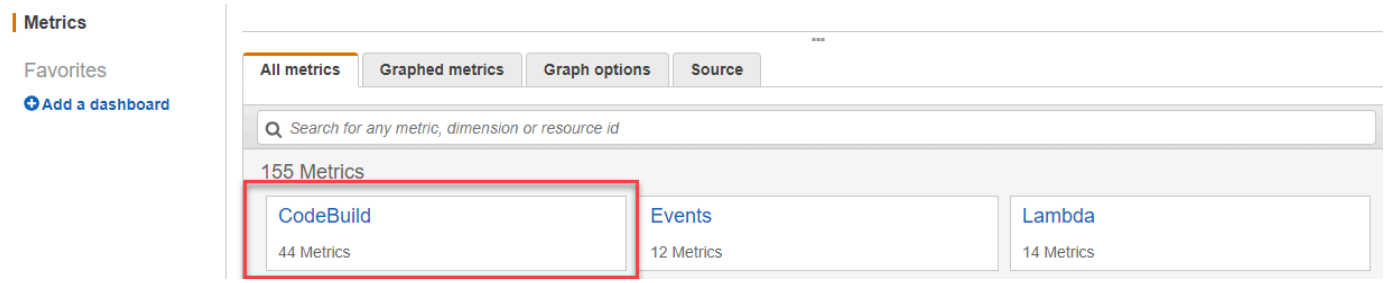


4. Wählen Sie Account Metrics (Konto-Metriken) aus.
5. Wählen Sie ein oder mehrere Projekte und eine oder mehrere Metriken. Sie können für jedes Projekt die Metriken SucceededBuilds, FailedBuilds, Builds und Duration auswählen. Alle ausgewählten Projekt- und Metrik-Kombinationen werden in dem Diagramm auf der Seite angezeigt.

Metriken auf Projektebene

Zugriff auf Metriken auf Projektebene

1. Melden Sie sich bei AWS Management Console an und öffnen Sie die CloudWatch-Konsole unter https://console.aws.amazon.com/cloudwatch/.
2. Wählen Sie im Navigationsbereich Metrics (Metriken) aus.
3. Wählen Sie auf der Registerkarte All metrics (Alle Metriken) auf CodeBuild.



4. Wählen Sie By Project (Nach Projekt).
5. Wählen Sie eine oder mehrere Projekt- und Metrikkombinationen. Sie können für jedes Projekt die Metriken SucceededBuilds, FailedBuilds, Builds und Duration auswählen. Alle ausgewählten Projekt- und Metrik-Kombinationen werden in dem Diagramm auf der Seite angezeigt.
6. (Optional) Sie können Ihre Metriken und Diagramme anpassen. Beispielsweise aus der Dropdown-Liste imStatistikSie können eine andere Statistik auswählen, die angezeigt werden soll. Alternativ können Sie auch aus dem Dropdown-Menü in der Spalte Period (Zeitraum) einen anderen Zeitraum für die Überwachung der Metriken auswählen.

   Weitere Informationen finden Sie unterMetriken grafisch darstellenundAnzeigen der verfügbaren MetrikenimAmazon CloudWatch-Benutzerhandbuchaus.

Überwachung von CodeBuild-Ressourcennutzungsmetriken

AWS CodeBuildüberwacht die Ressourcenauslastung in Ihrem Namen und meldet Metriken über Amazon CloudWatch. Dazu gehören Metriken wie CPU-, Arbeitsspeicher- und Speicherauslastung.

> **Note**
>
> CodeBuild-Ressourcennutzungsmetriken werden nur für Builds aufgezeichnet, die länger als eine Minute laufen.

Sie können die CodeBuild-Konsole oder die CloudWatch-Konsole verwenden, um Metriken für die Ressourcennutzung für CodeBuild zu überwachen.

> **Note**
>
> CodeBuild-Ressourcenauslastungsmetriken sind nur in folgenden Regionen verfügbar:
>
> - Asia Pacific (Tokyo) Region

- Asia Pacific (Seoul) Region
- Region Asien-Pazifik (Mumbai)
- Asia Pacific (Singapore) Region
- Asia Pacific (Sydney) Region
- Region Kanada (Zentral)
- Region Europa (Frankfurt)
- Europe (Ireland) Region
- Europe (London) Region
- Region Europa (Paris)
- South America (São Paulo) Region
- US East (N. Virginia) Region
- US East (Ohio) Region
- US West (N. California) Region
- US West (Oregon) Region

Die folgenden Verfahren zeigen Ihnen, wie Sie auf Ihre Ressourcenauslastungsmetriken zugreifen können.

Themen

- Zugriff auf Metriken zur Ressourcennutzung (CodeBuild-Konsole)
- Greifen Sie auf Metriken zur Ressourcennutzung zu (Amazon CloudWatch CloudWatch-Konsole)

Zugriff auf Metriken zur Ressourcennutzung (CodeBuild-Konsole)

> **Note**
>
> Sie können die Metriken oder die Diagramme für ihre Anzeige in der CodeBuild-Konsole nicht anpassen. Wenn Sie die Anzeige anpassen möchten, können Sie mithilfe der Amazon CloudWatch CloudWatch-Konsole Ihre Build-Metriken anzeigen.

Ressourcenauslastung auf Projektebene

So greifen Sie auf Ressourcenauslastungsmetriken auf Projektebene

1. Melden Sie sich beimAWS Management Consoleund öffnen SieAWS CodeBuildconsole unterhttps://console.aws.amazon.com/codesuite/codebuild/homeaus.
2. Wählen Sie im linken Navigationsbereich Build projects aus.
3. In der Liste der Build-Projekte finden Sie unterNameWählen Sie das Projekt aus, für das Sie die Auslastungsmetriken anzeigen möchten.
4. Wählen Sie den Tab Metrics. Die Metriken für die Ressourcennutzung werden imRessourcenauslastungsmetrikenAbschnitts erstellt.
5. Zum Anzeigen der Ressourcenauslastungsmetriken auf Projektebene in der CloudWatch-Konsole wählen SieAnzeigen in CloudWatchimRessourcenauslastungsmetrikenAbschnitts erstellt.

Ressourcenauslastungsmetriken auf Build-Level

So greifen Sie auf Ressourcenauslastungsmetriken auf Build-Ebene

1. Melden Sie sich beimAWS Management Consoleund öffnen SieAWS CodeBuildconsole unterhttps://console.aws.amazon.com/codesuite/codebuild/homeaus.
2. Wählen Sie im Navigationsbereich Build history aus.
3. In der Liste der Builds, imBuild RunWählen Sie den Build aus, für den Sie die Auslastungsmetriken anzeigen möchten.
4. Wählen Sie das SymbolRessourcenauslastung-Tab.
5. Zum Anzeigen der Ressourcenauslastungsmetriken auf Build-Level in der CloudWatch-Konsole wählen SieAnzeigen in CloudWatchimRessourcenauslastungsmetrikenAbschnitts erstellt.

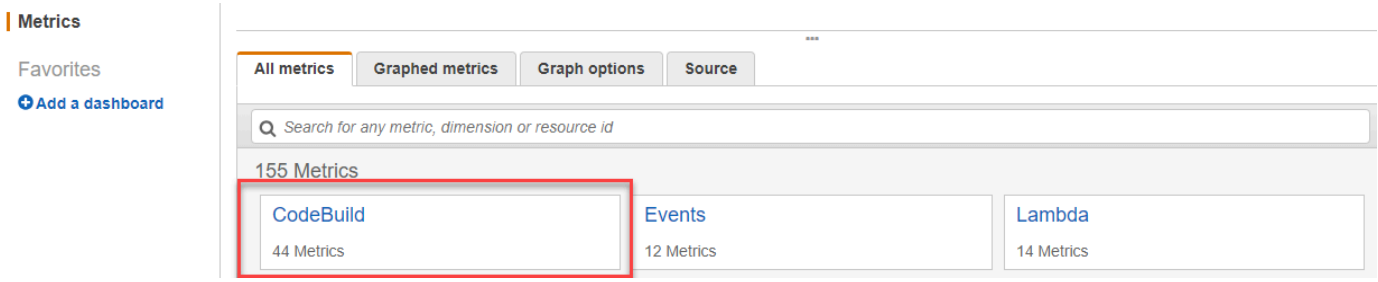
Greifen Sie auf Metriken zur Ressourcennutzung zu (Amazon CloudWatch CloudWatch-Konsole)

Die Amazon CloudWatch CloudWatch-Konsole kann verwendet werden, um auf CodeBuild-Ressourcennutzungsmetriken zuzugreifen.

Ressourcenauslastung auf Projektebene

So greifen Sie auf Ressourcenauslastungsmetriken auf Projektebene

1. Melden Sie sich bei AWS Management Console an und öffnen Sie die CloudWatch-Konsole unter https://console.aws.amazon.com/cloudwatch/.
2. Wählen Sie im Navigationsbereich Metrics (Metriken) aus.
3. Wählen Sie auf der Registerkarte All metrics (Alle Metriken) auf CodeBuild.



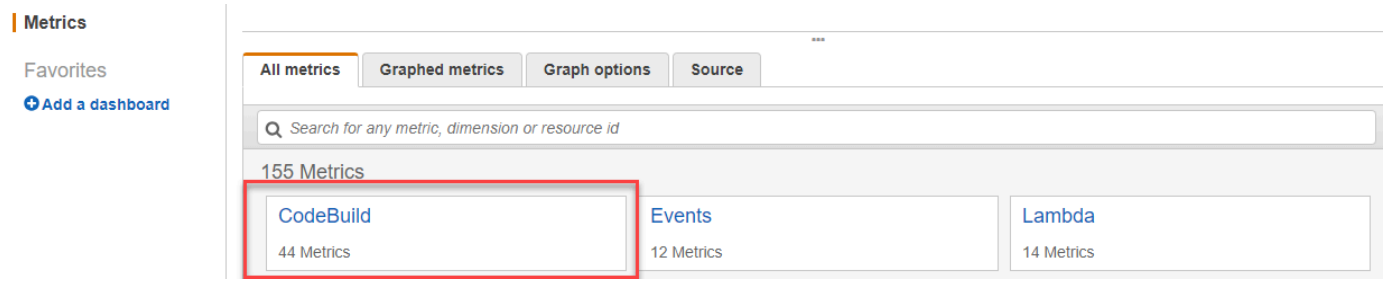
4. Wählen Sie By Project (Nach Projekt).
5. Wählen Sie eine oder mehrere Projekt- und Metrikkombinationen für das Diagramm. Alle ausgewählten Projekt- und Metrik-Kombinationen werden in dem Diagramm auf der Seite angezeigt.
6. (Optional) Sie können Ihre Metriken und Diagramme ausDiagrammte Metriken-Tab. Beispielsweise aus der Dropdown-Liste imStatistik-Spalte können Sie eine andere Statistik auswählen, die angezeigt werden soll. Alternativ können Sie auch aus dem Dropdown-Menü in der Spalte Period (Zeitraum) einen anderen Zeitraum für die Überwachung der Metriken auswählen.

   Weitere Informationen finden Sie unterDarstellen von MetrikenundAnzeigen der verfügbaren MetrikenimAmazon CloudWatch-Benutzerhandbuchaus.

Ressourcenauslastungsmetriken auf Build-Level

So greifen Sie auf Ressourcenauslastungsmetriken auf Build-Ebene

1. Melden Sie sich bei AWS Management Console an und öffnen Sie die CloudWatch-Konsole unter https://console.aws.amazon.com/cloudwatch/.
2. Wählen Sie im Navigationsbereich Metrics (Metriken) aus.
3. Wählen Sie auf der Registerkarte All metrics (Alle Metriken) auf CodeBuild.



4. Klicken Sie aufbuildNumber, buildNumber, ProjectNameaus.
5. Wählen Sie eine oder mehrere Build- und Metrikkombinationen für das Diagramm. Alle ausgewählten Build- und Metrikkombinationen werden im Diagramm auf der Seite angezeigt.
6. (Optional) Sie können Ihre Metriken und Diagramme ausDiagrammte Metriken-Tab. Beispielsweise aus der Dropdown-Liste imStatistik-Spalte können Sie eine andere Statistik auswählen, die angezeigt werden soll. Alternativ können Sie auch aus dem Dropdown-Menü in der Spalte Period (Zeitraum) einen anderen Zeitraum für die Überwachung der Metriken auswählen.

   Weitere Informationen finden Sie unterDarstellen von MetrikenundAnzeigen der verfügbaren MetrikenimAmazon CloudWatch-Benutzerhandbuchaus.

Überwachen von Builds mit CloudWatch-Alarmen

Sie können einen CloudWatch-Alarm für Ihre Builds erstellen. Ein Alarm überwacht eine Metrik über einen bestimmten, von Ihnen definierten Zeitraum und führt eine oder mehrere Aktionen durch, die vom Wert der Metrik im Verhältnis zu einem bestimmten Schwellenwert in einer Reihe von Zeiträumen abhängt. Mit der nativen CloudWatch-Alarmfunktionalität können Sie eine der von CloudWatch unterstützten Aktionen angeben, die ausgeführt werden soll, wenn ein Schwellenwert überschritten wird. Sie können beispielsweise festlegen, dass eine Amazon SNS SNS-Benachrichtigung gesendet wird, wenn mehr als drei Builds in Ihrem Konto innerhalb von 15 Minuten fehlschlagen.

So erstellen Sie einen CloudWatch-Alarm für eine CodeBuild-Metrik

1. Melden Sie sich bei AWS Management Console an und öffnen Sie die CloudWatch-Konsole unter https://console.aws.amazon.com/cloudwatch/.
2. Klicken Sie im Navigationsbereich auf Alarms (Alarme).
3. Wählen Sie Create Alarm (Alarm erstellen) aus.

4. Wählen Sie unter CloudWatch Metrics by Category (CloudWatch-Metriken nach Kategorie) die Option CodeBuild aus. Wenn Sie wissen, dass Sie nur Metriken auf Projektebene benötigen, wählen Sie By Project (Nach Projekt). Wenn Sie wissen, dass Sie nur Metriken auf Kontoebene benötigen, wählen Sie Account Metrics (Kontometriken).
5. Wählen Sie unter Create Alarm (Alarm erstellen) Select Metric (Metrik auswahlen), falls es noch nicht ausgewählt ist.
6. Wählen Sie eine Metrik aus, für die Sie einen Alarm erstellen möchten. Die Optionen sind By Project (Nach Projekt) oder Account Metrics (Konto-Metriken).
7. Wählen Sie Next (Weiter) und anschließend Define Alarm (Alarm definieren), und erstellen Sie dann Ihren Alarm. Weitere Informationen finden Sie unter Bearbeiten oder Löschen eines CloudWatch-Alarms im Amazon CloudWatch-Benutzerhandbuch. Weitere Informationen zum Einrichten von Amazon SNS SNS-Benachrichtigungen, wenn ein Alarm ausgelöst wird, finden Sie unterEinrichten von Amazon SNS SNS-BenachrichtigungenimEntwicklerhandbuch von Amazon SNSaus.
8. Wählen Sie Create Alarm (Alarm erstellen) aus.

Sicherheit in AWS CodeBuild

Cloud-Sicherheit AWS hat höchste Priorität. Als AWS Kunde profitieren Sie von einer Rechenzentrums- und Netzwerkarchitektur, die darauf ausgelegt sind, die Anforderungen der sicherheitssensibelsten Unternehmen zu erfüllen.

Sicherheit und Compliance sind eine gemeinsame Verantwortung von Ihnen AWS und Ihnen. Dieses gemeinsame Modell kann Ihnen helfen, Ihre betriebliche Belastung zu verringern: Es AWS betreibt, verwaltet und kontrolliert die Komponenten vom Host-Betriebssystem und der Virtualisierungsebene bis hin zur physischen Sicherheit der Serviceeinrichtungen. Sie übernehmen die Verantwortung und die Verwaltung für das Gastbetriebssystem (einschließlich Updates und Sicherheits-Patches) und andere damit verbundene Anwendungssoftware. Sie sind auch für die Konfiguration der AWS bereitgestellten Sicherheitsgruppen-Firewall verantwortlich. Ihre Aufgaben sind je nach genutzten Services, deren Integration in Ihre IT-Umgebung sowie den geltenden Gesetzen und Vorschriften unterschiedlich. Aus diesem Grund sollten Sie sich gut überlegen, welche Services in Ihrer Organisation verwendet werden sollen. Weitere Informationen finden Sie unter Modell der geteilten Verantwortung.

In den folgenden Themen erfahren Sie, wie Sie Ihre CodeBuild Ressourcen schützen können.

Themen

Datenschutz in AWS CodeBuild

Das AWS Modell der gilt für den Datenschutz in AWS CodeBuild. Wie in diesem Modell beschrieben, AWS ist verantwortlich für den Schutz der globalen Infrastruktur, auf der alle Systeme laufen AWS Cloud. Sie sind dafür verantwortlich, die Kontrolle über Ihre in dieser Infrastruktur gehosteten Inhalte zu behalten. Sie sind auch für die Sicherheitskonfiguration und die Verwaltungsaufgaben für die von

Ihnen verwendeten AWS-Services verantwortlich. Weitere Informationen zum Datenschutz finden Sie unter Häufig gestellte Fragen zum Datenschutz. Informationen zum Datenschutz in Europa finden Sie im Blog-Beitrag AWS -Modell der geteilten Verantwortung und in der DSGVO im AWS -Sicherheitsblog.

Aus Datenschutzgründen empfehlen wir, dass Sie AWS-Konto Anmeldeinformationen schützen und einzelne Benutzer mit AWS IAM Identity Center oder AWS Identity and Access Management (IAM) einrichten. So erhält jeder Benutzer nur die Berechtigungen, die zum Durchführen seiner Aufgaben erforderlich sind. Außerdem empfehlen wir, die Daten mit folgenden Methoden schützen:

- Verwenden Sie für jedes Konto die Multi-Faktor-Authentifizierung (MFA).
- Verwenden Sie SSL/TLS, um mit Ressourcen zu kommunizieren. AWS Wir benötigen TLS 1.2 und empfehlen TLS 1.3.
- Richten Sie die API und die Protokollierung von Benutzeraktivitäten mit ein. AWS CloudTrail
- Verwenden Sie AWS Verschlüsselungslösungen zusammen mit allen darin enthaltenen Standardsicherheitskontrollen AWS-Services.
- Verwenden Sie erweiterte verwaltete Sicherheitsservices wie Amazon Macie, die dabei helfen, in Amazon S3 gespeicherte persönliche Daten zu erkennen und zu schützen.
- Wenn Sie für den Zugriff AWS über eine Befehlszeilenschnittstelle oder eine API FIPS 140-2-validierte kryptografische Module benötigen, verwenden Sie einen FIPS-Endpunkt. Weitere Informationen über verfügbare FIPS-Endpunkte finden Sie unter Federal Information Processing Standard (FIPS) 140-2.

Wir empfehlen dringend, in Freitextfeldern, z. B. im Feld Name, keine vertraulichen oder sensiblen Informationen wie die E-Mail-Adressen Ihrer Kunden einzugeben. Dies gilt auch, wenn Sie mit der Konsole, der API CodeBuild oder den SDKs arbeiten oder diese anderweitig AWS-Services verwenden. AWS CLI AWS Alle Daten, die Sie in Tags oder Freitextfelder eingeben, die für Namen verwendet werden, können für Abrechnungs- oder Diagnoseprotokolle verwendet werden. Wenn Sie eine URL für einen externen Server bereitstellen, empfehlen wir dringend, keine Anmeldeinformationen zur Validierung Ihrer Anforderung an den betreffenden Server in die URL einzuschließen.

Um vertrauliche Informationen zu schützen, sind die folgenden Informationen in CodeBuild Protokollen versteckt:

- Zeichenketten, die mithilfe des Parameterspeichers in den Umgebungsvariablen des CodeBuild Projekts oder im Abschnitt `env/parameter-store` buildspec angegeben wurden. Weitere

Informationen finden Sie unter Systems Manager Parameter Store und Systems Manager Parameter Store Console Walkthrough im Amazon EC2 Systems Manager Manager-Benutzerhandbuch.

- Zeichenketten, die AWS Secrets Manager in den Umgebungsvariablen CodeBuild des Projekts oder im Abschnitt `env/secrets-manager` buildspec angegeben wurden. Weitere Informationen finden Sie unter Schlüsselverwaltung.

Weitere Informationen zum Datenschutz enthält der Blog-Beitrag AWS Shared Responsibility Model and GDPR im AWS -Sicherheitsblog.

Themen

- Datenverschlüsselung
- Schlüsselverwaltung
- Datenschutz für Datenverkehr

Datenverschlüsselung

Verschlüsselung ist ein wichtiger Teil der Sicherheit. CodeBuild Manche Verschlüsselungen, wie z. B. für Daten in der Übertragung, werden standardmäßig bereitgestellt – Sie müssen nichts zu tun. Andere Verschlüsselungsmöglichkeiten, wie z. B. für Daten im Ruhezustand, können Sie konfigurieren, wenn Sie ein Projekt erstellen oder aufbauen.

- Verschlüsselung ruhender Daten — Build-Artefakte wie Cache, Protokolle, exportierte Rohdatendateien für Testberichte und Buildergebnisse werden standardmäßig mit Von AWS verwaltete Schlüssel verschlüsselt. Wenn Sie diese KMS-Schlüssel nicht verwenden möchten, müssen Sie einen vom Kunden verwalteten Schlüssel erstellen und konfigurieren. Weitere Informationen finden Sie unter Erstellen von KMS-Schlüsseln und AWS Key Management Service-Konzepte im AWS Key Management Service -Benutzerhandbuch.
  - Sie können die Kennung des AWS KMS-Schlüssels, mit dem CodeBuild das Build-Ausgabeartefakt verschlüsselt wird, in der `CODEBUILD_KMS_KEY_ID` Umgebungsvariablen speichern. Weitere Informationen finden Sie unter Umgebungsvariablen in Build-Umgebungen.
  - Sie können einen vom Kunden verwalteten Schlüssel angeben, wenn Sie ein Build-Projekt erstellen. Weitere Informationen finden Sie unter Set the Encryption Key Using the Console und Einstellen des Verschlüsselungsschlüssels mithilfe der CLI.

Die Amazon Elastic Block Store-Volumes Ihrer Build-Flotte sind standardmäßig mit verschlüsselt Von AWS verwaltete Schlüssel.

- Verschlüsselung von Daten während der Übertragung — Die gesamte Kommunikation zwischen Kunden und zwischen CodeBuild CodeBuild und zwischen den nachgelagerten Abhängigkeiten wird durch TLS-Verbindungen geschützt, die mit dem Signature Version 4-Signaturverfahren signiert wurden. Alle CodeBuild Endgeräte verwenden SHA-256-Zertifikate, die von verwaltet werden. AWS Private Certificate Authority Weitere Informationen finden Sie unter Signaturprozess mit Signaturversion 4 und Was ist ACM PCA?.
- Verschlüsselung von Build-Artefakten — Die dem Build-Projekt zugeordnete CodeBuild Servicerolle benötigt Zugriff auf einen KMS-Schlüssel, um die Build-Ausgabeartefakte zu verschlüsseln. CodeBuild Verwendet standardmäßig eine Von AWS verwalteter Schlüssel für Amazon S3 in Ihrem AWS Konto. Wenn Sie dies nicht verwenden möchten Von AWS verwalteter Schlüssel, müssen Sie einen vom Kunden verwalteten Schlüssel erstellen und konfigurieren. Weitere Informationen finden Sie unter Einen kundenverwalteten Schlüssel erstellen Schlüssel erstellen im AWS KMS Entwicklerhandbuch.

Schlüsselverwaltung

Sie können Ihre Inhalte durch Verschlüsselung vor unberechtigtem Zugriff schützen. Speichern Sie Ihre Verschlüsselungsschlüssel in AWS Secrets Manager und erteilen Sie dann der mit dem Build-Projekt verknüpften CodeBuild Servicerolle die Erlaubnis, die Verschlüsselungsschlüssel von Ihrem Secrets Manager Manager-Konto abzurufen. Weitere Informationen finden Sie unter Erstellen und konfigurieren Sie einen kundenverwalteten Schlüssel für CodeBuild, Erstellen eines Build-Projekts in AWS CodeBuild, Ausführen eines Build in AWS CodeBuild und Tutorial: Speichern und Abrufen eines Secrets.

Verwenden Sie die `CODEBUILD_KMS_KEY_ID` Umgebungsvariable in einem Build-Befehl, um die AWS KMS Schlüssel-ID abzurufen. Weitere Informationen finden Sie unter Umgebungsvariablen in Build-Umgebungen.

Sie können Secrets Manager verwenden, um Anmeldeinformationen für eine private Registrierung zu schützen, in der ein Docker-Image gespeichert ist, das für Ihre Laufzeitumgebung verwendet wird. Weitere Informationen finden Sie unter Private Registrierung mit AWS Secrets Manager Beispiel für CodeBuild.

Datenschutz für Datenverkehr

Sie können die Sicherheit Ihrer Builds verbessern, indem Sie die Verwendung eines VPC-Endpunkts mit Schnittstelle konfigurieren CodeBuild . Dafür benötigen Sie kein Internet-Gateway, kein NAT-Gerät und kein virtuelles privates Gateway. Eine Konfiguration ist ebenfalls nicht erforderlich PrivateLink, wird jedoch empfohlen. Weitere Informationen finden Sie unter Verwenden von VPC-Endpunkten. Weitere Informationen zu PrivateLink VPC-Endpunkten finden Sie unter AWS PrivateLinkund Zugreifen auf AWS Dienste über. PrivateLink

Identitäts- und Zugriffsmanagement in AWS CodeBuild

Für den Zugriff auf AWS CodeBuild sind Anmeldeinformationen erforderlich. Diese Anmeldeinformationen müssen über Berechtigungen für den Zugriff auf AWS Ressourcen verfügen, z. B. zum Speichern und Abrufen von Build-Artefakten in S3-Buckets und zum Anzeigen von CloudWatch Amazon-Logs für Builds. In den folgenden Abschnitten wird beschrieben, wie Sie AWS Identity and Access Management(IAM) verwenden und wie Sie CodeBuild den Zugriff auf Ihre Ressourcen sichern können:

Überblick über die Verwaltung der Zugriffsberechtigungen für Ihre Ressourcen AWS CodeBuild

Jede AWS Ressource gehört einem AWS Konto, und die Berechtigungen zum Erstellen oder Zugreifen auf eine Ressource werden durch Berechtigungsrichtlinien geregelt. Ein Kontoadministrator kann Berechtigungsrichtlinien an IAM-Identitäten (Benutzer, Gruppen und Rollen) anfügen.

> **Note**
>
> Ein Kontoadministrator (oder Administratorbenutzer) ist ein Benutzer mit Administratorrechten. Weitere Informationen finden Sie unter Bewährte Methoden für IAM im IAM-Benutzerhandbuch.

Beim Erteilen von Berechtigungen entscheiden Sie, wer die Berechtigungen erhält, für welche Ressourcen diese gelten und welche Aktionen an diesen Ressourcen gestattet werden sollen.

Themen

- AWS CodeBuild Ressourcen und Abläufe

- Grundlegendes zum Eigentum an Ressourcen
- Verwalten des Zugriffs auf Ressourcen
- Festlegen der Richtlinienelemente: Aktionen, Effekte und Prinzipale

AWS CodeBuild Ressourcen und Abläufe

In AWS CodeBuild ist die primäre Ressource ein Build-Projekt. In einer Richtlinie identifizieren Sie die Ressource, für welche die Richtlinie gilt, mithilfe eines Amazon-Ressourcennamens (ARN). Builds sind ebenfalls Ressourcen und haben einen zugeordneten ARN. Weitere Informationen finden Sie unter Amazon Resource Names (ARN) und AWS Service Namespaces in der. Allgemeine Amazon Web Services-Referenz

Ressourcentyp	ARN-Format
Build-Projekt	`arn:aws:codebuild:` *region-ID* `:` *account-ID* `:project/` *project-name*
Entwicklung	`arn:aws:codebuild:` *region-ID* `:` *account-ID* `:build/` *build-ID*
Berichtsgruppe	`arn:aws:codebuild:` *region-ID* `:` *account-ID* `:report-group/` *report-group-name*
Bericht	`arn:aws:codebuild:` *region-ID* `:` *account-ID* `:report/` *report-ID*
Flotte	`arn:aws:codebuild:` *region-ID* `:` *account-ID* `:fleet/` *fleet-ID*
Alle Ressourcen CodeBuild	`arn:aws:codebuild:*`
Alle CodeBuild Ressourcen, die dem angegebenen Konto in der angegebenen AWS Region gehören	`arn:aws:codebuild:` *region-ID* `:` *account-ID* `:*`

> **Important**
>
> Wenn Sie die Funktion für reservierte Kapazität verwenden, können andere Projekte innerhalb desselben Kontos auf Daten zugreifen, die auf Flotteninstanzen zwischengespeichert sind, einschließlich Quelldateien, Docker-Layern und zwischengespeicherten Verzeichnissen, die in der Buildspec angegeben sind. Dies ist beabsichtigt und ermöglicht es Projekten innerhalb desselben Kontos, Flotteninstanzen gemeinsam zu nutzen.

> **Note**
>
> Die meisten AWS Dienste behandeln einen Doppelpunkt (:) oder einen Schrägstrich (/) als dasselbe Zeichen in ARNs. CodeBuild Verwendet jedoch eine exakte Übereinstimmung in den Ressourcenmustern und Regeln. Verwenden Sie also die richtigen Zeichen zum Erstellen von Ereignismustern, sodass sie mit der ARN-Syntax in der Ressource übereinstimmen.

Sie können in Ihrer Anweisung beispielsweise ein bestimmtes Build-Projekt (*myBuildProject*) mit seinem ARN wie folgt angeben:

```
"Resource": "arn:aws:codebuild:us-east-2:123456789012:project/myBuildProject"
```

Wenn Sie alle Ressourcen angeben möchten oder wenn eine API-Aktion keine ARNs unterstützt, verwenden Sie das Platzhalterzeichen (*) im `Resource`-Element, wie folgt:

```
"Resource": "*"
```

Einige CodeBuild API-Aktionen akzeptieren mehrere Ressourcen (z. B.`BatchGetProjects`). Um mehrere Ressourcen in einer einzigen Anweisung anzugeben, trennen Sie die ARNs mit Komma, wie folgt:

```
"Resource": [
  "arn:aws:codebuild:us-east-2:123456789012:project/myBuildProject",
  "arn:aws:codebuild:us-east-2:123456789012:project/myOtherBuildProject"
]
```

CodeBuild bietet eine Reihe von Operationen für die Arbeit mit den CodeBuild Ressourcen. Eine Liste finden Sie hier: AWS CodeBuild Referenz zu Berechtigungen.

Grundlegendes zum Eigentum an Ressourcen

Das AWS Konto besitzt die Ressourcen, die im Konto erstellt wurden, unabhängig davon, wer die Ressourcen erstellt hat. Insbesondere ist der Ressourcenbesitzer das AWS Konto der Prinzipalentität (d. h. das Root-Konto, ein Benutzer oder eine IAM-Rolle), das die Anfrage zur Ressourcenerstellung authentifiziert. Die Funktionsweise wird anhand der folgenden Beispiele deutlich:

- Wenn Sie die Root-Kontoanmeldeinformationen Ihres AWS Kontos verwenden, um eine Regel zu erstellen, ist Ihr AWS Konto der Eigentümer der CodeBuild Ressource.
- Wenn Sie in Ihrem AWS Konto einen Benutzer erstellen und diesem Benutzer Berechtigungen zum Erstellen von CodeBuild Ressourcen gewähren, kann der Benutzer CodeBuild Ressourcen erstellen. Ihr AWS Konto, zu dem der Benutzer gehört, besitzt jedoch die CodeBuild Ressourcen.
- Wenn Sie in Ihrem AWS Konto eine IAM-Rolle mit Berechtigungen zum Erstellen von CodeBuild Ressourcen erstellen, kann jeder, der die Rolle übernehmen kann, CodeBuild Ressourcen erstellen. Ihr AWS Konto, zu dem die Rolle gehört, besitzt die CodeBuild Ressourcen.

Verwalten des Zugriffs auf Ressourcen

Eine Berechtigungsrichtlinie beschreibt, wer Zugriff auf welche Ressourcen hat.

> **Note**
>
> Dieser Abschnitt beschäftigt sich mit der Verwendung von IAM in AWS CodeBuild. Er enthält keine detaillierten Informationen über den IAM-Service. Eine umfassende IAM-Dokumentation finden Sie unter Was ist IAM? im IAM-Benutzerhandbuch. Für Informationen über die Syntax und Beschreibungen von AWS -IAM-Richtlinien lesen Sie die IAM-Richtlinienreferenz im IAM-Benutzerhandbuch.

An eine IAM-Identität angefügte Richtlinien werden als identitätsbasierte Richtlinien (oder IAM-Richtlinien) bezeichnet. Mit einer Ressource verknüpfte Richtlinien werden als ressourcenbasierte Richtlinien bezeichnet. CodeBuild unterstützt identitätsbasierte Richtlinien und ressourcenbasierte Richtlinien für bestimmte schreibgeschützte APIs zum Zwecke der kontenübergreifenden gemeinsamen Nutzung von Ressourcen.

Sicherer Zugriff auf S3-Buckets

Wir empfehlen dringend, dass Sie die folgenden Berechtigungen in Ihre IAM-Rolle aufnehmen, um sicherzustellen, dass der mit Ihrem CodeBuild Projekt verknüpfte S3-Bucket Ihnen oder einer Person gehört, der Sie vertrauen. Diese Berechtigungen sind nicht in AWS verwalteten Richtlinien und Rollen enthalten. Sie müssen sie selbst hinzufügen.

- `s3:GetBucketAcl`
- `s3:GetBucketLocation`

Wenn sich der Besitzer eines von Ihrem Projekt verwendeten S3-Buckets ändert, müssen Sie überprüfen, ob Sie weiterhin Eigentümer des Buckets sind, und andernfalls die Berechtigungen in Ihrer IAM-Rolle aktualisieren. Weitere Informationen finden Sie unter Hinzufügen von CodeBuild Zugriffsberechtigungen zu einer IAM-Gruppe oder einem IAM-Benutzer und Erstellen Sie eine CodeBuild Servicerolle.

Festlegen der Richtlinienelemente: Aktionen, Effekte und Prinzipale

Für jede AWS CodeBuild Ressource definiert der Service eine Reihe von API-Vorgängen. CodeBuild Definiert eine Reihe von Aktionen, die Sie in einer Richtlinie angeben können, um Berechtigungen für diese API-Operationen zu gewähren. Einige API-Operationen erfordern möglicherweise Berechtigungen für mehr als eine Aktion, um die API-Operation auszuführen. Weitere Informationen finden Sie unter AWS CodeBuild Ressourcen und Abläufe und AWS CodeBuild Referenz zu Berechtigungen.

Grundlegende Richtlinienelemente:

- Ressource – Sie verwenden einen Amazon-Ressourcennamen (ARN), um die Ressource, für welche die Richtlinie gilt, zu identifizieren.
- Aktion — Sie verwenden Aktionsschlüsselwörter, um Ressourcenoperationen zu identifizieren, die Sie zulassen oder verweigern möchten. Die `codebuild:CreateProject`-Berechtigung erteilt dem Benutzer zum Beispiel Berechtigungen zum Ausführen der `CreateProject`-Operation.
- Wirkung — Sie geben den Effekt an, entweder zulassen oder verweigern, wenn der Benutzer die Aktion anfordert. Wenn Sie den Zugriff auf eine Ressource nicht ausdrücklich gestatten („Allow“), wird er automatisch verweigert. Sie können den Zugriff auf eine Ressource auch explizit verweigern. So können Sie zum Beispiel sicherstellen, dass Benutzer nicht auf eine Ressource zugreifen können, auch wenn der Zugriff durch eine andere Richtlinie gestattet wird.

- Principal — In identitätsbasierten Richtlinien (IAM-Richtlinien) ist der Benutzer, an den die Richtlinie angehängt ist, der implizite Prinzipal. In ressourcenbasierten Richtlinien müssen Sie den Benutzer, das Konto, den Service oder die sonstige Entität angeben, die die Berechtigungen erhalten soll.

Weitere Informationen zur Syntax und zu Beschreibungen von IAM-Richtlinien finden Sie in der AWS -IAM-Richtlinienreferenz im IAM-Benutzerhandbuch.

Eine Tabelle mit allen CodeBuild API-Aktionen und den Ressourcen, für die sie gelten, finden Sie unter. AWS CodeBuild Referenz zu Berechtigungen

Verwendung identitätsbasierter Richtlinien für AWS CodeBuild

Dieses Thema enthält Beispiele zu identitätsbasierten Richtlinien, die verdeutlichen, wie ein Kontoadministrator IAM-Identitäten (d. h. Benutzern, Gruppen und Rollen) Berechtigungsrichtlinien zuweisen und somit Berechtigungen zur Durchführung von Operationen für AWS CodeBuild -Ressourcen erteilen kann.

> **Important**
>
> Wir empfehlen Ihnen, zunächst die einführenden Themen zu lesen, in denen die grundlegenden Konzepte und Optionen für die Verwaltung des Zugriffs auf Ihre Ressourcen erläutert werden. CodeBuild Weitere Informationen finden Sie unter Überblick über die Verwaltung der Zugriffsberechtigungen für Ihre Ressourcen AWS CodeBuild.

Themen

- Erforderliche Berechtigungen für die Verwendung der AWS CodeBuild -Konsole
- Erforderliche Berechtigungen für AWS CodeBuild die Verbindung mit Amazon Elastic Container Registry
- Für die AWS CodeBuild Konsole sind Berechtigungen erforderlich, um eine Verbindung zu Quellanbietern herzustellen
- AWS verwaltete (vordefinierte) Richtlinien für AWS CodeBuild
- CodeBuild verwaltete Richtlinien und Benachrichtigungen
- CodeBuild Aktualisierungen der AWS verwalteten Richtlinien
- Beispiele für vom Kunden verwaltete Richtlinien

Nachfolgend sehen Sie ein Beispiel für eine Berechtigungsrichtlinie, die einem Benutzer ermöglicht, Informationen über Build-Projekte nur in der Region `us-east-2` für Konto `123456789012` für alle Build-Projekte, die mit dem Namen `my` beginnen, abzurufen:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "codebuild:BatchGetProjects",
      "Resource": "arn:aws:codebuild:us-east-2:123456789012:project/my*"
    }
  ]
}
```

Erforderliche Berechtigungen für die Verwendung der AWS CodeBuild -Konsole

Ein Benutzer, der die AWS CodeBuild Konsole verwendet, muss über Mindestberechtigungen verfügen, die es dem Benutzer ermöglichen, andere AWS Ressourcen für das AWS Konto zu beschreiben. Sie benötigen Berechtigungen für die folgenden Services:

- AWS CodeBuild
- Amazon CloudWatch
- CodeCommit (wenn Sie Ihren Quellcode in einem AWS CodeCommit Repository speichern)
- Amazon Elastic Container Registry (Amazon ECR) (wenn Sie eine Build-Umgebung verwenden, die auf einem Docker-Image in einem Amazon ECR-Repository basiert)

> **Note**
>
> Am 26. Juli 2022 wurde die Standard-IAM-Richtlinie aktualisiert. Weitere Informationen finden Sie unter Erforderliche Berechtigungen für AWS CodeBuild die Verbindung mit Amazon Elastic Container Registry.

- Amazon Elastic Container Service (Amazon ECS) (wenn Sie eine Build-Umgebung verwenden, die auf einem Docker-Image in einem Amazon ECR-Repository basiert)
- AWS Identity and Access Management (IAM)
- AWS Key Management Service (AWS KMS)
- Amazon-Simple-Storage-Service (Amazon-S3)

Wenn Sie eine IAM-Richtlinie erstellen, die restriktiver ist als die erforderlichen Mindestberechtigungen, funktioniert die Konsole nicht wie vorgesehen.

Erforderliche Berechtigungen für AWS CodeBuild die Verbindung mit Amazon Elastic Container Registry

AWS CodeBuild Hat am 26. Juli 2022 seine Standard-IAM-Richtlinie für die Amazon ECR-Genehmigung aktualisiert. Die folgenden Berechtigungen wurden aus der Standardrichtlinie entfernt:

```
"ecr:PutImage",
"ecr:InitiateLayerUpload",
"ecr:UploadLayerPart",
"ecr:CompleteLayerUpload"
```

Für CodeBuild Projekte, die vor dem 26. Juli 2022 erstellt wurden, empfehlen wir Ihnen, Ihre Richtlinie mit der folgenden Amazon ECR-Richtlinie zu aktualisieren:

```
"Action": [
    "ecr:BatchCheckLayerAvailability",
    "ecr:GetDownloadUrlForLayer",
    "ecr:BatchGetImage"
]
```

Weitere Informationen zur Aktualisierung Ihrer Richtlinie finden Sie unterHinzufügen von CodeBuild Zugriffsberechtigungen zu einer IAM-Gruppe oder einem IAM-Benutzer.

Für die AWS CodeBuild Konsole sind Berechtigungen erforderlich, um eine Verbindung zu Quellanbietern herzustellen

Die AWS CodeBuild Konsole verwendet die folgenden API-Aktionen, um eine Verbindung zu Quellanbietern (z. B. GitHub Repositorys) herzustellen.

- `codebuild:ListConnectedOAuthAccounts`
- `codebuild:ListRepositories`
- `codebuild:PersistOAuthToken`
- `codebuild:ImportSourceCredentials`

Mithilfe der Konsole können Sie Quellanbieter (wie GitHub Repositorys) Ihren Build-Projekten zuordnen. AWS CodeBuild Dazu müssen Sie zunächst die oben genannten API-Aktionen zu den

IAM-Zugriffsrichtlinien hinzufügen, die dem Benutzer zugeordnet sind, den Sie für den Zugriff auf die AWS CodeBuild Konsole verwenden.

Die API-Aktionen `ListConnected0AuthAccounts`, `ListRepositories` und `Persist0AuthToken` werden nicht von Ihrem Code aufgerufen. Daher sind diese API-Aktionen nicht in den AWS SDKs AWS CLI und enthalten.

AWS verwaltete (vordefinierte) Richtlinien für AWS CodeBuild

AWS adressiert viele gängige Anwendungsfälle durch die Bereitstellung eigenständiger IAM-Richtlinien, die von erstellt und verwaltet werden. AWS Diese AWS verwalteten Richtlinien gewähren die erforderlichen Berechtigungen für allgemeine Anwendungsfälle, sodass Sie nicht erst untersuchen müssen, welche Berechtigungen benötigt werden. Die verwalteten Richtlinien für gewähren CodeBuild auch Berechtigungen zur Ausführung von Vorgängen in anderen Diensten wie IAM AWS CodeCommit, Amazon EC2, Amazon ECR, Amazon SNS und Amazon CloudWatch Events, je nach den Verantwortlichkeiten der Benutzer, denen die betreffende Richtlinie erteilt wurde. Bei der `AWSCodeBuildAdminAccess` Richtlinie handelt es sich beispielsweise um eine Benutzerrichtlinie auf Administrationsebene, die es Benutzern mit dieser Richtlinie ermöglicht, CloudWatch Veranstaltungsregeln für Projekt-Builds und Amazon SNS SNS-Themen für Benachrichtigungen über projektbezogene Ereignisse (Themen, deren Namen mit einem Präfix versehen sind`arn:aws:codebuild:`) zu erstellen und zu verwalten sowie Projekte und Berichtsgruppen in zu verwalten. CodeBuild Weitere Informationen finden Sie unter AWS -verwaltete Richtlinien im IAM-Benutzerhandbuch.

Die folgenden AWS verwalteten Richtlinien, die Sie Benutzern in Ihrem Konto zuordnen können, sind spezifisch für. AWS CodeBuild

AWSCodeBuildAdminAccess

Bietet vollen Zugriff auf CodeBuild einschließlich Berechtigungen zur Verwaltung von CodeBuild Build-Projekten.

AWSCodeBuildDeveloperAccess

Bietet Zugriff auf die Verwaltung von Build-Projekten, erlaubt CodeBuild aber nicht.

AWSCodeBuildReadOnlyAccess

Bietet schreibgeschützten Zugriff auf. CodeBuild

Um auf die erstellten Build-Ausgabeartefakte zuzugreifen, müssen Sie auch die AWS verwaltete Richtlinie mit dem Namen anhängen. CodeBuild `AmazonS3ReadOnlyAccess`

Um CodeBuild Servicerollen zu erstellen und zu verwalten, müssen Sie auch die AWS verwaltete Richtlinie mit dem Namen anhängen`IAMFullAccess`.

Sie können auch Ihre eigenen benutzerdefinierten IAM-Richtlinien erstellen, um Berechtigungen für CodeBuild Aktionen und Ressourcen zu gewähren. Die benutzerdefinierten Richtlinien können Sie dann den -Benutzern oder -Gruppen zuweisen, die diese Berechtigungen benötigen.

Themen

- AWSCodeBuildAdminAccess
- AWSCodeBuildDeveloperAccess
- AWSCodeBuildReadOnlyAccess

AWSCodeBuildAdminAccess

Die `AWSCodeBuildAdminAccess` Richtlinie bietet vollen Zugriff auf CodeBuild Build-Projekte CodeBuild, einschließlich Berechtigungen zur Verwaltung. Wenden Sie diese Richtlinie nur auf Benutzer auf Administratorebene an, um ihnen die volle Kontrolle über CodeBuild Projekte, Berichtsgruppen und zugehörige Ressourcen in Ihrem AWS Konto zu gewähren, einschließlich der Möglichkeit, Projekte und Berichtsgruppen zu löschen.

Die Richtlinie `AWSCodeBuildAdminAccess` enthält die folgende Richtlinienanweisung:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AWSServicesAccess",
      "Action": [
        "codebuild:*",
        "codecommit:GetBranch",
        "codecommit:GetCommit",
        "codecommit:GetRepository",
        "codecommit:ListBranches",
        "codecommit:ListRepositories",
        "cloudwatch:GetMetricStatistics",
        "ec2:DescribeVpcs",
        "ec2:DescribeSecurityGroups",
```

```
        "ec2:DescribeSubnets",
        "ecr:DescribeRepositories",
        "ecr:ListImages",
        "elasticfilesystem:DescribeFileSystems",
        "events:DeleteRule",
        "events:DescribeRule",
        "events:DisableRule",
        "events:EnableRule",
        "events:ListTargetsByRule",
        "events:ListRuleNamesByTarget",
        "events:PutRule",
        "events:PutTargets",
        "events:RemoveTargets",
        "logs:GetLogEvents",
        "s3:GetBucketLocation",
        "s3:ListAllMyBuckets"
      ],
      "Effect": "Allow",
      "Resource": "*"
    },
    {
      "Sid": "CWLDeleteLogGroupAccess",
      "Action": [
        "logs:DeleteLogGroup"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:logs:*:*:log-group:/aws/codebuild/*:log-stream:*"
    },
    {
      "Sid": "SSMParameterWriteAccess",
      "Effect": "Allow",
      "Action": [
        "ssm:PutParameter"
      ],
      "Resource": "arn:aws:ssm:*:*:parameter/CodeBuild/*"
    },
    {
      "Sid": "SSMStartSessionAccess",
      "Effect": "Allow",
      "Action": [
        "ssm:StartSession"
      ],
      "Resource": "arn:aws:ecs:*:*:task/*/*"
    },
```

```
    {
      "Sid": "CodeStarConnectionsReadWriteAccess",
      "Effect": "Allow",
      "Action": [
        "codestar-connections:CreateConnection",
        "codestar-connections:DeleteConnection",
        "codestar-connections:UpdateConnectionInstallation",
        "codestar-connections:TagResource",
        "codestar-connections:UntagResource",
        "codestar-connections:ListConnections",
        "codestar-connections:ListInstallationTargets",
        "codestar-connections:ListTagsForResource",
        "codestar-connections:GetConnection",
        "codestar-connections:GetIndividualAccessToken",
        "codestar-connections:GetInstallationUrl",
        "codestar-connections:PassConnection",
        "codestar-connections:StartOAuthHandshake",
        "codestar-connections:UseConnection"
      ],
      "Resource": [
        "arn:aws:codestar-connections:*:*:connection/*",
        "arn:aws:codeconnections:*:*:connection/*"
      ]
    },
    {
      "Sid": "CodeStarNotificationsReadWriteAccess",
      "Effect": "Allow",
      "Action": [
        "codestar-notifications:CreateNotificationRule",
        "codestar-notifications:DescribeNotificationRule",
        "codestar-notifications:UpdateNotificationRule",
        "codestar-notifications:DeleteNotificationRule",
        "codestar-notifications:Subscribe",
        "codestar-notifications:Unsubscribe"
      ],
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "codestar-notifications:NotificationsForResource": "arn:aws:codebuild:*"
        }
      }
    },
    {
      "Sid": "CodeStarNotificationsListAccess",
```

```
      "Effect": "Allow",
      "Action": [
        "codestar-notifications:ListNotificationRules",
        "codestar-notifications:ListEventTypes",
        "codestar-notifications:ListTargets",
        "codestar-notifications:ListTagsforResource"
      ],
      "Resource": "*"
    },
    {
      "Sid": "CodeStarNotificationsSNSTopicCreateAccess",
      "Effect": "Allow",
      "Action": [
        "sns:CreateTopic",
        "sns:SetTopicAttributes"
      ],
      "Resource": "arn:aws:sns:*:*:codestar-notifications*"
    },
    {
      "Sid": "SNSTopicListAccess",
      "Effect": "Allow",
      "Action": [
        "sns:ListTopics",
        "sns:GetTopicAttributes"
      ],
      "Resource": "*"
    },
    {
      "Sid": "CodeStarNotificationsChatbotAccess",
      "Effect": "Allow",
      "Action": [
        "chatbot:DescribeSlackChannelConfigurations",
        "chatbot:ListMicrosoftTeamsChannelConfigurations"
      ],
      "Resource": "*"
    }
  ]
}
```

AWSCodeBuildDeveloperAccess

Die `AWSCodeBuildDeveloperAccess` Richtlinie ermöglicht den Zugriff auf alle Funktionen von Ressourcen im Zusammenhang mit Projekt CodeBuild - und Berichtsgruppen. Diese Richtlinie erlaubt

es Benutzern nicht, CodeBuild Projekte oder Berichtsgruppen oder verwandte Ressourcen in anderen AWS Diensten, wie z. B. CloudWatch Ereignisse, zu löschen. Wir empfehlen, dass diese Richtlinie auf die meisten Benutzer anzuwenden.

Die Richtlinie `AWSCodeBuildDeveloperAccess` enthält die folgende Richtlinienanweisung:

```
{
  "Statement": [
    {
      "Sid": "AWSServicesAccess",
      "Action": [
        "codebuild:StartBuild",
        "codebuild:StopBuild",
        "codebuild:StartBuildBatch",
        "codebuild:StopBuildBatch",
        "codebuild:RetryBuild",
        "codebuild:RetryBuildBatch",
        "codebuild:BatchGet*",
        "codebuild:GetResourcePolicy",
        "codebuild:DescribeTestCases",
        "codebuild:DescribeCodeCoverages",
        "codebuild:List*",
        "codecommit:GetBranch",
        "codecommit:GetCommit",
        "codecommit:GetRepository",
        "codecommit:ListBranches",
        "cloudwatch:GetMetricStatistics",
        "events:DescribeRule",
        "events:ListTargetsByRule",
        "events:ListRuleNamesByTarget",
        "logs:GetLogEvents",
        "s3:GetBucketLocation",
        "s3:ListAllMyBuckets"
      ],
      "Effect": "Allow",
      "Resource": "*"
    },
    {
      "Sid": "SSMParameterWriteAccess",
      "Effect": "Allow",
      "Action": [
        "ssm:PutParameter"
      ],
```

```
      "Resource": "arn:aws:ssm:*:*:parameter/CodeBuild/*"
    },
    {
      "Sid": "SSMStartSessionAccess",
      "Effect": "Allow",
      "Action": [
        "ssm:StartSession"
      ],
      "Resource": "arn:aws:ecs:*:*:task/*/*"
    },
    {
      "Sid": "CodeStarConnectionsUserAccess",
      "Effect": "Allow",
      "Action": [
        "codestar-connections:ListConnections",
        "codestar-connections:GetConnection"
      ],
      "Resource": [
        "arn:aws:codestar-connections:*:*:connection/*",
        "arn:aws:codeconnections:*:*:connection/*"
      ]
    },
    {
      "Sid": "CodeStarNotificationsReadWriteAccess",
      "Effect": "Allow",
      "Action": [
        "codestar-notifications:CreateNotificationRule",
        "codestar-notifications:DescribeNotificationRule",
        "codestar-notifications:UpdateNotificationRule",
        "codestar-notifications:Subscribe",
        "codestar-notifications:Unsubscribe"
      ],
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "codestar-notifications:NotificationsForResource": "arn:aws:codebuild:*"
        }
      }
    },
    {
      "Sid": "CodeStarNotificationsListAccess",
      "Effect": "Allow",
      "Action": [
        "codestar-notifications:ListNotificationRules",
```

```
        "codestar-notifications:ListEventTypes",
        "codestar-notifications:ListTargets",
        "codestar-notifications:ListTagsforResource"
      ],
      "Resource": "*"
    },
    {
      "Sid": "SNSTopicListAccess",
      "Effect": "Allow",
      "Action": [
        "sns:ListTopics",
        "sns:GetTopicAttributes"
      ],
      "Resource": "*"
    },
    {
      "Sid": "CodeStarNotificationsChatbotAccess",
      "Effect": "Allow",
      "Action": [
        "chatbot:DescribeSlackChannelConfigurations",
        "chatbot:ListMicrosoftTeamsChannelConfigurations"
      ],
      "Resource": "*"
    }
  ],
  "Version": "2012-10-17"
}
```

AWSCodeBuildReadOnlyAccess

Die `AWSCodeBuildReadOnlyAccess` Richtlinie gewährt nur Lesezugriff auf CodeBuild und verwandte Ressourcen in anderen AWS Diensten. Wenden Sie diese Richtlinie auf Benutzer an, die Builds anzeigen und ausführen, Projekte anzeigen und Berichtsgruppen anzeigen können, jedoch keine Änderungen für sie ausführen dürfen.

Die Richtlinie `AWSCodeBuildReadOnlyAccess` enthält die folgende Richtlinienanweisung:

```
{
  "Statement": [
    {
      "Sid": "AWSServicesAccess",
      "Action": [
        "codebuild:BatchGet*",
```

```
        "codebuild:GetResourcePolicy",
        "codebuild:List*",
        "codebuild:DescribeTestCases",
        "codebuild:DescribeCodeCoverages",
        "codecommit:GetBranch",
        "codecommit:GetCommit",
        "codecommit:GetRepository",
        "cloudwatch:GetMetricStatistics",
        "events:DescribeRule",
        "events:ListTargetsByRule",
        "events:ListRuleNamesByTarget",
        "logs:GetLogEvents"
      ],
      "Effect": "Allow",
      "Resource": "*"
    },
    {
      "Sid": "CodeStarConnectionsUserAccess",
      "Effect": "Allow",
      "Action": [
        "codestar-connections:ListConnections",
        "codestar-connections:GetConnection"
      ],
      "Resource": [
        "arn:aws:codestar-connections:*:*:connection/*",
        "arn:aws:codeconnections:*:*:connection/*"
      ]
    },
    {
      "Sid": "CodeStarNotificationsPowerUserAccess",
      "Effect": "Allow",
      "Action": [
        "codestar-notifications:DescribeNotificationRule"
      ],
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "codestar-notifications:NotificationsForResource": "arn:aws:codebuild:*"
        }
      }
    },
    {
      "Sid": "CodeStarNotificationsListAccess",
      "Effect": "Allow",
```

```
      "Action": [
        "codestar-notifications:ListNotificationRules",
        "codestar-notifications:ListEventTypes",
        "codestar-notifications:ListTargets"
      ],
      "Resource": "*"
    }
  ],
  "Version": "2012-10-17"
}
```

CodeBuild verwaltete Richtlinien und Benachrichtigungen

CodeBuild unterstützt Benachrichtigungen, mit denen Benutzer über wichtige Änderungen an Build-Projekten informiert werden können. Zu den verwalteten Richtlinien CodeBuild gehören auch Richtlinienerklärungen für die Benachrichtigungsfunktion. Weitere Informationen finden Sie unter Was sind Benachrichtigungen?.

Berechtigungen in Zusammenhang mit Benachrichtigungen in verwalteten Vollzugriffsrichtlinien

Die von `AWSCodeBuildFullAccess` verwaltete Richtlinie enthält die folgenden Anweisungen, um den vollständigen Zugriff auf Benachrichtigungen zu ermöglichen. Benutzer, für die diese verwaltete Richtlinie gilt, können auch Amazon SNS SNS-Themen für Benachrichtigungen erstellen und verwalten, Benutzer für Themen abonnieren und abbestellen, Themen auflisten, die als Ziele für Benachrichtigungsregeln ausgewählt werden sollen, und für Slack konfigurierte AWS Chatbot Clients auflisten.

```
    {
        "Sid": "CodeStarNotificationsReadWriteAccess",
        "Effect": "Allow",
        "Action": [
            "codestar-notifications:CreateNotificationRule",
            "codestar-notifications:DescribeNotificationRule",
            "codestar-notifications:UpdateNotificationRule",
            "codestar-notifications:DeleteNotificationRule",
            "codestar-notifications:Subscribe",
            "codestar-notifications:Unsubscribe"
        ],
        "Resource": "*",
        "Condition" : {
            "StringLike" : {"codestar-notifications:NotificationsForResource" :
 "arn:aws:codebuild:*"}
```

```
        }
    },
    {
        "Sid": "CodeStarNotificationsListAccess",
        "Effect": "Allow",
        "Action": [
            "codestar-notifications:ListNotificationRules",
            "codestar-notifications:ListTargets",
            "codestar-notifications:ListTagsforResource",
            "codestar-notifications:ListEventTypes"
        ],
        "Resource": "*"
    },
    {
        "Sid": "CodeStarNotificationsSNSTopicCreateAccess",
        "Effect": "Allow",
        "Action": [
            "sns:CreateTopic",
            "sns:SetTopicAttributes"
        ],
        "Resource": "arn:aws:sns:*:*:codestar-notifications*"
    },
    {
        "Sid": "SNSTopicListAccess",
        "Effect": "Allow",
        "Action": [
            "sns:ListTopics"
        ],
        "Resource": "*"
    },
    {
        "Sid": "CodeStarNotificationsChatbotAccess",
        "Effect": "Allow",
        "Action": [
            "chatbot:DescribeSlackChannelConfigurations",
            "chatbot:ListMicrosoftTeamsChannelConfigurations"
          ],
       "Resource": "*"
    }
```

Berechtigungen in Zusammenhang mit Benachrichtigungen in schreibgeschützten verwalteten Richtlinien

Die verwaltete Richtlinie `AWSCodeBuildReadOnlyAccess` enthält die folgenden Anweisungen, um schreibgeschützten Zugriff auf Benachrichtigungen zu ermöglichen. Benutzer mit dieser verwalteten Richtlinie können Benachrichtigungen für Ressourcen anzeigen, sie können sie jedoch nicht erstellen, verwalten oder abonnieren.

```
  {
       "Sid": "CodeStarNotificationsPowerUserAccess",
       "Effect": "Allow",
       "Action": [
           "codestar-notifications:DescribeNotificationRule"
       ],
       "Resource": "*",
       "Condition" : {
           "StringLike" : {"codestar-notifications:NotificationsForResource" :
"arn:aws:codebuild:*"}
       }
   },
   {
       "Sid": "CodeStarNotificationsListAccess",
       "Effect": "Allow",
       "Action": [
           "codestar-notifications:ListNotificationRules",
           "codestar-notifications:ListEventTypes",
           "codestar-notifications:ListTargets"
       ],
       "Resource": "*"
   }
```

Berechtigungen in Zusammenhang mit Benachrichtigungen in anderen verwalteten Richtlinien

Die verwaltete Richtlinie `AWSCodeBuildDeveloperAccess` enthält die folgenden Anweisungen, mit denen Sie Benutzern erlauben können, Benachrichtigungen zu erstellen, zu bearbeiten und zu abonnieren. Benutzer können Benachrichtigungsregeln nicht löschen und auch keine Tags für Ressourcen verwalten.

```
   {
       "Sid": "CodeStarNotificationsReadWriteAccess",
       "Effect": "Allow",
       "Action": [
```

```
            "codestar-notifications:CreateNotificationRule",
            "codestar-notifications:DescribeNotificationRule",
            "codestar-notifications:UpdateNotificationRule",
            "codestar-notifications:Subscribe",
            "codestar-notifications:Unsubscribe"
        ],
        "Resource": "*",
        "Condition" : {
            "StringLike" : {"codestar-notifications:NotificationsForResource" :
"arn:aws:codebuild*"}
        }
    },
    {
        "Sid": "CodeStarNotificationsListAccess",
        "Effect": "Allow",
        "Action": [
            "codestar-notifications:ListNotificationRules",
            "codestar-notifications:ListTargets",
            "codestar-notifications:ListTagsforResource",
            "codestar-notifications:ListEventTypes"
        ],
        "Resource": "*"
    },
    {
        "Sid": "SNSTopicListAccess",
        "Effect": "Allow",
        "Action": [
            "sns:ListTopics"
        ],
        "Resource": "*"
    },
    {
        "Sid": "CodeStarNotificationsChatbotAccess",
        "Effect": "Allow",
        "Action": [
            "chatbot:DescribeSlackChannelConfigurations",
            "chatbot:ListMicrosoftTeamsChannelConfigurations"
          ],
       "Resource": "*"
    }
```

Weitere Informationen zu IAM und Benachrichtigungen findest du unter Identity and Access Management Benachrichtigungen. AWS CodeStar

CodeBuild Aktualisierungen der AWS verwalteten Richtlinien

Hier finden Sie Informationen zu Aktualisierungen AWS verwalteter Richtlinien, die CodeBuild seit Beginn der Nachverfolgung dieser Änderungen durch diesen Dienst vorgenommen wurden. Abonnieren Sie den RSS-Feed auf, um automatische Benachrichtigungen über Änderungen an dieser Seite zu erhaltenAWS CodeBuild Dokumenthistorie des Benutzerhandbuches .

Änderung	Beschreibung	Datum
`AWSCodeBuildAdminAccess` `AWSCodeBuildDeveloperAccess` , und `AWSCodeBuildReadOnlyAccess` — Aktualisierung vorhandener Richtlinien	CodeBuild hat diesen Richtlinien eine Ressource hinzugefügt, um das AWS CodeConnections Rebranding zu unterstützen. Die `AWSCodeBuildReadOnlyAccess` Richtlinien `AWSCodeBuildAdminAccess` `AWSCodeBuildDeveloperAccess` , und wurden geändert, um eine Ressource hinzuzufügen,`arn:aws:codeconnections:*:*:connection/*` .	18. April 2024
`AWSCodeBuildAdminAccess` und `AWSCodeBuildDeveloperAccess` — Aktualisierung der bestehenden Richtlinien	CodeBuild diesen Richtlinien wurde eine Berechtigung hinzugefügt, um einen zusätzlichen Benachrichtigungstyp zu unterstützen, indem AWS Chatbot Die `AWSCodeBuildDeveloperAccess` Richtlinien `AWSCodeBuildAdminAccess` und wurden geändert,	16. Mai 2023

Änderung	Beschreibung	Datum
	um eine Berechtigung hinzuzufügen,`chatbot:ListMicrosoftTeamsChannelConfigurations` .	
CodeBuild hat begonnen, Änderungen zu verfolgen	CodeBuild hat begonnen, Änderungen für die AWS verwalteten Richtlinien zu verfolgen.	16. Mai 2021

Beispiele für vom Kunden verwaltete Richtlinien

In diesem Abschnitt finden Sie Beispiele für Benutzerrichtlinien, die Berechtigungen für diverse AWS CodeBuild -Aktionen erteilen. Diese Richtlinien funktionieren, wenn Sie die CodeBuild API, AWS SDKs oder verwenden. AWS CLI Bei Verwendung der Konsole müssen Sie zusätzliche konsolenspezifische Berechtigungen erteilen. Weitere Informationen finden Sie unter Erforderliche Berechtigungen für die Verwendung der AWS CodeBuild -Konsole.

Sie können die folgenden IAM-Beispielrichtlinien verwenden, um den CodeBuild Zugriff für Ihre Benutzer und Rollen einzuschränken.

Themen

- Benutzern das Abrufen von Informationen über Build-Projekte ermöglichen
- Ermöglicht es einem Benutzer, Informationen über Flotten abzurufen
- Benutzern das Abrufen von Informationen über Berichtsgruppen ermöglichen
- Benutzern das Abrufen von Informationen über Berichte ermöglichen
- Benutzern das Erstellen von Build-Projekten ermöglichen
- Erlaubt einem Benutzer, eine Flotte zu erstellen
- Benutzern das Erstellen einer Berichtsgruppe ermöglichen
- Erlaubt einem Benutzer, eine Flotte zu löschen
- Benutzern das Löschen einer Berichtsgruppe ermöglichen
- Benutzern das Löschen eines Berichts ermöglichen

- Benutzern das Löschen von Build-Projekten ermöglichen
- Benutzern das Abrufen einer Liste mit Build-Projektnamen ermöglichen
- Benutzern das Ändern von Informationen über Build-Projekte ermöglichen
- Erlaubt einem Benutzer, eine Flotte zu ändern
- Benutzern das Ändern einer Berichtsgruppe ermöglichen
- Benutzern das Abrufen von Informationen über Builds ermöglichen
- Benutzern das Abrufen einer Liste mit Build-IDs für ein Build-Projekt ermöglichen
- Benutzern das Abrufen einer Liste mit Build-IDs ermöglichen
- Erlaubt einem Benutzer, eine Liste von Flotten abzurufen
- Benutzern das Abrufen einer Liste von Berichtsgruppen ermöglichen
- Benutzern das Abrufen einer Liste von Berichten ermöglichen
- Benutzern das Abrufen einer Liste von Berichten für eine Berichtsgruppe ermöglichen
- Benutzern das Abrufen einer Liste von Testfällen für einen Bericht ermöglichen
- Benutzern das Ausführen eines Builds ermöglichen
- Benutzern das Stoppen von Builds ermöglichen
- Benutzern das Löschen von Builds ermöglichen
- Erlaubt einem Benutzer, Informationen über Docker-Images abzurufen, die verwaltet werden von CodeBuild
- Erlaubt einem Benutzer, eine Berechtigungsrichtlinie für eine Flottenservice-Rolle hinzuzufügen
- Erlauben Sie den CodeBuild Zugriff auf AWS Dienste, die zum Erstellen einer VPC-Netzwerkschnittstelle erforderlich sind
- Verwenden Sie eine Deny-Anweisung, um zu verhindern, dass die Verbindung zu den Quellanbietern AWS CodeBuild getrennt wird

Benutzern das Abrufen von Informationen über Build-Projekte ermöglichen

Nachfolgend sehen Sie ein Beispiel für eine Richtlinienanweisung, die einem Benutzer ermöglicht, Informationen über Build-Projekte in der Region `us-east-2` für Konto `123456789012` für alle Build-Projekte, die mit dem Namen `my` beginnen, abzurufen:

```
{
```

```
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "codebuild:BatchGetProjects",
      "Resource": "arn:aws:codebuild:us-east-2:123456789012:project/my*"
    }
  ]
}
```

Ermöglicht es einem Benutzer, Informationen über Flotten abzurufen

Das folgende Beispiel für eine Richtlinienanweisung ermöglicht es einem Benutzer, Informationen über Flotten in der `us-east-2` Region auf eigene Rechnung abzurufen: `123456789012`

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "codebuild:BatchGetFleets",
      "Resource": "arn:aws:codebuild:us-east-2:123456789012:fleet/*"
    }
  ]
}
```

Benutzern das Abrufen von Informationen über Berichtsgruppen ermöglichen

Die folgende Beispielrichtlinienanweisung ermöglicht es einem Benutzer, Informationen zu Berichtsgruppen in der `us-east-2`-Region für das Konto `123456789012` abzurufen:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "codebuild:BatchGetReportGroups",
      "Resource": "arn:aws:codebuild:us-east-2:123456789012:report-group/*"
    }
  ]
}
```

Benutzern das Abrufen von Informationen über Berichte ermöglichen

Die folgende Beispielrichtlinienanweisung ermöglicht es einem Benutzer, Informationen zu Berichten in der `us-east-2`-Region für das Konto `123456789012` abzurufen:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "codebuild:BatchGetReports",
      "Resource": "arn:aws:codebuild:us-east-2:123456789012:report-group/*"
    }
  ]
}
```

Benutzern das Erstellen von Build-Projekten ermöglichen

Die folgende Beispiel-Richtlinienanweisung ermöglicht es einem Benutzer, Build-Projekte mit einem beliebigen Namen zu erstellen, jedoch nur in der `us-east-2` Region für das Konto `123456789012` und nur unter Verwendung der angegebenen CodeBuild Servicerolle:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "codebuild:CreateProject",
      "Resource": "arn:aws:codebuild:us-east-2:123456789012:project/*"
    },
    {
      "Effect": "Allow",
      "Action": "iam:PassRole",
      "Resource": "arn:aws:iam::123456789012:role/CodeBuildServiceRole"
    }
  ]
}
```

Die folgende beispielhafte Richtlinienanweisung ermöglicht es einem Benutzer, Build-Projekte mit einem beliebigen Namen zu erstellen, jedoch nur in der `us-east-2` Region für das Konto `123456789012` und nur unter Verwendung der angegebenen CodeBuild Servicerolle. Außerdem

wird festgelegt, dass der Benutzer die angegebene Servicerolle nur zusammen mit AWS CodeBuild anderen AWS Diensten verwenden kann.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "codebuild:CreateProject",
      "Resource": "arn:aws:codebuild:us-east-2:123456789012:project/*"
    },
    {
      "Effect": "Allow",
      "Action": "iam:PassRole",
      "Resource": "arn:aws:iam::123456789012:role/CodeBuildServiceRole",
      "Condition": {
          "StringEquals": {"iam:PassedToService": "codebuild.amazonaws.com"}
      }
    }
  ]
}}
```

Erlaubt einem Benutzer, eine Flotte zu erstellen

Das folgende Beispiel für eine Richtlinienerklärung ermöglicht es einem Benutzer, eine Flotte in der `us-east-2` Region für ein Konto `123456789012` zu erstellen:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "codebuild:CreateFleet",
      "Resource": "arn:aws:codebuild:us-east-2:123456789012:fleet/*"
    }
  ]
}
```

Benutzern das Erstellen einer Berichtsgruppe ermöglichen

Die folgende Beispielrichtlinienanweisung ermöglicht es einem Benutzer, eine Berichtsgruppe in der Region `us-east-2` für das Konto `123456789012` zu erstellen:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "codebuild:CreateReportGroup",
      "Resource": "arn:aws:codebuild:us-east-2:123456789012:report-group/*"
    }
  ]
}
```

Erlaubt einem Benutzer, eine Flotte zu löschen

Das folgende Beispiel für eine Richtlinienerklärung ermöglicht es einem Benutzer, eine Flotte in der `us-east-2` Region für ein Konto `123456789012` zu löschen:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "codebuild:DeleteFleet",
      "Resource": "arn:aws:codebuild:us-east-2:123456789012:fleet/*"
    }
  ]
}
```

Benutzern das Löschen einer Berichtsgruppe ermöglichen

Die folgende Beispielrichtlinienanweisung ermöglicht es einem Benutzer, eine Berichtsgruppe in der `us-east-2`-Region für das Konto `123456789012` zu löschen:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "codebuild:DeleteReportGroup",
      "Resource": "arn:aws:codebuild:us-east-2:123456789012:report-group/*"
    }
  ]
```

```
}
```

Benutzern das Löschen eines Berichts ermöglichen

Die folgende Beispielrichtlinienanweisung ermöglicht es einem Benutzer, einen Bericht in der `us-east-2`-Region für das Konto `123456789012` zu löschen:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "codebuild:DeleteReport",
      "Resource": "arn:aws:codebuild:us-east-2:123456789012:report-group/*"
    }
  ]
}
```

Benutzern das Löschen von Build-Projekten ermöglichen

Nachfolgend sehen Sie ein Beispiel für eine Richtlinienanweisung, die einem Benutzer ermöglicht, Build-Projekte in der Region `us-east-2` für Konto `123456789012` für alle Build-Projekte, die mit dem Namen `my` beginnen, zu löschen:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "codebuild:DeleteProject",
      "Resource": "arn:aws:codebuild:us-east-2:123456789012:project/my*"
    }
  ]
}
```

Benutzern das Abrufen einer Liste mit Build-Projektnamen ermöglichen

Nachfolgend sehen Sie ein Beispiel für eine Richtlinienanweisung, die einem Benutzer ermöglicht, eine Liste mit Build-Projektnamen für dasselbe Konto abzurufen:

```
{
```

```
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "codebuild:ListProjects",
      "Resource": "*"
    }
  ]
}
```

Benutzern das Ändern von Informationen über Build-Projekte ermöglichen

Nachfolgend sehen Sie ein Beispiel für eine Richtlinienanweisung, die einem Benutzer ermöglicht, Informationen über Build-Projekte mit einem beliebigen Namen zu ändern, aber nur in der Region `us-east-2` für Konto `123456789012` und nur mit der angegebenen Servicerolle AWS CodeBuild :

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "codebuild:UpdateProject",
      "Resource": "arn:aws:codebuild:us-east-2:123456789012:project/*"
    },
    {
      "Effect": "Allow",
      "Action": "iam:PassRole",
      "Resource": "arn:aws:iam::123456789012:role/CodeBuildServiceRole"
    }
  ]
}
```

Erlaubt einem Benutzer, eine Flotte zu ändern

Das folgende Beispiel für eine Richtlinienerklärung ermöglicht es einem Benutzer, eine Flotte in der `us-east-2` Region für ein Konto `123456789012` zu ändern:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
```

```
      "Action": "codebuild:UpdateFleet",
      "Resource": "arn:aws:codebuild:us-east-2:123456789012:fleet/*"
    }
  ]
}
```

Benutzern das Ändern einer Berichtsgruppe ermöglichen

Die folgende Beispielrichtlinienanweisung ermöglicht es einem Benutzer, eine Berichtsgruppe in der `us-east-2`-Region für das Konto `123456789012` zu ändern:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "codebuild:UpdateReportGroup",
      "Resource": "arn:aws:codebuild:us-east-2:123456789012:report-group/*"
    }
  ]
}
```

Benutzern das Abrufen von Informationen über Builds ermöglichen

Nachfolgend sehen Sie ein Beispiel für eine Richtlinienanweisung, die einem Benutzer ermöglicht, Informationen über Builds in der Region `us-east-2` für Konto `123456789012` für Build-Projekte mit den Namen `my-build-project` und `my-other-build-project` abzurufen:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "codebuild:BatchGetBuilds",
      "Resource": [
        "arn:aws:codebuild:us-east-2:123456789012:project/my-build-project",
        "arn:aws:codebuild:us-east-2:123456789012:project/my-other-build-project"
      ]
    }
  ]
}
```

Benutzern das Abrufen einer Liste mit Build-IDs für ein Build-Projekt ermöglichen

Nachfolgend sehen Sie ein Beispiel für eine Richtlinienanweisung, die einem Benutzer ermöglicht, eine Liste mit Build-IDs in der Region `us-east-2` für Konto `123456789012` für Build-Projekte mit den Namen `my-build-project` und `my-other-build-project` abzurufen:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "codebuild:ListBuildsForProject",
      "Resource": [
        "arn:aws:codebuild:us-east-2:123456789012:project/my-build-project",
        "arn:aws:codebuild:us-east-2:123456789012:project/my-other-build-project"
      ]
    }
  ]
}
```

Benutzern das Abrufen einer Liste mit Build-IDs ermöglichen

Nachfolgend sehen Sie ein Beispiel für eine Richtlinienanweisung, die einem Benutzer ermöglicht, eine Liste aller Build-IDs für dasselbe Konto abzurufen:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "codebuild:ListBuilds",
      "Resource": "*"
    }
  ]
}
```

Erlaubt einem Benutzer, eine Liste von Flotten abzurufen

Die folgende beispielhafte Richtlinienanweisung ermöglicht es einem Benutzer, eine Liste von Flotten in der `us-east-2` Region für sein Konto abzurufen: `123456789012`

```
{
```

```
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "codebuild:ListFleets",
      "Resource": "*"
    }
  ]
}
```

Benutzern das Abrufen einer Liste von Berichtsgruppen ermöglichen

Mit der folgenden Beispielrichtlinienanweisung kann ein Benutzer eine Liste der Berichtsgruppen in der `us-east-2`-Region für das Konto `123456789012` abrufen:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "codebuild:ListReportGroups",
      "Resource": "*"
    }
  ]
}
```

Benutzern das Abrufen einer Liste von Berichten ermöglichen

Mit der folgenden Beispielrichtlinienanweisung kann ein Benutzer eine Liste von Berichten in der `us-east-2`-Region für das Konto `123456789012` abrufen:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "codebuild:ListReports",
      "Resource": "*"
    }
  ]
}
```

Benutzern das Abrufen einer Liste von Berichten für eine Berichtsgruppe ermöglichen

Mit der folgenden Beispielrichtlinienanweisung kann ein Benutzer eine Liste von Berichten für eine Berichtsgruppe in der `us-east-2`-Region für das Konto `123456789012` abrufen:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "codebuild:ListReportsForReportGroup",
      "Resource": "arn:aws:codebuild:us-east-2:123456789012:report-group/*"
    }
  ]
}
```

Benutzern das Abrufen einer Liste von Testfällen für einen Bericht ermöglichen

Mit der folgenden Beispielrichtlinienanweisung kann ein Benutzer eine Liste von Testfällen für einen Bericht in der `us-east-2`-Region für das Konto `123456789012` abrufen:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "codebuild:DescribeTestCases",
      "Resource": "arn:aws:codebuild:us-east-2:123456789012:report-group/*"
    }
  ]
}
```

Benutzern das Ausführen eines Builds ermöglichen

Nachfolgend sehen Sie ein Beispiel für eine Richtlinienanweisung, die einem Benutzer ermöglicht, Builds in der Region `us-east-2` für Konto `123456789012` für alle Build-Projekte, die mit dem Namen `my` beginnen, auszuführen:

```
{
  "Version": "2012-10-17",
  "Statement": [
```

```
    {
      "Effect": "Allow",
      "Action": "codebuild:StartBuild",
      "Resource": "arn:aws:codebuild:us-east-2:123456789012:project/my*"
    }
  ]
}
```

Benutzern das Stoppen von Builds ermöglichen

Nachfolgend sehen Sie ein Beispiel für eine Richtlinienanweisung, die einem Benutzer ermöglicht, ausgeführte Builds nur in der Region `us-east-2` für Konto `123456789012` für alle Build-Projekte, die mit dem Namen `my` beginnen, zu stoppen:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "codebuild:StopBuild",
      "Resource": "arn:aws:codebuild:us-east-2:123456789012:project/my*"
    }
  ]
}
```

Benutzern das Löschen von Builds ermöglichen

Nachfolgend finden Sie ein Beispiel für eine Richtlinienanweisung, die einem Benutzer ermöglicht, Builds nur in der Region `us-east-2` für Konto `123456789012` zu löschen, wenn der Name des Build-Projekts mit `my` beginnt:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "codebuild:BatchDeleteBuilds",
      "Resource": "arn:aws:codebuild:us-east-2:123456789012:project/my*"
    }
  ]
}
```

Erlaubt einem Benutzer, Informationen über Docker-Images abzurufen, die verwaltet werden von CodeBuild

Die folgende Beispiel-Richtlinienanweisung ermöglicht es einem Benutzer, Informationen über alle Docker-Images abzurufen, die von verwaltet werden: CodeBuild

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "codebuild:ListCuratedEnvironmentImages",
      "Resource": "*"
    }
  ]
}
```

Erlaubt einem Benutzer, eine Berechtigungsrichtlinie für eine Flottenservice-Rolle hinzuzufügen

Das folgende Beispiel für eine Ressourcenrichtlinienanweisung ermöglicht es einem Benutzer, eine Berechtigungsrichtlinie für eine Flottenservice-Rolle hinzuzufügen:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "CodeBuildFleetVpcCreateNI",
      "Effect": "Allow",
      "Action": [
        "ec2:CreateNetworkInterface"
      ],
      "Resource": [
        "arn:aws:ec2:region:account-id:subnet/subnet-id-1",
        "arn:aws:ec2:region:account-id:security-group/security-group-id-1",
        "arn:aws:ec2:region:account-id:network-interface/*"
      ]
    },
    {
      "Sid": "CodeBuildFleetVpcPermission",
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeDhcpOptions",
        "ec2:DescribeNetworkInterfaces",
```

```
        "ec2:DescribeSecurityGroups",
        "ec2:DescribeSubnets",
        "ec2:DescribeVpcs",
        "ec2:ModifyNetworkInterfaceAttribute",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    },
    {
      "Sid": "CodeBuildFleetVpcNIPermission",
      "Effect": "Allow",
      "Action": [
        "ec2:CreateNetworkInterfacePermission"
      ],
      "Resource": "arn:aws:ec2:region:account-id:network-interface/*",
      "Condition": {
        "StringEquals": {
          "ec2:Subnet": [
            "arn:aws:ec2:region:account-id:subnet/subnet-id-1"
          ]
        }
      }
    }
  ]
}
```

Das folgende Beispiel für eine Vertrauensrichtlinien-Erklärung ermöglicht es einem Benutzer, eine Berechtigungsrichtlinie für eine Flottenservice-Rolle hinzuzufügen:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "CodeBuildFleetVPCTrustPolicy",
      "Effect": "Allow",
      "Principal": {
        "Service": "codebuild.amazonaws.com"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "account-id"
        }
```

```
      }
    }
  ]
}
```

Erlauben Sie den CodeBuild Zugriff auf AWS Dienste, die zum Erstellen einer VPC-Netzwerkschnittstelle erforderlich sind

Die folgende Beispiel-Richtlinienanweisung erteilt die AWS CodeBuild Erlaubnis, eine Netzwerkschnittstelle in einer VPC mit zwei Subnetzen zu erstellen:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:CreateNetworkInterface",
        "ec2:DescribeDhcpOptions",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface",
        "ec2:DescribeSubnets",
        "ec2:DescribeSecurityGroups",
        "ec2:DescribeVpcs"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "ec2:CreateNetworkInterfacePermission"
      ],
      "Resource": "arn:aws:ec2:region:account-id:network-interface/*",
      "Condition": {
        "StringEquals": {
          "ec2:AuthorizedService": "codebuild.amazonaws.com"
        },
        "ArnEquals": {
          "ec2:Subnet": [
            "arn:aws:ec2:region:account-id:subnet/subnet-id-1",
            "arn:aws:ec2:region:account-id:subnet/subnet-id-2"
          ]
        }
```

```
      }
    }
  ]
}
```

Verwenden Sie eine Deny-Anweisung, um zu verhindern, dass die Verbindung zu den Quellanbietern AWS CodeBuild getrennt wird

Nachfolgend sehen Sie ein Beispiel für eine Richtlinienanweisung, die eine Deny-Anweisung verwendet, um die Trennung zwischen AWS CodeBuild und Quellanbietern zu verhindern. In der Anweisung wird `codebuild:DeleteOAuthToken`, die Umkehrung von `codebuild:PersistOAuthToken` und `codebuild:ImportSourceCredentials`, verwendet, um eine Verbindung zu Quellanbietern herzustellen. Weitere Informationen finden Sie unter Für die AWS CodeBuild Konsole sind Berechtigungen erforderlich, um eine Verbindung zu Quellanbietern herzustellen.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "codebuild:DeleteOAuthToken",
      "Resource": "*"
    }
  ]
}
```

AWS CodeBuild Referenz zu Berechtigungen

Sie können in Ihren AWS CodeBuild Richtlinien AWS Bedingungsschlüssel für alle Bereiche verwenden, um Bedingungen auszudrücken. Eine Liste finden Sie unter Verfügbare Schlüssel im IAM-Benutzerhandbuch.

Sie geben die Aktionen im Feld `Action` der Richtlinie an. Um eine Aktion anzugeben, verwenden Sie das Präfix `codebuild:` gefolgt vom Namen der API-Operation (z. B. `codebuild:CreateProject` und `codebuild:StartBuild`). Um mehrere Aktionen in einer einzigen Anweisung anzugeben, trennen Sie sie mit Komma (z. B. `"Action": [ "codebuild:CreateProject", "codebuild:StartBuild" ]`).

Verwenden von Platzhalterzeichen

Sie geben einen ARN mit oder ohne Platzhalterzeichen (*) als Ressourcenwert im Feld `Resource` der Richtlinie an. Sie können das Platzhalterzeichen verwenden, um mehrere Aktionen oder Ressourcen anzugeben. `codebuild:*`Gibt beispielsweise alle Aktionen an und `codebuild:Batch*` gibt alle CodeBuild CodeBuild Aktionen an, die mit dem Wort `Batch` beginnen. Im folgenden Beispiel wird der Zugriff auf alle Build-Projekte erteilt, deren Name mit `my` beginnt:

```
arn:aws:codebuild:us-east-2:123456789012:project/my*
```

CodeBuild API-Operationen und erforderliche Berechtigungen für Aktionen

BatchDeleteBuilds

Aktion: `codebuild:BatchDeleteBuilds`

Erforderlich zum Löschen von Builds.

Ressource: `arn:aws:codebuild:`*region-ID*`:`*account-ID*`:project/`*project-name*

BatchGetBuilds

Aktion: `codebuild:BatchGetBuilds`

Erforderlich, um Informationen über Builds abzurufen.

Ressource: `arn:aws:codebuild:`*region-ID*`:`*account-ID*`:project/`*project-name*

BatchGetProjects

Aktion: `codebuild:BatchGetProjects`

Erforderlich, um Informationen über Build-Projekte abzurufen.

Ressource: `arn:aws:codebuild:`*region-ID*`:`*account-ID*`:project/`*project-name*

BatchGetReportGroups

Aktion: `codebuild:BatchGetReportGroups`

Erforderlich, um Informationen zu Berichtsgruppen abzurufen.

Ressource: `arn:aws:codebuild:`*region-ID*`:`*account-ID*`:report-group/`*report-group-name*

BatchGetReports

Aktion: `codebuild:BatchGetReports`

Erforderlich, um Informationen über Berichte abzurufen.

Ressource: `arn:aws:codebuild:`*`region-ID`*`:`*`account-ID`*`:report-group/`*`report-group-name`*

BatchPutTestCases[1]

Aktion: `codebuild:BatchPutTestCases`

Erforderlich, um einen Testbericht zu erstellen oder zu aktualisieren.

Ressource: `arn:aws:codebuild:`*`region-ID`*`:`*`account-ID`*`:report-group/`*`report-group-name`*

CreateProject

Aktionen: `codebuild:CreateProject`, `iam:PassRole`

Erforderlich, um Build-Projekte zu erstellen.

Ressourcen:

- `arn:aws:codebuild:`*`region-ID`*`:`*`account-ID`*`:project/`*`project-name`*
- `arn:aws:iam::`*`account-ID`*`:role/`*`role-name`*

CreateReport[1]

Aktion: `codebuild:CreateReport`

Erforderlich, um einen Testbericht zu erstellen.

Ressource: `arn:aws:codebuild:`*`region-ID`*`:`*`account-ID`*`:report-group/`*`report-group-name`*

CreateReportGroup

Aktion: `codebuild:CreateReportGroup`

Erforderlich, um eine Berichtsgruppe zu erstellen.

Ressource: `arn:aws:codebuild:`*region-ID*`:`*account-ID*`:report-group/`*report-group-name*

CreateWebhook

Aktion: `codebuild:CreateWebhook`

Erforderlich zum Erstellen eines Webhooks.

Ressource: `arn:aws:codebuild:`*region-ID*`:`*account-ID*`:project/`*project-name*

DeleteProject

Aktion: `codebuild:DeleteProject`

Erforderlich, um ein CodeBuild Projekt zu löschen.

Ressource: `arn:aws:codebuild:`*region-ID*`:`*account-ID*`:project/`*project-name*

DeleteReport

Aktion: `codebuild:DeleteReport`

Erforderlich zum Löschen eines Berichts.

Ressource: `arn:aws:codebuild:`*region-ID*`:`*account-ID*`:report-group/`*report-group-name*

DeleteReportGroup

Aktion: `codebuild:DeleteReportGroup`

Erforderlich, um eine Berichtsgruppe zu löschen.

Ressource: `arn:aws:codebuild:`*region-ID*`:`*account-ID*`:report-group/`*report-group-name*

DeleteSourceCredentials

Aktion: `codebuild:DeleteSourceCredentials`

Erforderlich, um eine Reihe von `SourceCredentialsInfo` Objekten zu löschen, die Informationen über Anmeldeinformationen für ein GitHub GitHub Enterprise Server- oder Bitbucket-Repository enthalten.

Ressource: *

DeleteWebhook

Aktion: `codebuild:DeleteWebhook`

Erforderlich zum Erstellen eines Webhooks.

Ressource: `arn:aws:codebuild:`*`region-ID`*`:`*`account-ID`*`:project/`*`project-name`*

DescribeTestCases

Aktion: `codebuild:DescribeTestCases`

Erforderlich, um eine paginierte Liste von Testfällen zurückzugeben.

Ressource: `arn:aws:codebuild:`*`region-ID`*`:`*`account-ID`*`:report-group/`*`report-group-name`*

ImportSourceCredentials

Aktion: `codebuild:ImportSourceCredentials`

Erforderlich, um eine Reihe von `SourceCredentialsInfo` Objekten zu importieren, die Informationen über Anmeldeinformationen für ein GitHub GitHub Enterprise Server- oder Bitbucket-Repository enthalten.

Ressource: *

InvalidateProjectCache

Aktion: `codebuild:InvalidateProjectCache`

Erforderlich, um den Cache für ein Projekt zurückzusetzen.

Ressource: `arn:aws:codebuild:`*`region-ID`*`:`*`account-ID`*`:project/`*`project-name`*

ListBuildBatches

Aktion: `codebuild:ListBuildBatches`

Erforderlich, um eine Liste von Build-Batch-IDs abzurufen.

Ressource: *

ListBuildBatchesForProject

Aktion: `codebuild:ListBuildBatchesForProject`

Erforderlich, um eine Liste mit Build-Batch-IDs für ein bestimmtes Projekt abzurufen.

Ressource: `arn:aws:codebuild:`*`region-ID`*`:`*`account-ID`*`:project/`*`project-name`*

ListBuilds

Aktion: `codebuild:ListBuilds`

Erforderlich, um eine Liste mit Build-IDs abzurufen.

Ressource: *

ListBuildsForProject

Aktion: `codebuild:ListBuildsForProject`

Erforderlich, um eine Liste mit Build-IDs für ein Build-Projekt abzurufen.

Ressource: `arn:aws:codebuild:`*`region-ID`*`:`*`account-ID`*`:project/`*`project-name`*

ListCuratedEnvironmentImages

Aktion: `codebuild:ListCuratedEnvironmentImages`

Erforderlich, um Informationen über alle von AWS CodeBuild verwaltete Docker-Images abzurufen.

Ressource: * (erforderlich, aber verweist nicht auf eine aufrufbare AWS -Ressource)

ListProjects

Aktion: `codebuild:ListProjects`

Erforderlich, um eine Liste mit Build-Projektnamen abzurufen.

Ressource: *

ListReportGroups

Aktion: `codebuild:ListReportGroups`

Erforderlich, um eine Liste von Berichtsgruppen abrufen.

Ressource: *

ListReports

Aktion: `codebuild:ListReports`

Erforderlich zum Abrufen einer Liste von Berichten.

Ressource: *

ListReportsForReportGroup

Aktion: `codebuild:ListReportsForReportGroup`

Erforderlich, um eine Liste von Berichten für eine Berichtsgruppe abrufen zu können.

Ressource: `arn:aws:codebuild:`*`region-ID`*`:`*`account-ID`*`:report-group/`*`report-group-name`*

RetryBuild

Aktion: `codebuild:RetryBuild`

Erforderlich, um Builds erneut zu versuchen.

Ressource: `arn:aws:codebuild:`*`region-ID`*`:`*`account-ID`*`:project/`*`project-name`*

StartBuild

Aktion: `codebuild:StartBuild`

Erforderlich, um Builds auszuführen.

Ressource: `arn:aws:codebuild:`*`region-ID`*`:`*`account-ID`*`:project/`*`project-name`*

StopBuild

Aktion: `codebuild:StopBuild`

Erforderlich, um ausgeführte Builds zu stoppen.

Ressource: `arn:aws:codebuild:`*`region-ID`*`:`*`account-ID`*`:project/`*`project-name`*

UpdateProject

Aktionen: `codebuild:UpdateProject`, `iam:PassRole`

Erforderlich, um Informationen über Builds zu ändern.

Ressourcen:

- `arn:aws:codebuild:`*`region-ID`*`:`*`account-ID`*`:project/`*`project-name`*
- `arn:aws:iam::`*`account-ID`*`:role/`*`role-name`*

UpdateProjectVisibility

Aktionen: `codebuild:UpdateProjectVisibility`, `iam:PassRole`

Erforderlich, um die öffentliche Sichtbarkeit der Builds eines Projekts zu ändern.

Ressourcen:

- `arn:aws:codebuild:`*`region-ID`*`:`*`account-ID`*`:project/`*`project-name`*
- `arn:aws:iam::`*`account-ID`*`:role/`*`role-name`*

UpdateReport[1]

Aktion: `codebuild:UpdateReport`

Erforderlich, um einen Testbericht zu erstellen oder zu aktualisieren.

Ressource: `arn:aws:codebuild:`*`region-ID`*`:`*`account-ID`*`:report-group/`*`report-group-name`*

UpdateReportGroup

Aktion: `codebuild:UpdateReportGroup`

Erforderlich, um eine Berichtsgruppe zu aktualisieren.

Ressource: `arn:aws:codebuild:`*`region-ID`*`:`*`account-ID`*`:report-group/`*`report-group-name`*

UpdateWebhook

Aktion: `codebuild:UpdateWebhook`

Erforderlich zum Aktualisieren eines Webhooks.

Ressource: `arn:aws:codebuild:region-ID:account-ID:project/project-name`

[1] Wird nur zur Genehmigung verwendet. Für diese Aktion gibt es keine API.

Verwenden von Tags zur Steuerung des Zugriffs auf AWS CodeBuild -Ressourcen

Bedingungen in IAM-Richtlinienerklärungen sind Teil der Syntax, mit der Sie Berechtigungen für CodeBuild projektbasierte Aktionen angeben können. Sie können eine Richtlinie erstellen, die Aktionen für Projekte auf der Grundlage der mit diesen Projekten verknüpften Tags zulässt oder verweigert, und diese Richtlinien dann auf die IAM-Gruppen anwenden, die Sie für die Benutzerverwaltung konfigurieren. Informationen zum Anwenden von Tags auf ein Projekt mithilfe der Konsole oder finden Sie AWS CLI unter. Erstellen eines Build-Projekts in AWS CodeBuild Informationen zum Anwenden von Tags mithilfe des CodeBuild SDK finden Sie unter CreateProject und Tags in der CodeBuildAPI-Referenz. Informationen zur Verwendung von Tags zur Steuerung des Zugriffs auf AWS Ressourcen finden Sie unter Steuern des Zugriffs auf AWS Ressourcen mithilfe von Ressourcen-Tags im IAM-Benutzerhandbuch.

> **Important**
>
> Wenn Sie die Funktion für reservierte Kapazität verwenden, können andere Projekte innerhalb desselben Kontos auf Daten zugreifen, die auf Flotteninstanzen zwischengespeichert sind, einschließlich Quelldateien, Docker-Layern und zwischengespeicherten Verzeichnissen, die in der Buildspec angegeben sind. Dies ist beabsichtigt und ermöglicht es Projekten innerhalb desselben Kontos, Flotteninstanzen gemeinsam zu nutzen.

Example Beispiel 1: Beschränken Sie CodeBuild Projektaktionen auf der Grundlage von Ressourcen-Tags

Im folgenden Beispiel werden alle `Production`-Aktionen für Projekte verweigert, die mit dem Schlüssel `BatchGetProjects` mit dem Schlüsselwert `Environment` gekennzeichnet sind. Der Administrator eines Benutzers muss diese IAM-Richtlinie zusätzlich zur verwalteten Benutzerrichtlinie für nicht autorisierte Benutzer hinzufügen. Der Bedingungsschlüssel `aws:ResourceTag` wird verwendet, um den Zugriff auf Ressourcen basierend auf ihren Tags zu steuern.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": [
        "codebuild:BatchGetProjects"
      ],
      "Resource": "*",
      "Condition": {
        "ForAnyValue:StringEquals": {
          "aws:ResourceTag/Environment": "Production"
        }
      }
    }
  ]
}
```

Example Beispiel 2: Beschränken Sie CodeBuild Projektaktionen auf der Grundlage von Anforderungs-Tags

Die folgende Richtlinie verweigert Benutzern die Berechtigung für die Aktion `CreateProject`, wenn die Anforderung ein Tag mit dem Schlüssel `Environment` und dem Schlüsselwert `Production` enthält. Darüber hinaus hindert die Richtlinie diese nicht autorisierten Benutzer daran, Projekte zu ändern, indem sie den Bedingungsschlüssel `aws:TagKeys` verwenden, um `UpdateProject` nicht zuzulassen, wenn die Anforderung ein Tag mit dem Schlüssel `Environment` enthält. Ein Administrator muss diese IAM-Richtlinie zusätzlich zu der Richtlinie für verwaltete Benutzer hinzufügen, die nicht berechtigt sind, diese Aktionen auszuführen. Der `aws:RequestTag` Bedingungsschlüssel wird verwendet, um zu steuern, welche Tags in einer IAM-Anfrage übergeben werden können

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": [
        "codebuild:CreateProject"
      ],
      "Resource": "*",
      "Condition": {
```

```
        "ForAnyValue:StringEquals": {
          "aws:RequestTag/Environment": "Production"
        }
      }
    },
    {
      "Effect": "Deny",
      "Action": [
        "codebuild:UpdateProject"
      ],
      "Resource": "*",
      "Condition": {
        "ForAnyValue:StringEquals": {
          "aws:TagKeys": ["Environment"]
        }
      }
    }
  ]
}
```

Example Beispiel 3: Ablehnen oder Zulassen von Aktionen für Berichtsgruppen basierend auf Ressourcen-Tags

Sie können eine Richtlinie erstellen, die Aktionen für CodeBuild Ressourcen (Projekte und Berichtsgruppen) auf der Grundlage der mit diesen Ressourcen verknüpften AWS Tags zulässt oder verweigert, und diese Richtlinien dann auf die IAM-Gruppen anwenden, die Sie für die Benutzerverwaltung konfigurieren. *Sie können beispielsweise eine Richtlinie erstellen, die alle CodeBuild Aktionen für eine Berichtsgruppe mit dem AWS Tag-Schlüssel Status und dem Schlüsselwert von verweigertSecret, und diese Richtlinie dann auf die IAM-Gruppe anwenden, die Sie für allgemeine Entwickler (Entwickler) erstellt haben.* Anschließend müssen Sie sicherstellen, dass die Entwickler, die an diesen markierten Berichtsgruppen arbeiten, nicht Mitglieder dieser allgemeinen *Entwicklergruppe* sind, sondern einer anderen IAM-Gruppe angehören, auf die die restriktive Richtlinie nicht angewendet wurde (). `SecretDevelopers`

Im folgenden Beispiel werden alle CodeBuild Aktionen für Berichtsgruppen verweigert, die mit dem Schlüssel `Status` und dem Schlüsselwert von gekennzeichnet sind: `Secret`

```
{
  "Version": "2012-10-17",
  "Statement" : [
```

```
    {
      "Effect" : "Deny",
      "Action" : [
        "codebuild:BatchGetReportGroups,"
        "codebuild:CreateReportGroup",
        "codebuild:DeleteReportGroup",
        "codebuild:ListReportGroups",
        "codebuild:ListReportsForReportGroup",
        "codebuild:UpdateReportGroup"
       ]
      "Resource" : "*",
      "Condition" : {
         "StringEquals" : "aws:ResourceTag/Status": "Secret"
        }
    }
  ]
}
```

Example Beispiel 4: Beschränken Sie CodeBuild Aktionen auf die AWSCodeBuildDeveloperAccess Grundlage von Ressourcen-Tags

Sie können Richtlinien erstellen, die CodeBuild Aktionen für alle Berichtsgruppen und Projekte zulassen, die nicht mit bestimmten Tags gekennzeichnet sind. Die folgende Richtlinie lässt z. B. das Äquivalent von AWSCodeBuildDeveloperAccess-Berechtigungen für alle Berichtsgruppen und Projekte zu. Ausgenommen sind jedoch Berichtsgruppen und Projekte, die mit den angegebenen Tags markiert sind:

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Allow",
         "Action": [
            "codebuild:StartBuild",
            "codebuild:StopBuild",
            "codebuild:BatchGet*",
            "codebuild:GetResourcePolicy",
            "codebuild:DescribeTestCases",
            "codebuild:List*",
            "codecommit:GetBranch",
            "codecommit:GetCommit",
            "codecommit:GetRepository",
            "codecommit:ListBranches",
```

```
            "cloudwatch:GetMetricStatistics",
            "events:DescribeRule",
            "events:ListTargetsByRule",
            "events:ListRuleNamesByTarget",
            "logs:GetLogEvents",
            "s3:GetBucketLocation",
            "s3:ListAllMyBuckets"
         ],
         "Resource": "*",
         "Condition": {
            "StringNotEquals": {
               "aws:ResourceTag/Status": "Secret",
               "aws:ResourceTag/Team": "Saanvi"
            }
         }
      }
   ]
}
```

Anzeigen von Ressourcen in der Konsole

Die AWS CodeBuild Konsole benötigt die `ListRepositories` Erlaubnis, eine Liste von Repositorys für Ihr AWS Konto in der AWS Region anzuzeigen, in der Sie angemeldet sind. Die Konsole umfasst auch eine Funktion Go to resource (Zu Ressource wechseln), um schnell ohne Berücksichtigung der Groß- und Kleinschreibung nach Ressourcen zu suchen. Diese Suche wird in Ihrem AWS Konto in der AWS Region durchgeführt, in der Sie angemeldet sind. Die folgenden Ressourcen werden für die folgenden Services angezeigt:

- AWS CodeBuild: Build-Projekte
- AWS CodeCommit: Repositorys
- AWS CodeDeploy: Anwendungen
- AWS CodePipeline: Pipelines

Für diese Suche unter den Ressourcen in allen Services müssen Sie über die folgenden Berechtigungen verfügen:

- CodeBuild: `ListProjects`
- CodeCommit: `ListRepositories`
- CodeDeploy: `ListApplications`

- CodePipeline: `ListPipelines`

Es werden keine Ergebnisse für die Ressourcen eines Service zurückgegeben, wenn Sie nicht über Berechtigungen für diesen Service verfügen. Auch wenn Sie über Berechtigungen zum Anzeigen von Ressourcen verfügen, werden manche Ressourcen nicht zurückgegeben, wenn die Anzeige dieser Ressourcen ausdrücklich verweigert wird (`Deny`).

Überprüfung der Einhaltung der Vorschriften für AWS CodeBuild

Externe Prüfer bewerten die Sicherheit und Einhaltung von Vorschriften im AWS CodeBuild Rahmen mehrerer AWS Compliance-Programme. Hierzu zählen unter anderem SOC, PCI, FedRAMP und HIPAA.

Eine Liste der AWS Dienstleistungen im Rahmen bestimmter Compliance-Programme finden Sie unter AWS Dienstleistungen im Umfang der einzelnen Compliance-Programme. Allgemeine Informationen finden Sie unter AWS -Compliance-Programme.

Sie können Prüfberichte von Drittanbietern unter herunterladen AWS Artifact. Weitere Informationen finden Sie unter Berichte in AWS Artifact herunterladen.

Ihre Verantwortung für die Einhaltung der Vorschriften bei der Nutzung CodeBuild hängt von der Vertraulichkeit Ihrer Daten, den Compliance-Zielen Ihres Unternehmens und den geltenden Gesetzen und Vorschriften ab. Wenn Ihre Verwendung von der CodeBuild Einhaltung von Standards wie HIPAA, PCI oder FedRAMP unterliegt, AWS bietet Ihnen Ressourcen, die Ihnen helfen:

- Schnellstartanleitungen zu Sicherheit und Compliance — In diesen Bereitstellungsleitfäden werden architektonische Überlegungen erörtert und Schritte für die Implementierung von Umgebungen beschrieben, auf denen Sicherheit und Compliance im Vordergrund stehen. AWS
- Whitepaper „Architecting for HIPAA Security and Compliance“ — In diesem Whitepaper wird beschrieben, wie Unternehmen HIPAA-konforme Anwendungen entwickeln können. AWS
- AWS Ressourcen zur Einhaltung von Vorschriften — Diese Sammlung von Arbeitsmappen und Leitfäden könnte für Ihre Branche und Ihren Standort gelten.
- AWS Config— Dieser AWS Service bewertet, wie gut Ihre Ressourcenkonfigurationen den internen Praktiken, Branchenrichtlinien und Vorschriften entsprechen.
- AWS Security Hub— Überwachen Sie Ihre Nutzung von AWS CodeBuild in Bezug auf bewährte Sicherheitsverfahren mithilfe AWS Security Hubvon. Security Hub verwendet Sicherheitskontrollen für die Bewertung von Ressourcenkonfigurationen und Sicherheitsstandards, um Sie bei der

Einhaltung verschiedener Compliance-Frameworks zu unterstützen. Weitere Informationen zur Verwendung von Security Hub zur Bewertung von CodeBuild Ressourcen finden Sie unter AWS CodeBuild Kontrollen im AWS Security Hub Benutzerhandbuch.

Resilienz in AWS CodeBuild

Die AWS globale Infrastruktur basiert auf AWS Regionen und Availability Zones. AWS Regionen bieten mehrere physisch getrennte und isolierte Availability Zones, die über Netzwerke mit niedriger Latenz, hohem Durchsatz und hoher Redundanz miteinander verbunden sind. Mithilfe von Availability Zones können Sie Anwendungen und Datenbanken erstellen und ausführen, die automatisch Failover zwischen Availability Zones ausführen, ohne dass es zu Unterbrechungen kommt. Availability Zones sind besser hoch verfügbar, fehlertoleranter und skalierbarer als herkömmliche Infrastrukturen mit einem oder mehreren Rechenzentren.

Weitere Informationen zu AWS Regionen und Availability Zones finden Sie unter AWS Globale Infrastruktur.

Sicherheit der Infrastruktur in AWS CodeBuild

Als verwalteter Dienst AWS CodeBuild ist er durch AWS globale Netzwerksicherheit geschützt. Informationen zu AWS Sicherheitsdiensten und zum AWS Schutz der Infrastruktur finden Sie unter AWS Cloud-Sicherheit. Informationen zum Entwerfen Ihrer AWS Umgebung unter Verwendung der bewährten Methoden für die Infrastruktursicherheit finden Sie unter Infrastructure Protection in Security Pillar AWS Well-Architected Framework.

Sie verwenden AWS veröffentlichte API-Aufrufe für den Zugriff CodeBuild über das Netzwerk. Kunden müssen Folgendes unterstützen:

- Transport Layer Security (TLS). Wir benötigen TLS 1.2 und empfehlen TLS 1.3.
- Verschlüsselungs-Suiten mit Perfect Forward Secrecy (PFS) wie DHE (Ephemeral Diffie-Hellman) oder ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Die meisten modernen Systeme wie Java 7 und höher unterstützen diese Modi.

Außerdem müssen Anforderungen mit einer Zugriffsschlüssel-ID und einem geheimen Zugriffsschlüssel signiert sein, der einem IAM-Prinzipal zugeordnet ist. Alternativ können Sie mit AWS Security Token Service (AWS STS) temporäre Sicherheitsanmeldeinformationen erstellen, um die Anforderungen zu signieren.

Greifen Sie auf Ihren Quellanbieter zu in CodeBuild

Für GitHub unseren GitHub Enterprise Server verwendest du ein persönliches Zugriffstoken oder eine OAuth-App, um auf den Quellanbieter zuzugreifen. Für Bitbucket verwendest du entweder ein Zugriffstoken, ein App-Passwort oder eine OAuth-App, um auf den Quellanbieter zuzugreifen.

> **Note**
>
> GitLab und Self Managed GitLab Source Provider werden nicht direkt von, CodeBuild sondern stattdessen über aufgerufen. AWS CodeConnections

Themen

GitHub und GitHub Enterprise Server-Zugriffstoken

Voraussetzungen für Zugriffstoken

Bevor Sie beginnen, müssen Sie Ihrem GitHub Zugriffstoken die richtigen Berechtigungsbereiche hinzufügen.

Denn GitHub Ihr persönliches Zugriffstoken muss die folgenden Bereiche haben.

- repo gewährt volle Kontrolle über private Repositorys.
- repo:status: Gewährt Lese-/Schreibzugriff auf den Commit-Status eines öffentlichen und privaten Repositorys.
- admin:repo_hook gewährt volle Kontrolle über Repository-Hooks. Dieser Bereich ist nicht erforderlich, wenn Ihr Token den Bereich `repo` aufweist.

Weitere Informationen finden Sie auf der Website unter Grundlegendes zu Bereichen für OAuth-Apps. GitHub

Wenn Sie differenzierte persönliche Zugriffstoken verwenden, benötigt Ihr persönliches Zugriffstoken je nach Anwendungsfall möglicherweise die folgenden Berechtigungen:

- Inhalt: Schreibgeschützt: Gewährt Zugriff auf private Repositorys. Diese Berechtigung ist erforderlich, wenn Sie private Repositorys als Quelle verwenden.
- Commit-Status: Lesen und Schreiben: Erteilt die Berechtigung zum Erstellen von Commit-Status. Diese Berechtigung ist erforderlich, wenn für Ihr Projekt ein Webhook eingerichtet ist oder wenn Sie die Funktion zum Erstellen des Berichtsstatus aktiviert haben.
- Webhooks: Lesen und Schreiben: Erteilt die Berechtigung zur Verwaltung von Webhooks. Diese Berechtigung ist erforderlich, wenn für Ihr Projekt ein Webhook eingerichtet ist.
- Pull-Requests: Schreibgeschützt: Erteilt die Erlaubnis, auf Pull-Requests zuzugreifen. Diese Berechtigung ist erforderlich, wenn dein Webhook über einen `FILE_PATH` Filter für Pull-Request-Ereignisse verfügt.
- Administration: Lesen und Schreiben: Diese Berechtigung ist erforderlich, wenn Sie die selbst gehostete GitHub Actions Runner-Funktion mit verwenden. CodeBuild Weitere Informationen finden Sie unter Registrierungstoken für ein Repository erstellen undRichten Sie selbst gehostete GitHub Action-Runner ein in AWS CodeBuild.

> **Note**
>
> Wenn Sie auf Organisations-Repositorys zugreifen möchten, stellen Sie sicher, dass Sie die Organisation als Ressourcenbesitzer des Zugriffstokens angeben.

Weitere Informationen finden Sie auf der Website unter Erforderliche Berechtigungen für detaillierte persönliche Zugriffstoken. GitHub

GitHub Mit einem Zugriffstoken (Konsole) Connect

Um Ihr Projekt GitHub mithilfe eines Zugriffstokens über die Konsole mit einem Zugriffstoken zu verbinden, gehen Sie beim Erstellen eines Projekts wie folgt vor. Weitere Informationen finden Sie unter Erstellen Sie ein Build-Projekt (Konsole).

1. Wählen Sie als Quellanbieter die Option GitHub.
2. Wählen Sie für Repository die Option Mit einem GitHub persönlichen Zugriffstoken Connect aus.
3. Geben Sie im Feld GitHub Persönliches Zugriffstoken Ihr GitHub persönliches Zugriffstoken ein.

4. Wählen Sie Save token (Token speichern) aus.

GitHub Mit einem Zugriffstoken (CLI) Connect

Gehen Sie wie folgt vor, um Ihr Projekt GitHub mithilfe eines Zugriffstokens mit dem zu verbinden. AWS CLI Informationen zur Verwendung von AWS CLI with AWS CodeBuild finden Sie unterBefehlszeilenreferenz.

1. Führen Sie den Befehl import-source-credentials aus:

```
aws codebuild import-source-credentials --generate-cli-skeleton
```

Daten im JSON-Format werden in der Ausgabe angezeigt. Kopieren Sie die Daten in eine Datei (z. B.*import-source-credentials.json*) an einem Speicherort auf dem lokalen Computer oder in der Instanz, in der der installiert AWS CLI ist. Ändern Sie die kopierten Daten wie im Folgenden dargestellt und speichern Sie die Ergebnisse.

```
{
  "serverType": "server-type",
  "authType": "auth-type",
  "shouldOverwrite": "should-overwrite",
  "token": "token",
  "username": "username"
}
```

Ersetzen Sie Folgendes:

- *server-type* ist ein erforderlicher Wert. Der Quellanbieter für diese Anmeldeinformationen. Gültige Werte sind GITHUB oder GITHUB_ENTERPRISE.
- *auth-type* ist ein erforderlicher Wert. Der Authentifizierungstyp, der für die Verbindung zu einem GitHub oder GitHub Enterprise Server-Repository verwendet wird. Gültige Werte sind PERSONAL_ACCESS_TOKEN und BASIC_AUTH. Sie können die CodeBuild API nicht verwenden, um eine OAUTH-Verbindung herzustellen. Sie müssen stattdessen die CodeBuild Konsole verwenden.
- *should-overwrite*: Optionaler Wert. Setzen Sie diesen auf `false`, um zu verhindern, dass die Repository-Quellanmeldeinformationen überschrieben werden. Legen Sie den Wert auf `true` fest, um die Repository-Quellanmeldeinformationen zu überschreiben. Der Standardwert ist `true`.

- *token* ist ein erforderlicher Wert. Für GitHub unseren GitHub Enterprise Server ist dies das persönliche Zugriffstoken.
- *username* ist ein optionaler Wert. Dieser Parameter wird für GitHub GitHub Enterprise Server-Quellanbieter ignoriert.

2. Um Ihr Konto mit einem Zugriffstoken zu verbinden, wechseln Sie in das Verzeichnis mit der Datei `import-source-credentials.json`, die Sie in Schritt 1 gespeichert haben, und führen den Befehl import-source-credentials erneut aus.

```
aws codebuild import-source-credentials --cli-input-json file://import-source-credentials.json
```

Daten im JSON-Format werden in der Ausgabe mit einem Amazon-Ressourcennamen (ARN) angezeigt.

```
{
  "arn": "arn:aws:codebuild:region:account-id:token/server-type"
}
```

> **Note**
>
> Wenn Sie den Befehl import-source-credentials mit demselben Server- und Auth-Typ ein zweites Mal ausführen, wird das gespeicherte Zugriffstoken aktualisiert.

Nachdem Ihr Konto mit einem Zugriffstoken verbunden wurde, können Sie es verwenden, `create-project` um Ihr CodeBuild Projekt zu erstellen. Weitere Informationen finden Sie unter Erstellen eines Build-Projekts (AWS CLI).

3. Rufen Sie zum Anzeigen der verbundenen Zugriffstoken den Befehl list-source-credentials auf.

```
aws codebuild list-source-credentials
```

Ein Objekt vom Typ `sourceCredentialsInfos` wird im JSON-Format in der Ausgabe angezeigt:

```
{
    "sourceCredentialsInfos": [
        {
```

```
            "authType": "auth-type",
            "serverType": "server-type",
            "arn": "arn"
        }
    ]
}
```

`sourceCredentialsObject` enthält eine Liste der verbundenen Quell-Anmeldeinformationen:

- `authType` ist der Typ der Authentifizierung für die Anmeldeinformationen. Mögliche Werte sind `OAUTH`, `BASIC_AUTH` oder `PERSONAL_ACCESS_TOKEN`.
- `serverType` ist der Typ des Quellanbieters. Mögliche Werte sind `GITHUB`, `GITHUB_ENTERPRISE` oder `BITBUCKET`.
- `arn` ist der ARN des Tokens.

4. Rufen Sie den Befehl delete-source-credentials mit dem zugehörigen ARN auf, um einen Quellanbieter zu trennen und dessen Zugriffstoken zu entfernen.

```
aws codebuild delete-source-credentials --arn arn-of-your-credentials
```

Die Ausgabe im JSON-Format enthält einen ARN der gelöschten Anmeldeinformationen.

```
{
  "arn": "arn:aws:codebuild:region:account-id:token/server-type"
}
```

GitHub OAuth-App

Connect GitHub über OAuth (Konsole) herstellen

Um Ihr Projekt über die Konsole mit einer GitHub OAuth-App zu verbinden, gehen Sie beim Erstellen eines Projekts wie folgt vor. Weitere Informationen finden Sie unter Erstellen Sie ein Build-Projekt (Konsole).

1. Wählen Sie als Quellanbieter die Option. GitHub
2. Wählen Sie für Repository Connect using OAuth.
3. Wählen Sie Connect GitHub, melden Sie sich an und autorisieren Sie Ihr Konto.
4. Wählen Sie Bestätigen, um eine Verbindung CodeBuild zu Ihrem GitHub Konto herzustellen.

5. Geben Sie im GitHub Repository Ihren GitHub Repository-Link ein.

Um Ihre autorisierten OAuth-Apps zu überprüfen, navigieren Sie zu Applications on und vergewissern Sie sich GitHub, dass eine Anwendung mit dem Namen `AWS CodeBuild (region)` aws-codesuite aufgeführt ist.

Passwort oder Zugriffstoken für die Bitbucket-App

Voraussetzungen

Bevor du anfängst, musst du deinem Bitbucket-App-Passwort oder Zugriffstoken die richtigen Berechtigungsbereiche hinzufügen.

Für Bitbucket muss dein App-Passwort oder dein Zugriffstoken die folgenden Bereiche haben.

- repository:read gewährt Lesezugriff auf alle Repositorys, auf die der autorisierende Benutzer Zugriff hat.
- pullrequest:read gewährt Lesezugriff auf Pull-Anforderungen. Wenn dein Projekt einen Bitbucket-Webhook hat, muss dein App-Passwort oder dein Zugriffstoken diesen Bereich haben.
- webhook gewährt Zugriff auf Webhooks. Wenn dein Projekt über einen Webhook-Vorgang verfügt, muss dein App-Passwort oder dein Zugriffstoken diesen Bereich haben.

Weitere Informationen finden Sie unter Bereiche für die Bitbucket-Cloud-REST-API und OAuth für Bitbucket Cloud auf der Bitbucket-Website.

Bitbucket mit einem App-Passwort Connect (Konsole)

Um dein Projekt mithilfe eines App-Passworts über die Konsole mit Bitbucket zu verbinden, gehe wie folgt vor, wenn du ein Projekt erstellst. Weitere Informationen finden Sie unter Erstellen Sie ein Build-Projekt (Konsole).

1. Wählen Sie für Source provider (Quellanbieter) die Option Bitbucket aus.

   > **Note**
   >
   > CodeBuild unterstützt Bitbucket Server nicht.

2. Wählen Sie für Repository die Option Connect with a Bitbucket app password (Mit Bitbucket-App-Passwort verbinden) aus.

3. Geben Sie in Bitbucket username (Bitbucket-Benutzername) Ihren Bitbucket-Benutzernamen an.
4. Geben Sie in Bitbucket app password (Bitbucket-App-Passwort) Ihr Bitbucket-App-Passwort an.
5. Wählen Sie Save Bitbucket credentials (Bitbucket-Anmeldeinformationen speichern) aus.

Connect Bitbucket mit einem Zugriffstoken (Konsole)

Um dein Projekt mithilfe eines Zugriffstoken über die Konsole mit Bitbucket zu verbinden, gehe beim Erstellen eines Projekts wie folgt vor. Weitere Informationen finden Sie unter Erstellen Sie ein Build-Projekt (Konsole).

1. Wählen Sie für Source provider (Quellanbieter) die Option Bitbucket aus.

   > **Note**
   >
   > CodeBuild unterstützt Bitbucket Server nicht.

2. Wähle für Repository die Option Mit einem Bitbucket-Zugriffstoken Connect.
3. Gib unter Bitbucket-Zugriffstoken dein Bitbucket-Zugriffstoken ein.
4. Wählen Sie Save token (Token speichern) aus.

Connect Bitbucket mit einem App-Passwort oder einem Zugriffstoken (CLI)

Folge diesen Schritten, um dein Projekt mithilfe eines App-Passworts oder Zugriffstoken mit Bitbucket zu verbinden. AWS CLI Informationen zur Verwendung von AWS CLI with findest AWS CodeBuild du unterBefehlszeilenreferenz.

1. Führen Sie den Befehl import-source-credentials aus:

   ```
   aws codebuild import-source-credentials --generate-cli-skeleton
   ```

   Daten im JSON-Format werden in der Ausgabe angezeigt. Kopieren Sie die Daten in eine Datei (z. B.*import-source-credentials.json*) an einem Speicherort auf dem lokalen Computer oder in der Instanz, in der der installiert AWS CLI ist. Ändern Sie die kopierten Daten wie im Folgenden dargestellt und speichern Sie die Ergebnisse.

   ```
   {
     "serverType": "BITBUCKET",
   ```

```
  "authType": "auth-type",
  "shouldOverwrite": "should-overwrite",
  "token": "token",
  "username": "username"
}
```

Ersetzen Sie Folgendes:

- *auth-type* ist ein erforderlicher Wert. Der Authentifizierungstyp, der für die Verbindung mit einem Bitbucket-Repository verwendet wird. Gültige Werte sind PERSONAL_ACCESS_TOKEN und BASIC_AUTH. Du kannst die CodeBuild API nicht verwenden, um eine OAUTH-Verbindung herzustellen. Sie müssen stattdessen die CodeBuild-Konsole verwenden.
- *should-overwrite*: Optionaler Wert. Setzen Sie diesen auf `false`, um zu verhindern, dass die Repository-Quellanmeldeinformationen überschrieben werden. Legen Sie den Wert auf `true` fest, um die Repository-Quellanmeldeinformationen zu überschreiben. Der Standardwert ist `true`.
- *token* ist ein erforderlicher Wert. Für Bitbucket ist dies entweder das Zugriffstoken oder das App-Passwort.
- *username* ist ein optionaler Wert. Wenn der Bitbucket-Benutzername BASIC_AUTH `authType` ist. Dieser Parameter wird für andere Arten von Quellanbietern oder Verbindungen ignoriert.

2. Um dein Konto mit einem App-Passwort oder einem Zugriffstoken zu verbinden, wechsle zu dem Verzeichnis, das die in Schritt 1 gespeicherte `import-source-credentials.json` Datei enthält, und führe den Befehl erneut aus. import-source-credentials

```
aws codebuild import-source-credentials --cli-input-json file://import-source-
credentials.json
```

Daten im JSON-Format werden in der Ausgabe mit einem Amazon-Ressourcennamen (ARN) angezeigt.

```
{
  "arn": "arn:aws:codebuild:region:account-id:token/server-type"
}
```

> **Note**
>
> Wenn Sie den Befehl import-source-credentials mit demselben Server- und Auth-Typ ein zweites Mal ausführen, wird das gespeicherte Zugriffstoken aktualisiert.

Nachdem Ihr Konto mit einem App-Passwort verbunden wurde, können Sie es verwenden, `create-project` um Ihr CodeBuild Projekt zu erstellen. Weitere Informationen finden Sie unter Erstellen eines Build-Projekts (AWS CLI).

3. Führen Sie den list-source-credentials Befehl aus, um die Passwörter oder Zugriffstoken der verbundenen Apps anzuzeigen.

```
aws codebuild list-source-credentials
```

Ein Objekt vom Typ `sourceCredentialsInfos` wird im JSON-Format in der Ausgabe angezeigt:

```
{
    "sourceCredentialsInfos": [
        {
            "authType": "auth-type",
            "serverType": "BITBUCKET",
            "arn": "arn"
        }
    ]
}
```

`sourceCredentialsObject` enthält eine Liste der verbundenen Quell-Anmeldeinformationen:

- `authType` ist der Typ der Authentifizierung für die Anmeldeinformationen. Mögliche Werte sind `OAUTH`, `BASIC_AUTH` oder `PERSONAL_ACCESS_TOKEN`.
- `arn` ist der ARN des Tokens.

4. Um die Verbindung zu einem Quellanbieter zu trennen und dessen App-Passwort oder Zugriffstoken zu entfernen, führen Sie den delete-source-credentials Befehl mit seinem ARN aus.

```
aws codebuild delete-source-credentials --arn arn-of-your-credentials
```

Die Ausgabe im JSON-Format enthält einen ARN der gelöschten Anmeldeinformationen.

```
{
  "arn": "arn:aws:codebuild:region:account-id:token/server-type"
}
```

Bitbucket OAuth-App

Bitbucket mit OAuth Connect (Konsole)

Um dein Projekt mithilfe einer OAuth-App über die Konsole mit Bitbucket zu verbinden, gehe beim Erstellen eines Projekts wie folgt vor. Weitere Informationen finden Sie unter Erstellen Sie ein Build-Projekt (Konsole).

1. Wählen Sie für Source provider (Quellanbieter) die Option Bitbucket aus.
2. Wählen Sie für Repository Connect using OAuth.
3. Wähle Mit Bitbucket verbinden, melde dich an und autorisiere dein Konto.
4. Wähle Bestätigen, um dich mit deinem CodeBuild Bitbucket-Konto zu verbinden.
5. Gib im Bitbucket-Repository deinen Bitbucket-Repository-Link ein.

Um deine autorisierten OAuth-Apps zu überprüfen, navigiere zu Anwendungsautorisierungen auf Bitbucket und vergewissere dich, dass eine Anwendung mit dem Namen aufgeführt ist. `AWS CodeBuild (region)`

Serviceübergreifende Confused-Deputy-Prävention

Das Confused-Deputy-Problem ist ein Sicherheitsproblem, bei dem eine juristische Stelle, die nicht über die Berechtigung zum Ausführen einer Aktion verfügt, eine privilegiertere juristische Stelle zwingen kann, die Aktion auszuführen. In AWS kann ein dienstübergreifender Identitätswechsel zu dem Problem mit dem verwirrten Stellvertreter führen. Ein dienstübergreifender Identitätswechsel kann auftreten, wenn ein Dienst (der Anruf-Dienst) einen anderen Dienst anruft (den aufgerufenen Dienst). Der aufrufende Service kann manipuliert werden, um seine Berechtigungen zu verwenden, um Aktionen auf die Ressourcen eines anderen Kunden auszuführen, für die er sonst keine Zugriffsberechtigung haben sollte. Um dies zu verhindern, bietet AWS Tools, mit denen Sie Ihre

Daten für alle Services mit Serviceprinzipalen schützen können, die Zugriff auf Ressourcen in Ihrem Konto erhalten haben.

Wir empfehlen, die Kontextschlüssel `aws:SourceArn`und die `aws:SourceAccount`globalen Bedingungsschlüssel in Ressourcenrichtlinien zu verwenden, um die Berechtigungen einzuschränken, die der AWS CodeBuild Ressource einen anderen Dienst gewähren. Verwenden Sie `aws:SourceArn`, wenn Sie nur eine Ressource mit dem betriebsübergreifenden Zugriff verknüpfen möchten. Verwenden Sie `aws:SourceAccount`, wenn Sie zulassen möchten, dass Ressourcen in diesem Konto mit der betriebsübergreifenden Verwendung verknüpft werden.

Der effektivste Weg, um sich vor dem Confused-Deputy-Problem zu schützen, ist die Verwendung des globalen Bedingungskontext-Schlüssels `aws:SourceArn` mit dem vollständigen ARN der Ressource. Wenn Sie den vollständigen ARN der Ressource nicht kennen oder wenn Sie mehrere Ressourcen angeben, verwenden Sie den globalen Kontextbedingungsschlüssel `aws:SourceArn` mit Platzhalterzeichen (*) für die unbekannten Teile des ARN. z. B. `arn:aws:codebuild:*:`*`123456789012`*`:*`.

Wenn der `aws:SourceArn`-Wert die Konto-ID nicht enthält, z. B. einen Amazon-S3-Bucket-ARN, müssen Sie beide globale Bedingungskontextschlüssel verwenden, um Berechtigungen einzuschränken.

Der Wert von `aws:SourceArn` muss der CodeBuild Projekt-ARN sein.

Das folgende Beispiel zeigt, wie Sie die Kontextschlüssel `aws:SourceArn` und die `aws:SourceAccount` globale Bedingung verwenden können, CodeBuild um das Problem des verwirrten Stellvertreters zu vermeiden.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "codebuild.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "aws:SourceArn": "arn:aws:codebuild:region-ID:account-
ID:project/project-name"
                }
```

```
            }
        }
    ]
}
```

Fortschrittliche Themen

Dieser Abschnitt enthält eine Reihe erweiterter Themen für erfahrene AWS CodeBuild-Benutzer.

Themen

Erweiterte Einstellungen

Wenn Sie die Schritte unter Erste Schritte mit der Konsole für den erstmaligen Zugriff auf AWS CodeBuild befolgen, werden Sie die Informationen in diesem Thema höchstwahrscheinlich nicht benötigen. Wenn Sie die Nutzung jedoch fortsetzen CodeBuild, möchten Sie möglicherweise beispielsweise IAM-Gruppen und Benutzern in Ihrer Organisation Zugriff CodeBuild auf bestehende Servicerollen in IAM oderAWS KMS keys zum Zugriff gewähren CodeBuild oder diese für den ZugriffAWS CLI auf alle Arbeitsstationen Ihrer Organisation einrichten CodeBuild. In diesem Thema wird beschrieben, wie Sie die zugehörigen Einrichtungsschritte durchführen.

Wir gehen davon aus, dass Sie bereits über ein AWS-Konto verfügen. Falls Sie jedoch noch keine haben, gehen Sie zu http://aws.amazon.com, wählen Sie In der Konsole anmelden und folgen Sie den Online-Anweisungen.

Themen

- Installieren und Konfigurieren der AWS CLI.

Hinzufügen von CodeBuild Zugriffsberechtigungen zu einer IAM-Gruppe oder einem IAM-Benutzer

Für den ZugriffAWS CodeBuild mit einer IAM-Gruppe oder einem IAM-Benutzer müssen Sie Zugriffsberechtigungen hinzufügen. In diesem Abschnitt wird beschrieben, wie Sie dies mit der IAM-Konsole oder der tunAWS CLI.

Wenn Sie CodeBuild mit IhremAWS Root-Konto (nicht empfohlen) oder einem Administratorbenutzer in IhremAWS Konto darauf zugreifen möchten, müssen Sie diese Anweisungen nicht befolgen.

Informationen zuAWS Root-Konten und Administratorbenutzern finden Sie unter DerAWS-Konto Root-Benutzer und Erstellen Ihres erstenAWS-Konto Root-Benutzers und Ihrer ersten Root-Gruppe im Benutzerhandbuch.

Um einer IAM-Gruppe oder einem IAM-Benutzer CodeBuild Zugriffsberechtigungen hinzuzufügen (Konsole)

1. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.

   Sie sollten bereits unter Verwendung eines der folgenden Konten bzw. Benutzer bei der AWS Management Console angemeldet sein:

   - Ihr AWS-Stammkonto. Dies wird nicht empfohlen. Weitere Informationen finden Sie im Benutzerhandbuch unter DerAWS-KontoRoot-Benutzer.
   - Ein Administratorbenutzer in IhremAWS Konto. Weitere Informationen finden Sie im Benutzerhandbuch unter Erstellen Ihres erstenAWS-Konto Root-Benutzers und Ihrer ersten Root-Gruppe.
   - Ein Benutzer in IhremAWS Konto mit der Erlaubnis, die folgenden Mindestaktionen auszuführen:

     ```
     iam:AttachGroupPolicy
     iam:AttachUserPolicy
     iam:CreatePolicy
     iam:ListAttachedGroupPolicies
     iam:ListAttachedUserPolicies
     iam:ListGroups
     iam:ListPolicies
     ```

```
iam:ListUsers
```

Weitere Informationen finden Sie im Benutzerhandbuch unter Überblick über die IAM-Richtlinien.

2. Wählen Sie im Navigationsbereich Policies aus.
3. Wenn Sie einer IAM-Gruppe oder einem IAM-Benutzer einen benutzerdefinierten Satz vonAWS CodeBuild Zugriffsberechtigungen zu einer IAM-Gruppe oder einem IAM-Benutzer einen benutzerdefinierten Satz von Zugriffsberechtigungen zu einer IAM-Gruppe oder einem IAM-Benutzer einen benutzerdefinierten Satz von Zugriffsberechtigungen zu einer I

   Um einer IAM-Gruppe oder einem IAM-Benutzer einen Standardsatz von CodeBuild Zugriffsberechtigungen hinzuzufügen, wählen Sie Richtlinientyp, AWSVerwaltet aus, und gehen Sie dann wie folgt vor:

   - Um Vollzugriffsberechtigungen hinzuzufügen CodeBuild, markieren Sie das benannte Feld AWSCodeBuildAdminAccess, wählen Sie Policy Actions und dann Attach aus. Wählen Sie das Feld neben der IAM-Zielgruppe oder dem Zielbenutzer aus, und wählen Sie dann Attach Policy aus. Wiederholen Sie dies für die Richtlinien AmazonS3ReadOnlyAccess und IAMFullAccess.
   - Um Zugriffsberechtigungen CodeBuild für alles außer der Build-Projektverwaltung hinzuzufügen, markieren Sie das benannte Feld AWSCodeBuildDeveloperAccess, wählen Sie Policy Actions und dann Attach aus. Wählen Sie das Feld neben der IAM-Zielgruppe oder dem Zielbenutzer aus, und wählen Sie dann Attach Policy aus. Wiederholen Sie dies für die Richtlinie mit dem Namen AmazonS3ReadOnlyAccess.
   - Um nur Lesezugriffsberechtigungen hinzuzufügen CodeBuild, wählen Sie die genannten Felder aus AWSCodeBuildReadOnlyAccess. Wählen Sie das Feld neben der IAM-Zielgruppe oder dem Zielbenutzer aus, und wählen Sie dann Attach Policy aus. Wiederholen Sie dies für die Richtlinie mit dem Namen AmazonS3ReadOnlyAccess.

   Sie haben jetzt einer IAM-Gruppe oder einem IAM-Benutzer einen Standardsatz von CodeBuild Zugriffsberechtigungen hinzugefügt. Lassen Sie die restlichen Schritte in dieser Anleitung aus.
4. Wählen Sie Create Policy (Richtlinie erstellen) aus.
5. Klicken Sie auf der Seite Create Policy neben Create Your Own Policy auf Select.
6. Geben Sie auf der Seite Review Policy (Richtlinie überprüfen) für Policy Name (Richtlinienname) einen neuen Namen für die Richtlinie ein (z. B. **`CodeBuildAccessPolicy`**). Wenn Sie einen anderen Namen verwenden, müssen Sie diesen während der gesamten Anleitung verwenden.

7. Geben Sie unter Policy Document (Richtliniendokument) Folgendes ein und klicken Sie anschließend auf Create Policy (Richtlinie erstellen):

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "CodeBuildAccessPolicy",
      "Effect": "Allow",
      "Action": [
        "codebuild:*"
      ],
      "Resource": "*"
    },
    {
      "Sid": "CodeBuildRolePolicy",
      "Effect": "Allow",
      "Action": [
        "iam:PassRole"
      ],
      "Resource": "arn:aws:iam::account-ID:role/role-name"
    },
    {
      "Sid": "CloudWatchLogsAccessPolicy",
      "Effect": "Allow",
      "Action": [
        "logs:FilterLogEvents",
        "logs:GetLogEvents"
      ],
      "Resource": "*"
    },
    {
      "Sid": "S3AccessPolicy",
      "Effect": "Allow",
      "Action": [
        "s3:CreateBucket",
        "s3:GetObject",
        "s3:List*",
        "s3:PutObject"
      ],
      "Resource": "*"
    },
    {
```

```
      "Sid": "S3BucketIdentity",
      "Effect": "Allow",
      "Action": [
        "s3:GetBucketAcl",
        "s3:GetBucketLocation"
      ],
      "Resource": "*"
    }
  ]
}
```

> **Note**
>
> Diese Richtlinie ermöglicht den Zugriff auf alle CodeBuild Aktionen und auf eine potenziell große Anzahl vonAWS Ressourcen. Um die Berechtigungen auf bestimmte CodeBuild Aktionen zu beschränken, ändern Sie den Wert von`codebuild:*` in der CodeBuild Richtlinienerklärung. Weitere Informationen finden Sie unter Identity and Access Management. Um den Zugriff auf bestimmte AWS-Ressourcen zu beschränken, ändern Sie den Wert des `Resource`-Objekts. Weitere Informationen finden Sie unter Identity and Access Management.
> Die`CodeBuildRolePolicy` Anweisung ist erforderlich, damit ein Build-Projekt erstellt oder geändert werden kann.

8. Klicken Sie im Navigationsbereich auf Groups oder Users.
9. Wählen Sie in der Liste der Gruppen oder Benutzer den Namen der IAM-Gruppe oder des IAM-Benutzers aus, dem Sie CodeBuild Zugriffsberechtigungen hinzufügen möchten.
10. Erweitern Sie bei einer Gruppe auf der Seite mit den Gruppeneinstellungen auf der Registerkarte Permissions (Berechtigungen) den Eintrag Managed Policies (Verwaltete Richtlinien) und klicken Sie dann auf Attach Policy (Richtlinie anfügen).

    Bei einem Benutzer wählen Sie auf der Seite für die Benutzereinstellungen auf der Registerkarte Permissions die Option Add permissions aus.
11. Wählen CodeBuildAccessPolicySie für eine Gruppe auf der Seite Richtlinie anhängen die Option Richtlinie anhängen aus und wählen Sie dann Richtlinie anhängen aus.

    Wählen Sie für einen Benutzer auf der Seite Berechtigungen hinzufügen die Option Bestehende Richtlinien direkt anhängen aus. Wählen Sie aus CodeBuildAccessPolicy, wählen Sie Weiter: Überprüfen und wählen Sie dann Berechtigungen hinzufügen.

Um einer IAM-Gruppe oder einem IAM-Benutzer CodeBuild Zugriffsberechtigungen hinzuzufügen (AWS CLI)

1. Stellen Sie sicher, dass Sie denAWS CLI mit demAWS Zugriffsschlüssel und demAWS geheimen Zugriffsschlüssel konfiguriert haben, die einer der IAM-Entitäten entsprechen, wie im vorherigen Verfahren beschrieben. Weitere Informationen finden Sie unter Installieren der AWS Command Line Interface im AWS Command Line Interface-Benutzerhandbuch.
2. Um einer IAM-Gruppe oder einem IAM-Benutzer einen benutzerdefinierten Satz vonAWS CodeBuild Zugriffsberechtigungen hinzuzufügen, fahren Sie mit Schritt 3 dieses Verfahrens fort.

   Gehen Sie wie folgt vor, um einer IAM-Gruppe oder einem IAM-Benutzer einen Standardsatz von CodeBuild Zugriffsberechtigungen hinzuzufügen:

   Verwenden Sie je nachdem, ob Sie einer IAM-Gruppe oder einem IAM-Benutzer Berechtigungen für eine IAM-Gruppe oder einen der folgenden Befehle für einen der folgenden Befehle festlegen, je nachdem, ob Sie einer IAM-Gruppe oder einem Benutzer

   ```
   aws iam attach-group-policy --group-name group-name --policy-arn policy-arn

   aws iam attach-user-policy --user-name user-name --policy-arn policy-arn
   ```

   Sie müssen den Befehl dreimal ausführen, wobei Sie den *Gruppennamen* oder den *Benutzernamen* durch den IAM-Gruppennamen oder den Benutzernamen und *Policy-arn* einmal für jede der folgenden Richtlinien (Amazon Resource Names, ARNs) ersetzen:

   - Verwenden Sie die folgenden Richtlinien-ARNs CodeBuild, um vollständige Zugriffsberechtigungen hinzuzufügen:
     - `arn:aws:iam::aws:policy/AWSCodeBuildAdminAccess`
     - `arn:aws:iam::aws:policy/AmazonS3ReadOnlyAccess`
     - `arn:aws:iam::aws:policy/IAMFullAccess`
   - Verwenden Sie die folgenden Richtlinien-ARNs, um Zugriffsberechtigungen CodeBuild für alles außer der Build-Projektverwaltung hinzuzufügen:
     - `arn:aws:iam::aws:policy/AWSCodeBuildDeveloperAccess`
     - `arn:aws:iam::aws:policy/AmazonS3ReadOnlyAccess`
   - Verwenden Sie die folgenden Richtlinien-ARNs CodeBuild, um schreibgeschützte Zugriffsberechtigungen hinzuzufügen:

- `arn:aws:iam::aws:policy/AWSCodeBuildReadOnlyAccess`
- `arn:aws:iam::aws:policy/AmazonS3ReadOnlyAccess`

Sie haben jetzt einer IAM-Gruppe oder einem IAM-Benutzer einen Standardsatz von CodeBuild Zugriffsberechtigungen hinzugefügt. Lassen Sie die restlichen Schritte in dieser Anleitung aus.

3. Erstellen Sie in einem leeren Verzeichnis oder auf der lokalen Workstation oder Instance, auf der die AWS CLI installiert ist, eine Datei mit dem Namen `put-group-policy.json` oder `put-user-policy.json`. Wenn Sie einen anderen Dateinamen verwenden, müssen Sie diesen während der gesamten Anleitung verwenden.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "CodeBuildAccessPolicy",
      "Effect": "Allow",
      "Action": [
        "codebuild:*"
      ],
      "Resource": "*"
    },
    {
      "Sid": "CodeBuildRolePolicy",
      "Effect": "Allow",
      "Action": [
        "iam:PassRole"
      ],
      "Resource": "arn:aws:iam::account-ID:role/role-name"
    },
    {
      "Sid": "CloudWatchLogsAccessPolicy",
      "Effect": "Allow",
      "Action": [
        "logs:FilterLogEvents",
        "logs:GetLogEvents"
      ],
      "Resource": "*"
    },
    {
      "Sid": "S3AccessPolicy",
      "Effect": "Allow",
```

```
      "Action": [
        "s3:CreateBucket",
        "s3:GetObject",
        "s3:List*",
        "s3:PutObject"
      ],
      "Resource": "*"
    },
    {
      "Sid": "S3BucketIdentity",
      "Effect": "Allow",
      "Action": [
        "s3:GetBucketAcl",
        "s3:GetBucketLocation"
      ],
      "Resource": "*"
    }
  ]
}
```

> **Note**
>
> Diese Richtlinie ermöglicht den Zugriff auf alle CodeBuild Aktionen und auf eine potenziell große Anzahl vonAWS Ressourcen. Um die Berechtigungen auf bestimmte CodeBuild Aktionen zu beschränken, ändern Sie den Wert von`codebuild:*` in der CodeBuild Richtlinienerklärung. Weitere Informationen finden Sie unter Identity and Access Management. Um den Zugriff auf bestimmte AWS-Ressourcen zu beschränken, ändern Sie den Wert des zugehörigen `Resource`-Objekts. Weitere Informationen finden Sie unter Identity and Access Management oder in der Sicherheitsdokumentation zu dem betreffenden AWS-Service.
> Die`CodeBuildRolePolicy` Anweisung ist erforderlich, damit ein Build-Projekt erstellt oder geändert werden kann.

4. Wechseln Sie in das Verzeichnis, in dem Sie die Datei gespeichert haben, und führen Sie einen der folgenden Befehle aus. Sie können verschiedene Werte für `CodeBuildGroupAccessPolicy` und `CodeBuildUserAccessPolicy` verwenden. Wenn Sie verschiedene Werte verwenden, achten Sie darauf, diese hier zu verwenden.

   Für eine IAM-Gruppe:

```
aws iam put-group-policy --group-name group-name --policy-name
 CodeBuildGroupAccessPolicy --policy-document file://put-group-policy.json
```

Für einen -Benutzer:

```
aws iam put-user-policy --user-name user-name --policy-name
 CodeBuildUserAccessPolicy --policy-document file://put-user-policy.json
```

Ersetzen Sie in den vorangegangenen Befehlen den *Gruppennamen oder* den Benutzernamen durch den Namen der IAM-Zielgruppe oder des Zielbenutzers.

Erstellen Sie eine CodeBuild Servicerolle

Sie benötigen eineAWS CodeBuild Servicerolle, damit CodeBuild Sie in Ihrem Namen mit abhängigenAWS Diensten interagieren können. Sie können eine CodeBuild Servicerolle mithilfe der CodeBuild AWS CodePipeline OR-Konsolen erstellen. Weitere Informationen finden Sie hier:

- Erstellen Sie ein Build-Projekt (Konsole)
- Erstellen einer Pipeline, die CodeBuild (CodePipeline-Konsole) verwendet
- Hinzufügen einer CodeBuild-Build-Aktion zu einer Pipeline (CodePipeline-Konsole)
- Ändern der Einstellungen eines Build-Projekts (Konsole)

Wenn Sie diese Konsolen nicht verwenden möchten, wird in diesem Abschnitt beschrieben, wie Sie eine CodeBuild Servicerolle mit der IAM-Konsole oder der erstellenAWS CLI.

Important

CodeBuild verwendet die Service-Rolle für alle in Ihrem Namen ausgeführten Vorgänge für alle in Ihrem Namen ausgeführten Vorgänge in Ihrem Namen ausgeführt werden. Wenn die Rolle Berechtigungen umfasst, die der Benutzer nicht haben sollte, können Sie die Berechtigungen eines Benutzers versehentlich weiterleiten. Stellen Sie sicher, dass die Rolle die geringsten Rechte zugesteht.
Die auf dieser Seite beschriebene Servicerolle enthält eine Richtlinie, die Mindestberechtigungen zur Verwendung von CodeBuild gewährt. Sie müssen möglicherweise je nach Anwendungsfall zusätzliche Berechtigungen festlegen, je nach

Anwendungsfall einen zusätzlichen Zugriff auf den Benutzer zu dem Anwendungsfall zugeteilt wird.

So erstellen Sie eine CodeBuild Service-Rolle (Konsole)

1. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.

   Sie sollten bereits unter Verwendung eines der folgenden Konten bzw. Benutzer bei der Konsole angemeldet sein:

   - Ihr AWS-Stammkonto. Dies wird nicht empfohlen. Weitere Informationen finden Sie im Benutzerhandbuch unter DerAWS-KontoRoot-Benutzer.
   - Ein Administratorbenutzer in IhremAWS Konto. Weitere Informationen finden Sie im Benutzerhandbuch unter Erstellen Ihres erstenAWS-Konto Root-Benutzers und Ihrer ersten Root-Gruppe.
   - Ein Benutzer in IhremAWS Konto mit der Erlaubnis, die folgenden Mindestaktionen auszuführen:

     ```
     iam:AddRoleToInstanceProfile
     iam:AttachRolePolicy
     iam:CreateInstanceProfile
     iam:CreatePolicy
     iam:CreateRole
     iam:GetRole
     iam:ListAttachedRolePolicies
     iam:ListPolicies
     iam:ListRoles
     iam:PassRole
     iam:PutRolePolicy
     iam:UpdateAssumeRolePolicy
     ```

     Weitere Informationen finden Sie im Benutzerhandbuch unter Überblick über die IAM-Richtlinien.
2. Wählen Sie im Navigationsbereich Policies aus.
3. Wählen Sie Create Policy (Richtlinie erstellen) aus.
4. Wählen Sie auf der Seite Create Policy die Option JSON aus.

5. Geben Sie für die JSON-Richtlinie das Folgende ein und wählen Sie dann Review Policy (Richtlinie prüfen) aus:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "CloudWatchLogsPolicy",
      "Effect": "Allow",
      "Action": [
        "logs:CreateLogGroup",
        "logs:CreateLogStream",
        "logs:PutLogEvents"
      ],
      "Resource": "*"
    },
    {
      "Sid": "CodeCommitPolicy",
      "Effect": "Allow",
      "Action": [
        "codecommit:GitPull"
      ],
      "Resource": "*"
    },
    {
      "Sid": "S3GetObjectPolicy",
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:GetObjectVersion"
      ],
      "Resource": "*"
    },
    {
      "Sid": "S3PutObjectPolicy",
      "Effect": "Allow",
      "Action": [
        "s3:PutObject"
      ],
      "Resource": "*"
    },
    {
      "Sid": "ECRPullPolicy",
```

```
      "Effect": "Allow",
      "Action": [
        "ecr:BatchCheckLayerAvailability",
        "ecr:GetDownloadUrlForLayer",
        "ecr:BatchGetImage"
      ],
      "Resource": "*"
    },
    {
      "Sid": "ECRAuthPolicy",
      "Effect": "Allow",
      "Action": [
        "ecr:GetAuthorizationToken"
      ],
      "Resource": "*"
    },
    {
      "Sid": "S3BucketIdentity",
      "Effect": "Allow",
      "Action": [
        "s3:GetBucketAcl",
        "s3:GetBucketLocation"
      ],
      "Resource": "*"
    }
  ]
}
```

> **Note**
>
> Diese Richtlinie enthält Anweisungen, die den Zugriff auf eine möglicherweise große Anzahl von AWS-Ressourcen ermöglichen. Um den AWS CodeBuild-Zugriff auf bestimmte AWS-Ressourcen zu beschränken, ändern Sie den Wert des `Resource`-Arrays. Weitere Informationen finden Sie in der Sicherheitsdokumentation zu dem AWS-Service.

6. Geben Sie auf der Seite Review Policy (Richtlinie prüfen) unter Policy Name (Richtlinienname) einen Namen für die Richtlinie ein (z. B. **`CodeBuildServiceRolePolicy`**), und wählen Sie dann Create policy (Richtlinie erstellen).

> **Note**
>
> Wenn Sie einen anderen Namen verwenden, müssen Sie diesen während der gesamten Anleitung verwenden.

7. Wählen Sie im Navigationsbereich Roles aus.
8. Wählen Sie Create role (Rolle erstellen) aus.
9. Wählen Sie auf der Seite Rolle erstellen, wobei AWSService bereits ausgewählt ist CodeBuild, die Option und dann Weiter:Berechtigungen aus.
10. Wählen CodeBuildServiceRolePolicySie auf der Seite „Zugriffsrichtlinien anhängen“ die Option aus und wählen Sie dann Weiter: Überprüfen.
11. Geben Sie auf der Seite Create role and review (Rolle erstellen und prüfen) für Role name (Rollenname) einen Namen für die Rolle ein (z. B. **`CodeBuildServiceRole`**), und wählen Sie dann Create role (Rolle erstellen).

Um eine CodeBuild Servicerolle zu erstellen (AWS CLI)

1. Stellen Sie sicher, dass Sie denAWS CLI mit demAWS Zugriffsschlüssel und demAWS geheimen Zugriffsschlüssel konfiguriert haben, die einer der IAM-Entitäten entsprechen, wie im vorherigen Verfahren beschrieben. Weitere Informationen finden Sie unter Installieren der AWS Command Line Interface im AWS Command Line Interface-Benutzerhandbuch.
2. Erstellen Sie in einem leeren Verzeichnis oder auf der lokalen Workstation oder Instance, auf der die AWS CLI installiert ist, zwei Dateien mit den Namen `create-role.json` und `put-role-policy.json`. Wenn Sie andere Dateinamen wählen, achten Sie darauf, diese in dieser gesamten Anleitung zu verwenden.

   `create-role.json`:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "codebuild.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
```

```
    }
  ]
}
```

> **Note**
>
> Wir empfehlen Ihnen, die `aws:SourceAccount`- und `aws:SourceArn`-Bedingungsschlüssel zu verwenden, um sich vor dem Problem des verwirrten Stellvertreters zu schützen. Sie können beispielsweise die vorherige Vertrauensrichtlinie mit den folgenden Bedingungsblöcken bearbeiten. Das`aws:SourceAccount` ist der Eigentümer des CodeBuild Projekts und das`aws:SourceArn` ist das CodeBuild Projekt ARN.

Wenn Sie Ihre Servicerolle auf einAWS Konto beschränken möchten,`create-role.json` könnte das in etwa so aussehen:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "codebuild.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": [
                        "account-ID"
                    ]
                }
            }
        }
    ]
}
```

Wenn Sie Ihre Servicerolle auf ein bestimmtes CodeBuild Projekt beschränken möchten,`create-role.json` könnte das etwa so aussehen:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "codebuild.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "aws:SourceArn": "arn:aws:codebuild:region-ID:account-
ID:project/project-name"
                }
            }
        }
    ]
}
```

> **Note**
>
> Wenn Sie keinen Namen für Ihr Projekt nicht kennen oder sich nicht für einen Namen für einen Namen für Ihr CodeBuild Projekt entschieden haben und eine Vertrauensrichtlinie für ein bestimmtes ARN ein bestimmtes ARN Muster festlegen möchten, können Sie diesen Teil des ARN durch einen Platzhalter (*) ersetzen. Nachdem Sie Ihr Projekt erstellt haben, können Sie die Vertrauensrichtlinie aktualisieren.

`put-role-policy.json`:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "CloudWatchLogsPolicy",
      "Effect": "Allow",
      "Action": [
        "logs:CreateLogGroup",
        "logs:CreateLogStream",
        "logs:PutLogEvents"
```

```
      ],
      "Resource": "*"
    },
    {
      "Sid": "CodeCommitPolicy",
      "Effect": "Allow",
      "Action": [
        "codecommit:GitPull"
      ],
      "Resource": "*"
    },
    {
      "Sid": "S3GetObjectPolicy",
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:GetObjectVersion"
      ],
      "Resource": "*"
    },
    {
      "Sid": "S3PutObjectPolicy",
      "Effect": "Allow",
      "Action": [
        "s3:PutObject"
      ],
      "Resource": "*"
    },
    {
      "Sid": "S3BucketIdentity",
      "Effect": "Allow",
      "Action": [
        "s3:GetBucketAcl",
        "s3:GetBucketLocation"
      ],
      "Resource": "*"
    }
  ]
}
```

> **Note**
>
> Diese Richtlinie enthält Anweisungen, die den Zugriff auf eine möglicherweise große Anzahl von AWS-Ressourcen ermöglichen. Um den AWS CodeBuild-Zugriff auf bestimmte AWS-Ressourcen zu beschränken, ändern Sie den Wert des `Resource`-Arrays. Weitere Informationen finden Sie in der Sicherheitsdokumentation zu dem AWS-Service.

3. Wechseln Sie in das Verzeichnis, in dem Sie die obigen Dateien gespeichert haben, und führen Sie die folgenden Befehle einzeln und in der angegebenen Reihenfolge aus. Sie können andere Werte für `CodeBuildServiceRole` und `CodeBuildServiceRolePolicy` verwenden. In diesem Fall müssen Sie sie hier verwenden.

```
aws iam create-role --role-name CodeBuildServiceRole --assume-role-policy-document
 file://create-role.json
```

```
aws iam put-role-policy --role-name CodeBuildServiceRole --policy-name
 CodeBuildServiceRolePolicy --policy-document file://put-role-policy.json
```

Erstellen und konfigurieren Sie einen kundenverwalteten Schlüssel für CodeBuild

AWS CodeBuildUm seine Build-Ausgabeartefakte zu verschlüsseln, benötigt es Zugriff auf einen KMS-Schlüssel. CodeBuild Verwendet standardmäßig dieVon AWS verwalteter Schlüssel für Amazon S3 in IhremAWS Konto.

Wenn Sie den nicht verwenden möchtenVon AWS verwalteter Schlüssel, müssen Sie einen kundenverwalteten Schlüssel selbst erstellen und konfigurieren einen kundenverwalteten Schlüssel selbst erstellen und konfigurieren. In diesem Abschnitt wird beschrieben, wie Sie dies mit der IAM-Konsole tun können, wie Sie dies mit der IAM-Konsole tun.

Informationen zu vom Kunden verwalteten Schlüsseln finden Sie im AWS KMSEntwicklerhandbuch unter AWS Key Management ServiceKonzepte und Erstellung von Schlüsseln.

Um einen vom Kunden verwalteten Schlüssel für die Verwendung durch zu konfigurieren CodeBuild, folgen Sie den Anweisungen im Abschnitt „So ändern Sie eine Schlüsselrichtlinie“ unter Ändern einer Schlüsselrichtlinie im AWS KMSEntwicklerhandbuch. Fügen Sie der Schlüsselrichtlinie dann

zwischen *### BEGIN ADDING STATEMENTS HERE ###* und *### END ADDING STATEMENTS HERE ###* die folgenden Anweisungen hinzu. Auslassungspunkte (`...`) werden zur Abkürzung verwendet und weisen auf die Stellen hin, an denen die Anweisungen hinzugefügt werden müssen. Entfernen Sie keine Anweisungen und geben Sie die Auslassungspunkte nicht in die Schlüsselrichtlinie ein.

```
{
  "Version": "2012-10-17",
  "Id": "...",
  "Statement": [
    ### BEGIN ADDING STATEMENTS HERE ###
    {
      "Sid": "Allow access through Amazon S3 for all principals in the account that are
 authorized to use Amazon S3",
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
      },
      "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:DescribeKey"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "kms:ViaService": "s3.region-ID.amazonaws.com",
          "kms:CallerAccount": "account-ID"
        }
      }
    },
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::account-ID:role/CodeBuild-service-role"
      },
      "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
```

```
        "kms:DescribeKey"
      ],
      "Resource": "*"
    },
    ### END ADDING STATEMENTS HERE ###
    {
      "Sid": "Enable IAM User Permissions",
      ...
    },
    {
      "Sid": "Allow access for Key Administrators",
      ...
    },
    {
      "Sid": "Allow use of the key",
      ...
    },
    {
      "Sid": "Allow attachment of persistent resources",
      ...
    }
  ]
}
```

- *Region-ID* steht für die ID derAWS Region, in der sich die zugehörigen Amazon S3-Buckets CodeBuild befinden (z. B.us-east-1).
- Die *Konto-IF steht für die Lebens-ID* desAWS Kontos, das zu dem vom Kunden verwalteten Schlüssel gehört, das zu dem vom Kunden verwalteten Schlüssel gehört, das zu dem vom Kunden
- *CodeBuild-service-role* steht für den Namen der CodeBuild Servicerolle, die Sie zuvor in diesem Thema erstellt oder identifiziert haben.

> **Note**
>
> Um einen vom Kunden verwalteten Schlüssel über die IAM-Konsole zu erstellen oder zu konfigurieren, müssen Sie sichAWS Management Console zunächst mit einer der folgenden Methoden bei der anmelden:
>
> - Ihr AWS-Stammkonto. Dies wird nicht empfohlen. Weitere Informationen finden Sie unter Der Account-Root-Benutzer im Benutzerhandbuch.

- Ein Administratorbenutzer in IhremAWS Konto. Weitere Informationen finden Sie im Benutzerhandbuch unter Erstellen Ihres erstenAWS-Konto Root-Benutzers und Ihrer ersten Root-Gruppe.
- Ein Benutzer in IhremAWS Konto mit der Berechtigung, den vom Kunden verwalteten Schlüssel zu erstellen oder zu ändern. Weitere Informationen finden Sie unter Erforderliche Berechtigungen für die Verwendung der AWS KMS-Konsole im AWS KMS-Entwicklerhandbuch.

Installieren und Konfigurieren der AWS CLI.

Für den ZugriffAWS CodeBuild können Sie dieAWS CLI mit — oder anstelle von — der CodeBuild Konsole, die CodePipeline Konsole oder dieAWS SDKs verwenden. Informationen zur Installation und Konfiguration von finden SieAWS Command Line Interface im AWS Command Line InterfaceBenutzerhandbuch unter Getting Set Up with the.AWS CLI

1. Bestätigen Sie durch Ausführen des folgenden Befehls, ob Ihre Installation des die folgenden BefehleAWS CLI unterstützt CodeBuild, die folgenden Befehle zu bestätigen, dass Ihre Installation

   ```
   aws codebuild list-builds
   ```

   Ist der Befehl erfolgreich, gibt er als Ausgabe Informationen zurück, die wie folgt aussehen sollten:

   ```
   {
     "ids": []
   }
   ```

   Die leeren eckigen Klammern weisen darauf hin, dass Sie noch keine Builds ausgeführt haben.

2. Wenn ein Fehler ausgegeben wird, müssen Sie Ihre aktuelle Version der AWS CLI deinstallieren und dann die neueste Version installieren. Weitere Informationen zum Deinstallieren der AWS CLI und Installieren der AWS Command Line Interface finden Sie im Benutzerhandbuch für AWS Command Line Interface.

Befehlszeilenreferenz für AWS CodeBuild

Die AWS CLI bietet zwei Befehle für das Automatisieren von AWS CodeBuild. Verwenden Sie die Informationen in diesem Thema als Ergänzung zuAWS Command Line Interface-Benutzerhandbuchund dieAWS CLI-Referenz fürAWS CodeBuildaus.

Nicht das, wonach Sie gesucht haben? Wenn Sie dieAWSSDKs zum Aufrufen von CodeBuild finden Sie imAWS SDKs- und Tools-Referenzaus.

Zur Verwendung der Informationen in diesem Thema sollten Sie bereits dieAWS CLIund konfigurierte es für die Verwendung mit CodeBuild, wie unterInstallieren und Konfigurieren der AWS CLI.aus.

So verwenden Sie denAWS CLIInformationen zur Angabe des Endpunkts für CodeBuild finden Sie unterAngeben des AWS CodeBuild-Endpunkts (AWS CLI)aus.

Führen Sie diesen Befehl aus, um eine Liste von CodeBuild-Befehlen abzurufen.

```
aws codebuild help
```

Führen Sie diesen Befehl aus, um Informationen zu einem CodeBuild-Befehl abzurufen, wo*befehl-name*Der Name des Befehls.

```
aws codebuild command-name help
```

Die CodeBuild-Befehle umfassen:

- `batch-delete-builds`: Löscht ein oder mehrere Builds in CodeBuild. Weitere Informationen finden Sie unter Löschen von Builds (AWS CLI).
- `batch-get-builds`: Ruft Informationen zu mehreren Builds in CodeBuild ab. Weitere Informationen finden Sie unter Anzeigen von Build-Details (AWS CLI).
- `batch-get-projects`: Ruft Informationen über ein oder mehrere angegebene Build-Projekte ab. Weitere Informationen finden Sie unter Anzeigen der Details eines Build-Projekts (AWS CLI).
- `create-project`: Erstellt ein Build-Projekt. Weitere Informationen finden Sie unter Erstellen eines Build-Projekts (AWS CLI).
- `delete-project`: Löscht ein Build-Projekt. Weitere Informationen finden Sie unter Löschen eines Build-Projekts (AWS CLI).
- `list-builds`: Listet Amazon-Ressourcenamen (ARNs) für Builds in CodeBuild auf. Weitere Informationen finden Sie unter Anzeigen einer Liste mit Build-IDs (AWS CLI).

- `list-builds-for-project`: Ruft eine Liste von Build-IDs auf, die mit einem angegebenen Build-Projekt verbunden sind. Weitere Informationen finden Sie unter Anzeigen einer Liste mit Build-IDs für ein Build-Projekt (AWS CLI).
- `list-curated-environment-images`: Ruft eine Liste von Docker-Images ab, die von CodeBuild verwaltet werden, die Sie für die Builds einsetzen können. Weitere Informationen finden Sie unter Docker-Images bereitgestellt von CodeBuild.
- `list-projects`: Anzeigen einer Liste mit Build-Projektnamen. Weitere Informationen finden Sie unter Anzeigen einer Liste mit Build-Projektnamen (AWS CLI).
- `start-build`: Startet die Build-Ausführung. Weitere Informationen finden Sie unter Ausführen eines Build (AWS CLI).
- `stop-build`: Versucht, die Ausführung des angegebenen Builds zu stoppen. Weitere Informationen finden Sie unter Stoppen eines Builds (AWS CLI).
- `update-project`: Ändert Informationen über das angegebene Build-Projekt. Weitere Informationen finden Sie unter Ändern der Einstellungen eines Build-Projekts (AWS CLI).

AWS SDKs- und Tools-Referenz für AWS CodeBuild

Wenn Sie eines der AWS SDKs oder Tools zur Automatisierung von AWS CodeBuild verwenden möchten, finden Sie weitere Informationen in den folgenden Ressourcen.

Wenn Sie dieAWS CLIum CodeBuild auszuführen, lesen Sie dieBefehlszeilenreferenzaus.

Unterstützte AWS SDKs und Tools für AWS CodeBuild

FolgendesAWSSDKs und Tools unterstützen CodeBuild:

- Das AWS SDK für C++. Weitere Informationen finden Sie im Abschnitt über den Aws::CodeBuild-Namespace derAWS SDK für C++ API-Referenz.
- Das AWS SDK für Go. Weitere Informationen finden Sie im .codebuild-Abschnitt imAWSSDK for Go — API-Referenzaus.
- Das AWS SDK für Java. Weitere Informationen finden Sie in den Abschnitten `com.amazonaws.services.codebuild` und `com.amazonaws.services.codebuild.model` der AWS-SDK für Java API-Referenz.
- Das AWS-SDK für JavaScript im Browser und das AWS-SDK für JavaScript in Node.js. Weitere Informationen finden Sie im .-Klasse: AWS.CodeBuild-Abschnitt imAWSSDK for JavaScript — API-Referenzaus.

- Das AWS SDK für .NET. Weitere Informationen finden Sie im .Amazon.CodeBuildundAmazon.CodeBuild.ModelNamespace-Abschnitte desAWSSDK for .NET — API-Referenzaus.
- Das AWS SDK für PHP. Weitere Informationen finden Sie im .Namespace Aws\ CodeBuild-Abschnitt imAWSSDK for PHP — API-Referenzaus.
- Das AWS SDK für Python (Boto3). Weitere Informationen finden Sie im .CodeBuild-Abschnitt imBoto 3 — Dokumentationaus.
- Das AWS SDK für Ruby. Weitere Informationen finden Sie im .Modul: Aws:: CodeBuild-Abschnitt imAWSSDK for Ruby — API-Referenzaus.
- Die AWS Tools für PowerShell. Weitere Informationen finden Sie im Abschnitt AWS CodeBuild der AWS Tools für PowerShell Cmdlet-Referenz.

Angeben des AWS CodeBuild-Endpunkts

Sie können die AWS Command Line Interface (AWS CLI) oder eines der AWS-SDKs verwenden, um den von AWS CodeBuild verwendeten Endpunkt anzugeben. Es ist ein Endpunkt für jede Region vorhanden, in der CodeBuild verfügbar ist. Zusätzlich zu einem regionalen Endpunkt, verfügen vier Regionen zudem über einen Endpunkt mit Federal Information Processing Standard (FIPS). Weitere Informationen zu FIPS-Endpunkten finden Sie unter FIPS 140-2-Übersicht.

Die Angabe eines Endpunkts ist optional. Wenn Sie nicht explizit angeben,CodeBuild welcher Endpunkt verwendet werden soll, verwendet der Dienst den Endpunkt, der der Region zugeordnet ist, die IhrAWS Konto verwendet. CodeBuildverwendet standardmäßig nie einen FIPS-Endpunkt. Wenn Sie einen FIPS-Endpunkt verwenden möchten, müssen Sie CodeBuild mit einer der folgenden Methoden diesem Endpunkt zuordnen.

> **Note**
>
> Sie können einen Alias oder einen Regionsnamen verwenden, um einen Endpunkt mit einem AWS-SDK anzugeben. Wenn Sie die AWS CLI nutzen, müssen Sie den vollständige Namen des Endpunkts verwenden.

Informationen zu Endpunkten, die mit CodeBuild verwendet werden können, finden Sie unter CodeBuild-Regionen und Endpunkte.

Themen

Angeben des AWS CodeBuild-Endpunkts (AWS CLI)

Mit der AWS CLI können Sie den Endpunkt angeben, über den mit dem `--endpoint-url`-Argument in einem beliebigen CodeBuild-Befehl auf AWS CodeBuild zugegriffen wird. Führen Sie beispielsweise diesen Befehl aus, um eine Liste der Projekt-Buildnamen (FIPS) in der Region USA Ost (Nord-Virginia) abzurufen:

```
aws codebuild list-projects --endpoint-url https://codebuild-fips.us-
east-1.amazonaws.com
```

Fügen Sie am Anfang des Endpunkts `https://` ein.

Das AWS CLI-Argument `--endpoint-url` steht allen AWS-Services zur Verfügung. Weitere Informationen zu diesen und anderenAWS CLI Argumenten finden Sie unter AWS CLIBefehlsreferenz.

Angeben des AWS CodeBuild-Endpunkts (AWS SDK)

Sie können den Endpunkt für den Zugriff auf AWS CodeBuild mithilfe eines AWS-SDKs angeben. Obwohl in diesem Beispiel das AWS-SDK für Java verwendet wird, können Sie den Endpunkt mit den anderen AWS-SDKs angeben.

Verwenden Sie die`withEndpointConfiguration` Methode, wenn Sie denAWSCodeBuild Client erstellen. Verwenden Sie dieses Format:

```
AWSCodeBuild awsCodeBuild = AWSCodeBuildClientBuilder.standard().
    withEndpointConfiguration(new AwsClientBuilder.EndpointConfiguration("endpoint",
 "region")).
    withCredentials(new AWSStaticCredentialsProvider(sessionCredentials)).
    build();
```

Informationen`AWSCodeBuildClientBuilder` dazu finden Sie unter KlasseAWSCodeBuildClientBuilder.

Die in `withCredentials` verwendeten Anmeldeinformationen müssen vom Typ `AWSCredentialsProvider` sein. Weitere Informationen finden Sie unter Arbeiten mit AWS-Anmeldeinformationen.

Fügen Sie am Anfang des Endpunkts kein `https://` ein.

Wenn Sie einen Endpunkt ohne FIPS angeben möchten, können Sie die Region anstelle des tatsächlichen Endpunkts verwenden. Um beispielsweise den Endpunkt in der Region USA Ost (Nord-Virginia) anzugeben, können Sie`us-east-1` anstelle des vollständigen Endpunktes den Namen, verwenden`codebuild.us-east-1.amazonaws.com`.

Wenn Sie einen FIPS-Endpunkt angeben möchten, können Sie Ihren Code mit einem Alias vereinfachen. Nur FIPS-Endpunkte verfügen über einen Alias. Andere Endpunkte müssen mithilfe ihrer Region oder des vollständigen Namens angegeben werden.

In der folgenden Tabelle werden die Alias für jeden der vier verfügbaren FIPS-Endpunkte aufgeführt:

Name der Region	Region	Endpunkt	Alias
USA Ost (Nord-Virginia)	us-east-1	codebuild-fips.us-east-1.amazonaws.com	us-east-1-fips
USA Ost (Ohio)	us-east-2	codebuild-fips.us-east-2.amazonaws.com	us-east-2-fips
USA West (Nordkalifornien)	us-west-1	codebuild-fips.us-west-1.amazonaws.com	us-west-1-fips
USA West (Oregon)	us-west-2	codebuild-fips.us-west-2.amazonaws.com	us-west-2-fips

Um die Verwendung des FIPS Endpunktes in der Region USA West (Oregon) mit einem Alias zu spezifizieren:

```
AWSCodeBuild awsCodeBuild = AWSCodeBuildClientBuilder.standard().
    withEndpointConfiguration(new AwsClientBuilder.EndpointConfiguration("us-west-2-
fips", "us-west-2")).
    withCredentials(new AWSStaticCredentialsProvider(sessionCredentials)).
    build();
```

Um die Verwendung des Nicht-FIPS Endpunktes in der Region USA Ost (Nord-Virginia) zu spezifizieren, gehen Sie wie folgt vor:

```
AWSCodeBuild awsCodeBuild = AWSCodeBuildClientBuilder.standard().
    withEndpointConfiguration(new AwsClientBuilder.EndpointConfiguration("us-east-1",
 "us-east-1")).
    withCredentials(new AWSStaticCredentialsProvider(sessionCredentials)).
    build();
```

Um die Verwendung des Nicht-FIPS Endpunktes in der Region Asien-Pazifik (Mumbai) zu spezifizieren:

```
AWSCodeBuild awsCodeBuild = AWSCodeBuildClientBuilder.standard().
    withEndpointConfiguration(new AwsClientBuilder.EndpointConfiguration("ap-south-1",
 "ap-south-1")).
    withCredentials(new AWSStaticCredentialsProvider(sessionCredentials)).
    build();
```

Verwenden von AWS CodePipeline mit AWS CodeBuild zum Testen von Codes und zum Ausführen von Builds

Sie können Ihr Freigabeverfahren durch den Einsatz von AWS CodePipeline automatisieren, um Ihren Code zu testen und Builds mit AWS CodeBuild auszuführen.

In der folgenden Tabelle werden Aufgaben und Methoden aufgeführt, die zur Ausführung zur Verfügung stehen. Die Durchführung dieser Aufgaben mit AWS SDKs wird in diesem Thema nicht beschrieben.

Aufgabe	Verfügbare Ansätze	In diesem Thema beschriebene Ansätze
Erstellen Sie eine Continuous Delivery (CD)-Pipeline mit CodePipeline , die Builds mit automatisiert CodeBuild	• CodePipeline - Konsole • AWS CLI • AWS SDKs	• Verwenden der CodePipeline-Konsole • Verwenden der AWS CLI • Sie können die Informationen in diesem Thema für AWS SDKs anpassen. Weitere Informationen finden Sie in der `create-pipeline` Aktionsdokumentation für Ihre Programmiersprache im Abschnitt SDKs von Tools for Amazon Web Services oder unter `CreatePipeline` in der APIAWS CodePipeline-Referenz zu .
Hinzufügen der Test- und Build-Automatisierung mit CodeBuild zu einer bestehenden Pipeline in CodePipeline	• CodePipeline - Konsole • AWS CLI • AWS SDKs	• Verwenden der CodePipeline Konsole zum Hinzufügen von Build-Automatisierung • Verwenden der CodePipeline Konsole zum Hinzufügen der Testautomatisierung • Für die können Sie die Informationen in diesem Thema anpassenAWS CLI, um eine Pipeline zu erstellen, die eine CodeBuild Build-Aktion oder Testaktion enthält. Weitere Informationen finden Sie unter Bearbeiten einer Pipeline (AWS CLI) und in der CodePipeline Referenz zur Pipeline-Struktur im AWS CodePipeline - Benutzerhandbuch. • Sie können die Informationen in diesem Thema für AWS SDKs anpassen. Weitere Informationen finden Sie in der `update-pipeline` Aktionsdokumentation für Ihre Programmiersprache über den Abschnitt SDKs unter Tools für Amazon Web Services oder unter `UpdatePipeline` in der APIAWS CodePipeline-Referenz zu .

Voraussetzungen

1. Beantworten Sie die Fragen in Planen eines Builds.

2. Wenn Sie anstelle eines -AWSStammkonto CodePipeline s oder eines Administratorbenutzers einen -Benutzer für den Zugriff auf verwenden, fügen Sie die verwaltete Richtlinie mit dem Namen `AWSCodePipelineFullAccess` dem Benutzer (oder der IAM-Gruppe, zu der der Benutzer gehört) an. Es wird nicht empfohlen, ein AWS-Stammkonto zu verwenden. Diese Richtlinie erteilt dem Benutzer die Berechtigung zum Erstellen der Pipeline in CodePipeline. Weitere Informationen finden Sie unter Anfügen verwalteter Richtlinien im -Benutzerhandbuch.

   > **Note**
   >
   > Die IAM-Entität, die die Richtlinie an den Benutzer anfügt (oder an die IAM-Gruppe, zu der der Benutzer gehört), muss in IAM über die Berechtigung zum Anfügen von Richtlinien verfügen. Weitere Informationen finden Sie unter Delegieren von Berechtigungen zur Verwaltung von IAM-Benutzern, -Gruppen und -Anmeldeinformationen im -Benutzerhandbuch.

3. Erstellen Sie eine CodePipeline Servicerolle, falls noch keine in Ihrem AWS Konto verfügbar ist. CodePipeline verwendet diese Servicerolle, um in Ihrem Namen mit anderen -AWSServices, einschließlich AWS CodeBuild, zu interagieren. Um beispielsweise die AWS CLI zum Erstellen einer CodePipeline Servicerolle zu verwenden, führen Sie den IAM-`create-role`Befehl aus:

   Für Linux, macOS oder Unix:

   ```
   aws iam create-role --role-name AWS-CodePipeline-CodeBuild-Service-Role
    --assume-role-policy-document '{"Version":"2012-10-17","Statement":
   {"Effect":"Allow","Principal":
   {"Service":"codepipeline.amazonaws.com"},"Action":"sts:AssumeRole"}}'
   ```

   Für Windows:

   ```
   aws iam create-role --role-name AWS-CodePipeline-CodeBuild-Service-Role --assume-
   role-policy-document "{\"Version\":\"2012-10-17\",\"Statement\":{\"Effect\":
   \"Allow\",\"Principal\":{\"Service\":\"codepipeline.amazonaws.com\"},\"Action\":
   \"sts:AssumeRole\"}}"
   ```

   > **Note**
   >
   > Die IAM-Entität, die diese CodePipeline Servicerolle erstellt, muss in IAM über die Berechtigung zum Erstellen von Servicerollen verfügen.

4. Nachdem Sie eine CodePipeline Servicerolle erstellt oder eine vorhandene identifiziert haben, müssen Sie der Servicerolle die Standard CodePipeline -Servicerollenrichtlinie hinzufügen, wie unter Überprüfen der Standard CodePipeline -Servicerollenrichtlinie im AWS CodePipeline -Benutzerhandbuch beschrieben, sofern sie noch nicht Teil der Richtlinie für die Rolle ist.

   > **Note**
   >
   > Die IAM-Entität, die diese CodePipeline Servicerollenrichtlinie hinzufügt, muss in IAM über die Berechtigung verfügen, Servicerollenrichtlinien zu Servicerollen hinzuzufügen.

5. Erstellen Sie den Quellcode und laden Sie ihn in einen Repository-Typ hoch, der von CodeBuild und unterstützt wird CodePipeline, z. B. CodeCommit, Amazon S3, Bitbucket oder GitHub. Der Quellcode muss eine Build-Spezifikationsdatei enthalten. Sie können eine deklarieren, wenn Sie später in diesem Thema ein Build-Projekt definieren. Weitere Informationen hierzu finden Sie unter Build-Spezifikationsreferenz.

   > **Important**
   >
   > Wenn Sie planen, die Pipeline zur Bereitstellung des Quellcodes einzusetzen, muss das Build-Ausgabeartifakt mit dem von Ihnen verwendeten Bereitstellungssystem kompatibel sein.
   >
   > - Informationen AWS OpsWorkszu finden Sie unter Anwendungsquelle und Verwenden von CodePipeline mit AWS OpsWorks im AWS OpsWorks -Benutzerhandbuch.

Themen

- Erstellen einer Pipeline, die CodeBuild (CodePipeline-Konsole) verwendet
- Erstellen einer Pipeline unter Verwendung von CodeBuild (AWS CLI)
- Hinzufügen einer CodeBuild-Build-Aktion zu einer Pipeline (CodePipeline-Konsole)
- Hinzufügen einer CodeBuild-Testaktion zu einer Pipeline (CodePipeline-Konsole)

Erstellen einer Pipeline, die CodeBuild (CodePipeline-Konsole) verwendet

Setzen Sie das folgende Verfahren ein, um eine Pipeline zu erstellen, die zum Aufbau und zur Bereitstellung des Quellcodes CodeBuild verwendet.

So erstellen Sie eine Pipeline, die nur Ihren Quellcode testet:

- Setzen Sie die das folgende Verfahren ein, um eine Pipeline zu erstellen, und anschließend löschen Sie die Build- und Beta-Stufen aus der Pipeline. Fügen Sie dann mithilfe des Hinzufügen einer CodeBuild-Testaktion zu einer Pipeline (CodePipeline-Konsole)-Verfahrens dieses Themas eine Testaktion mit CodeBuild zur Pipeline hinzu.
- Erstellen Sie die Pipeline mithilfe eines anderen Verfahrens dieses Themas und fügen Sie dann mit dem Hinzufügen einer CodeBuild-Testaktion zu einer Pipeline (CodePipeline-Konsole)-Verfahren dieses Themas eine Testaktion mit CodeBuild zur Pipeline hinzu.

Erstellen einer Pipeline mit CodeBuild mithilfe des Assistenten zum Erstellen einer Pipeline in CodePipeline

1. Melden Sie sich an der AWS Management Console Konsole an, indem Sie Folgendes verwenden:

   - Ihr AWS-Stammkonto. Dies wird nicht empfohlen. Weitere Informationen finden Sie im Benutzerhandbuch unter Der Root-Benutzer des Kontos.
   - Ein Administratorbenutzer in IhremAWS Konto. Weitere Informationen finden Sie im Benutzerhandbuch unter Erstellen Ihres erstenAWS-Konto Root-Benutzers und Ihrer ersten Root-Gruppe.
   - Ein Benutzer in IhremAWS Konto mit der Erlaubnis, die folgenden Mindestaktionen zu verwenden:

     ```
     codepipeline:*
     iam:ListRoles
     iam:PassRole
     s3:CreateBucket
     s3:GetBucketPolicy
     s3:GetObject
     s3:ListAllMyBuckets
     s3:ListBucket
     s3:PutBucketPolicy
     codecommit:ListBranches
     codecommit:ListRepositories
     codedeploy:GetApplication
     codedeploy:GetDeploymentGroup
     codedeploy:ListApplications
     codedeploy:ListDeploymentGroups
     ```

```
elasticbeanstalk:DescribeApplications
elasticbeanstalk:DescribeEnvironments
lambda:GetFunctionConfiguration
lambda:ListFunctions
opsworks:DescribeStacks
opsworks:DescribeApps
opsworks:DescribeLayers
```

2. Öffnen Sie dieAWS CodePipeline Konsole unter https://console.aws.amazon.com/codesuite/codepipeline/home.
3. Wählen Sie in derAWS Regionsauswahl dieAWS Region aus, in der sich IhreAWS Build-Projektressourcen befinden. Dies muss eineAWS Region sein, in derCodeBuild es unterstützt wird. Weitere Informationen finden Sie unter AWS CodeBuild im Allgemeine Amazon Web Services-Referenz.
4. Erstellen Sie eine Pipeline. Wenn eine CodePipeline-Informationsseite angezeigt wird, wählen Sie Create Pipeline (Pipeline erstellen) aus. Wenn eine Pipelines-Seite angezeigt wird, wählen Sie die Option Create pipeline (Pipeline erstellen) aus.
5. Geben Sie auf der Seite Step 1: Choose pipeline settings (Schritt 1: Pipeline-Einstellungen auswählen) unter Pipeline name (Pipeline-Name) einen Namen für die Pipeline ein (z. B. **`CodeBuildDemoPipeline`**). Wenn Sie einen anderen Namen auswählen, müssen Sie diesen während der gesamten Anleitung verwenden.
6. Führen Sie für Role name (Rollenname) einen der folgenden Schritte aus:

   Wählen Sie New service role (Neue Servicerolle) aus und geben Sie in Role Name (Rollenname) den Namen für Ihre neue Servicerolle aus.

   Wählen Sie Existing service role (Vorhandene Servicerolle) und klicken Sie dann auf die CodePipeline-Servicerolle, die Sie erstellt oder als Bestandteil der Voraussetzungen für dieses Thema ermittelt haben.
7. Für Artifact store (Artefact speichern) führen Sie einen der folgenden Schritte aus:

   - Wählen Sie den Standardspeicherort, um den standardmäßigen Artefaktspeicher, z. B. den als Standard festgelegten S3-Artifact-Bucket, für Ihre Pipeline in derAWS Region zu verwenden, die Sie für Ihre Pipeline ausgewählt haben.
   - Wählen Sie Benutzerdefinierter Standort, wenn Sie bereits über einen vorhandenen Artefaktspeicher verfügen, den Sie erstellt haben, z. B. einen S3-Artifact-Bucket, in derselbenAWS Region wie Ihre Pipeline.

> **Note**
>
> Dies ist nicht der Quell-Bucket für den Quellcode Ihrer Pipeline. sondern um den Artefaktspeicher für Ihre Pipeline. Für jede Pipeline benötigen Sie einen separaten Artefaktspeicher, z. B. einen S3 Bucket. Sie benötigen einen separaten Artefaktspeicher, z.AWS B. einen S3 Bucket.

8. Wählen Sie Weiter.
9. Führen Sie auf der Seite Step 2: Add source stage für Source provider einen der folgenden Schritte aus:

   - Wenn Ihr Quellcode in einem S3 Bucket gespeichert benötigen, wählen Sie Amazon S3 aus. Wählen Sie für Bucket, den S3-Bucket aus, der Ihren Quellcode enthält. Geben Sie für S3 object key (S3-Objektschlüssel) den Namen der Datei an, die den Quellcode enthält (z. B. `file-name.zip`). Wählen Sie Weiter.
   - Wenn Ihr Quellcode in einem AWS CodeCommit-Repository gespeichert ist, wählen Sie CodeCommit aus. Wählen Sie für Repository name den Namen des Repositories aus, das den Quellcode enthält. Wähen Sie für Branch name (Name der Verzweigung) den Namen der Verzweigung aus, die die Version des Quellcodes enthält, die Sie erstellen möchten. Wählen Sie Weiter.
   - Wenn Ihr Quellcode in einemGitHub Repository gespeichert ist, wählen Sie GitHub. Wählen SieGitHub Connect mit und folgen Sie den Anweisungen zur AuthentifizierungGitHub. Wählen Sie für Repository den Namen des Repositories aus, das den Quellcode enthält. Wähen Sie für Branch (Verzweigung) den Namen der Verzweigung aus, die die Version des Quellcodes enthält, die Sie erstellen möchten.

   Wählen Sie Weiter.
10. Wählen Sie auf der Seite Step 3: Add build stage für Build provider CodeBuild aus.
11. Wenn Sie bereits ein Build-Projekt haben, das Sie verwenden möchten, wählen Sie als Projektname den Namen des Build-Projekts aus und fahren Sie mit dem nächsten Schritt in diesem Verfahren fort.

    Wenn Sie ein neuesCodeBuild Build-Projekt erstellen müssen, folgen Sie den Anweisungen unterErstellen Sie ein Build-Projekt (Konsole) und kehren Sie zu diesem Verfahren zurück.

Wenn Sie ein bestehendes Build-Projekt auswählen, muss es bereits über definierte Einstellungen für Build-Ausgabeartefakte verfügen (auch wenn CodePipeline diese überschreibt). Weitere Informationen finden Sie unter Ändern der Einstellungen eines Build-Projekts (Konsole).

> **Important**
>
> Wenn Sie Webhooks für ein CodeBuild-Projekt aktivieren und das Projekt als Build-Schritt in CodePipeline verwendet wird, werden zwei identische Builds für jeden Commit erstellt. Ein Build wird über Webhooks ausgelöst und einer durch CodePipeline. Da die Fakturierung pro Build erfolgt, werden Ihnen beide Builds in Rechnung gestellt. Wenn Sie also CodePipeline verwenden, sollten Sie Webhooks in CodeBuild deaktivieren. Deaktivieren Sie in der AWS CodeBuild-Konsole das Webhook-Feld. Weitere Informationen finden Sie unter Ändern der Einstellungen eines Build-Projekts (Konsole).

12. Führen Sie auf der Seite Step 4: Add deploy stage einen der folgenden Schritte aus:

    - Wenn Sie das Build-Ausgabeartefakt nicht bereitstellen möchten, wählen Sie Skip (Überspringen) und bestätigen diese Auswahl auf Anforderung.
    - Wenn Sie Build-Ausgabeartefakte bereitstellen möchten, wählen Sie für Deployment provider (bereitstellungs-Provider) einen Bereitstellungs-Provider aus und geben dann die Einstellungen an, wenn Sie dazu aufgefordert werden.

    Wählen Sie Weiter.

13. Überprüfen Sie auf der Seite Review (Überprüfen) Ihre Einstellungen und wählen Sie dann Create pipeline (Pipeline erstellen) aus.

14. Wenn die Pipeline erfolgreich läuft, können Sie die Build-Ausgabeartefakte abrufen. Wählen Sie, wenn die Pipeline in der CodePipeline-Konsole angezeigt wird, in der Build-Aktion den Tooltip aus. Notieren Sie sich den Wert für das Ausgabe-Artefakt (z. B. MyAppBuild).

> **Note**
>
> Sie können das Build-Ausgabeartefakt auch unter dem Link Build artifacts auf der Build-Detailseite in der CodeBuild-Konsole abrufen. Lassen Sie die restlichen Schritte in diesem Verfahren aus, um zu dieser Seite zu gelangen, und sehen Sie sich diese unter Anzeigen von Build-Details (Konsole) an.

15. Öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
16. Öffnen Sie in der Liste der Buckets auf den von der Pipeline verwendeten Bucket. Der Name des Buckets sollte das Format `codepipeline-`*`region-ID`*`-`*`random-number`* aufweisen. Sie können den verwendenAWS CLI, um denCodePipelineget-pipeline Befehl auszuführen, um den Namen des Buckets abzurufen, wobei *`my-pipeline-name`*sich der Anzeigename Ihrer Pipeline befindet:

    ```
    aws codepipeline get-pipeline --name my-pipeline-name
    ```

    In der Ausgabe enthalt das Objekt `pipeline` ein Objekt mit Namen `artifactStore`, das einen Wert `location` mit dem Namen des Buckets enthält.
17. Öffnen Sie den Ordner, der dem Namen Ihrer Pipeline entspricht (abhängig von der Länge des Pipeline-Namens kann der Name des Ordners verkürzt sein), und dann öffnen Sie den Ordner, der dem Wert für Output artifact (Ausgabeartefakt) entspricht, den Sie zuvor notiert haben.
18. Extrahieren Sie den Inhalt der Datei . Wenn in diesem Ordner mehrere Dateien enthalten sind, extrahieren Sie die Inhalte der Datei mit dem neuesten Zeitstempel Last Modified. (Sie müssen die Datei möglicherweise mit der Erweiterung `.zip` versehen, damit Sie vom ZIP-Dienstprogramm Ihres Systems unterstützt wird.) Das Build-Ausgabeartefakt ist in den extrahierten Inhalten der Datei vorhanden.
19. Wenn Sie CodePipeline angewiesen haben den Build-Ausgabeartefakt bereitzustellen, verwenden Sie die Anweisungen des Bereitstellungs-Providers, um zum Build-Ausgabeartefakt über die Bereitstellungsziele zu gelangen.

Erstellen einer Pipeline unter Verwendung von CodeBuild (AWS CLI)

Setzen Sie das folgende Verfahren ein, um eine Pipeline zu erstellen, die zum Aufbau Ihres Quellcodes CodeBuild verwendet.

Um dieAWS CLI zu verwenden, um eine Pipeline zu erstellen, die Ihren erstellten Quellcode bereitstellt oder die nur Ihren Quellcode testet, können Sie die Anweisungen unter Pipeline bearbeiten (AWS CLI) und die CodePipelinePipeline-Strukturreferenz im AWS CodePipelineBenutzerhandbuch anpassen.

1. Erstellen oder Ermitteln eines Build-Projekts in CodeBuild. Weitere Informationen finden Sie unter Erstellen eines Build-Projekts.

> **Important**
>
> Das Build-Projekt muss die Einstellungen des Build-Ausgabeartefakts definieren (auch wenn CodePipeline diese überschreibt). Weitere Informationen finden Sie in der Beschreibung von `artifacts` in Erstellen eines Build-Projekts (AWS CLI).

2. Stellen Sie sicher, dass Sie denAWS CLI mit demAWS Zugriffsschlüssel und demAWS geheimen Zugriffsschlüssel konfiguriert haben, die einer der in diesem Thema beschriebenen IAM-Entitäten entsprechen. Weitere Informationen finden Sie unter Einrichtung der AWS Command Line Interface im AWS Command Line Interface-Benutzerhandbuch.
3. Erstellen Sie eine JSON-formatiere Datei, die die Struktur der Pipeline darstellt. Benennen Sie die Datei `create-pipeline.json` oder ähnlich. So erstellt diese JSON-formatierte Struktur beispielsweise eine Pipeline mit einer Quellaktion, die sich auf ein S3-Eingabe-Bucket und eine Build-Aktion bezieht, die CodeBuild verwendet:

```
{
  "pipeline": {
    "roleArn": "arn:aws:iam::<account-id>:role/<AWS-CodePipeline-service-role-name>",
    "stages": [
      {
        "name": "Source",
        "actions": [
          {
            "inputArtifacts": [],
            "name": "Source",
            "actionTypeId": {
              "category": "Source",
              "owner": "AWS",
              "version": "1",
              "provider": "S3"
            },
            "outputArtifacts": [
              {
                "name": "MyApp"
              }
            ],
            "configuration": {
              "S3Bucket": "<bucket-name>",
              "S3ObjectKey": "<source-code-file-name.zip>"
```

```
            },
            "runOrder": 1
          }
        ]
      },
      {
        "name": "Build",
        "actions": [
          {
            "inputArtifacts": [
              {
                "name": "MyApp"
              }
            ],
            "name": "Build",
            "actionTypeId": {
              "category": "Build",
              "owner": "AWS",
              "version": "1",
              "provider": "CodeBuild"
            },
            "outputArtifacts": [
              {
                "name": "default"
              }
            ],
            "configuration": {
              "ProjectName": "<build-project-name>"
            },
            "runOrder": 1
          }
        ]
      }
    ],
    "artifactStore": {
      "type": "S3",
      "location": "<CodePipeline-internal-bucket-name>"
    },
    "name": "<my-pipeline-name>",
    "version": 1
  }
}
```

In diesen JSON-formatierten Daten:

- Der Wert von `roleArn` muss dem ARN der CodePipeline-Servicerolle entsprechen, die Sie als Teil der Voraussetzungen erstellt oder ermittelt haben.
- Die Werte von `S3Bucket` und `S3ObjectKey` in `configuration` gehen davon aus, dass der Quellcode in einem S3-Bucket gespeichert ist. Informationen zu den Einstellungen für andere Repository-Typen von Quellcodes erhalten Sie unter Referenz der CodePipeline-Pipeline-Struktur im AWS CodePipeline-Benutzerhandbuch.
- Der Wert von `ProjectName` ist der Name des von Ihnen zu einem früheren Zeitpunkt erstellten CodeBuild-Build-Projekts.
- Der Wert von `location` ist der Name des S3-Buckets, der von dieser Pipeline verwendet wird. Weitere Informationen finden Sie unter Erstellen einer Richtlinie für einen S3-Bucket zur Verwendung als Artefaktspeicher für CodePipeline im AWS CodePipeline-Benutzerhandbuch.
- Der Wert von `name` ist der Name dieser Pipeline. Alle Pipeline-Namen müssen in Ihrem Konto eindeutig sein.

Auch wenn diese Daten ausschließlich eine Quellaktion und eine Build-Aktion beschreiben, können Sie Aktionen für Aktivitäten in Bezug auf das Testen, Bereitstellen des Build-Ausgabeartefakts, Weiterleiten von AWS Lambda-Funktionen und viele weitere ergänzen. Weitere Informationen finden Sie unter der Referenz der AWS CodePipeline-Pipeline-Struktur im AWS CodePipeline-Benutzerhandbuch.

4. Wechseln Sie zu dem Ordner, der die JSON-Datei enthält, und führen Sie dann den CodePipeline-Befehl create-pipeline unter Angabe des Dateinamens aus:

```
aws codepipeline create-pipeline --cli-input-json file://create-pipeline.json
```

> **Note**
>
> Sie müssen die Pipeline in einer AWS-Region erstellen, in der CodeBuild unterstützt wird. Weitere Informationen finden Sie unter AWS CodeBuild im Allgemeine Amazon Web Services-Referenz.

Die Daten im JSON-Format werden in der Ausgabe angezeigt und CodePipeline erstellt die Pipeline.

5. Um Informationen über den Status der Pipeline zu erhalten, führen Sie den CodePipeline Befehl get-pipeline-state unter Angabe des Namens der Pipeline aus:

```
aws codepipeline get-pipeline-state --name <my-pipeline-name>
```

Suchen Sie in der Ausgabe nach Informationen, die bestätigen, dass der Build erfolgreich war. Auslassungspunkte (`...`) werden verwendet, um zu zeigen, dass stehen für Daten zur Abkürzung ausgelassen wurden.

```
{
  ...
  "stageStates": [
    ...
    {
      "actionStates": [
        {
          "actionName": "CodeBuild",
          "latestExecution": {
            "status": "SUCCEEDED",
            ...
          },
          ...
        }
      ]
    }
  ]
}
```

Wenn Sie diesen Befehl zu früh ausführen, bekommen Sie die Informationen über die Build-Aktion möglichweise nicht angezeigt. Sie müssen diesen Befehl möglicherweise mehrere Male ausführen, bis die Pipeline die Ausführung der Build-Aktion abgeschlossen hat.

6. Befolgen Sie nach einem erfolgreichen Build die Anweisungen, um den Build-Ausgabeartefakt abzurufen. Öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.

> **Note**
>
> Sie können das Build-Ausgabeartefakt auch unter dem Link Build artifacts auf der dazugehörigen Build-Detailseite in der CodeBuild-Konsole abrufen. Lassen Sie die

restlichen Schritte in diesem Verfahren aus, um zu dieser Seite zu gelangen, und sehen Sie sich diese unter Anzeigen von Build-Details (Konsole) an.

7. Öffnen Sie in der Liste der Buckets auf den von der Pipeline verwendeten Bucket. Der Name des Buckets sollte das Format `codepipeline-<region-ID>-<random-number>` aufweisen. Sie können den Bucket-Namen von der Datei `create-pipeline.json` abrufen oder den CodePipeline get-pipeline-Befehl zum Abruf des Bucket-Namens ausführen.

```
aws codepipeline get-pipeline --name <pipeline-name>
```

   In der Ausgabe enthalt das Objekt `pipeline` ein Objekt mit Namen `artifactStore`, das einen Wert `location` mit dem Namen des Buckets enthält.
8. Öffnen Sie den Ordner, der dem Namen Ihrer Pipeline entspricht (z. B. *<pipeline-name>*).
9. Öffnen Sie in diesem Ordner den Ordner mit Namen `default`.
10. Extrahieren Sie den Inhalt der Datei . Wenn in diesem Ordner mehrere Dateien enthalten sind, extrahieren Sie die Inhalte der Datei mit dem neuesten Zeitstempel Last Modified. (Sie müssen die Datei möglicherweise mit der Erweiterung `.zip` versehen, damit Sie vom ZIP-Dienstprogramm Ihres Systems unterstützt wird.) Das Build-Ausgabeartefakt ist in den extrahierten Inhalten der Datei vorhanden.

Hinzufügen einer CodeBuild-Build-Aktion zu einer Pipeline (CodePipeline-Konsole)

1. Melden Sie sich an der AWS Management Console Konsole an, indem Sie Folgendes verwenden:

   - Ihr AWS-Stammkonto. Dies wird nicht empfohlen. Weitere Informationen finden Sie im Benutzerhandbuch unter Der Root-Benutzer des Kontos.
   - Ein Administratorbenutzer in IhremAWS Konto. Weitere Informationen finden Sie im Benutzerhandbuch unter Erstellen Ihres erstenAWS-Konto Root-Benutzers und Ihrer ersten Root-Gruppe.
   - Ein Benutzer in IhremAWS Konto mit der Erlaubnis, die folgenden Mindestaktionen auszuführen:

```
codepipeline:*
iam:ListRoles
```

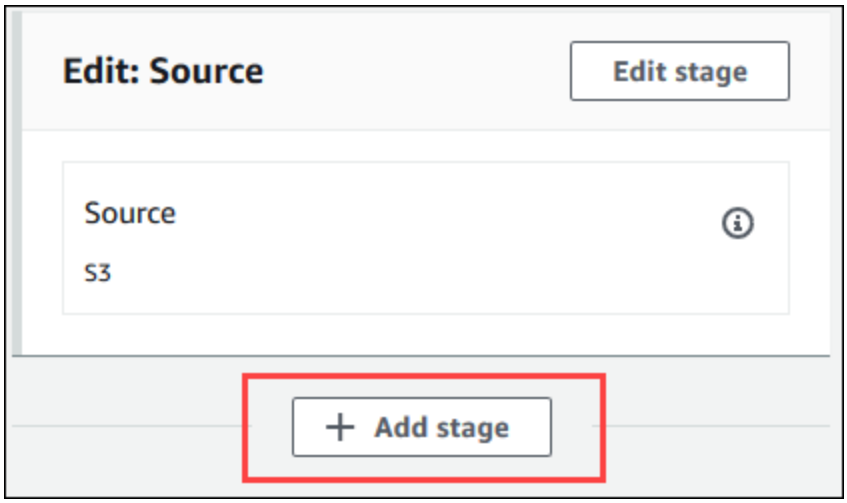
```
iam:PassRole
s3:CreateBucket
s3:GetBucketPolicy
s3:GetObject
s3:ListAllMyBuckets
s3:ListBucket
s3:PutBucketPolicy
codecommit:ListBranches
codecommit:ListRepositories
codedeploy:GetApplication
codedeploy:GetDeploymentGroup
codedeploy:ListApplications
codedeploy:ListDeploymentGroups
elasticbeanstalk:DescribeApplications
elasticbeanstalk:DescribeEnvironments
lambda:GetFunctionConfiguration
lambda:ListFunctions
opsworks:DescribeStacks
opsworks:DescribeApps
opsworks:DescribeLayers
```

2. Öffnen Sie dieCodePipeline Konsole unter https://console.aws.amazon.com/codesuite/codepipeline/home.
3. Wählen Sie in derAWS Regionsauswahl dieAWS Region aus, in der sich Ihre Pipeline befindet. Dies muss eine Region sein, in der CodeBuild unterstützt wird. Weitere Informationen finden Sie unter CodeBuild im Allgemeine Amazon Web Services-Referenz.
4. Wählen Sie auf der Seite All Pipelines (Alle Pipelines) den Namen der Pipeline aus.
5. Wählen Sie auf der Detailseite für die Pipeline für die Aktion Source (Quelle) den Tooltip aus. Notieren Sie sich den Wert für das Ausgabe-Artefakt (z. B. MyApp).

> **Note**
>
> Dieses Verfahren verdeutlicht, wie Sie eine Build-Aktion innerhalb einer Build-Stufe zwischen den Stufen Source (Quelle) und Beta einfügen. Wenn Sie die Build-Aktion an einer anderen Stelle hinzufügen möchten, wählen Sie den Tooltip für die Aktion direkt vor der Stelle aus, an der Sie die Build-Aktion hinzufügen möchten und notieren Sie sich den Wert für das Output artifact (Ausgabeartefakt).

6. Wählen Sie Edit (Bearbeiten) aus.
7. Wählen Sie zwischen den Stufen Source (Quelle) und Beta Add stage (Stufe hinzufügen) aus.

> **Note**
>
> Dieses Verfahren verdeutlicht, wie Sie Ihrer Pipeline eine Build-Stufe zwischen den Stufen Source (Quelle) und Beta hinzufügen. Um einer bestehenden Stufe eine Build-Aktion hinzuzufügen, klicken Sie in der bestehenden Stufe auf Edit stage (Stufe bearbeiten). Fahren Sie dann mit Schritt 8 dieses Verfahrens fort. Um die Build-Stufe an einer anderen Stelle hinzuzufügen, klicken Sie an der gewünschten Stelle auf Add stage (Stufe hinzufügen).



8. Geben Sie unter Stage name (Stufenname) den Namen der Build-Stufe ein (z. B. **`Build`**). Wenn Sie einen anderen Namen wählen, verwenden Sie diesen im gesamten Verfahren.
9. Wählen Sie in der ausgewählten Stufe Add action (Aktion hinzufügen).

> **Note**
>
> Dieses Verfahren verdeutlicht, wie Sie die Build-Aktion innerhalb einer Build-Stufe einfügen. Um die Build-Aktion an einer anderen Stelle hinzuzufügen, klicken Sie an der gewünschten Stelle auf Add stage (Stufe hinzufügen). Sie müssen möglicherweise zuerst Edit stage (Stufe bearbeiten) in der bestehenden Stufe an der Stelle wählen, an der Sie die Build-Aktion hinzufügen möchten.

10. Geben Sie in Edit action (Aktion bearbeiten) unter Action name (Aktionsname) einen Namen für die Aktion ein (z. B. **`CodeBuild`**). Wenn Sie einen anderen Namen wählen, verwenden Sie diesen im gesamten Verfahren.
11. Wählen Sie für Action provider (Aktions-Provider) die Option CodeBuild.
12. Wenn Sie bereits ein Build-Projekt haben, das Sie verwenden möchten, wählen Sie als Projektname den Namen des Build-Projekts aus und fahren Sie mit dem nächsten Schritt in diesem Verfahren fort.

    Wenn Sie ein neuesCodeBuild Build-Projekt erstellen müssen, folgen Sie den Anweisungen unterErstellen Sie ein Build-Projekt (Konsole) und kehren Sie zu diesem Verfahren zurück.

    Wenn Sie ein bestehendes Build-Projekt auswählen, muss es bereits über definierte Einstellungen für Build-Ausgabeartefakte verfügen (auch wenn CodePipeline diese überschreibt). Weitere Informationen finden Sie in der Beschreibung von Artifacts (Artefakte) in Erstellen Sie ein Build-Projekt (Konsole) oder Ändern der Einstellungen eines Build-Projekts (Konsole).

    > **Important**
    >
    > Wenn Sie Webhooks für ein CodeBuild-Projekt aktivieren und das Projekt als Build-Schritt in CodePipeline verwendet wird, werden zwei identische Builds für jeden Commit erstellt. Ein Build wird über Webhooks ausgelöst und einer durch CodePipeline. Da die Fakturierung pro Build erfolgt, werden Ihnen beide Builds in Rechnung gestellt. Wenn Sie also CodePipeline verwenden, sollten Sie Webhooks in CodeBuild deaktivieren. Deaktivieren Sie in der CodeBuild-Konsole das Webhook-Feld. Weitere Informationen finden Sie unter Ändern der Einstellungen eines Build-Projekts (Konsole)
13. Wählen Sie für Input artifacts (Eingabeartefakte) das Ausgabeartefakt aus, das Sie in zuvor dieser Anleitung notiert haben.
14. Geben Sie für Output artifacts (Ausgabe-Artefakte) einen Namen für das Ausgabeartefakt ein (z. B. **`MyAppBuild`**).
15. Wählen Sie Add action aus.
16. Wählen Sie Save (Speichern) und Save (Speichern) aus, um die Änderungen an der Pipeline zu speichern.
17. Klicken Sie auf Release change.

18. Wenn die Pipeline erfolgreich läuft, können Sie die Build-Ausgabeartefakte abrufen. Wählen Sie, wenn die Pipeline in der CodePipeline-Konsole angezeigt wird, in der Build-Aktion den Tooltip aus. Notieren Sie sich den Wert für das Ausgabe-Artefakt (z. B. MyAppBuild).

    > **Note**
    >
    > Sie können das Build-Ausgabeartefakt auch unter dem Link Build artifacts auf der Build-Detailseite in der CodeBuild-Konsole abrufen. Um zu dieser Seite zu gelangen, gehen Sie auf Anzeigen von Build-Details (Konsole) und dann springen Sie zu Schritt 31 in diesem Verfahren.

19. Öffnen Sie die Amazon-S3-Konsole unter https://console.aws.amazon.com/s3/.
20. Öffnen Sie in der Liste der Buckets auf den von der Pipeline verwendeten Bucket. Der Name des Buckets sollte das Format `codepipeline-`*`region-ID`*`-`*`random-number`* aufweisen. Führen Sie mithilfe der AWS CLI den CodePipeline get-pipeline-Befehl aus, um den Bucket-Namen abzurufen:

    ```
    aws codepipeline get-pipeline --name my-pipeline-name
    ```

    In der Ausgabe enthält das Objekt `pipeline` ein Objekt mit Namen `artifactStore`, das einen Wert `location` mit dem Namen des Buckets enthält.
21. Öffnen Sie den Ordner, der dem Namen Ihrer Pipeline entspricht (abhängig von der Länge des Pipeline-Namens kann der Name des Ordners verkürzt sein), und dann öffnen Sie den Ordner, der dem Wert für Output artifact (Ausgabeartefakt) entspricht, den Sie zuvor in dieser Anleitung notiert haben.
22. Extrahieren Sie den Inhalt der Datei . Wenn in diesem Ordner mehrere Dateien enthalten sind, extrahieren Sie die Inhalte der Datei mit dem neuesten Zeitstempel Last Modified. (Sie müssen die Datei möglicherweise mit der Erweiterung `.zip` versehen, damit Sie vom ZIP-Dienstprogramm Ihres Systems unterstützt wird.) Das Build-Ausgabeartefakt ist in den extrahierten Inhalten der Datei vorhanden.
23. Wenn Sie CodePipeline angewiesen haben den Build-Ausgabeartefakt bereitzustellen, verwenden Sie die Anweisungen des Bereitstellungs-Providers, um zum Build-Ausgabeartefakt über die Bereitstellungsziele zu gelangen.

Hinzufügen einer CodeBuild-Testaktion zu einer Pipeline (CodePipeline-Konsole)

1. Melden Sie sich an der AWS Management Console Konsole an, indem Sie Folgendes verwenden:

   - Ihr AWS-Stammkonto. Dies wird nicht empfohlen. Weitere Informationen finden Sie im Benutzerhandbuch unter Der Root-Benutzer des Kontos.
   - Ein Administratorbenutzer in IhremAWS Konto. Weitere Informationen finden Sie im Benutzerhandbuch unter Erstellen Ihres erstenAWS-Konto Root-Benutzers und Ihrer ersten Root-Gruppe.
   - Ein Benutzer in IhremAWS Konto mit der Erlaubnis, die folgenden Mindestaktionen auszuführen:

   ```
   codepipeline:*
   iam:ListRoles
   iam:PassRole
   s3:CreateBucket
   s3:GetBucketPolicy
   s3:GetObject
   s3:ListAllMyBuckets
   s3:ListBucket
   s3:PutBucketPolicy
   codecommit:ListBranches
   codecommit:ListRepositories
   codedeploy:GetApplication
   codedeploy:GetDeploymentGroup
   codedeploy:ListApplications
   codedeploy:ListDeploymentGroups
   elasticbeanstalk:DescribeApplications
   elasticbeanstalk:DescribeEnvironments
   lambda:GetFunctionConfiguration
   lambda:ListFunctions
   opsworks:DescribeStacks
   opsworks:DescribeApps
   opsworks:DescribeLayers
   ```

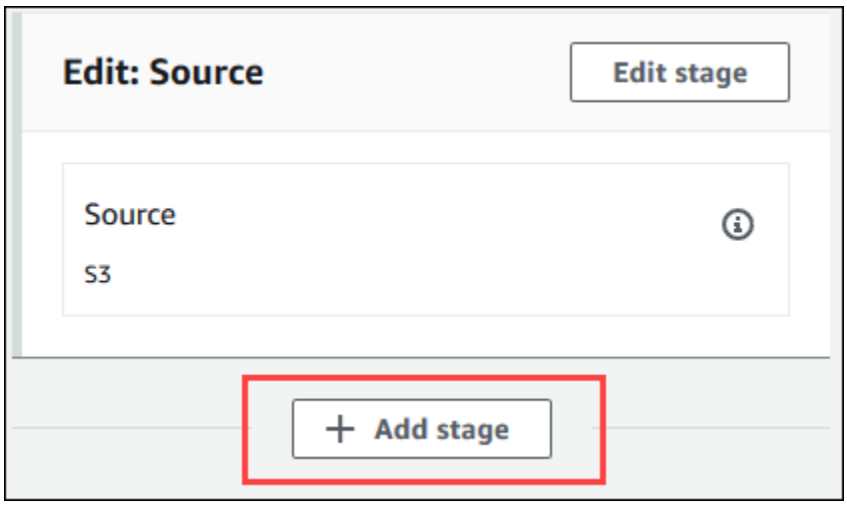
2. Öffnen Sie dieCodePipeline Konsole unter https://console.aws.amazon.com/codesuite/codepipeline/home.

3. Wählen Sie in derAWS Regionsauswahl dieAWS Region aus, in der sich Ihre Pipeline befindet. Dies muss eineAWS Region sein, in derCodeBuild es unterstützt wird. Weitere Informationen finden Sie unter AWS CodeBuild im Allgemeine Amazon Web Services-Referenz.
4. Wählen Sie auf der Seite All Pipelines (Alle Pipelines) den Namen der Pipeline aus.
5. Wählen Sie auf der Detailseite für die Pipeline für die Aktion Source (Quelle) den Tooltip aus. Notieren Sie sich den Wert für das Ausgabe-Artefakt (z. B. MyApp).

> **Note**
>
> Dieses Verfahren verdeutlicht, wie Sie eine Testaktion innerhalb einer Teststufe zwischen den Stufen Source (Quelle) und Beta einfügen. Wenn Sie eine Testaktion an einer anderen Stelle hinzufügen möchten, lassen Sie Ihren Mauszeiger auf der Aktion unmittelbar vor der Stelle ruhen, an der Sie die Build-Aktion hinzufügen möchten und notieren Sie sich den Wert für den Output artifact.

6. Wählen Sie Edit (Bearbeiten) aus.
7. Wählen Sie unmittelbar nach der Stufe Source (Quelle) die Option Add stage (Stufe hinzufügen).

> **Note**
>
> Dieses Verfahren verdeutlicht, wie Sie Ihrer Pipeline eine Teststufe unmittelbar nach der Stufe Source (Quelle) hinzufügen. Um einer bestehenden Stufe eine Testaktion hinzuzufügen, klicken Sie in der bestehenden Stufe auf Edit stage (Stufe bearbeiten). Fahren Sie dann mit Schritt 8 dieses Verfahrens fort. Um die Teststufe an einer anderen Stelle hinzuzufügen, klicken Sie an der gewünschten Stelle auf Add stage (Stufe hinzufügen).



8. Geben Sie unter Stage name (Stufenname) den Namen der Teststufe ein (z. B. **`Test`**). Wenn Sie einen anderen Namen wählen, verwenden Sie diesen im gesamten Verfahren.
9. Wählen Sie in der ausgewählten Stufe Add action (Aktion hinzufügen) aus.

> **Note**
>
> Dieses Verfahren verdeutlicht, wie Sie die Testaktion in einer Teststufe hinzufügen. Um die Testaktion an einer anderen Stelle hinzuzufügen, klicken Sie an der gewünschten Stelle auf Add stage (Stufe hinzufügen). Sie müssen möglicherweise zuerst Edit (Bearbeiten) in der bestehenden Stufe an der Stelle wählen, an der Sie die Testaktion hinzufügen möchten.

10. Geben Sie in Edit action (Aktion bearbeiten) unter Action name (Aktionsname) einen Namen für die Aktion ein (z. B. **`Test`**). Wenn Sie einen anderen Namen wählen, verwenden Sie diesen im gesamten Verfahren.
11. Wählen Sie für Action provider (Aktions-Provider) unter Test CodeBuild aus.
12. Wenn Sie bereits ein Build-Projekt haben, das Sie verwenden möchten, wählen Sie als Projektname den Namen des Build-Projekts aus und fahren Sie mit dem nächsten Schritt in diesem Verfahren fort.

    Wenn Sie ein neuesCodeBuild Build-Projekt erstellen müssen, folgen Sie den Anweisungen unterErstellen Sie ein Build-Projekt (Konsole) und kehren Sie zu diesem Verfahren zurück.

> **Important**
>
> Wenn Sie Webhooks für ein CodeBuild-Projekt aktivieren und das Projekt als Build-Schritt in CodePipeline verwendet wird, werden zwei identische Builds für jeden Commit erstellt. Ein Build wird über Webhooks ausgelöst und einer durch CodePipeline. Da die Fakturierung pro Build erfolgt, werden Ihnen beide Builds in Rechnung gestellt. Wenn Sie also CodePipeline verwenden, sollten Sie Webhooks in CodeBuild deaktivieren. Deaktivieren Sie in der CodeBuild-Konsole das Webhook-Feld. Weitere Informationen finden Sie unter Ändern der Einstellungen eines Build-Projekts (Konsole)

13. Wählen Sie für Input artifacts (Eingabeartefakte) den Wert für das Output artifact (Ausgabeartefakt) aus, den Sie in zuvor in diesem Verfahren notiert haben.
14. (Optional) Wenn Sie möchten, dass Ihre Testaktion ein Ausgabeartefakt erstellt, richten Sie Ihre Build-Spezifikation dementsprechend ein. Geben Sie dann für Output artifact (Ausgabeartefakt) den Wert ein, den Sie dem Ausgabeartefakt zuweisen möchten.
15. Wählen Sie Speichern.
16. Klicken Sie auf Release change.
17. Wenn die Pipeline erfolgreich läuft, können Sie die Testergebnisse abrufen. Klicken Sie in der Test-Stufe der Pipeline auf den Hyperlink CodeBuild, um die dazugehörige Seite des Build-Projekts in der CodeBuild-Konsole zu öffnen.
18. Klicken Sie auf der Seite des Build-Projekts im Bereich Build history (Build-Verlauf) auf den Hyperlink Build run (Build ausführen).
19. Wählen Sie auf der Seite Build-Run unter Build-Logs den Hyperlink Gesamtes Protokoll anzeigen aus, um das Build-Log in derCloudWatch Amazon-Konsole zu öffnen.
20. Scrollen Sie durch das Build-Protokoll, um die Testergebnisse anzuzeigen.

Verwenden von AWS CodeBuild mit Jenkins

Sie können das Jenkins-Plugin verwenden,AWS CodeBuild um es in Ihre Jenkins-Build-Jobs zu integrieren CodeBuild . Dadurch werden Ihre Build-Aufträge mithilfe des Plug-Ins an CodeBuild gesendet, statt diese an Jenkins-Build-Knoten zu senden. Sie müssen somit keine Jenkins-Build-Knoten bereitstellen, konfigurieren und verwalten.

Einrichten von Jenkins

Informationen zur Einrichtung von Jenkins mit demAWS CodeBuild Plugin und zum Herunterladen des Plugin-Quellcodes finden Sie unter https://github.com/awslabs/aws-codebuild-jenkins-plugin.

Installieren des Plugins

Wenn Sie einen Jenkins-Server bereits eingerichtet haben und nur das Plugin für AWS CodeBuild installieren möchten, suchen Sie in Ihrer Jenkins-Instance im Plugin Manager nach **`CodeBuild Plugin for Jenkins`**.

Verwenden des Plug-Ins

So verwenden Sie AWS CodeBuild mit Quellen außerhalb einer VPC.

1. Erstellen Sie ein Projekt in der CodeBuild Konsole. Weitere Informationen finden Sie unter Erstellen Sie ein Build-Projekt (Konsole).
   - Wählen Sie dieAWS -Regionen aus, in denen Sie den Build ausführen möchten.
   - (Optional) Stellen Sie die Amazon VPC-Konfiguration so ein, dass der CodeBuild Build-Container auf Ressourcen in Ihrer VPC zugreifen kann.
   - Notieren Sie den Namen Ihres Projekts. Sie benötigen ihn in Schritt 3.
   - (Optional) Wenn Ihr Quell-Repository nicht nativ unterstützt wird CodeBuild, können Sie Amazon S3 als Eingabequelltyp für Ihr Projekt festlegen.
2. Erstellen Sie in der IAMConsole einen Benutzer, der vom Jenkins-Plugin verwendet werden soll.
   - Wählen Sie während der Erstellung der Anmeldeinformationen für den Benutzer Programmatic Access.
   - Erstellen Sie eine Richtlinie ähnlich der folgenden und fügen Sie die Richtlinie dem Benutzer an.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Resource": ["arn:aws:logs:{{region}}:{{awsAccountId}}:log-group:/aws/
codebuild/{{projectName}}:*"],
      "Action": ["logs:GetLogEvents"]
```

```
    },
    {
      "Effect": "Allow",
      "Resource": ["arn:aws:s3:::{{inputBucket}}"],
      "Action": ["s3:GetBucketVersioning"]
    },
    {
      "Effect": "Allow",
      "Resource": ["arn:aws:s3:::{{inputBucket}}/{{inputObject}}"],
      "Action": ["s3:PutObject"]
    },
    {
      "Effect": "Allow",
      "Resource": ["arn:aws:s3:::{{outputBucket}}/*"],
      "Action": ["s3:GetObject"]
    },
    {
      "Effect": "Allow",
      "Resource": ["arn:aws:codebuild:{{region}}:{{awsAccountId}}:project/
{{projectName}}"],
      "Action": ["codebuild:StartBuild",
        "codebuild:BatchGetBuilds",
        "codebuild:BatchGetProjects"]
    }
  ]
}
```

3. Erstellen Sie in Jenkins ein Freestyle-Projekt.

   - Wählen Sie auf der Seite Configure (Konfigurieren) die Option Add build step (Build-Schritt hinzufügen) und anschließend Run build on CodeBuild (Build auf ACB ausführen).
   - Konfigurieren Sie Ihren Build-Schritt.
     - Geben Sie für Region, Credentials und Project Name Werte an.
     - Wählen Sie Use Project source.
     - Speichern Sie die Konfiguration und führen Sie über Jenkins einen Build aus.
4. Wählen Sie in Source Code Management die Art des Abrufs Ihrer Quelle. Möglicherweise müssen Sie das GitHub Plugin (oder das Jenkins-Plugin für Ihren Quell-Repository-Anbieter) auf Ihrem Jenkins-Server installieren.

   - Wählen Sie auf der Seite Configure (Konfigurieren) die Option Add build step (Build-Schritt hinzufügen) und anschließend Run build on AWS CodeBuild (Build auf ACB ausführen).

- Konfigurieren Sie Ihren Build-Schritt.
  - Geben Sie für Region, Credentials und Project Name Werte an.
  - Wählen Sie Use Jenkins source.
  - Speichern Sie die Konfiguration und führen Sie über Jenkins einen Build aus.

So verwenden Sie das Plugin für AWS CodeBuild mit dem Jenkins-Pipeline-Plugin

- Verwenden Sie auf Ihrer Jenkins-Pipeline-Projektseite den Snippet-Generator, um ein Pipeline-Skript zu generieren, das Ihrer Pipeline CodeBuild als Schritt hinzugefügt wird. Hierdurch sollte ein Skript generiert werden, das dem folgenden Skript ähnlich ist:

```
awsCodeBuild projectName: 'project', credentialsType: 'keys', region: 'us-west-2',
 sourceControlType: 'jenkins'
```

Verwendung von AWS CodeBuild mit Codecov

Codecov ist ein Tool, das den Testumfang Ihres Codes misst. Codecov ermittelt, welche Methoden und Anweisungen in Ihrem Code nicht getestet werden. Verwenden Sie die Ergebnisse, um zu bestimmen, wo Tests geschrieben werden sollen, um die Qualität Ihres Codes zu verbessern. Codecov steht für drei der Quellrepositorys zur Verfügung, die von CodeBuild unterstützt werden: GitHub, GitHub Enterprise Server und Bitbucket. Wenn für Ihr Build-Projekt GitHub Enterprise Server verwendet wird, müssen Sie Codecov Enterprise verwenden.

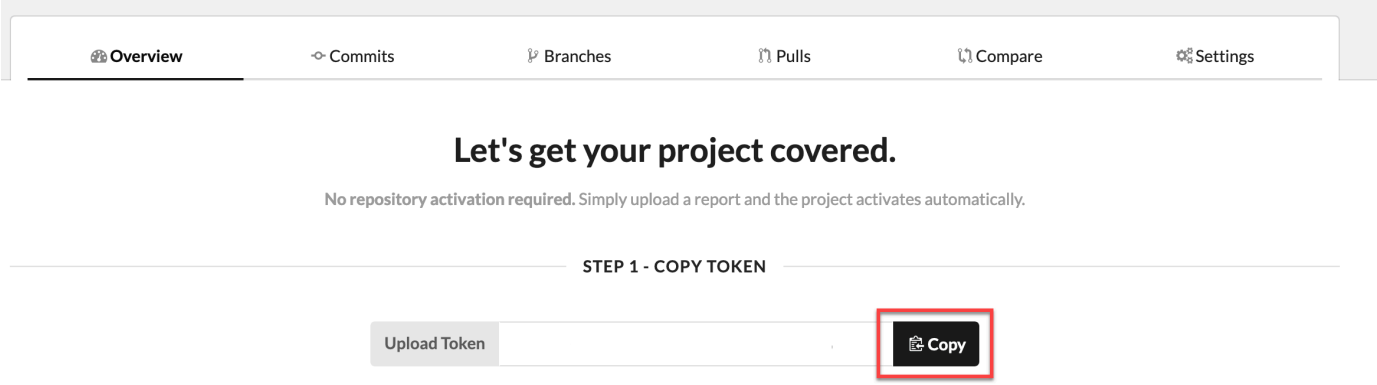
Wenn Sie einen Build eines CoBuild-Projekts ausführen, das in Codecov integriert ist, werden Codecov-Berichte, die den Code in Ihrem Repository analysieren, in Codecov hochgeladen. Die Build-Protokolle enthalten einen Link zu den Berichten. Dieses Beispiel zeigt Ihnen, wie Sie ein Python- und ein Java-Build-Projekt in Codecov integrieren. Eine Liste der von Codecov unterstützten Sprachen finden Sie auf der Codecov-Website unter Codecov Supported Languages.

Integrieren von Codecov in ein Build-Projekt

So integrieren Sie Codecov in Ihr Build-Projekt

1. Gehen Sie zu https://codecov.io/signup und melden Sie sich für ein GitHub- oder Bitbucket-Quell-Repository an. Wenn Sie GitHub Enterprise verwenden, finden Sie auf der Codecov-Website Informationen unter Codecov Enterprise.

2. Fügen Sie in Codecov das Repository hinzu, das mit einbezogen werden soll.
3. Wenn Token-Informationen angezeigt werden, wählen Sie Copy (Kopieren).



4. Fügen Sie das kopierte Token als eine Umgebungsvariable mit dem Namen `CODECOV_TOKEN` zu Ihrem Build-Projekt hinzu. Weitere Informationen finden Sie unter Ändern der Einstellungen eines Build-Projekts (Konsole).
5. Erstellen Sie in Ihrem Repository eine Textdatei mit dem Namen `my_script.sh`. Kopieren Sie Folgendes in die Datei:

```
#/bin/bash
bash <(curl -s https://codecov.io/bash) -t $CODECOV_TOKEN
```

6. Wählen Sie die Registerkarte Python oder Java, je nachdem, was für Ihr Build-Projekt verwendet wird, und befolgen Sie diese Schritten.

   Java

   1. Fügen Sie in Ihrem Repository das folgende JaCoCo-Plugin `pom.xml` hinzu.

```
<build>
  <plugins>
    <plugin>
      <groupId>org.jacoco</groupId>
      <artifactId>jacoco-maven-plugin</artifactId>
      <version>0.8.2</version>
      <executions>
          <execution>
              <goals>
                  <goal>prepare-agent</goal>
              </goals>
          </execution>
          <execution>
```

```
                <id>report</id>
                <phase>test</phase>
                <goals>
                    <goal>report</goal>
                </goals>
            </execution>
        </executions>
    </plugin>
  </plugins>
</build>
```

2. Geben Sie die folgenden Befehle in die Build-Spezifikationsdatei ein. Weitere Informationen finden Sie unter Syntax der Build-Spezifikation.

```
build:
  - mvn test -f pom.xml -fn
postbuild:
  - echo 'Connect to CodeCov'
  - bash my_script.sh
```

Python

Geben Sie die folgenden Befehle in die Build-Spezifikationsdatei ein. Weitere Informationen finden Sie unter Syntax der Build-Spezifikation.

```
build:
  - pip install coverage
  - coverage run -m unittest discover
postbuild:
  - echo 'Connect to CodeCov'
  - bash my_script.sh
```

7. Führen Sie einen Build Ihres Build-Projekts aus. Ein Link zu den für Ihr Projekt generierten Codecov-Berichten wird in Ihren Build-Protokollen angezeigt. Verwenden Sie den Link, um sich die Codecov-Berichte anzeigen zu lassen. Weitere Informationen finden Sie unter Ausführen eines Build in AWS CodeBuild und Protokollierung von AWS CodeBuild-API-Aufrufen mit AWS CloudTrail. Die Codecov-Informationen in den Build-Protokollen sehen wie folgt aus:

```
[Container] 2020/03/09 16:31:04 Running command bash my_script.sh

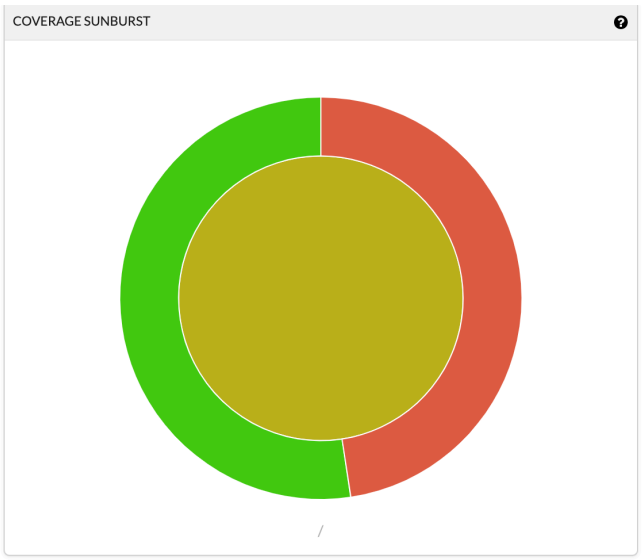

  _____          _
```

```
 / ____|        | |
| |     ___   __| | ___  ___ _____   __
| |    / _ \ / _` |/ _ \/ __/ _ \ \ / /
| |___| (_) | (_| |  __/ (_| (_) \ V /
 \_____\___/ \__,_|\___|\___\___/ \_/
                              Bash-20200303-bc4d7e6

·[0;90m==>·[0m AWS CodeBuild detected.
... The full list of Codecov log entries has been omitted for brevity ...
    .
    ·[0;32m->·[0m View reports at ·[0;36mhttps://codecov.io/github/user/test_py/
commit/commit-id·[0m

[Container] 2020/03/09 16:31:07 Phase complete: POST_BUILD State: SUCCEEDED
```

Die Berichte sehen wie folgt aus:



Files					Coverage
code.py	10	7	0	3	70.00%
tests.py	11	11	0	0	100.00%
Project Totals (2 files)	21	18	0	3	85.71%

Verwenden vonAWS CodeBuildbei serverlosen Anwendungen

DieAWS Serverless Application Model(AWS SAM) ist ein Open-Source-Framework für die Erstellung von serverlosen Anwendungen. Weitere Informationen finden Sie unter AWS Serverless Application Model auf GitHub.

Sie könnenAWS CodeBuildum serverlose Anwendungen zu verpacken und bereitzustellen, dieAWS SAMstandard. Für den Bereitstellungsschritt kann CodeBuild verwendenAWS CloudFormationaus. Zur Automatisierung der Erstellung und Bereitstellung von serverlosen Anwendungen mit CodeBuild undAWS CloudFormationkönnen Sie verwenden.AWS CodePipelineaus.

Weitere Informationen finden Sie unterBereitstellen von serverlosen AnwendungenimAWS Serverless Application ModelEntwicklerhandbuchaus.

Zugehörige Ressourcen

- Informationen zu den ersten Schritten mit AWS CodeBuild finden Sie unter Erste Schritte mit AWS CodeBuild unter Verwendung der Konsole.
- Informationen zur Fehlerbehebung in CodeBuild finden Sie unterFehlerbehebung AWS CodeBuildaus.
- Informationen zu Kontingenten in CodeBuild finden Sie unterKontingente für AWS CodeBuildaus.

Fehlerbehebung AWS CodeBuild

Verwenden Sie die Informationen in diesem Thema, um Fehler zu identifizieren, zu diagnostizieren und zu beheben. Informationen zum Protokollieren und Überwachen von CodeBuild Builds zur Behebung von Problemen finden Sie unter Protokollierung und Überwachung.

Themen

- Fehler: „The bucket you are attempting to access must be addressed using the specified endpoint” beim Ausführen eines Builds
- Fehler: "This build image requires selecting at least one runtime version."
- Fehler: "QUEUED: INSUFFICIENT_SUBNET", wenn ein Build in einer Build-Warteschlange fehlschlägt
- Fehler: „Cache konnte nicht heruntergeladen werden: RequestError: Die Sendeanforderung ist fehlgeschlagen, verursacht durch: x509: System-Roots konnten nicht geladen werden und es wurden keine Roots angegeben“
- Fehler: „Zertifikat konnte nicht von S3 heruntergeladen werden AccessDenied.“
- Fehler: „Unable to locate credentials“
- RequestError Timeout-Fehler beim Ausführen von CodeBuild auf einem Proxy-Server
- Die Bourne-Shell (sh) muss in Build-Images vorhanden sein
- Warnung: „Skipping install of runtimes. runtime version selection is not supported by this build image“ beim Ausführen eines Builds
- Fehler: „ JobWorker Identität konnte nicht überprüft werden“ beim Öffnen der CodeBuild Konsole
- Build konnte nicht gestartet werden
- Zugreifen auf GitHub Metadaten in lokal zwischengespeicherten Builds
- AccessDenied: Der Bucket-Eigentümer für die Berichtsgruppe stimmt nicht mit dem Eigentümer des S3-Buckets überein...

Apache Maven erstellt Referenzartefakte aus dem falschen Repository

Problem: Wenn Sie Maven mit einer von bereitgestellten Java AWS CodeBuild-Build-Umgebung verwenden, ruft Maven Build- und Plugin-Abhängigkeiten aus dem sicheren zentralen Maven-Repository unter https://repo1.maven.org/maven2 ab. Das passiert auch dann, wenn die Datei `pom.xml` des Build-Projekts explizit die Verwendung anderer Verzeichnisse angibt.

Mögliche Ursache: CodeBuild Von bereitgestellte Java-Build-Umgebungen enthalten eine Datei mit dem Namen `settings.xml`, die im `/root/.m2` Verzeichnis der Build-Umgebung vorinstalliert ist. Diese Datei `settings.xml` enthält die folgenden Deklarationen, die Maven anweisen, Build- und Plugin-Abhängigkeiten aus dem sicheren zentralen Maven-Repository unter https://repo1.maven.org/maven2 abzurufen.

```
<settings>
  <activeProfiles>
    <activeProfile>securecentral</activeProfile>
  </activeProfiles>
  <profiles>
    <profile>
      <id>securecentral</id>
      <repositories>
        <repository>
          <id>central</id>
          <url>https://repo1.maven.org/maven2</url>
          <releases>
            <enabled>true</enabled>
          </releases>
        </repository>
      </repositories>
      <pluginRepositories>
        <pluginRepository>
          <id>central</id>
          <url>https://repo1.maven.org/maven2</url>
          <releases>
            <enabled>true</enabled>
          </releases>
        </pluginRepository>
      </pluginRepositories>
    </profile>
  </profiles>
</settings>
```

Empfohlene Lösung: Gehen Sie wie folgt vor:

1. Fügen Sie eine Datei `settings.xml` zu Ihrem Quellcode hinzu.
2. Verwenden Sie in dieser Datei `settings.xml` das oben angegebene Format für `settings.xml` als Richtlinie zur Deklaration der Repositorys, aus denen Maven stattdessen die Build- und Plugin-Abhängigkeiten abrufen soll.
3. Weisen Sie in der -`install`Phase Ihres Build-Projekts CodeBuild an, Ihre `settings.xml` Datei in das `/root/.m2` Verzeichnis der Build-Umgebung zu kopieren. Ein Beispiel ist der folgende Codeausschnitt aus der Datei `buildspec.yml`, der dieses Verhalten veranschaulicht.

   ```
   version 0.2
   ```

```
phases:
  install:
    commands:
      - cp ./settings.xml /root/.m2/settings.xml
```

Build-Befehle werden standardmäßig als Root-Benutzer ausgeführt

Issue: AWS CodeBuild Führt Ihre Build-Befehle als Root-Benutzer aus. Dies passiert, auch wenn das Dockerfile des entsprechenden Build-Image die `USER`-Anweisungen auf einen anderen Benutzer umstellt.

Ursache: Standardmäßig CodeBuild führt alle Build-Befehle als Root-Benutzer aus.

Empfohlene Lösung: Keine.

Builds können fehlschlagen, wenn Dateinamen nicht in den USA liegen. Englische Zeichen

Problem: Wenn Sie einen Build ausführen, der Dateien mit Dateinamen verwendet, die Nicht-USA enthalten. Englische Zeichen (z. B. chinesische Zeichen), der Build schlägt fehl.

Mögliche Ursache: Bei Build-Umgebungen, die von bereitgestellt AWS CodeBuild werden, ist ihr Standardgebietsschema auf festgelegt`POSIX`. -`POSIX`Standorteinstellungen sind weniger kompatibel mit - CodeBuild und -Dateinamen, die Nicht-USA enthalten. Englische Zeichen und können dazu führen, dass zugehörige Builds fehlschlagen.

Empfohlene Lösung: Fügen Sie die folgenden Befehle zum Abschnitt `pre_build` Ihrer Build-Spezifikationsdatei hinzu. Diese Befehle machen die Build-Umgebung zur Verwendung von UTF-8 in US-Englisch für ihre Lokalisierungseinstellungen, das mit - CodeBuild und -Dateinamen, die Nicht-USA enthalten, besser kompatibel ist. Englische Zeichen.

Für Build-Umgebungen basierend auf Ubuntu:

```
pre_build:
  commands:
    - export LC_ALL="en_US.UTF-8"
    - locale-gen en_US en_US.UTF-8
    - dpkg-reconfigure locales
```

Für Build-Umgebungen, die auf Amazon Linux basieren:

```
pre_build:
  commands:
    - export LC_ALL="en_US.utf8"
```

Builds können fehlschlagen, wenn Parameter aus dem Amazon EC2 Parameter Store abgerufen werden

Problem: Wenn ein Build versucht, den Wert eines oder mehrerer Parameter abzurufen, die im Amazon EC2 Parameter Store gespeichert sind, schlägt der Build in der -DOWNLOAD_SOURCEPhase mit dem Fehler fehlParameter does not exist.

Mögliche Ursache: Die Servicerolle, auf die das Build-Projekt angewiesen ist, verfügt nicht über die Berechtigung zum Aufrufen der `ssm:GetParameters` Aktion oder das Build-Projekt verwendet eine Servicerolle, die von generiert wird AWS CodeBuild und den Aufruf der `ssm:GetParameters` Aktion ermöglicht, aber die Parameter haben Namen, die nicht mit beginnen`/CodeBuild/`.

Empfohlene Lösungen:

- Wenn die Servicerolle nicht von generiert wurde CodeBuild, aktualisieren Sie ihre Definition, damit die `ssm:GetParameters` Aktion CodeBuild aufrufen kann. Die folgende Richtlinienanweisung erlaubt es beispielsweise, die Aktion `ssm:GetParameters` auszuführen und Parameter, deren Namen mit `/CodeBuild/` beginnen, abzurufen:

  ```
  {
    "Version": "2012-10-17",
    "Statement": [
      {
        "Action": "ssm:GetParameters",
        "Effect": "Allow",
        "Resource": "arn:aws:ssm:REGION_ID:ACCOUNT_ID:parameter/CodeBuild/*"
      }
    ]
  }
  ```

- Wenn die Servicerolle von generiert wurde CodeBuild, aktualisieren Sie ihre Definition, um den Zugriff auf Parameter im Amazon EC2 Parameter Store CodeBuild mit anderen Namen als denen zu ermöglichen, die mit beginnen`/CodeBuild/`. Die folgende Richtlinienanweisung

erlaubt es beispielsweise, die Aktion `ssm:GetParameters` auszuführen und Parameter mit dem angegebenen Namen abzurufen:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": "ssm:GetParameters",
      "Effect": "Allow",
      "Resource": "arn:aws:ssm:REGION_ID:ACCOUNT_ID:parameter/PARAMETER_NAME"
    }
  ]
}
```

Zugriff auf Verzweigungsfilter in der CodeBuild -Konsole nicht möglich

Problem: Die Verzweigungsfilteroption ist in der Konsole nicht verfügbar, wenn Sie ein - AWS CodeBuild Projekt erstellen oder aktualisieren.

Mögliche Ursache: Die Verzweigungsfilter-Option ist veraltet. Sie wurde durch Webhook-Filtergruppen ersetzt, die eine bessere Kontrolle über die Webhook-Ereignisse bieten, die einen neuen CodeBuild-Build auslösen.

Empfohlene Lösung: Um einen Verzweigungsfilter zu migrieren, den Sie vor der Einführung von Webhook-Filtern erstellt haben, erstellen Sie eine Webhook-Filtergruppe mit einem `HEAD_REF`-Filter mit dem regulären Ausdruck `^refs/heads/branchName$`. Wenn der reguläre Ausdruck Ihres Verzweigungsfilters beispielsweise `^branchName$` lautete, ist der aktualisierte reguläre Ausdruck, den Sie in den `HEAD_REF`-Filter eingeben, `^refs/heads/branchName$`. Weitere Informationen finden Sie unter Bitbucket-Webhook-Ereignisse und GitHub Webhook-Ereignisse filtern (Konsole).

Erfolg oder Misserfolg der Build-Erstellung wird nicht angezeigt

Problem: Erfolg oder Misserfolg einer wiederholten Build-Erstellung wird nicht angezeigt.

Mögliche Ursache: Die Option zum Melden des Build-Status ist nicht aktiviert.

Empfohlene Lösungen: Aktivieren Sie den Status der Berichtserstellung, wenn Sie ein CodeBuild Projekt erstellen oder aktualisieren. Diese Option weist CodeBuild an, den Status zurückzugeben,

wenn Sie eine Build-Erstellung auslösen. Weitere Informationen finden Sie unter reportBuildStatus in der AWS CodeBuild -API-Referenz.

Build-Status wird nicht an den Quellanbieter gemeldet

Problem: Nachdem Sie einem Quellanbieter wie GitHub oder Bitbucket Build-Statusberichte gewährt haben, wird der Build-Status nicht aktualisiert.

Mögliche Ursache: Der dem Quellanbieter zugeordnete Benutzer hat keinen Schreibzugriff auf das Repo.

Empfohlene Lösungen: Um den Build-Status an den Quellanbieter melden zu können, muss der dem Quellanbieter zugeordnete Benutzer Schreibzugriff auf das Repo haben. Wenn der Benutzer keinen Schreibzugriff hat, kann der Build-Status nicht aktualisiert werden. Weitere Informationen finden Sie unter Zugriff auf den Quellanbieter.

Das Basis-Image der Windows Server Core 2019-Plattform kann nicht gefunden und ausgewählt werden

Problem: Sie können das Basis-Image der Windows Server Core 2019-Plattform nicht finden oder auswählen.

Mögliche Ursache: Sie verwenden eine - AWS Region, die dieses Image nicht unterstützt.

Empfohlene Lösungen: Verwenden Sie eine der folgenden AWS Regionen, in denen das Basis-Image der Windows Server Core 2019-Plattform unterstützt wird:

- USA Ost (Nord-Virginia)
- USA Ost (Ohio)
- USA West (Oregon)
- Europa (Irland)

Vorherige Befehle in den Build-Spezifikationsdateien werden von nachfolgenden Befehlen nicht erkannt

Problem: Die Ergebnisse von einem oder mehreren Befehlen in der Build-Spezifikationsdatei werden von nachfolgenden Befehlen in derselben Build-Spezifikationsdatei nicht erkannt. Beispielsweise legt

ein Befehl eine lokale Umgebungsvariable fest, aber ein späterer Befehl ruft den Wert dieser lokalen Umgebungsvariable nicht ab.

Mögliche Ursache: In der Build-Spezifikationsdateiversion 0.1 führt AWS CodeBuild jeden Befehl in einer separaten Instance der Standard-Shell in der Build-Umgebung aus. d. h., dass jeder Befehl unabhängig von allen anderen Befehlen ausgeführt wird. Daher können Sie keinen Einzelbefehl ausführen, der auf dem Status eines vorherigen Befehls basiert.

Empfohlene Lösungen: Wir empfehlen die Verwendung der Build-Spezifikationsversion 0.2, die dieses Problem behebt. Falls Sie doch die Build-Spezifikationsversion 0.1 verwenden müssen, empfehlen wir die Verwendung des Shell-Befehlsverkettungsoperators (z. B. && in Linux), um mehrere Befehle zu einem einzigen Befehl zu kombinieren. Oder schließen Sie ein Shell-Skript in den Quellcode ein, das mehrere Befehle enthält, und rufen Sie dann dieses Shell-Skript über einen Einzelbefehl in der Build-Spezifikationsdatei auf. Weitere Informationen finden Sie unter Shells und Befehle in Build-Umgebungen und Umgebungsvariablen in Build-Umgebungen.

Fehler: „Access denied“ (Zugriff verweigert) beim Versuch, den Cache herunterzuladen

Problem: Beim Versuch, den Cache für ein Build-Projekt herunterzuladen, das den Cache aktiviert hat, wird eine `Access denied`-Fehlermeldung angezeigt.

Mögliche Ursachen:

- Sie haben gerade Caching als Teil Ihres Build-Projekts konfiguriert.
- Der Cache wurde vor Kurzem durch die `InvalidateProjectCache`-API ungültig gemacht.
- Die von verwendete Servicerolle CodeBuild verfügt nicht über - `s3:GetObject` und `-s3:PutObject`Berechtigungen für den S3-Bucket, der den Cache enthält.

Empfohlene Lösung: Bei der ersten Verwendung ist es normal, diese Fehlermeldung direkt nach der Aktualisierung der Cache-Konfiguration zu erhalten. Wenn das Problem weiterhin besteht, sollten Sie prüfen, ob Ihre Service-Rolle über die Berechtigungen `s3:GetObject` und `s3:PutObject` für den S3-Bucket verfügt, in dem sich der Cache befindet. Weitere Informationen finden Sie unter Angeben von S3-Berechtigungen im Amazon S3-Entwicklerhandbuch.

Fehler: „BUILD_CONTAINER_UNABLE_TO_PULL_IMAGE” bei der Verwendung eines benutzerdefinierten Build-Image

Problem: Wenn Sie versuchen, Builds auszuführen, die benutzerdefinierte Build-Images verwenden, erhalten Sie die Fehlermeldung `BUILD_CONTAINER_UNABLE_TO_PULL_IMAGE`.

Mögliche Ursache: Die Gesamtgröße des Build-Images ist größer als der verfügbare Speicherplatz des Datenverarbeitungstyps der Build-Umgebung. Zum Prüfen der Größe des Build-Image nutzen Sie Docker und führen den Befehl *`docker images REPOSITORY:TAG`* aus. Eine Liste der für die jeweiligen Datenverarbeitungstypen verfügbaren Speicherplätze finden Sie unter Berechnungsmodi und Typen der Build-Umgebung.

> Empfohlene Lösung: Verwenden Sie einen größeren Datenverarbeitungstyp mit mehr verfügbarem Speicherplatz oder reduzieren Sie die Größe Ihres benutzerdefinierten Build-Images.

Mögliche Ursache: AWS CodeBuild ist nicht berechtigt, das Build-Image aus Ihrer Amazon Elastic Container Registry (Amazon ECR) abzurufen.

> Empfohlene Lösung: Aktualisieren Sie die Berechtigungen in Ihrem Repository in Amazon ECR, damit Ihr benutzerdefiniertes Build-Image in die Build-Umgebung ziehen CodeBuild kann. Weitere Informationen hierzu finden Sie unter Amazon ECR-Beispiel.

Mögliche Ursache: Das angeforderte Amazon-ECR-Image ist in der AWS Region, die Ihr AWS Konto verwendet, nicht verfügbar.

> Empfohlene Lösung: Verwenden Sie ein Amazon-ECR-Image, das sich in derselben AWS Region befindet wie das, das Ihr AWS Konto verwendet.

Mögliche Ursache: Sie verwenden eine private Registrierung in einer VPC, die keinen öffentlichen Internetzugang hat. CodeBuild kann kein Image von einer privaten IP-Adresse in einer VPC abrufen. Weitere Informationen finden Sie unter Private Registrierung mit AWS Secrets Manager Beispiel für CodeBuild.

> Empfohlene Lösung: Wenn Sie eine private Registrierung in einer VPC verwenden, stellen Sie sicher, dass die VPC über öffentlichen Internetzugang verfügt.

Mögliche Ursache: Wenn die Fehlermeldung „toomanyrequests“ enthält und das Image von Docker Hub abgerufen wird, bedeutet dieser Fehler, dass das Docker-Hub-Pull-Limit erreicht wurde.

> Empfohlene Lösung: Verwenden Sie eine private Docker-Hub-Registrierung oder erhalten Sie Ihr Image von Amazon ECR. Weitere Informationen zur Verwendung einer privaten Registrierung

finden Sie unter Private Registrierung mit AWS Secrets Manager Beispiel für CodeBuild. Weitere Informationen zur Verwendung von Amazon ECR finden Sie unter Amazon ECR-Beispiel für CodeBuild .

Fehler: „Build-Container vor Abschluss des Builds tot gefunden. Der Build-Container ist gestorben, weil er nicht genügend Speicher hatte oder das Docker ErrorCode-Image nicht unterstützt wird.“

Problem: Wenn Sie versuchen, einen Microsoft-Windows- oder Linux-Container in zu verwenden AWS CodeBuild, tritt dieser Fehler während der PROVISIONING-Phase auf.

Mögliche Ursachen:

- Die Container-Betriebssystemversion wird von nicht unterstützt CodeBuild.
- `HTTP_PROXY`, `HTTPS_PROXY` oder beides werden im Container angegeben.

Empfohlene Lösungen:

- Für Microsoft Windows verwenden Sie einen Windows-Container mit einem Container-Betriebssystem der Version microsoft/windowsservercore:10.0.x (z. B. microsoft/windowsservercore:10.0.14393.2125).
- Löschen Sie für Linux die `HTTP_PROXY`- und `HTTPS_PROXY`-Einstellungen in Ihrem Docker Image oder geben Sie die VPC-Konfiguration in Ihrem Build-Projekt an.

Fehler: „Es kann keine Verbindung mit dem Docker-Daemon hergestellt werden“ beim Ausführen eines Builds

Problem: Ihr Build ist nicht erfolgreich und Sie erhalten eine Fehlermeldung ähnlich `Cannot connect to the Docker daemon at unix:/var/run/docker.sock. Is the docker daemon running?` im Build-Protokoll.

Mögliche Ursache: Sie führen den Build nicht im privilegierten Modus aus.

Empfohlene Lösung: Um diesen Fehler zu beheben, müssen Sie den privilegierten Modus aktivieren und Ihre Build-Spezifikation mithilfe der folgenden Anweisungen aktualisieren.

Gehen Sie folgendermaßen vor, um Ihren Build im privilegierten Modus auszuführen:

1. Öffnen Sie die - CodeBuild Konsole unter https://console.aws.amazon.com/codebuild/.
2. Wählen Sie im Navigationsbereich Build projects und dann Ihr Build-Projekt aus.
3. Wählen Sie in Edit (Bearbeiten) Environment (Umgebung) aus.
4. Wählen Sie Additional configuration (Zusätzliche Konfiguration).
5. Wählen Sie unter Privilegiert die Option Dieses Flag aktivieren aus, wenn Sie Docker-Images erstellen möchten oder möchten, dass Ihre Builds erhöhte Berechtigungen erhalten.
6. Wählen Sie Update environment (Umgebung aktualisieren).
7. Wählen Sie Start build (Build starten) aus, um erneut zu versuchen, den Build zu erstellen.

Sie müssen auch den Docker-Daemon in Ihrem Container starten. Die `install` Phase Ihrer Build-Spezifikation könnte in etwa so aussehen.

```
phases:
  install:
    commands:
      - nohup /usr/local/bin/dockerd --host=unix:///var/run/docker.sock --
host=tcp://127.0.0.1:2375 --storage-driver=overlay2 &
      - timeout 15 sh -c "until docker info; do echo .; sleep 1; done"
```

Weitere Informationen über die OverlayFS-Speichertreiber, auf die in der Build-Spezifikationsdatei verwiesen wird, finden Sie auf der Docker-Website unter Verwenden des OverlayFS-Speichertreibers.

> **Note**
>
> Wenn das Basis-Betriebssystem Alpine Linux ist,fügen Sie in `buildspec.yml` das Argument `-t` zu `timeout` hinzu:
>
> ```
> - timeout -t 15 sh -c "until docker info; do echo .; sleep 1; done"
> ```

Weitere Informationen zum Erstellen und Ausführen eines Docker-Images mithilfe von finden Sie AWS CodeBuild unter Docker im Beispiel eines benutzerdefinierten Images für CodeBuild .

Fehler: „CodeBuild ist nicht zur Ausführung autorisiert: sts:AssumeRole“ beim Erstellen oder Aktualisieren eines Build-Projekts

Problem: Wenn Sie versuchen, ein Build-Projekt zu erstellen oder zu aktualisieren, wird der Fehler `Code:InvalidInputException, Message:CodeBuild is not authorized to perform: sts:AssumeRole on arn:aws:iam::`*`account-ID`*`:role/`*`service-role-name`* angezeigt.

Mögliche Ursachen:

- Die AWS Security Token Service (AWS STS) wurde für die AWS Region deaktiviert, in der Sie versuchen, das Build-Projekt zu erstellen oder zu aktualisieren.
- Die dem Build-Projekt zugeordnete AWS CodeBuild Servicerolle ist nicht vorhanden oder verfügt nicht über ausreichende Berechtigungen, um zu vertrauen CodeBuild.

Empfohlene Lösungen:

- Stellen Sie sicher, dass für die AWS Region aktiviert AWS STS ist, in der Sie versuchen, das Build-Projekt zu erstellen oder zu aktualisieren. Weitere Informationen finden Sie unter Aktivieren und Deaktivieren von AWS STS in einer - AWS Region im IAM-Benutzerhandbuch.
- Stellen Sie sicher, dass die CodeBuild Zielservicerolle in Ihrem AWS Konto vorhanden ist. Wenn Sie nicht die Konsole verwenden, achten Sie darauf, dass Sie den Amazon-Ressourcenname (ARN) der Servicerolle beim Erstellen oder Aktualisieren des Build-Projekts richtig geschrieben haben.
- Stellen Sie sicher, dass die CodeBuild Zielservicerolle über ausreichende Berechtigungen verfügt, um zu vertrauen CodeBuild. Weitere Informationen finden Sie in der Vertrauensbeziehung-Richtlinienanweisung unter Erstellen Sie eine CodeBuild Servicerolle.

Fehler: „Fehler beim Aufrufen von GetBucketAcl: Entweder hat sich der Bucket-Eigentümer geändert oder die Servicerolle hat keine Berechtigung mehr zum Aufrufen von s3:GetBucketAcl“

Problem: Ihnen wird beim Ausführen eines Builds eine Fehlermeldung in Bezug auf die Änderung eines S3-Bucket-Eigentümers und von `GetBucketAcl`-Berechtigungen angezeigt.

Mögliche Ursache: Sie haben die `s3:GetBucketLocation` Berechtigungen `s3:GetBucketAcl` und zu Ihrer IAM-Rolle hinzugefügt. Diese Berechtigungen sichern den S3-Bucket Ihres Projekts und stellen Sie sicher, dass nur Sie Zugriff auf diesen haben. Nachdem Sie diese Berechtigungen hinzugefügt haben, hat sich der Besitzer des S3-Buckets geändert.

Empfohlene Lösung: Überprüfen Sie, ob Sie Besitzer des S3-Buckets sind, und fügen Sie dann Ihrer IAM-Rolle erneut Berechtigungen hinzu. Weitere Informationen finden Sie unter Sicherer Zugriff auf S3-Buckets.

Fehler: „Failed to upload artifacts: Invalid arn" beim Ausführen eines Builds

Problem: Beim Ausführen eines Builds schlägt die Build-Phase `UPLOAD_ARTIFACTS` mit der Fehlermeldung `Failed to upload artifacts: Invalid arn` fehl.

Mögliche Ursache: Ihr S3-Ausgabe-Bucket (der Bucket, in dem seine Ausgabe aus dem Build AWS CodeBuild speichert) befindet sich in einer anderen - AWS Region als das CodeBuild Build-Projekt.

Empfohlene Lösung: Aktualisieren Sie die Einstellungen des Build-Projekts so, dass sie auf einen Ausgabe-Bucket verweisen, der sich in derselben AWS Region wie das Build-Projekt befindet.

Fehler: „Git clone failed: Unable to access **`'your-repository-URL'`**: SSL certificate problem: Self signed certificate"

Problem: Beim Versuch, ein Build-Projekt auszuführen, schlägt der Build mit dieser Fehlermeldung fehl.

Mögliche Ursache: Ihr Quell-Repository verfügt über ein selbstsigniertes Zertifikat, Sie haben jedoch nicht angegeben, dass das Zertifikat aus Ihrem S3-Bucket als Teil Ihres Build-Projekts installiert werden soll.

Empfohlene Lösungen:

- Bearbeiten Sie Ihr Projekt. Für Certificate wählen Sie Install certificate from S3. Für Bucket of certificate wählen Sie den S3-Bucket, in dem Ihr SSL-Zertifikat gespeichert ist. Für Object key of certificate (Objektschlüssel des Zertifikats) geben Sie den Namen Ihres S3-Objektschlüssels ein.
- Bearbeiten Sie Ihr Projekt. Wählen Sie SSL unsicher aus, um SSL-Warnungen zu ignorieren, während Sie eine Verbindung zu Ihrem GitHub Enterprise Server-Projekt-Repository herstellen.

> **Note**
>
> Wir empfehlen, dass Sie Insecure SSL nur für Tests verwenden. Es sollte nicht in einer Produktionsumgebung verwendet werden.

Fehler: „The bucket you are attempting to access must be addressed using the specified endpoint” beim Ausführen eines Builds

Problem: Beim Ausführen eines Builds schlägt die Build-Phase `DOWNLOAD_SOURCE` mit der Fehlermeldung `The bucket you are attempting to access must be addressed using the specified endpoint. Please send all future requests to this endpoint` fehl.

Mögliche Ursache: Ihr vorgefertigter Quellcode wird in einem S3-Bucket gespeichert und dieser Bucket befindet sich in einer anderen - AWS Region als das AWS CodeBuild Build-Projekt.

Empfohlene Lösung: Aktualisieren Sie die Einstellungen des Build-Projekts, sodass diese auf einen Bucket verweisen, der Ihren vorgefertigten Quellcode enthält. Stellen Sie sicher, dass sich der Bucket in derselben AWS Region wie das Build-Projekt befindet.

Fehler: "This build image requires selecting at least one runtime version."

Problem: Beim Ausführen eines Builds schlägt die Build-Phase `DOWNLOAD_SOURCE` mit der Fehlermeldung `YAML_FILE_ERROR: This build image requires selecting at least one runtime version` fehl.

Mögliche Ursache: Ihr Build verwendet Version 1.0 oder höher des Amazon Linux 2 (AL2)-Standardabbilds oder Version 2.0 oder höher des Ubuntu-Standardabbilds und in der buildspec-Datei wird keine Laufzeit angegeben.

Empfohlene Lösung: Wenn Sie das `aws/codebuild/standard:2.0` CodeBuild verwaltete Image verwenden, müssen Sie eine Laufzeitversion im `runtime-versions` Abschnitt der buildspec-Datei

angeben. Sie können beispielsweise die folgende Build-Spezifikationsdatei für ein Projekt mit PHP verwenden:

```
version: 0.2

phases:
  install:
    runtime-versions:
        php: 7.3
  build:
    commands:
      - php --version
artifacts:
  files:
    -  README.md
```

> **Note**
>
> Wenn Sie einen `runtime-versions` Abschnitt angeben und ein anderes Image als Ubuntu Standard Image 2.0 oder höher oder das Amazon Linux 2 (AL2) Standard-Image 1.0 oder höher verwenden, gibt der Build die Warnung „“ aus`Skipping install of runtimes. Runtime version selection is not supported by this build image`.

Weitere Informationen finden Sie unter Specify runtime versions in the buildspec file.

Fehler: "QUEUED: INSUFFICIENT_SUBNET", wenn ein Build in einer Build-Warteschlange fehlschlägt

Problem: Ein Build in einer Build-Warteschlange schlägt fehl und es wird eine Fehlermeldung ähnlich wie `QUEUED: INSUFFICIENT_SUBNET` angezeigt.

Mögliche Ursachen: Der für Ihre VPC angegebene IPv4-CIDR-Block verwendet eine reservierte IP-Adresse. Die ersten vier sowie auch die letzte IP-Adresse in jedem Subnetz CIDR-Block stehen nicht zu Ihrer Verfügung und können daher keiner Instance zugewiesen werden. So sind beispielsweise in einem Subnetz mit dem CIDR-Block `10.0.0.0/24` die folgenden fünf IP-Adressen reserviert:

- `10.0.0.0:`: Netzwerkadresse.
- `10.0.0.1`: Reserviert von AWS für den VPC-Router.

- `10.0.0.2`: Reserviert von AWS. Die IP-Adresse des DNS-Servers ist immer die Basis des VPC-Netzwerk-Bereichs plus zwei. Wir reservieren jedoch auch die Basis jedes Subnetzbereichs plus zwei. Für VPCs mit mehreren CIDR-Blöcken befindet sich die IP-Adresse des DNS-Servers im primären CIDR. Weitere Informationen finden Sie unter Amazon DNS-Server im Amazon VPC Benutzerhandbuch.
- `10.0.0.3`: Reserviert von AWS für die zukünftige Verwendung.
- `10.0.0.255`: Broadcast Adresse des Netzwerks. Wir unterstützen keinen Broadcast in eine VPC. Diese Adresse ist reserviert.

Empfohlene Lösungen: Überprüfen Sie, ob Ihre VPC eine reservierte IP-Adresse verwendet. Ersetzen Sie eine reservierte IP-Adresse durch eine IP-Adresse, die nicht reserviert ist. Weitere Informationen finden Sie unter Dimensionierung der VPC und der Subnetze im Amazon VPC Benutzerhandbuch.

Fehler: „Cache konnte nicht heruntergeladen werden: RequestError: Die Sendeanforderung ist fehlgeschlagen, verursacht durch: x509: System-Roots konnten nicht geladen werden und es wurden keine Roots angegeben“

Problem: Beim Versuch, ein Build-Projekt auszuführen, schlägt der Build mit dieser Fehlermeldung fehl.

Mögliche Ursache: Sie haben Caching als Teil Ihres Build-Projekts konfiguriert und verwenden ein älteres Docker-Image mit einem abgelaufenen Stammzertifikat.

Empfohlene Lösung: Aktualisieren Sie das Docker-Image, AWS CodeBuild das in Ihrem Projekt verwendet wird. Weitere Informationen finden Sie unter Docker-Images bereitgestellt von CodeBuild.

Fehler: „Zertifikat konnte nicht von S3 heruntergeladen werden AccessDenied.“

Problem: Beim Versuch, ein Build-Projekt auszuführen, schlägt der Build mit dieser Fehlermeldung fehl.

Mögliche Ursachen:

- Sie haben den falschen S3-Bucket für Ihr Zertifikat gewählt.
- Sie haben den falschen Objektschlüssel für Ihr Zertifikat eingegeben.

Empfohlene Lösungen:

- Bearbeiten Sie Ihr Projekt. Für Bucket of certificate wählen Sie den S3-Bucket, in dem Ihr SSL-Zertifikat gespeichert ist.
- Bearbeiten Sie Ihr Projekt. Für Object key of certificate (Objektschlüssel des Zertifikats) geben Sie den Namen Ihres S3-Objektschlüssels ein.

Fehler: „Unable to locate credentials“

Problem: Wenn Sie versuchen, die auszuführen AWS CLI, ein AWS SDK zu verwenden oder eine andere ähnliche Komponente als Teil eines Builds aufzurufen, erhalten Sie Build-Fehler, die sich direkt auf die AWS CLI, das AWS SDK oder die Komponente beziehen. Sie könnten zum Beispiel eine Build-Fehlermeldung wie `Unable to locate credentials` erhalten.

Mögliche Ursachen:

- Die Version von AWS CLI, AWS SDK oder Komponente in der Build-Umgebung ist nicht mit kompatibel AWS CodeBuild.
- Sie führen einen Docker-Container in einer Build-Umgebung aus, die Docker verwendet, und der Container hat standardmäßig keinen Zugriff auf die AWS Anmeldeinformationen.

Empfohlene Lösungen:

- Stellen Sie sicher, dass Ihre Build-Umgebung über die folgende Version oder höher des AWS CLI, AWS SDK oder der Komponente verfügt.
  - AWS CLI: 1.10.47
  - AWS SDK für C++: 0.2.19
  - AWS SDK für Go: 1.2.5
  - AWS SDK für Java: 1.11.16
  - AWS SDK für JavaScript: 2.4.7
  - AWS SDK für PHP: 3.18.28
  - AWS SDK für Python (Boto3): 1.4.0

- AWS SDK für Ruby: 2.3.22
- Botocore: 1.4.37
- CoreCLR: 3.2.6-beta
- Node.js: 2.4.7

- Wenn Sie einen Docker-Container in einer Build-Umgebung ausführen müssen und der Container AWS Anmeldeinformationen erfordert, müssen Sie die Anmeldeinformationen von der Build-Umgebung an den Container übergeben. Fügen Sie in Ihrer Build-Spezifikationsdatei wie in dem nachfolgenden Beispiel einen Docker-`run`-Befehl ein. In diesem Beispiel werden die verfügbaren S3-Buckets mit dem `aws s3 ls` Befehl aufgeführt. Die `-e` Option durchläuft die Umgebungsvariablen, die Ihr Container für den Zugriff auf AWS Anmeldeinformationen benötigt.

```
docker run -e AWS_DEFAULT_REGION -e AWS_CONTAINER_CREDENTIALS_RELATIVE_URI your-image-tag aws s3 ls
```

- Wenn Sie ein Docker-Image erstellen und der Build AWS Anmeldeinformationen erfordert (z. B. um eine Datei von Amazon S3 herunterzuladen), müssen Sie die Anmeldeinformationen wie folgt aus der Build-Umgebung an den Docker-Build-Prozess übergeben.

  1. Geben Sie in Ihrem Quellcode für das Dockerfile des Docker-Image die folgenden `ARG`-Anweisungen ein.

     ```
     ARG AWS_DEFAULT_REGION
     ARG AWS_CONTAINER_CREDENTIALS_RELATIVE_URI
     ```

  2. Fügen Sie in Ihrer Build-Spezifikationsdatei wie in dem nachfolgenden Beispiel einen Docker-`build`-Befehl ein. Die `--build-arg` Optionen legen die Umgebungsvariablen fest, die für Ihren Docker-Build-Prozess für den Zugriff auf die AWS Anmeldeinformationen erforderlich sind.

     ```
     docker build --build-arg AWS_DEFAULT_REGION=$AWS_DEFAULT_REGION --build-arg
      AWS_CONTAINER_CREDENTIALS_RELATIVE_URI=$AWS_CONTAINER_CREDENTIALS_RELATIVE_URI -
     t your-image-tag .
     ```

RequestError Timeout-Fehler beim Ausführen von CodeBuild auf einem Proxy-Server

Problem: Sie erhalten eine `RequestError`-Fehlermeldung ähnlich einer der folgenden:

- `RequestError: send request failed caused by: Post https://logs.<your-region>.amazonaws.com/: dial tcp 52.46.158.105:443: i/o timeout` aus CloudWatch Logs.
- `Error uploading artifacts: RequestError: send request failed caused by: Put https://your-bucket.s3.your-aws-region.amazonaws.com/*: dial tcp 52.219.96.208:443: connect: connection refused` aus Amazon S3.

Mögliche Ursachen:

- `ssl-bump` ist nicht ordnungsgemäß konfiguriert.
- Die Sicherheitsrichtlinie Ihrer Organisation lässt nicht zu, dass Sie `ssl_bump` verwenden.
- Für Ihre Buildspec-Datei sind keine Proxy-Einstellungen mit einem `proxy` -Element angegeben.

Empfohlene Lösungen:

- Stellen Sie sicher, dass `ssl-bump` korrekt konfiguriert ist. Wenn Sie Squid als Proxy-Server verwenden, beachten Sie die Informationen im Abschnitt Konfigurieren von Squid als expliziter Proxy-Server.
- Gehen Sie wie folgt vor, um private Endpunkte für Amazon S3 und CloudWatch Logs zu verwenden:

  1. Entfernen Sie in der Routing-Tabelle Ihres privaten Subnetzes die von Ihnen hinzugefügte Regel, die für das Internet bestimmten Datenverkehr an Ihren Proxy-Server weiterleitet. Weitere Informationen finden Sie unter Erstellen eines Subnetzes in Ihrer VPC im Amazon-VPC-Benutzerhandbuch.
  2. Erstellen Sie einen privaten Amazon S3-Endpunkt und einen CloudWatch Logs-Endpunkt und verknüpfen Sie sie mit dem privaten Subnetz Ihrer Amazon VPC. Weitere Informationen finden Sie unter VPC-Endpunktservices im Amazon-VPC-Benutzerhandbuch.
  3. Bestätigen Sie, dass Private DNS-Name in Ihrer Amazon VPC aktivieren ausgewählt ist. Weitere Informationen finden Sie unter Erstellung eines Schnittstellenendpunkts im Benutzerhandbuch für Amazon VPC.
- Wenn Sie für einen expliziten Proxy-Server kein `ssl-bump` verwenden, fügen Sie Ihrer Buildspec-Datei mithilfe eines `proxy` -Elements eine Proxy-Konfiguration hinzu. Weitere Informationen finden Sie unter Führen Sie CodeBuild in einem expliziten Proxy-Server aus und Syntax der Build-Spezifikation.

```
version: 0.2
proxy:
  upload-artifacts: yes
  logs: yes
phases:
  build:
    commands:
```

Die Bourne-Shell (sh) muss in Build-Images vorhanden sein

Problem: Sie verwenden ein Build-Image, das nicht von bereitgestellt wird AWS CodeBuild, und Ihre Builds schlagen mit der Nachricht fehl`Build container found dead before completing the build`.

Mögliche Ursache: Die Bourne-Shell (`sh`) ist nicht in Ihrem Build-Image enthalten. CodeBuild benötigt `sh`, um Build-Befehle und Skripts auszuführen.

Empfohlene Lösung: Wenn `sh` in Ihrem Build-Image nicht vorhanden ist, müssen Sie es unbedingt einschließen, bevor Sie weitere Builds starten, die Ihr Image verwenden. (schließt CodeBuild bereits `sh` in die Build-Images ein.)

Warnung: „Skipping install of runtimes. runtime version selection is not supported by this build image“ beim Ausführen eines Builds

Problem: Wenn Sie einen Build ausführen, enthält das Build-Protokoll diesen Warnhinweis.

Mögliche Ursache: Ihr Build verwendet nicht Version 1.0 oder höher des Amazon Linux 2 (AL2)-Standard-Images oder Version 2.0 oder höher des Ubuntu-Standard-Images und eine Laufzeit ist in einem `runtime-versions` Abschnitt in Ihrer buildspec-Datei angegeben.

Empfohlene Lösung: Stellen Sie sicher, dass Ihre buildspec-Datei keinen `runtime-versions`-Abschnitt enthält. Der `runtime-versions` Abschnitt ist nur erforderlich, wenn Sie das Amazon Linux 2 (AL2)-Standard-Image oder höher oder die Ubuntu-Standard-Image-Version 2.0 oder höher verwenden.

Fehler: „ JobWorker Identität konnte nicht überprüft werden“ beim Öffnen der CodeBuild Konsole

Problem: Wenn Sie die CodeBuild Konsole öffnen, wird die Fehlermeldung „ JobWorker Identität kann nicht überprüft werden“ angezeigt.

Mögliche Ursache: Die IAM-Rolle, die für den Konsolenzugriff verwendet wird, hat ein Tag mit `jobId` als Schlüssel. Dieser Tag-Schlüssel ist für reserviert CodeBuild und verursacht diesen Fehler, wenn er vorhanden ist.

Empfohlene Lösung: Ändern Sie alle benutzerdefinierten IAM-Rollen-Tags, die den Schlüssel haben, `jobId` so, dass sie einen anderen Schlüssel haben, z. B. `jobIdentifier`.

Build konnte nicht gestartet werden

Problem: Beim Starten eines Builds erhalten Sie die Fehlermeldung Build konnte nicht gestartet werden.

Mögliche Ursache: Die Anzahl der gleichzeitigen Builds wurde erreicht.

Empfohlene Lösungen: Warten Sie, bis andere Builds abgeschlossen sind, oder erhöhen Sie das Limit für gleichzeitige Builds für das Projekt und starten Sie den Build erneut. Weitere Informationen finden Sie unter Konfiguration des Projekts.

Zugreifen auf GitHub Metadaten in lokal zwischengespeicherten Builds

Problem: In einigen Fällen ist das Git-Verzeichnis in einem zwischengespeicherten Build eine Textdatei und kein Verzeichnis.

Mögliche Ursachen: Wenn das lokale Quell-Caching für einen Build aktiviert ist, CodeBuild erstellt einen Gitlink für das `.git` Verzeichnis . Das bedeutet, dass das `.git` Verzeichnis tatsächlich eine Textdatei ist, die den Pfad zum Verzeichnis enthält.

Empfohlene Lösungen: Verwenden Sie in allen Fällen den folgenden Befehl, um das Git-Metadatenverzeichnis abzurufen. Dieser Befehl funktioniert unabhängig vom Format von `.git`:

```
git rev-parse --git-dir
```

AccessDenied: Der Bucket-Eigentümer für die Berichtsgruppe stimmt nicht mit dem Eigentümer des S3-Buckets überein...

Problem: Beim Hochladen von Testdaten in einen Amazon S3-Bucket CodeBuild kann die Testdaten nicht in den Bucket schreiben.

Mögliche Ursachen:

- Das für den Bucket-Eigentümer der Berichtsgruppe angegebene Konto stimmt nicht mit dem Eigentümer des Amazon S3-Buckets überein.
- Die Servicerolle hat keinen Schreibzugriff auf den Bucket.

Empfohlene Lösungen:

- Ändern Sie den Bucket-Eigentümer der Berichtsgruppe so, dass er mit dem Eigentümer des Amazon S3-Buckets übereinstimmt.
- Ändern Sie die Servicerolle, um Schreibzugriff auf den Amazon S3-Bucket zu gewähren.

Kontingente für AWS CodeBuild

In den folgenden Tabellen sind die aktuellen Kontingente in aufgeführt AWS CodeBuild. Diese Kontingente gelten für jede unterstützte AWS Region und jedes AWS Konto, sofern nicht anders angegeben.

Servicekontingente

Im Folgenden sind die Standardkontingente für den AWS CodeBuild Dienst aufgeführt.

Name	Standard	Anpas	Beschreibung
Zugewiesene Tags pro Projekt	Jede unterstützte Region: 50	Nein	Maximale Anzahl von Tags, die Sie einem Build-Projekt zuordnen können
Build-Projekte	Jede unterstützte Region: 5 000	Ja	Maximale Anzahl Build-Projekte
Build-Timeout in Minuten	Jede unterstützte Region: 2.160	Nein	Maximale Build-Zei tüberschreitung in Minuten
Gleichzeitige Anforderung nach Informationen über Builds	Jede unterstützte Region: 100	Nein	Maximale Anzahl von Builds, über die Sie mithilfe der AWS CLI oder eines AWS SDK gleichzeitig Informationen anfordern können.
Gleichzeitige Anforderungen nach Informationen über Build-Projekte	Jede unterstützte Region: 100	Nein	Maximale Anzahl von Build-Projekten, zu denen Sie mithilfe der AWS CLI oder eines AWS SDK jederzeit Informationen anfordern können.

Name	Standard	Anpas	Beschreibung
Gleichzeitige Ausführung von Builds für die ARM Lambda/10-GB-Umgebung	Jede unterstützte Region: 1	Ja	Maximale Anzahl gleichzeitig ausgeführter Builds für die ARM Lambda/10-GB-Umgebung
Gleichzeitige Ausführung von Builds für die ARM Lambda/1-GB-Umgebung	Jede unterstützte Region: 1	Ja	Maximale Anzahl gleichzeitig ausgeführter Builds für die ARM Lambda/1-GB-Umgebung
Gleichzeitige Ausführung von Builds für die ARM Lambda/2-GB-Umgebung	Jede unterstützte Region: 1	Ja	Maximale Anzahl gleichzeitig ausgeführter Builds für die ARM Lambda/2-GB-Umgebung
Gleichzeitige Ausführung von Builds für die ARM Lambda/4-GB-Umgebung	Jede unterstützte Region: 1	Ja	Maximale Anzahl gleichzeitig ausgeführter Builds für die ARM Lambda/4-GB-Umgebung
Gleichzeitige Ausführung von Builds für die ARM Lambda/8-GB-Umgebung	Jede unterstützte Region: 1	Ja	Maximale Anzahl gleichzeitig ausgeführter Builds für die ARM Lambda/8-GB-Umgebung
Gleichzeitig ausgeführte Builds für eine ARM/Large-Umgebung	Jede unterstützte Region: 1	Ja	Maximale Anzahl gleichzeitig ausgeführter Builds für eine ARM/Large-Umgebung
Gleichzeitig ausgeführte Builds für ARM/kleine Umgebungen	Jede unterstützte Region: 1	Ja	Maximale Anzahl gleichzeitig ausgeführter Builds für eine ARM/Small-Umgebung

Name	Standard	Anpas	Beschreibung
Gleichzeitig ausgeführte Builds für eine große Linux-GPU-Umgebung	Jede unterstützte Region: 0	Ja	Maximale Anzahl gleichzeitig ausgeführter Builds für Linux-GPU-/Large-Umgebungen
Gleichzeitig ausgeführte Builds für eine kleine Linux-GPU-Umgebung	Jede unterstützte Region: 0	Ja	Maximale Anzahl gleichzeitig ausgeführter Builds für Linux-GPU/kleine Umgebungen
Gleichzeitige Ausführung von Builds für die Linux Lambda/10-GB-Umgebung	Jede unterstützte Region: 1	Ja	Maximale Anzahl gleichzeitig ausgeführter Builds für die Linux Lambda/10-GB-Umgebung
Gleichzeitige Ausführung von Builds für die Linux Lambda/1-GB-Umgebung	Jede unterstützte Region: 1	Ja	Maximale Anzahl gleichzeitig ausgeführter Builds für die Linux Lambda/1-GB-Umgebung
Gleichzeitige Ausführung von Builds für die Linux Lambda/2-GB-Umgebung	Jede unterstützte Region: 1	Ja	Maximale Anzahl gleichzeitig ausgeführter Builds für die Linux Lambda/2-GB-Umgebung
Gleichzeitige Ausführung von Builds für die Linux Lambda/4-GB-Umgebung	Jede unterstützte Region: 1	Ja	Maximale Anzahl gleichzeitig ausgeführter Builds für die Linux Lambda/4-GB-Umgebung
Gleichzeitige Ausführung von Builds für die Linux Lambda/8-GB-Umgebung	Jede unterstützte Region: 1	Ja	Maximale Anzahl gleichzeitig ausgeführter Builds für die Linux Lambda/8-GB-Umgebung

Name	Standard	Anpas	Beschreibung
Gleichzeitig ausgeführte Builds für eine Linux/2xLarge-Umgebung	Jede unterstützte Region: 0	Ja	Maximale Anzahl gleichzeitig ausgeführter Builds für eine Linux/2xLarge-Umgebung
Gleichzeitig ausgeführte Builds für eine Linux/Large-Umgebung	Jede unterstützte Region: 1	Ja	Maximale Anzahl gleichzeitig ausgeführter Builds für eine Linux/Large-Umgebung
Gleichzeitig ausgeführte Builds für eine Linux/Medium-Umgebung	Jede unterstützte Region: 1	Ja	Maximale Anzahl gleichzeitig ausgeführter Builds für eine Linux/Medium-Umgebung
Gleichzeitig ausgeführte Builds für Linux/Small-Umgebungen	Jede unterstützte Region: 1	Ja	Maximale Anzahl gleichzeitig ausgeführter Builds für eine Linux/Small-Umgebung
Gleichzeitig ausgeführte Builds für eine Linux/XLarge-Umgebung	Jede unterstützte Region: 1	Ja	Maximale Anzahl gleichzeitig ausgeführter Builds für eine Linux/XLarge-Umgebung
Gleichzeitig ausgeführte Builds für Windows Server 2019/Large-Umgebungen	Jede unterstützte Region: 1	Ja	Maximale Anzahl gleichzeitig ausgeführter Builds für eine Windows Server 2019/Large-Umgebung
Gleichzeitig ausgeführte Builds für die Windows Server 2019/Medium-Umgebung	Jede unterstützte Region: 1	Ja	Maximale Anzahl gleichzeitig ausgeführter Builds für die Windows Server 2019/Medium-Umgebung

Name	Standard	Anpas	Beschreibung
Gleichzeitig ausgeführte Builds für Windows/Large-Umgebungen	Jede unterstützte Region: 1	Ja	Maximale Anzahl gleichzeitig ausgeführter Builds für die Windows/Large-Umgebung
Gleichzeitig ausgeführte Builds für eine Windows/Medium-Umgebung	Jede unterstützte Region: 1	Ja	Maximale Anzahl gleichzeitig ausgeführter Builds für die Windows/Medium-Umgebung
Mindestzeitraum für Build-Timeout in Minuten	Jede unterstützte Region: 5	Nein	Minimale Build-Zeitüberschreitung in Minuten
Sicherheitsgruppen unter VPC-Konfiguration	Jede unterstützte Region: 5	Nein	Für die VPC-Konfiguration verfügbare Sicherheitsgruppen
Subnetze unter VPC-Konfiguration	Jede unterstützte Region: 16	Nein	Für die VPC-Konfiguration verfügbare Subnetze

> **Note**
>
> Interne Metriken bestimmen die Standardkontingente für gleichzeitig ausgeführte Builds.

Die Kontingente für die maximale Anzahl gleichzeitig ausgeführter Builds variieren je nach Compute-Typ. Für einige Plattformen und Datenverarbeitungstypen lautet der Standardwert 20. Um ein höheres Kontingent für gleichzeitige Builds anzufordern oder wenn Sie die Fehlermeldung „Es können nicht mehr als X aktive Builds für das Konto vorhanden sein“ angezeigt wird, verwenden Sie den obigen Link, um die Anfrage zu stellen. Weitere Informationen zur Preisgestaltung finden Sie unter AWS CodeBuild Preise.

Weitere Beschränkungen

Build-Projekte

Ressource	Standard
Zulässige Zeichen in der Build-Projektbeschreibung	Any
Zulässige Zeichen in Build-Projektnamen	Die Buchstaben `A-Z` und `a-z`, die Zahlen `0-9` und die Sonderzeichen `-` und `_`
Länge des Build-Projektnamens	2 bis einschließlich 150 Zeichen
Maximale Länge der Build-Projektbeschreibung	255 Zeichen
Maximale Anzahl an Berichten, die Sie einem Projekt hinzufügen können	5
Anzahl der Minuten, die Sie in einem Build-Projekt für die Build-Zeitbeschränkung für alle zugehörigen Builds angeben können	5 bis 2160 (36 Stunden)

Builds

Ressource	Standard
Maximale Zeit, die der Verlauf eines Builds beibehalten wird	1 Jahr
Anzahl der Minuten, die Sie für die Build-Zeitbeschränkung für einen einzelnen Build angeben können	5 bis 2160 (36 Stunden)

Computerflotten

Ressource	Standard
Gleichzeitige Anzahl von Rechenflotten	10
Gleichzeitig ausgeführte Instanzen für Flotten mit ARM-/kleinen Umgebungen	1
Gleichzeitige Ausführung von Instances für Flotten mit ARM-/großen Umgebungen	1
Gleichzeitige Ausführung von Instances für Flotten in Linux/kleinen Umgebungen	1
Gleichzeitige Ausführung von Instances für Flotten in Linux/Medium-Umgebungen	1
Gleichzeitige Ausführung von Instances für Flotten in Linux/Large-Umgebungen	1
Gleichzeitige Ausführung von Instances für Flotten in Linux/XLarge-Umgebungen	1
Gleichzeitige Ausführung von Instances für Linux/2xLarge-Umgebungsflotten	0
Gleichzeitige Ausführung von Instances für Linux-GPU/Flotten in kleinen Umgebungen	0
Gleichzeitige Ausführung von Instances für Linux-GPU/große Umgebungsflotten	0
Gleichzeitige Ausführung von Instanzen für Windows Server 2019/Flotten in mittleren Umgebungen	1
Gleichzeitig ausgeführte Instanzen für Windows Server 2019/Große Umgebungsflotten	1

Ressource	Standard
Gleichzeitig ausgeführte Instanzen für Windows Server 2022/Flotten in mittleren Umgebungen	1
Gleichzeitig ausgeführte Instanzen für Windows Server 2022/Flotten in großen Umgebungen	1

Berichte

Ressource	Standard
Maximale Dauer, für die ein Testbericht nach der Erstellung verfügbar ist	30 Tage
Maximale Länge einer Testfallnachricht	5.000 Zeichen
Maximale Länge eines Testfallnamens	1.000 Zeichen
Maximale Anzahl von Berichtsgruppen pro AWS Konto	5,000
Maximale Anzahl der Testfälle pro Bericht	500

Tags

Tag-Limits gelten für Stichwörter in CodeBuild Build-Projekten und CodeBuild Berichtsgruppenressourcen.

Ressource	Standard
Ressourcen-Tag-Schlüsselnamen	Jede Kombination aus Unicode-Buchstaben , Zahlen, Leerzeichen und zulässigen UTF-8-Zeichen, die zwischen 1 und 127 Zeichen enthalten. Zulässige Zeichen sind `+ - = . _ : / @`

Ressource	Standard
	Tag-Schlüsselnamen müssen eindeutig sein. Jeder Schlüssel darf nur einen Wert haben. Ein Tag-Schlüsselname darf nicht: • mit `aws:` beginnen • nur aus Leerstellen bestehen • mit einem Leerzeichen enden • Emojis oder eines der folgenden Zeichen enthalten: `? ^ * [ \ ~ ! # $ % & * ( ) > < | " ' ` [ ] { } ;`
Ressourcen-Tag-Werte	Jede Kombination aus Unicode-Buchstaben, Zahlen, Leerzeichen und zulässigen UTF-8-Zeichen, die zwischen 0 und 255 Zeichen enthalten. Zulässige Zeichen sind `+ - = . _ : / @` Ein Schlüssel darf nur einen Wert haben, viele Schlüssel können aber dasselbe Tag aufweisen. Ein Tag-Schlüsselwert darf keine Emojis oder eines der folgenden Zeichen enthalten: `? ^ * [ \ ~ ! # $ % & * ( ) > < | " ' ` [ ] { } ;`

Hinweise von Drittanbietern für AWS CodeBuild für Windows

Wenn Sie CodeBuild für Windows-Builds verwenden, haben Sie die Möglichkeit, einige Pakete und Module von Drittanbietern zu verwenden, damit Ihre erstellte Anwendung auf Microsoft-Windows-Betriebssystemen ausgeführt werden kann und mit einigen Drittanbieterprodukten zusammenarbeiten kann. Die folgende Liste enthält die anwendbaren rechtlichen Bestimmungen von Drittanbietern, die Ihre Nutzung der angegebenen Pakete und Module von Drittanbietern regeln.

Themen

1) Docker-Basis-Image –windowsservercore

(Lizenzbedingungen verfügbar unter: https://hub.docker.com/_/microsoft-windows-servercore)

Lizenz: Durch die Anforderung und Verwendung dieses Container OS Image für Windows Container erklären Sie sich mit den folgenden zusätzlichen Lizenzbedingungen einverstanden:

ZUSÄTZLICHE LIZENZBEDINGUNGEN FÜR MICROSOFT SOFTWARE

CONTAINER OS IMAGE

Microsoft Corporation (oder abhängig von Ihrem Wohnort eines ihrer Tochterunternehmen) (als „wir“ oder “Microsoft“ bezeichnet) lizenziert diese Container OS Image-Ergänzung für Sie

(„Ergänzung“). Sie erhalten die Lizenz, diese Ergänzung in Verbindung mit der zugrunde liegenden Host-Betriebssystem-Software („Host-Software“) ausschließlich zur Unterstützung der Ausführung der Container-Funktion in der Host-Software zu verwenden. Die Host-Software-Lizenzbedingungen gelten für Ihre Nutzung der Ergänzung. Sie dürfen sie nicht verwenden, wenn Sie keine Lizenz für die Host-Software besitzen. Sie dürfen diese Ergänzung mit jeder gültig lizenzierten Kopie der Host-Software verwenden.

ZUSÄTZLICHE LIZENZANFORDERUNGEN UND/ODER NUTZUNGSRECHTE

Ihre Nutzung der Ergänzung, wie im vorhergehenden Absatz beschrieben, kann zur Erstellung oder Änderung eines Container-Images („Container-Image“) führen, das bestimmte Komponenten der Ergänzung enthält. Der Deutlichkeit halber sei darauf hingewiesen, dass ein Container-Image separat von einem Image für eine virtuelle Maschine oder eine virtuelle Appliance ist und sich davon unterscheidet. Gemäß diesen Lizenzbedingungen gewähren wir Ihnen ein eingeschränktes Recht, diese Komponenten der Ergänzung unter den folgenden Bedingungen weiterzugeben:

(i) Sie dürfen die Komponenten der Ergänzung nur so verwenden, wie sie in Ihrem und als Teil Ihres Container-Images verwendet werden,

(ii) Sie dürfen solche Komponenten der Ergänzung in Ihrem Container-Image verwenden, solange Sie über eine signifikante Primärfunktionalität in Ihrem Container-Image verfügen, die sich wesentlich von der Ergänzung unterscheidet; und

(iii) Sie stimmen zu, diese Lizenzbedingungen (oder ähnliche Bedingungen, die von uns oder einem Hoster verlangt werden) in Ihr Container-Image aufzunehmen, um die mögliche Nutzung der Komponenten der Ergänzung durch Ihre Endbenutzer ordnungsgemäß zu lizenzieren.

Wir behalten uns alle anderen Rechte vor, die hier nicht ausdrücklich gewährt werden.

Durch die Verwendung dieser Ergänzung akzeptieren Sie diese Bedingungen. Wenn Sie sie nicht akzeptieren, verwenden Sie diese Ergänzung nicht.

Als Teil der ergänzenden Lizenzbedingungen für dieses Container OS Image für Windows-Container unterliegen Sie auch den zugrundeliegenden Lizenzbedingungen für Windows Server-Host-Software, bereitgestellt unter: https://www.microsoft.com/en-us/useterms.

2) Docker-Image auf Windows-Basis – Choco

(Lizenzbedingungen verfügbar unter: https://github.com/chocolatey/choco/blob/master/LICENSE)

3) Docker-Image auf Windows-Basis – Git – Version 2.16.2

(Lizenzbedingungen finden Sie unter: https://chocolatey.org/packages/git/2.16.2)

Lizenziert unter der GNU General Public License, Version 2, verfügbar unter: https://www.gnu.org/licenses/old-licenses/gpl-2.0.html.

4) Docker-Image auf Windows-Basis microsoft-build-tools – Version 15.0.26320.2

(Lizenzbedingungen finden Sie unter: https://www.visualstudio.com/license-terms/mt171552/)

MICROSOFT VISUAL STUDIO 2015 ERWEITERUNGEN, VISUAL STUDIO SHELLS und C++ REDISTRIBUTABLE

Diese Lizenzbedingungen sind eine Vereinbarung zwischen der Microsoft Corporation (oder abhängig von Ihrem Wohnort einer ihrer Tochtergesellschaften) und Ihnen. Sie gelten für die oben genannte Software. Die Bedingungen gelten auch für alle Microsoft-Dienste oder Updates für die Software, es sei denn, diese enthalten zusätzliche Bedingungen.

WENN SIE DIESE LIZENZBEDINGUNGEN EINHALTEN, HABEN SIE DIE FOLGENDEN RECHTE.

1. INSTALLATIONS- UND NUTZUNGSRECHTE. Sie dürfen beliebig viele Kopien der Software installieren und verwenden.
2. BEDINGUNGEN FÜR SPEZIFISCHE KOMPONENTEN.
   a. Dienstprogramme. Die Software kann einige Elemente in der Utilities List unter https://docs.microsoft.com/en-us/visualstudio/productinfo/2015-redistribution-vs enthalten. Sie können diese Elemente, sofern sie in der Software enthalten sind, kopieren und auf Ihren oder anderen Computern von Drittanbietern installieren, um Ihre Anwendungen und Datenbanken, die Sie mit der Software entwickelt haben, zu debuggen und bereitzustellen. Bitte beachten Sie, dass Dienstprogramme für den temporären Einsatz konzipiert sind, dass Microsoft möglicherweise nicht in der Lage ist, Dienstprogramme getrennt von der restlichen Software zu patchen oder zu aktualisieren, und dass einige Dienstprogramme aufgrund ihrer Natur es anderen ermöglichen können, auf Maschinen zuzugreifen, auf denen sie installiert sind. Daher sollten Sie alle Dienstprogramme löschen, die Sie installiert haben, nachdem Sie das Debugging oder die Bereitstellung Ihrer Anwendungen und Datenbanken abgeschlossen haben. Microsoft ist nicht verantwortlich für die Nutzung von Dienstprogrammen, die Sie auf einem Computer installieren, oder den Zugriff auf diese, durch Dritte.
   b. Microsoft-Plattformen. Die Software kann Komponenten von Microsoft Windows, Microsoft Windows Server, Microsoft SQL Server, Microsoft Exchange, Microsoft Office und Microsoft enthalten SharePoint. Diese Komponenten unterliegen separaten Vereinbarungen und eigenen Produktsupport-Richtlinien, wie sie in den Lizenzbedingungen im Installationsverzeichnis dieser Komponente oder im Ordner „Lizenzen“ der Software beschrieben sind.
   c. Komponenten von Drittanbietern. Die Software kann Komponenten von Drittanbietern mit separaten rechtlichen Hinweisen enthalten oder durch andere Vereinbarungen geregelt werden, wie in der ThirdPartyNotices Datei beschrieben, die die Software mit sich bringt. Auch wenn diese Komponenten durch andere Vereinbarungen geregelt sind, gelten die nachstehenden Haftungsausschlüsse und Schadensbegrenzungen. Die Software kann auch Komponenten enthalten, die unter Open Source-Lizenzen mit Quellcode-Verfügbarkeitsverpflichtungen lizenziert sind. Kopien dieser Lizenzen sind gegebenenfalls in der ThirdPartyNotices Datei enthalten. Sie können diesen Quellcode von uns erhalten, wenn und soweit dies unter den entsprechenden Open-Source-Lizenzen erforderlich ist, indem Sie eine Zahlungsanweisung oder einen Scheck über USD 5,00 an: Quellcode-Compliance-Team, Microsoft Corporation, 1 Microsoft Way, Redmond, WA 98052, senden. Bitte schreiben Sie „Quellcode“ für eine oder mehrere der unten aufgeführten Komponenten in der Belegposition Ihrer Zahlung:
      - Remote Tools for Visual Studio 2015;
      - Standalone Profiler for Visual Studio 2015;

- IntelliTraceCollector für Visual Studio 2015;
- Microsoft VC++ Redistributable 2015;
- Multibyte MFC Library for Visual Studio 2015;
- Microsoft Build Tools 2015;
- Feedback Client;
- Visual Studio 2015 Integrated Shell; oder
- Visual Studio 2015 Isolated Shell.

Wir können auch eine Kopie des Quellcodes zur Verfügung stellen unter http://thirdpartysource.microsoft.com.

3. DATA. Die Software kann Informationen über Sie und Ihre Nutzung der Software erfassen und an Microsoft senden. Microsoft kann diese Informationen verwenden, um Dienstleistungen anzubieten und unsere Produkte und Dienstleistungen zu verbessern. Sie können viele dieser Szenarien ablehnen, aber nicht alle, wie in der Produktdokumentation beschrieben. Es gibt auch einige Funktionen in der Software, die es Ihnen ermöglichen, Daten von Benutzern Ihrer Anwendungen zu erfassen. Wenn Sie diese Funktionen nutzen, um eine Datenerfassung in Ihren Anwendungen zu ermöglichen, müssen Sie die geltenden Gesetze einhalten, einschließlich entsprechender Hinweise für die Benutzer Ihrer Anwendungen. Weitere Informationen zur Datenerfassung und -verwendung finden Sie in der Hilfedokumentation und in der Datenschutzerklärung unter https://privacy.microsoft.com/en-us/privacystatement. Ihre Nutzung der Software gilt als Ihre Zustimmung zu diesen Praktiken.
4. LIZENZUMFANG. Die Software wird lizenziert, nicht verkauft. Diese Vereinbarung gibt Ihnen nur gewisse Rechte zur Nutzung der Software. Microsoft behält sich alle anderen Rechte vor. Sofern das geltende Recht Ihnen trotz dieser Einschränkung keine weiteren Rechte einräumt, dürfen Sie die Software nur so nutzen, wie es in dieser Vereinbarung ausdrücklich erlaubt ist. Dabei müssen Sie alle technischen Einschränkungen der Software beachten, die nur eine bestimmte Nutzung erlauben. Die folgenden Dinge sind verboten:
   - technische Einschränkungen in der Software zu umgehen;
   - die Software zurückzuentwickeln, zu dekompilieren oder zu disassemblieren oder dies zu versuchen, es sei denn und nur in dem Umfang, wie es die Lizenzbedingungen Dritter für die Verwendung bestimmter Open-Source-Komponenten, die in der Software enthalten sein können, erfordern;
   - Hinweise von Microsoft oder seinen Lieferanten in der Software zu entfernen, zu minimieren, zu blockieren oder zu ändern;

- die Software in einer Weise zu nutzen, die gegen das Gesetz verstößt; oder
- die Software zu teilen, zu veröffentlichen, zu vermieten oder zu verpachten oder die Software als eigenständige, gehostete Lösung für andere zur Verfügung zu stellen.

5. EXPORTBESCHRÄNKUNGEN. Sie müssen alle nationalen und internationalen Exportgesetze und -bestimmungen einhalten, die für die Software gelten, einschließlich der Beschränkungen für Bestimmungsorte, Endbenutzer und Endbenutzer. Weitere Informationen zu Exportbeschränkungen finden Sie unter (aka.ms/exporting).
6. SUPPORT-SERVICES. Da dieser Software „ohne Mängelgewähr" ist, stellen wir möglicherweise keine Support-Services dafür bereit.
7. GESAMTE VEREINBARUNG. Diese Vereinbarung und die Bedingungen für Ergänzungen, Aktualisierungen, internetbasierte Dienste und Supportleistungen, die Sie nutzen, sind die gesamte Vereinbarung für die Software und Supportleistungen.
8. ANWENDBARES RECHT. Wenn Sie die Software in den Vereinigten Staaten erworben haben, gilt das Washingtoner Recht für die Auslegung dieser Vereinbarung und für Ansprüche wegen Verletzung dieser Vereinbarung, und die Gesetze des Staates, in dem Sie leben, gelten für alle anderen Ansprüche. Wenn Sie die Software in einem anderen Land erworben haben, gelten dessen Gesetze.
9. VERBRAUCHERRECHTE; REGIONALE ABWEICHUNGEN. Diese Vereinbarung beschreibt bestimmte gesetzliche Rechte. Sie haben möglicherweise andere Rechte, einschließlich Verbraucherrechte, nach den Gesetzen Ihres Staates oder Landes. Abgesehen von Ihrer Beziehung zu Microsoft haben Sie möglicherweise auch Rechte in Bezug auf die Partei, von der Sie die Software erworben haben. Diese Vereinbarung ändert diese anderen Rechte nicht, wenn die Gesetze Ihres Staates oder Landes dies nicht zulassen. Wenn Sie die Software beispielsweise in einer der unten aufgeführten Regionen erworben haben oder zwingendes Landesrecht gilt, gelten für Sie die folgenden Bestimmungen:
   a. Australien Sie haben gesetzliche Garantien nach dem australischen Verbrauchergesetz, und nichts in dieser Vereinbarung ist darauf ausgelegt, diese Rechte zu beeinträchtigen.
   b. Kanada. Wenn Sie diese Software in Kanada erworben haben, können Sie den Erhalt von Updates beenden, indem Sie die automatische Update-Funktion deaktivieren, Ihr Gerät vom Internet trennen (wenn Sie sich jedoch wieder mit dem Internet verbinden, wird die Software die Überprüfung und Installation von Updates fortsetzen) oder die Software deinstallieren. In der Produktdokumentation, falls vorhanden, kann auch angegeben werden, wie Sie Updates für Ihr spezielles Gerät oder Ihre Software deaktivieren können.
   c. Deutschland und Österreich.

i. Garantie. Die ordnungsgemäß lizenzierte Software funktioniert im Wesentlichen wie in allen Unterlagen von Microsoft beschrieben, die der Software beiliegen. Microsoft gibt jedoch keine vertragliche Garantie in Bezug auf die lizenzierte Software.

ii. Haftungsbeschränkung. Bei Vorsatz, grober Fahrlässigkeit, Ansprüchen nach dem Produkthaftungsgesetz sowie bei Verletzung des Lebens, des Körpers oder der Gesundheit haftet Microsoft nach den gesetzlichen Bestimmungen; vorbehaltlich der vorstehenden Ziffer (ii) haftet Microsoft für leichte Fahrlässigkeit nur, wenn Microsoft solche wesentlichen Vertragspflichten verletzt, deren Erfüllung die ordnungsgemäße Durchführung des Vertrages überhaupt erst ermöglicht, deren Verletzung den Vertragszweck gefährdet und auf deren Einhaltung eine Partei regelmäßig vertrauen darf (sogenannte „Kardinalpflichten“). In anderen Fällen leichter Fahrlässigkeit haftet Microsoft nicht für leichte Fahrlässigkeit.

10 AUSSCHLÜSSELUNG DER FEHLERUNG. DIE SOFTWARE WIRD „OHNE MÄNGELGEWÄHR“ LIZENZIERT. SIE TRAGEN DAS RISIKO IHRER NUTZUNG. MICROSOFT ERTEILT KEINE AUSDRÜCKLICHEN GARANTIEN, GEWÄHRLEISTUNGEN ODER ZUSAGEN. MICROSOFT SCHLIEßT IM RAHMEN IHRER LOKALEN GESETZE DIE IMPLIZITEN GARANTIEN DER MARKTGÄNGIGKEIT, DER EIGNUNG FÜR EINEN BESTIMMTEN ZWECK UND DER NICHTVERLETZUNG VON VERSTÖßEN AUS.

11. EINSCHRÄNKUNG UND AUSSCHLUSS VON ERSTATTUNGEN. VON MICROSOFT UND WIEDERHERSTELLUNG SIE DIE LIEFERANTEN ONLY DIREKTER SCHÄDEN BIS ZU US-5,00 USD. SIE KÖNNEN KEINE ANDEREN BEHANDLUNGEN WIEDERHERSTELLEN, EINSCHLIEßLICH DER ARZT, DER VERLORENEN, DER BESONDEREN, INDIREKTEN ODER DER KER. Diese Einschränkung gilt für (a) alles, was mit der Software, Dienstleistungen, Inhalten (einschließlich Code) auf Internetseiten Dritter oder Anwendungen Dritter zusammenhängt; und (b) Ansprüche wegen Vertragsverletzung, Verletzung von Gewährleistung, Garantie oder Beschaffenheit, verschuldensunabhängiger Haftung, Fahrlässigkeit oder sonstiger unerlaubter Handlung, soweit dies nach geltendem Recht zulässig ist.

Sie gilt auch dann, wenn Microsoft von der Möglichkeit des Schadens wusste oder hätte wissen müssen. Die obige Einschränkung oder der Ausschluss gelten möglicherweise nicht für Sie, da Ihr Land den Ausschluss oder die Beschränkung von Neben-, Folge- oder anderen Schäden nicht zulässt.

EULA-ID: VS2015_Update3_ShellsRedist_<ENU>

5) Docker-Image auf Windows-Basis –nuget.commandline – Version 4.5.1

(Lizenzbedingungen verfügbar unter: https://github.com/NuGet/Home/blob/dev/LICENSE.txt)

Copyright (c) .NET Foundation. Alle Rechte vorbehalten.

Lizenziert unter der Apache-Lizenz, Version 2.0 (die „Lizenz“); Sie dürfen diese Dateien nur in Übereinstimmung mit der Lizenz verwenden. Sie finden eine Kopie der Lizenz unter

http://www.apache.org/licenses/LICENSE-2.0

Sofern nicht gesetzlich vorgeschrieben oder schriftlich vereinbart, wird die unter der Lizenz vertriebene Software auf einer „OHNE MÄNGELGEWÄHR“-BASIS, OHNE GARANTIEN ODER BEDINGUNGEN JEGLICHER ART, weder ausdrücklich noch stillschweigend, vertrieben. Informationen zu den landesspezifischen Berechtigungen und Einschränkungen finden Sie in der Lizenz.

7) Windows-basiertes Docker-Image –netfx-4.6.2-devpack

ZUSÄTZLICHE LIZENZBEDINGUNGEN FÜR MICROSOFT SOFTWARE

.NET FRAMEWORK UND BEGLEITENDE LANGUAGE PACKS FÜR DAS BETRIEBSSYSTEM MICROSOFT WINDOWS

Die Microsoft Corporation (oder abhängig von Ihrem Wohnort: eine ihrer Tochtergesellschaften) lizenziert diese Ergänzung an Sie. Wenn Sie eine Lizenz zur Verwendung von Microsoft Windows-Betriebssystem-Software (die „Software“) besitzen, können Sie diese Ergänzung verwenden. Sie dürfen sie nicht verwenden, wenn Sie keine Lizenz für die Software besitzen. Sie dürfen diese Ergänzung mit jeder gültig lizenzierten Kopie der Software verwenden.

Die folgenden Lizenzbedingungen beschreiben zusätzliche Nutzungsbedingungen für diese Ergänzung. Diese Bedingungen und die Lizenzbedingungen für die Software gelten für Ihre Nutzung der Ergänzung. Wenn ein Konflikt besteht, gelten diese zusätzlichen Lizenzbedingungen.

DURCH DIE VERWENDUNG DIESER ERGÄNZUNG AKZEPTIEREN SIE DIESE BEDINGUNGEN. WENN SIE SIE NICHT AKZEPTIEREN, VERWENDEN SIE DIESE ERGÄNZUNG NICHT.

Wenn Sie diese Lizenzbedingungen einhalten, haben Sie die folgenden Rechte.

1. VERteilBARER CODE. Die Ergänzung besteht aus verteilbarem Code. „Weitergabefähiger Code“ ist Code, den Sie in von Ihnen entwickelten Programmen verbreiten dürfen, wenn Sie die folgenden Bedingungen einhalten.
   a. Recht zur Nutzung und Weitergabe.
      - Sie dürfen die Objektcode-Form der Ergänzung kopieren und verteilen.
      - Weitergabe durch Dritte. Sie können Händlern Ihrer Programme erlauben, den weitergabefähigen Code als Teil dieser Programme zu kopieren und zu verteilen.
   b. Verteilungsanforderungen. Für jeden verteilbaren Code, den Sie verteilen, müssen Sie
      - Sie müssen diesem in Ihren Programmen wichtige Primärfunktionen hinzufügen;
      - für jeden weitergabefähigen Code mit der Dateinamenerweiterung .lib, geben Sie nur die Ergebnisse der Ausführung dieses weitergabefähigen Codes über einen Linker mit Ihrem Programm weiter;
      - geben Sie weitergabefähigen Code, der in einem Setup-Programm enthalten ist, nur als Teil dieses Setup-Programms ohne Änderungen weiter;
      - verlangen Sie von Händlern und externen Endverbrauchern, dass sie sich mit Bedingungen einverstanden erklären, die ihn mindestens genauso gut schützen wie diese Vereinbarung;
      - zeigen Ihren gültigen Copyright-Vermerk auf Ihren Programmen an; und
      - halten Sie Microsoft von jeglichen Ansprüchen frei, einschließlich Anwaltskosten, die mit dem Vertrieb oder der Nutzung Ihrer Programme zusammenhängen, verteidigen Sie es und halten Sie es schadlos.
   c. Verteilungsbeschränkungen. Möglicherweise nicht
      - Urheberrechts-, Marken- oder Patenthinweise im weitergabefähigen Code zu ändern;
      - Marken von Microsoft in den Namen Ihrer Programme oder in einer Weise zu verwenden, die darauf hindeutet, dass Ihre Programme von Microsoft stammen oder von Microsoft unterstützt werden;
      - weitergabefähigen Code weiterzugeben, der auf einer anderen Plattform als der Windows-Plattform ausgeführt wird;
      - weitergabefähigen Code in bösartige, irreführende oder rechtswidrige Programme einzubinden; oder

- den Quellcode des weitergabefähigen Codes so zu verändern oder weiterzugeben, dass irgendein Teil davon unter den Bestimmungen einer Ausschlusslizenz unterworfen ist. Eine Ausschlusslizenz ist eine Lizenz, die als Bedingung für die Nutzung, Änderung oder Weitergabe fordert,
  - dass der Code im Quellformat offen gelegt oder weitergegeben wird; oder
  - andere haben das Recht, ihn zu modifizieren.

2. SUPPORT-SERVICES FÜR DIE ERGÄNZUNG. Microsoft bietet Support-Services für diese Software an, wie unter www.support.microsoft.com/common/international.aspx beschrieben.

8) Windows-basiertes Docker-Image – Visualfsharptools, v 4.0

(Lizenzbedingungen verfügbar unter: https://github.com/dotnet/fsharp/blob/main/License.txt)

Copyright (c) Microsoft Corporation. Alle Rechte vorbehalten.

Lizenziert unter der Apache-Lizenz, Version 2.0 (die „Lizenz“); Sie dürfen diese Dateien nur in Übereinstimmung mit der Lizenz verwenden. Sie finden eine Kopie der Lizenz unter

http://www.apache.org/licenses/LICENSE-2.0

Sofern nicht gesetzlich vorgeschrieben oder schriftlich vereinbart, wird die unter der Lizenz vertriebene Software auf einer „OHNE MÄNGELGEWÄHR“-BASIS, OHNE GARANTIEN ODER BEDINGUNGEN JEGLICHER ART, weder ausdrücklich noch stillschweigend, vertrieben. Informationen zu den landesspezifischen Berechtigungen und Einschränkungen finden Sie in der Lizenz.

9) Docker-Image auf Windows-Basis – netfx-pcl-reference-assemblies-4.6

LIZENZBEDINGUNGEN FÜR MICROSOFT SOFTWARE

MICROSOFT .NET PORTABLE CLASS LIBRARY REFERENCE ASSEMBLIES – 4.6

Diese Lizenzbedingungen sind eine Vereinbarung zwischen der Microsoft Corporation (oder abhängig von Ihrem Wohnort einer ihrer Tochtergesellschaften) und Ihnen. Bitte lesen. Sie gelten für die oben genannte Software. Die Bedingungen gelten auch für alle Microsoft

- Aktualisierungen,
- Ergänzungen,
- Internet-basierte Services und
- Support-Services

für diese Software, es sei denn, andere Bedingungen begleiten dieser Elemente. Wenn dies der Fall ist, sind diese Bedingungen anzuwenden.

DURCH DIE VERWENDUNG DER SOFTWARE AKZEPTIEREN SIE DIESE BEDINGUNGEN. WENN SIE SIE NICHT AKZEPTIEREN, VERWENDEN SIE DIE SOFTWARE NICHT.

WENN SIE DIESE LIZENZBEDINGUNGEN EINHALTEN, HABEN SIE DIE FOLGENDEN UNBEFRISTETEN RECHTE.

1. INSTALLATIONS- UND NUTZUNGSRECHTE. Sie dürfen beliebig viele Kopien der Software installieren und verwenden, um Ihre Programme zu entwerfen, zu entwickeln und zu testen.
2. ZUSÄTZLICHE LIZENZANFORDERUNGEN UND/ODER NUTZUNGSRECHTE.
   a. Weitergabefähiger Code. Sie dürfen die Software in von Ihnen entwickelten Entwickler-Tool-Programmen weitergeben, um Kunden Ihrer Programme zu ermöglichen, portable Bibliotheken zur Verwendung mit jedem Gerät oder Betriebssystem zu entwickeln, wenn Sie die nachstehenden Bedingungen einhalten.
      i. Recht zur Nutzung und Weitergabe. Die Software ist „Verteilbarer Code“.
         - Weitergabefähiger Code. Sie dürfen die Objektcode-Form der Software kopieren und verteilen.
         - Weitergabe durch Dritte. Sie können Händlern Ihrer Programme erlauben, den weitergabefähigen Code als Teil dieser Programme zu kopieren und zu verteilen.
      ii. Verteilungsanforderungen. Für jeden verteilbaren Code, den Sie verteilen, müssen Sie
         - Sie müssen diesem in Ihren Programmen wichtige Primärfunktionen hinzufügen;
         - verlangen Sie von Händlern und Ihren Kunden, dass sie sich mit Bedingungen einverstanden erklären, die ihn mindestens genauso gut schützen wie diese Vereinbarung;
         - zeigen Ihren gültigen Copyright-Vermerk auf Ihren Programmen an; und

- halten Sie Microsoft von jeglichen Ansprüchen frei, einschließlich Anwaltskosten, die mit dem Vertrieb oder der Nutzung Ihrer Programme zusammenhängen, verteidigen Sie es und halten Sie es schadlos.

iii. Verteilungsbeschränkungen. Möglicherweise nicht

- Urheberrechts-, Marken- oder Patenthinweise im weitergabefähigen Code zu ändern;
- Marken von Microsoft in den Namen Ihrer Programme oder in einer Weise zu verwenden, die darauf hindeutet, dass Ihre Programme von Microsoft stammen oder von Microsoft unterstützt werden;
- weitergabefähigen Code in bösartige, irreführende oder rechtswidrige Programme einzubinden; oder
- den weitergabefähigen Code so zu verändern oder weiterzugeben, dass irgendein Teil davon unter den Bestimmungen einer Ausschlusslizenz unterworfen ist. Eine Ausschlusslizenz ist eine Lizenz, die als Bedingung für die Nutzung, Änderung oder Weitergabe fordert,
  - dass der Code im Quellformat offen gelegt oder weitergegeben wird; oder
  - andere haben das Recht, ihn zu modifizieren.

3. LIZENZUMFANG. Die Software wird lizenziert, nicht verkauft. Diese Vereinbarung gibt Ihnen nur gewisse Rechte zur Nutzung der Software. Microsoft behält sich alle anderen Rechte vor. Sofern das geltende Recht Ihnen trotz dieser Einschränkung keine weiteren Rechte einräumt, dürfen Sie die Software nur so nutzen, wie es in dieser Vereinbarung ausdrücklich erlaubt ist. Dabei müssen Sie alle technischen Einschränkungen der Software beachten, die nur eine bestimmte Nutzung erlauben. Die folgenden Dinge sind verboten:
   - technische Einschränkungen in der Software zu umgehen;
   - die Software zurückzuentwickeln, zu dekompilieren oder zu disassemblieren, es sei denn, das geltende Recht erlaubt dies trotz dieser Einschränkung ausdrücklich;
   - die Software zu veröffentlichen, sodass andere sie kopieren können; oder
   - die Software vermieten, verpachten oder verleihen.
4. FEEDBACK. Sie können Feedback über die Software abgeben. Wenn Sie Microsoft Feedback über die Software geben, erteilen Sie Microsoft kostenlos das Recht, Ihr Feedback in jedweder Weise und für jeden Zweck zu nutzen, weiterzugeben und zu vermarkten. Sie erteilen auch Dritten unentgeltlich alle Patentrechte, die für ihre Produkte, Technologien und Dienstleistungen erforderlich sind, um bestimmte Teile einer Microsoft-Software oder eines Microsoft-Dienstes, die das Feedback enthalten, zu nutzen oder mit diesen zu verbinden. Sie geben kein Feedback, das einer Lizenz unterliegt, die von Microsoft verlangt, seine Software oder Dokumentation an

Dritte zu lizenzieren, wenn wir Ihr Feedback darin aufnehmen. Diese Rechte überdauern diese Vereinbarung.

5. ÜBERTRAGUNG AN EINEN DRITTEN. Der erste Nutzer der Software darf diese und diese Vereinbarung direkt an einen Dritten übertragen. Vor der Übertragung muss diese Partei zustimmen, dass diese Vereinbarung für die Übertragung und Nutzung der Software gilt. Der erste Benutzer muss die Software deinstallieren, bevor er sie getrennt vom Gerät übertragen darf. Der erste Benutzer darf keine Kopien zurückbehalten.
6. EXPORTBESCHRÄNKUNGEN. Die Software unterliegt den Exportgesetzen und -regelungen der USA. Sie müssen alle inländischen und internationalen Exportgesetze und -vorschriften einhalten, die für die Software gelten. Zu diesen Gesetzen gehören Einschränkungen im Hinblick auf Bestimmungsorte, Endbenutzer und Endnutzung. Weitere Informationen finden Sie unter www.microsoft.com/exporting.
7. SUPPORT-SERVICES. Da dieser Software „ohne Mängelgewähr" ist, stellen wir möglicherweise keine Support-Services dafür bereit.
8. GESAMTE VEREINBARUNG. Diese Vereinbarung und die Bedingungen für Ergänzungen, Aktualisierungen, internetbasierte Dienste und Supportleistungen, die Sie nutzen, sind die gesamte Vereinbarung für die Software und Supportleistungen, die wir bieten.
9. ANWENDBARES RECHT.
   a. Vereinigte Staaten. Wenn Sie die Software in den Vereinigten Staaten erworben haben, regelt das Recht des Bundesstaates Washington die Auslegung dieser Vereinbarung und gilt für Ansprüche wegen Verletzung dieser Vereinbarung, ungeachtet von Konflikten der Gesetzesgrundlagen. Die Gesetze des Staates, in dem Sie leben, regeln alle anderen Ansprüche, einschließlich der Ansprüche aus staatlichen Verbraucherschutzgesetzen, Gesetzen des unlauteren Wettbewerbs und aus unerlaubter Handlung.
   b. Außerhalb der Vereinigten Staaten. Wenn Sie die Software in einem anderen Land erworben haben, gelten dessen Gesetze.
10. ANWENDUNG VON GESETZEN. Diese Vereinbarung beschreibt bestimmte gesetzliche Rechte. Möglicherweise haben Sie anderen Rechte gemäß den Gesetzen Ihres Landes. Sie haben möglicherweise auch Rechte gegenüber der Partei, von der Sie die Software erworben haben. Diese Vereinbarung ändert Ihre Rechte unter den Gesetzen Ihres Landes nicht, wenn die Gesetze Ihres Staates oder Landes dies nicht zulassen.
11. AUSSCHLÜSSELUNG DER FEHLERUNG. DIE SOFTWARE WIRD „OHNE MÄNGELGEWÄHR" LIZENZIERT. SIE TRAGEN DAS RISIKO IHRER NUTZUNG. MICROSOFT ERTEILT KEINE AUSDRÜCKLICHEN GARANTIEN, GEWÄHRLEISTUNGEN ODER ZUSAGEN. SIE HABEN MÖGLICHERWEISE ZUSÄTZLICHE VERBRAUCHERRECHTE ODER GESETZLICHE

GARANTIEN NACH IHREN LOKALEN GESETZEN, DIE DURCH DIESE VEREINBARUNG NICHT GEÄNDERT WERDEN KÖNNEN. SOFERN DIES NACH IHREN LOKALEN GESETZEN ZULÄSSIG IST, SCHLIEßT MICROSOFT DIE IMPLIZITEN GARANTIEN DER MARKTGÄNGIGKEIT, DER EIGNUNG FÜR EINEN BESTIMMTEN ZWECK UND DER NICHTVERLETZUNG VON VERSTÖßEN AUS.

FÜR AUSTRALIEN - SIE HABEN GESETZLICHE GARANTIEN NACH DEM AUSTRALISCHEN VERBRAUCHERRECHT UND NICHTS IN DIESEN BEDINGUNGEN IST DARAUF AUSGELEGT, DIESE RECHTE ZU BEEINTRÄCHTIGEN.

12 EINSCHRÄNKUNG UND AUSSCHLUSS VON MITTELN UND MITTELN. VON MICROSOFT UND WIEDERHERSTELLUNG SIE DIE LIEFERANTEN ONLY DIREKTER SCHÄDEN BIS ZU US-5,00 USD. SIE KÖNNEN KEINE ANDEREN BEHANDLUNGEN WIEDERHERSTELLEN, EINSCHLIEßLICH DER ARZT, DER VERLORENEN, DER BESONDEREN, INDIREKTEN ODER DER ARZT.

Diese Einschränkung gilt für

- alles im Zusammenhang mit der Software, Services, Inhalten (einschließlich Code) auf Internetseiten von Dritten oder Programmen von Dritten; und
- Ansprüche aus Vertragsbruch, Verstoß gegen Garantie, Gewährleistung oder Bedingungen, Gefährdungshaftung, Fahrlässigkeit oder andere deliktische Handlungen, soweit dies gemäß geltendem Recht gestattet ist.

Sie gilt auch dann, wenn Microsoft von der Möglichkeit des Schadens wusste oder hätte wissen müssen. Die obige Einschränkung oder der Ausschluss gelten möglicherweise nicht für Sie, da Ihr Land den Ausschluss oder die Beschränkung von Neben-, Folge- oder anderen Schäden nicht zulässt.

10) Windows-basiertes Docker-Image – Visualcppbuildtools v 14.0.25420.1

(Lizenzbedingungen finden Sie unter: https://www.visualstudio.com/license-terms/mt644918/)

MICROSOFT VISUAL C++ BUILD TOOLS

LIZENZBEDINGUNGEN FÜR MICROSOFT SOFTWARE

MICROSOFT VISUAL C++ BUILD TOOLS

Diese Lizenzbedingungen sind eine Vereinbarung zwischen der Microsoft Corporation (oder abhängig von Ihrem Wohnort einer ihrer Tochtergesellschaften) und Ihnen. Sie gelten für die oben genannte Software. Die Bedingungen gelten auch für alle Microsoft-Dienste oder Updates für die Software, es sei denn, diese enthalten andere Bedingungen.

WENN SIE DIESE LIZENZBEDINGUNGEN EINHALTEN, HABEN SIE DIE FOLGENDEN RECHTE.

1. INSTALLATIONS- UND NUTZUNGSRECHTE.
   a. Ein Benutzer darf Kopien der Software verwenden, um seine Anwendungen entwickeln und testen.
2. DATA. Die Software kann Informationen über Sie und Ihre Nutzung der Software erfassen und an Microsoft senden. Microsoft kann diese Informationen verwenden, um Dienstleistungen anzubieten und unsere Produkte und Dienstleistungen zu verbessern. Sie können viele dieser Szenarien ablehnen, aber nicht alle, wie in der Produktdokumentation beschrieben. Es gibt auch einige Funktionen in der Software, die es Ihnen ermöglichen, Daten von Benutzern Ihrer Anwendungen zu erfassen. Wenn Sie diese Funktionen nutzen, um eine Datenerfassung in Ihren Anwendungen zu ermöglichen, müssen Sie die geltenden Gesetze einhalten, einschließlich entsprechender Hinweise für die Benutzer Ihrer Anwendungen. Weitere Informationen zur Datenerfassung und -verwendung finden Sie in der Hilfe-Dokumentation und in der Datenschutzerklärung unter http://go.microsoft.com/fwlink/?LinkID=528096. Ihre Nutzung der Software gilt als Ihre Zustimmung zu diesen Praktiken.
3. BEDINGUNGEN FÜR SPEZIFISCHE KOMPONENTEN.
   a. Build Server. Die Software kann einige Build Server-Komponenten enthalten, die in BuildServerTXT-Dateien aufgeführt sind, und/oder alle Dateien, die in der BuildeServer Liste aufgeführt sind, die sich nach diesen Microsoft Software License Terms befinden. Sie dürfen diese Elemente, sofern sie in der Software enthalten sind, auf Ihre Build-Maschinen kopieren und installieren. Sie und andere in Ihrem Unternehmen dürfen diese Elemente auf Ihren Build-Maschinen ausschließlich zum Kompilieren, Erstellen, Verifizieren und Archivieren Ihrer Anwendungen oder zum Ausführen von Qualitäts- oder Leistungstests als Teil des Build-Prozesses verwenden.
   b. Microsoft-Plattformen. Die Software kann Komponenten von Microsoft Windows, Microsoft Windows Server, Microsoft SQL Server, Microsoft Exchange, Microsoft Office und Microsoft enthalten SharePoint. Diese Komponenten unterliegen separaten Vereinbarungen und eigenen

Produktsupport-Richtlinien, wie sie in den Lizenzbedingungen im Installationsverzeichnis dieser Komponente oder im Ordner „Lizenzen“ der Software beschrieben sind.

c. Komponenten von Drittanbietern. Die Software kann Komponenten von Drittanbietern mit separaten rechtlichen Hinweisen enthalten oder durch andere Vereinbarungen geregelt werden, wie in der ThirdPartyNotices Datei beschrieben, die die Software mit sich bringt. Auch wenn diese Komponenten durch andere Vereinbarungen geregelt sind, gelten die nachstehenden Haftungsausschlüsse und Schadensbegrenzungen.

d. Package Manager. Die Software kann Package Manager wie Nuget enthalten, die Ihnen die Möglichkeit geben, andere Softwarepakete von Microsoft und Drittanbietern zur Verwendung mit Ihrer Anwendung herunterzuladen. Diese Pakete unterliegen ihrer eigenen Lizenz und nicht dieser Vereinbarung. Microsoft vertreibt, lizenziert oder gibt keine Garantien für die Pakete von Drittanbietern.

4. LIZENZUMFANG. Die Software wird lizenziert, nicht verkauft. Diese Vereinbarung gibt Ihnen nur gewisse Rechte zur Nutzung der Software. Microsoft behält sich alle anderen Rechte vor. Sofern das geltende Recht Ihnen trotz dieser Einschränkung keine weiteren Rechte einräumt, dürfen Sie die Software nur so nutzen, wie es in dieser Vereinbarung ausdrücklich erlaubt ist. Dabei müssen Sie alle technischen Einschränkungen der Software beachten, die nur eine bestimmte Nutzung erlauben. Weitere Informationen finden Sie unter https://docs.microsoft.com/en-us/legal/information-protection/software-license-terms#1-installation-and-use-rights. Die folgenden Dinge sind verboten:

   - technische Einschränkungen in der Software zu umgehen;
   - die Software zurückzuentwickeln, zu dekompilieren oder zu disassemblieren oder dies zu versuchen, es sei denn und nur in dem Umfang, wie es die Lizenzbedingungen Dritter für die Verwendung bestimmter Open-Source-Komponenten, die in der Software enthalten sein können, erfordern;
   - Hinweise von Microsoft oder seinen Lieferanten zu entfernen, zu minimieren, zu blockieren oder zu ändern;
   - die Software in einer Weise zu nutzen, die gegen das Gesetz verstößt; oder
   - die Software zu teilen, zu veröffentlichen, zu vermieten oder zu verpachten oder die Software als eigenständige, gehostete Lösung für andere zur Verfügung zu stellen.

5. EXPORTBESCHRÄNKUNGEN. Sie müssen alle nationalen und internationalen Exportgesetze und -bestimmungen einhalten, die für die Software gelten, einschließlich der Beschränkungen für Bestimmungsorte, Endbenutzer und Endbenutzer. Weitere Informationen zu Exportbeschränkungen finden Sie unter (aka.ms/exporting).

6. SUPPORT-SERVICES. Da dieser Software „ohne Mängelgewähr” ist, stellen wir möglicherweise keine Support-Services dafür bereit.
7. GESAMTE VEREINBARUNG. Diese Vereinbarung und die Bedingungen für Ergänzungen, Aktualisierungen, internetbasierte Dienste und Supportleistungen, die Sie nutzen, sind die gesamte Vereinbarung für die Software und Supportleistungen.
8. ANWENDBARES RECHT. Wenn Sie die Software in den Vereinigten Staaten erworben haben, gilt das Washingtoner Recht für die Auslegung dieser Vereinbarung und für Ansprüche wegen Verletzung dieser Vereinbarung, und die Gesetze des Staates, in dem Sie leben, gelten für alle anderen Ansprüche. Wenn Sie die Software in einem anderen Land erworben haben, gelten dessen Gesetze.
9. VERBRAUCHERRECHTE; REGIONALE ABWEICHUNGEN. Diese Vereinbarung beschreibt bestimmte gesetzliche Rechte. Sie haben möglicherweise andere Rechte, einschließlich Verbraucherrechte, nach den Gesetzen Ihres Staates oder Landes. Abgesehen von Ihrer Beziehung zu Microsoft haben Sie möglicherweise auch Rechte in Bezug auf die Partei, von der Sie die Software erworben haben. Diese Vereinbarung ändert diese anderen Rechte nicht, wenn die Gesetze Ihres Staates oder Landes dies nicht zulassen. Wenn Sie die Software beispielsweise in einer der unten aufgeführten Regionen erworben haben oder zwingendes Landesrecht gilt, gelten für Sie die folgenden Bestimmungen:
   - Australien Sie haben gesetzliche Garantien nach dem australischen Verbrauchergesetz, und nichts in dieser Vereinbarung ist darauf ausgelegt, diese Rechte zu beeinträchtigen.
   - Kanada. Wenn Sie diese Software in Kanada erworben haben, können Sie den Erhalt von Updates beenden, indem Sie die automatische Update-Funktion deaktivieren, Ihr Gerät vom Internet trennen (wenn Sie sich jedoch wieder mit dem Internet verbinden, wird die Software die Überprüfung und Installation von Updates fortsetzen) oder die Software deinstallieren. In der Produktdokumentation, falls vorhanden, kann auch angegeben werden, wie Sie Updates für Ihr spezielles Gerät oder Ihre Software deaktivieren können.
   - Deutschland und Österreich.
     - Garantie. Die ordnungsgemäß lizenzierte Software funktioniert im Wesentlichen wie in allen Unterlagen von Microsoft beschrieben, die der Software beiliegen. Microsoft gibt jedoch keine vertragliche Garantie in Bezug auf die lizenzierte Software.
     - Haftungsbeschränkung. Bei Vorsatz, grober Fahrlässigkeit, Ansprüchen nach dem Produkthaftungsgesetz sowie bei Verletzung des Lebens, des Körpers oder der Gesundheit haftet Microsoft nach den gesetzlichen Bestimmungen.

Gemäß der vorstehenden Ziffer (ii) haftet Microsoft für leichte Fahrlässigkeit nur, wenn Microsoft gegen solche wesentlichen Vertragspflichten verstößt, deren Erfüllung die ordnungsgemäße Durchführung dieses Vertrages erleichtert, deren Verletzung den Zweck dieses Vertrages gefährdet und auf deren Einhaltung eine Partei ständig vertrauen darf (sogenannte „Kardinalpflichten“). In anderen Fällen leichter Fahrlässigkeit haftet Microsoft nicht für leichte Fahrlässigkeit.

10 ANWENDUNG VON GESETZEN. Diese Vereinbarung beschreibt bestimmte gesetzliche Rechte. Möglicherweise haben Sie anderen Rechte gemäß den Gesetzen Ihres Landes. Diese Vereinbarung ändert Ihre Rechte unter den Gesetzen Ihres Landes nicht, wenn die Gesetze Ihres Staates oder Landes dies nicht zulassen. Ohne Einschränkung der obigen Aussagen gilt für Australien: SIE HABEN GESETZLICHE GARANTIEN NACH DEM AUSTRALISCHEN VERBRAUCHERRECHT UND NICHTS IN DIESEN BEDINGUNGEN IST DARAUF AUSGELEGT, DIESE RECHTE ZU BEEINTRÄCHTIGEN.

11 AUSSCHLÜSSELUNG DER FEHLERUNG. DIE SOFTWARE WIRD „OHNE MÄNGELGEWÄHR“ LIZENZIERT. SIE TRAGEN DAS RISIKO IHRER NUTZUNG. MICROSOFT ERTEILT KEINE AUSDRÜCKLICHEN GARANTIEN, GEWÄHRLEISTUNGEN ODER ZUSAGEN. SOFERN DIES NACH IHREN LOKALEN GESETZEN ZULÄSSIG IST, SCHLIEßT MICROSOFT DIE IMPLIZITEN GARANTIEN DER MARKTGÄNGIGKEIT, DER EIGNUNG FÜR EINEN BESTIMMTEN ZWECK UND DER NICHTVERLETZUNG VON VERSTÖßEN AUS.

12. EINSCHRÄNKUNG UND AUSSCHLUSS VON ERSTATTUNGEN. VON MICROSOFT UND WIEDERHERSTELLUNG SIE DIE LIEFERANTEN ONLY DIREKTER SCHÄDEN BIS ZU US-5,00 USD. SIE KÖNNEN KEINE ANDEREN BEHANDLUNGEN WIEDERHERSTELLEN, EINSCHLIEßLICH DER ARZT, DER VERLORENEN, DER BESONDEREN, INDIREKTEN ODER DER KER.

Diese Einschränkung gilt für (a) alles, was mit der Software, Dienstleistungen, Inhalten (einschließlich Code) auf Internetseiten Dritter oder Anwendungen Dritter zusammenhängt; und (b) Ansprüche wegen Vertragsverletzung, Verletzung von Gewährleistung, Garantie oder Beschaffenheit, verschuldensunabhängiger Haftung, Fahrlässigkeit oder sonstiger unerlaubter Handlung, soweit dies nach geltendem Recht zulässig ist.

Sie gilt auch dann, wenn Microsoft von der Möglichkeit des Schadens wusste oder hätte wissen müssen. Die obige Einschränkung oder der Ausschluss gelten möglicherweise nicht für Sie, da Ihr Land den Ausschluss oder die Beschränkung von Neben-, Folge- oder anderen Schäden nicht zulässt.

11) Docker-Image auf Windows-Basis – microsoft-windows-netfx3-ondemand-package.cab

ZUSÄTZLICHE LIZENZBEDINGUNGEN FÜR MICROSOFT SOFTWARE

MICROSOFT .NET FRAMEWORK 3.5 SP1 FÜR DAS BETRIEBSSYSTEM MICROSOFT WINDOWS

Die Microsoft Corporation (oder abhängig von Ihrem Wohnort: eine ihrer Tochtergesellschaften) lizenziert diese Ergänzung an Sie. Wenn Sie eine Lizenz zur Verwendung von Microsoft Windows-Betriebssystem-Software (für die diese Ergänzung anwendbar ist) (die „Software“) besitzen, können Sie diese Ergänzung verwenden. Sie dürfen sie nicht verwenden, wenn Sie keine Lizenz für die Software besitzen. Sie dürfen eine Kopie dieser Ergänzung mit jeder gültig lizenzierten Kopie der Software verwenden.

Die folgenden Lizenzbedingungen beschreiben zusätzliche Nutzungsbedingungen für diese Ergänzung. Diese Bedingungen und die Lizenzbedingungen für die Software gelten für Ihre Nutzung der Ergänzung. Wenn ein Konflikt besteht, gelten diese zusätzlichen Lizenzbedingungen.

DURCH DIE VERWENDUNG DIESER ERGÄNZUNG AKZEPTIEREN SIE DIESE BEDINGUNGEN. WENN SIE SIE NICHT AKZEPTIEREN, VERWENDEN SIE DIESE ERGÄNZUNG NICHT.

Wenn Sie diese Lizenzbedingungen einhalten, haben Sie die folgenden Rechte.

1. SUPPORT-SERVICES FÜR DIE ERGÄNZUNG. Microsoft bietet Support-Services für diese Software an, wie unter www.support.microsoft.com/common/international.aspx beschrieben.
2. MICROSOFT .NET BENCHMARK-TESTS. Die Software enthält das .NET Framework, Windows Communication Foundation-, Windows Presentation Foundation- und Windows Workflow Foundation-Komponenten der Windows-Betriebssysteme (.NET-Komponenten). Sie dürfen interne Benchmark-Tests der .NET-Komponenten durchführen. Sie dürfen die Ergebnisse von Benchmark-Tests der .NET-Komponenten offenlegen, vorausgesetzt, dass Sie die Bedingungen unter http://go.microsoft.com/fwlink/?LinkID=66406 erfüllen.

   Wenn Sie Ergebnisse von Benchmark-Tests offenlegen hat Microsoft ungeachtet anderer Vereinbarungen, die Sie mit Microsoft getroffen haben, das Recht, die Ergebnisse von Benchmark-Tests, die es für Ihre Produkte durchführt, die mit der anwendbaren .NET-Komponente

konkurrieren, offen zu legen, vorausgesetzt, es erfüllt die gleichen Bedingungen, wie unter http://go.microsoft.com/fwlink/?LinkID=66406 festgelegt.

12) Windows-Basis-Docker-Image – dotnet-sdk

(verfügbar unter https://github.com/dotnet/core/blob/main/LICENSE.TXT)

Die MIT-Lizenz (MIT)

Copyright (c) Microsoft Corporation

Hiermit wird jeder Person, die eine Kopie dieser Software und der zugehörigen Dokumentationsdateien (die „Software“) erhält, die Erlaubnis erteilt, die Software ohne Einschränkung zu nutzen, zu kopieren, zu modifizieren, zusammenzuführen, zu veröffentlichen, zu verteilen, zu unterlizenzieren und/oder zu verkaufen, und Personen, denen die Software zur Verfügung gestellt wird, dies unter den folgenden Bedingungen zu gestatten:

Der obige Urheberrechtshinweis und dieser Genehmigungshinweis sind in allen Kopien oder wesentlichen Teilen der Software enthalten.

DIE SOFTWARE WIRD OHNE JEDE AUSDRÜCKLICHE ODER IMPLIZIERTE GARANTIE BEREITGESTELLT, EINSCHLIESSLICH DER GARANTIE ZUR BENUTZUNG FÜR DEN VORGESEHENEN ODER EINEM BESTIMMTEN ZWECK SOWIE JEGLICHER RECHTSVERLETZUNG, JEDOCH NICHT DARAUF BESCHRÄNKT. IN KEINEM FALL SIND DIE AUTOREN ODER COPYRIGHTINHABER FÜR JEGLICHEN SCHADEN ODER SONSTIGE ANSPRÜCHE HAFTBAR ZU MACHEN, OB INFOLGE DER ERFÜLLUNG EINES VERTRAGES, EINES DELIKTES ODER ANDERS IM ZUSAMMENHANG MIT DER SOFTWARE ODER SONSTIGER VERWENDUNG DER SOFTWARE ENTSTANDEN.

AWS CodeBuild Dokumenthistorie des Benutzerhandbuches

In der folgenden Tabelle werden die wichtigen Änderungen an der Dokumentation seit der letzten Version von beschrieben AWS CodeBuild. Um Benachrichtigungen über Aktualisierungen dieser Dokumentation zu erhalten, können Sie einen RSS-Feed abonnieren.

- Letzte API-Version: 2016-10-06

Änderung	Beschreibung	Datum
Festplattenspeicher aktualisiert	Die Typen `ARM Small` und `ARM Large` Compute verfügen jetzt über mehr Festplattenspeicher.	4. Juni 2024
Neuer Inhalt: GitHub manuelle Webhooks	Unterstützung für GitHub manuelle Webhooks hinzugefügt.	23. Mai 2024
Aktualisierter Inhalt: Reservierte Kapazität	CodeBuild unterstützt jetzt VPC-Konnektivität für Flotten mit reservierter Kapazität auf Amazon Linux.	15. Mai 2024
Aktualisierter Inhalt: Lambda-Computing-Bilder	Lambda-Unterstützung für .NET 8 hinzufügen (`al-lambda/aarch64/dotnet8` und `al-lambda/x86_64/dotnet8`)	8. Mai 2024
Aktualisiertes Kontingent: Timeout beim Erstellen	Aktualisieren Sie das maximale Build-Timeout-Kontingent auf 2160 Minuten (36 Stunden).	1. Mai 2024

Aktualisierter Inhalt: AWS verwaltete (vordefinierte) Richtlinien für AWS CodeBuild	Die AWSCodeBuildReadOnlyAccess Richtlinien AWSCodeBuildAdminAccess AWSCodeBuildDeveloperAccess, und wurden aktualisiert, um dem AWS CodeConnections Rebranding Rechnung zu tragen.	30. April 2024
Neue Inhalte: Passwort oder Zugriffstoken für die Bitbucket-App	Unterstützung für Bitbucket-Zugriffstoken hinzugefügt.	11. April 2024
Neue Inhalte: Automatische Erkennung melden	CodeBuild unterstützt jetzt die automatische Erkennung von Berichten.	4. April 2024
Neuer Inhalt: Richte selbst gehostete GitHub Actions-Runner ein	Füge neue Inhalte für selbst gehostete GitHub Actions-Runner hinzu	2. April 2024
Neuer Inhalt: GitLab Verbindungen	Unterstützung für GitLab und GitHub selbstverwaltete Verbindungen hinzufügen.	25. März 2024
Neuer Inhalt: Fügen Sie neue Webhook-Ereignisse und Filtertypen hinzu	Unterstützung für neue Webhook-Ereignisse (`RELEASED`und`PRERELEASED`) und Filtertypen (`TAG_NAME`und`RELEASE_NAME`) hinzugefügt.	15. März 2024
Neuer Inhalt: Füge ein neues Webhook-Event hinzu: `PULL_REQUEST_CLOSED`	Unterstützung für ein neues Webhook-Ereignis hinzufügen:. `PULL_REQUEST_CLOSED`	20. Februar 2024

Aktualisierter Inhalt: Docker-Images bereitgestellt von CodeBuild	Unterstützung für Windows Server Core 2019 hinzufügen (`windows-base:2019-3.0`)	7. Februar 2024
Aktualisierter Inhalt: Docker-Images bereitgestellt von CodeBuild	Unterstützung für neue Laufzeiten für Amazon Linux 2023 hinzufügen () `al2/aarch64/standard/3.0`	29. Januar 2024
Neuer Inhalt: Reservierte Kapazität	CodeBuild unterstützt jetzt Flotten mit reservierter Kapazität in CodeBuild.	18. Januar 2024
Neuer Compute-Typ	CodeBuild unterstützt jetzt einen Linux XLarge-Compute-Typ. Weitere Informationen finden Sie unter Erstellen von Umgebungs-Compute-Typen.	8. Januar 2024
Aktualisierter Inhalt: Docker-Images bereitgestellt von CodeBuild	Unterstützung für neue Laufzeiten für Amazon Linux 2 (`al2/standard/5.0`) und Ubuntu (`ubuntu/standard/7.0`) hinzugefügt	14. Dezember 2023
Aktualisierter Inhalt: Docker-Images bereitgestellt von CodeBuild	Unterstützung für neue Lambda-Compute-Images hinzufügen	8. Dezember 2023
Neuer Inhalt: AWS Lambda Compute	Füge neuen Inhalt für die AWS Lambda Berechnung hinzu	6. November 2023
Neuer Inhalt: Verwendung der GitHub Action-Syntax in einer Buildspec	Fügen Sie neuen Inhalt für die Syntax using GitHub Actions in einer Buildspec hinzu	6. Juli 2023

Aktualisierter Inhalt: Docker-Images bereitgestellt von CodeBuild	Unterstützung für Amazon Linux 2 hinzufügen (`al2/standard/5.0`)	17. Mai 2023
Änderungen an verwalteten Richtlinien für CodeBuild	Einzelheiten zu Aktualisierungen der AWS verwalteten Richtlinien für CodeBuild sind jetzt verfügbar. Weitere Informationen finden Sie unter CodeBuild Aktualisierungen AWS verwalteter Richtlinien.	16. Mai 2023
Aktualisierter Inhalt: Docker-Images bereitgestellt von CodeBuild	Unterstützung für Amazon Linux 2 (`al2/standard/3.0`) entfernen und Unterstützung für Amazon Linux 2 (`al2/standard/corretto8`) und Amazon Linux 2 (`al2/standard/corretto11`) hinzufügen	09. Mai 2023
Aktualisierter Inhalt: Docker-Images bereitgestellt von CodeBuild	Unterstützung für Ubuntu 22.04 hinzufügen () `ubuntu/standard/7.0`	13. April 2023
Aktualisierter Inhalt: Docker-Images bereitgestellt von CodeBuild	Unterstützung für Ubuntu 18.04 (`ubuntu/standard/4.0`) und Amazon Linux 2 () entfernen `al2/aarch64/standard/1.0`	31. März 2023

Aktualisierter Inhalt: VPC-Beschränkung entfernen	Aufhebung der folgenden Einschränkung: Wenn Sie für die Verwendung mit einer VPC konfigurieren CodeBuild , wird lokales Caching nicht unterstützt. Ab dem 28.02.22 dauert Ihr VPC-Build länger, da für jeden Build eine neue Amazon EC2 EC2-Instance verwendet wird.	1. März 2023
Aktualisierter Inhalt: Docker-Images bereitgestellt von CodeBuild	Unterstützung für Ubuntu 18.04 (`ubuntu/standard/3.0`) und Amazon Linux 2 () entfernen `al2/standard/2.0`	30. Juni 2022
Amazon ECR-Beispiel: Beschränken Sie den Bildzugriff	Wenn CodeBuild Anmeldeinformationen zum Abrufen eines Amazon ECR-Images verwendet werden, können Sie den Bildzugriff auf ein bestimmtes CodeBuild Projekt einschränken. Weitere Informationen finden Sie unter Amazon ECR-Beispiel.	10. März 2022
Zusätzlicher Support für Regionen	Der `ARM_CONTAINER` Berechnungstyp wird jetzt in den folgenden zusätzlichen Regionen unterstützt: Asien-Pazifik (Seoul), Kanada (Zentral), Europa (London) und Europa (Paris). Weitere Informationen finden Sie unter Erstellen von Umgebungs-Compute-Typen.	10. März 2022

Neue VPC-Beschränkung	Wenn Sie CodeBuild für die Arbeit mit einer VPC konfiguri eren, wird lokales Caching nicht unterstützt. Ab dem 28.02.22 dauert Ihr VPC-Build länger, da für jeden Build eine neue Amazon EC2 EC2-Insta nce verwendet wird.	25. Februar 2022
Batch-Berichtsmodus	CodeBuild ermöglicht es Ihnen jetzt auszuwählen, wie Batch-Build-Status für ein Projekt an den Quellanbieter gesendet werden. Weitere Informationen finden Sie unter Batch-Ber ichtsmodus.	4. Oktober 2021
Neuer Berechnungstyp	CodeBuild unterstützt jetzt einen kleinen ARM-Compute-Typ. Weitere Informationen finden Sie unter Erstellen von Umgebungs-Compute-Typen.	13. September 2021
Öffentliche Bauprojekte	CodeBuild ermöglicht es Ihnen jetzt, die Build-Ergebnisse für Ihre Build-Projekte der Öffentlichkeit zugänglich zu machen, ohne Zugriff auf ein AWS Konto zu benötigen. Weitere Informationen finden Sie unter Öffentliche Build-Pro jekte.	11. August 2021

Sitzungsdebugging für Batch-Builds	CodeBuild unterstützt jetzt das Session-Debugging für Batch-Builds. Weitere Informationen finden Sie unter build-graph und build-list.	3. März 2021
Limit für gleichzeitige Builds auf Projektebene	CodeBuild ermöglicht es Ihnen jetzt, die Anzahl gleichzeitiger Builds für ein Build-Projekt zu begrenzen. Weitere Informationen finden Sie unter Projektkonfiguration und concurrentBuildLimit.	16. Februar 2021
Neue Buildspec-Eigenschaft: s3-prefix	CodeBuild stellt jetzt die Eigenschaft `s3-prefix` buildspec für Artefakte bereit, mit der Sie ein Pfadpräfix für Artefakte angeben können, die auf Amazon S3 hochgeladen werden. Weitere Informationen finden Sie unter s3-Präfix.	9. Februar 2021
Neue Buildspec-Eigenschaft: on-failure	CodeBuild stellt jetzt die Eigenschaft `on-failure` buildspec für Buildphasen bereit, mit der Sie bestimmen können, was passiert, wenn eine Buildphase fehlschlägt. Weitere Informationen finden Sie unter On-Failure.	9. Februar 2021

Neue Buildspec-Eigenschaft: exclude-paths	CodeBuild stellt jetzt die Eigenschaft `exclude-paths` buildspec für Artefakte bereit, mit der Sie Pfade von Ihren Build-Artefakten ausschließen können. Weitere Informationen finden Sie unter exclude-paths.	9. Februar 2021
Neue buildspec-Eigenschaft: enable-symlinks	CodeBuild stellt jetzt die Eigenschaft `enable-symlinks` buildspec für Artefakte bereit, mit der Sie symbolische Links in einem ZIP-Artefakt beibehalten können. Weitere Informationen finden Sie unter enable-symlinks.	9. Februar 2021
Erweiterung der Namen von Buildspec-Artefakten	CodeBuild ermöglicht es der `artifacts/name` Eigenschaft jetzt, Pfadinformationen zu enthalten. Weitere Informationen finden Sie unter Name.	9. Februar 2021
Berichterstattung über die Codeabdeckung	CodeBuild bietet jetzt Berichte zur Codeabdeckung. Weitere Informationen finden Sie unter Berichte zur Codeabdeckung.	30. Juli 2020
Batch-Builds	CodeBuild unterstützt jetzt das Ausführen gleichzeitiger und koordinierter Builds eines Projekts. Weitere Informationen finden Sie unter Batch-Builds in CodeBuild.	30. Juli 2020

Windows Server 2019-Bild	CodeBuild bietet jetzt ein Windows Server Core 2019-Build-Image. Weitere Informationen finden Sie unter Docker-Images, bereitgestellt von CodeBuild.	20. Juli 2020
Sitzungsmanager	CodeBuild ermöglicht es Ihnen jetzt, einen laufenden Build anzuhalten und dann den AWS Systems Manager Sitzungsmanager zu verwenden, um eine Verbindung zum Build-Container herzustellen und den Status des Containers anzuzeigen. Weitere Informationen finden Sie unter Session Manager.	20. Juli 2020
Das Thema wurde aktualisiert	CodeBuild unterstützt jetzt die Angabe einer Shell, die in ihren Build-Umgebungen verwendet werden soll, in der Buildspec-Datei. Weitere Informationen finden Sie unter Referenz zur Build-Spezifikation.	25. Juni 2020

Testberichterstattung mit Testframeworks	Es wurden mehrere Themen hinzugefügt, in denen beschrieben wird, wie CodeBuild Testberichte mit verschiedenen Testframeworks generiert werden. Weitere Informationen finden Sie unter Testberichte mit Test-Frameworks.	29. Mai 2020
Aktualisierte Themen	CodeBuild unterstützt jetzt das Hinzufügen von Tags zu Berichtsgruppen. Weitere Informationen finden Sie unter ReportGroup.	21. Mai 2020
Support für Testberichte	CodeBuild Unterstützung für Testberichte ist jetzt allgemein verfügbar.	21. Mai 2020
Aktualisierte Themen	CodeBuild unterstützt jetzt das Erstellen von Webhook-Filtern für Github und Bitbucket, die Builds nur auslösen, wenn die Head-Commit-Nachricht mit dem angegebenen Ausdruck übereinstimmt. Weitere Informationen findest du unter Beispiel für GitHub Pull-Requests und Webhook-Filter und Beispiel für Bitbucket-Pull-Requests und Webhook-Filter.	6. Mai 2020

Neue Themen	CodeBuild unterstützt jetzt die gemeinsame Nutzung von Ressourcen für Build-Projekte und Berichtsgruppen. Weitere Informationen finden Sie unter Arbeiten mit freigegebenen Projekten und Arbeiten mit freigegebenen Berichtsgruppen.	13. Dezember 2019
Neue und aktualisierte Themen	CodeBuild unterstützt jetzt Testberichte während der Ausführung eines Build-Projekts. Weitere Informationen finden Sie unter Arbeiten mit Testberichten, Erstellen eines Testberichts und Erstellen eines Testberichts anhand des AWS CLI Beispiels.	25. November 2019
Das Thema wurde aktualisiert	CodeBuild unterstützt jetzt die Linux-Umgebungstypen GPU und ARM sowie den `2xlarge` Compute-Typ. Weitere Informationen finden Sie unter Erstellen von Umgebungs-Compute-Typen.	19. November 2019
Aktualisierte Themen	CodeBuild unterstützt jetzt Buildnummern für alle Builds, den Export von Umgebungsvariablen und die AWS Secrets Manager Integration. Weitere Informationen finden Sie unter Exportierte Variablen und Secrets Manager in Syntax der Build-Spezifikation.	6. November 2019

Neues Thema	CodeBuild unterstützt jetzt Benachrichtigungsregeln. Sie können Benachric htigungsregeln verwenden , um Benutzer über wichtige Änderungen in Build-Projekten zu benachrichtigen. Weitere Informationen finden Sie unter Erstellen einer Benachric htigungsregel.	5. November 2019
Aktualisierte Themen	CodeBuild unterstützt jetzt die Laufzeiten Android Version 29 und Go Version 1.13. Weitere Informationen finden Sie unter Docker-Images, bereitgestellt von CodeBuild und Buildspec-Syntax.	10. September 2019
Aktualisierte Themen	Wenn Sie ein Projekt erstellen, können Sie jetzt das verwaltet e Abbild von Amazon Linux 2 (AL2) auswählen. Weitere Informationen finden Sie unter Docker-Images bereitgestellt von CodeBuild und Runtime-Versionen in der Buildspec-Datei (Beispiel für). CodeBuild	16. August 2019

Das Thema wurde aktualisiert	Beim Erstellen eines Projekts können Sie jetzt die Verschlüsselung von S3-Protokollen deaktivieren und, falls Sie ein Git-basiertes Quellrepository verwenden, Git-Submodule einbeziehen. Weitere Informationen finden Sie unter Erstellen eines Build-Projekts in CodeBuild.	8. März 2019
Neues Thema	CodeBuild unterstützt jetzt lokales Caching. Beim Erstellen eines Builds können Sie lokales Caching in einem oder mehreren von vier Modi angeben. Weitere Informationen finden Sie unter Build Caching in. CodeBuild	21. Februar 2019
Neue Themen	CodeBuild unterstützt jetzt Webhook-Filtergruppen zur Angabe von Ereignissen, die einen Build auslösen. Weitere Informationen findest du unter Webhook-Ereignisse filtern und GitHub Bitbucket-Webhook-Ereignisse filtern.	8. Februar 2019
Neues Thema	Das CodeBuild Benutzerhandbuch zeigt nun, wie die Verwendung CodeBuild mit einem Proxyserver funktioniert. Weitere Informationen finden Sie unter Verwendung CodeBuild mit einem Proxyserver.	4. Februar 2019

Aktualisierte Themen	CodeBuild unterstützt jetzt die Verwendung eines Amazon ECR-Images, das sich in einem anderen AWS Konto befindet. Verschiedene Themen wurden aktualisiert, um dieser Änderung Rechnung zu tragen, darunter das Amazon ECR-Beispiel für CodeBuild, Erstellen eines Build-Projekts und Erstellen einer CodeBuild Servicerolle.	24. Januar 2019
Support für private Docker-Registries	CodeBuild unterstützt jetzt die Verwendung eines Docker-Images, das in einer privaten Registrierung gespeichert ist, als Laufzeitumgebung. Weitere Informationen finden Sie unter Private Registrierung mit AWS Secrets Manager Beispiel.	24. Januar 2019
Das Thema wurde aktualisiert	CodeBuild unterstützt jetzt die Verwendung eines Zugriffstokens für die Verbindung zu GitHub Repositorys (mit einem persönlichen Zugriffstoken) und Bitbucket-Repositorys (mit einem App-Passwort). Weitere Informationen finden Sie unter Erstellen eines Build-Projekts (Konsole) und Verwenden von Zugriffstoken mit Ihrem Quellanbieter.	6. Dezember 2018

Das Thema wurde aktualisiert	CodeBuild unterstützt jetzt neue Build-Metriken, mit denen die Dauer jeder Phase in einem Build gemessen wird. Weitere Informationen finden Sie unter CodeBuild CloudWatch Metriken.	15. November 2018
Thema VPC-Endpunktrichtlinie	Amazon VPC-Endpunkte unterstützen CodeBuild derzeit Richtlinien. Weitere Informationen finden Sie unter Erstellen einer VPC-Endpunktrichtlinie für CodeBuild.	9. November 2018
Aktualisierter Inhalt	Themen wurde aktualisiert, um die neue Konsolenumgebung wiederzugeben.	30. Oktober 2018
Amazon EFS-Beispiel	CodeBuild kann ein Amazon EFS-Dateisystem während eines Builds mithilfe von Befehlen in der Buildspec-Datei eines Projekts mounten. Weitere Informationen finden Sie unter Amazon EFS-Beispiel für CodeBuild.	26. Oktober 2018
Bitbucket-Webhooks	CodeBuild unterstützt jetzt Webhooks, wenn du Bitbucket für dein Repository verwendest. Weitere Informationen findest du unter Bitbucket-Pull-Request-Beispiel für. CodeBuild	2. Oktober 2018

S3-Protokolle	CodeBuild unterstützt jetzt Build-Logs in einem S3-Bucket . Bisher konnten Sie Logs nur mithilfe von CloudWatch Logs erstellen. Weitere Informationen finden Sie unter Erstellen eines Projekts.	17. September 2018
Mehrere Eingabequellen und mehrere Ausgabeartefakte	CodeBuild unterstützt jetzt Projekte, die mehr als eine Eingabequelle verwenden und mehr als einen Satz von Artefakten veröffentlichen. Weitere Informationen finden Sie unter Beispiel für mehrere Eingabequellen und Eingabeartefakte und CodePipeline Integration mit CodeBuild und Beispiel für mehrere Eingabequellen und Ausgabeartefakte.	30. August 2018
Beispiel für semantische Versionierung	Das CodeBuild Benutzerhandbuch enthält jetzt ein auf Anwendungsfällen basierendes Beispiel, das zeigt, wie semantische Versionierung verwendet werden kann, um Artefaktnamen während der Erstellung zu erstellen. Weitere Informationen finden Sie unter Beispiel zur Verwendung des semantischen Versionings zum Benennen von Build-Artefakten.	14. August 2018

Neues Beispiel für eine statische Website	Das CodeBuild Benutzerh andbuch enthält jetzt ein Anwendungsfallbeispiel, das zeigt, wie Build-Ausgaben in einem S3-Bucket gehostet werden. Das Beispiel nutzt die kürzlich ergänzte Unterstüt zung unverschlüsselter Build-Artefakte. Weitere Informati onen finden Sie unter Erstellen einer statischen Website mit in einem S3-Bucket gehosteter Build-Ausgabe.	14. August 2018
Support für das Überschre iben eines Artefaktnamens mit semantischer Versionierung	Sie können jetzt die semantisc he Versionierung verwenden , um ein Format anzugeben, das zur Benennung von Build-Artefakten verwendet wird. CodeBuild Dies ist praktisch, da ein Build-Artefakt mit einem hartcodierten Namen frühere Build-Artefakte überschreibt, die denselben hartcodierten Namen haben. Wird beispiels weise ein Build mehrfach pro Tag ausgelöst, können Sie jetzt einen Zeitstempel zu seinem Artifact-Namen hinzufügen. Jeder Build-Artefakt-Name ist eindeutig und überschreibt nicht die Artefakte des vorherigen Builds.	7. August 2018

Support von unverschlüsselten Build-Artefakten	CodeBuild unterstützt jetzt Builds mit unverschlüsselten Build-Artefakten. Weitere Informationen finden Sie unter Erstellen eines Build-Projekts (Konsole).	26. Juli 2018
Support für CloudWatch Amazon-Metriken und -Alarme	CodeBuild bietet jetzt die Integration mit CloudWatch Metriken und Alarmen. Sie können die CodeBuild CloudWatch OR-Konsole verwenden, um Builds auf Projekt- und Kontoebene zu überwachen. Weitere Informationen finden Sie unter Überwachung von Builds.	19. Juli 2018
Support für die Meldung des Status eines Builds	CodeBuild kann jetzt den Status des Beginns und des Abschlusses eines Builds an Ihren Quellanbieter melden. Weitere Informationen finden Sie unter Erstellen eines Build-Projekts in CodeBuild.	10. Juli 2018
Umgebungsvariablen wurden der CodeBuild Dokumentation hinzugefügt	Die Seite Umgebungsvariablen in Build-Umgebungen wurde durch die Umgebungsvariablen CODEBUILD_BUILD_ID, CODEBUILD_LOG_PATH und CODEBUILD_START_TIME aktualisiert.	9. Juli 2018

Support für einen `finally` Block in der Buildspec-Datei	Die CodeBuild Dokumentation wurde mit Details zum optionalen `finally` Block in einer Buildspec-Datei aktualisiert. Befehle im Finaly-Block werden immer nach den Befehlen im entsprechenden Befehlsblock ausgeführt. Weitere Informationen finden Sie unter Syntax der Build-Spezifikation.	20. Juni 2018
CodeBuild Benachrichtigungen über Agenten-Updates	Die CodeBuild Dokumentation wurde mit Einzelheiten darüber aktualisiert, wie Sie Amazon SNS verwenden können, um benachrichtigt zu werden, wenn neue Versionen des CodeBuild Agenten veröffentlicht werden. Weitere Informationen finden Sie unter Empfangen von Benachrichtigungen für neue AWS CodeBuild Agentenversionen.	15. Juni 2018

Frühere Aktualisierungen

In der folgenden Tabelle sind wichtige Änderungen in jeder Version des AWS CodeBuild - Benutzerhandbuchs vor Juni 2018 beschrieben.

Änderung	Beschreibung	Datum
Unterstützung für Windows-Builds	CodeBuild unterstützt jetzt Builds für die Microsoft Windows Server-Plattform, einschließlich einer vorkonfig	25. Mai 2018

Änderung	Beschreibung	Datum
	urierten Build-Umgebung für den.NET Core 2.0 unter Windows. Weitere Informationen finden Sie unter Microsoft Windows-Beispiele für CodeBuild.	
Unterstützung für Build-Idempotenz	Wenn Sie den Befehl `start-build` über die AWS Command Line Interface (AWS CLI) ausführen, können Sie angeben, dass das Build idempotent ist. Weitere Informationen finden Sie unter Ausführen eines Build (AWS CLI).	15. Mai 2018
Unterstützung für das Überschreiben mehrerer Build-Projekteinstellungen	Sie können nun mehrere Build-Projekteinstellungen außer Kraft setzen, wenn Sie ein Build erstellen. Die Überschreibungen gelten nur für dieses Build. Weitere Informationen finden Sie unter Ausführen eines Build in AWS CodeBuild.	15. Mai 2018
Unterstützung von VPC-Endpunkt	Sie können die Sicherheit Ihrer Builds nun mithilfe von VPC-Endpunkten verbessern. Weitere Informationen finden Sie unter Verwenden von VPC-Endpunkten.	18. März 2018

Änderung	Beschreibung	Datum
Unterstützung von Auslösern	Sie können jetzt Auslöser erstellen, um in regelmäßigen Abständen Builds zu planen. Weitere Informationen finden Sie unter AWS CodeBuild Trigger erstellen.	28. März 2018
FIPS-Endpunktdokumentation	Sie können jetzt lernen, wie Sie mit dem AWS Command Line Interface (AWS CLI) oder einem AWS SDK angeben, dass CodeBuild Sie einen von vier FIPS-Endpunkten (Federal Information Processing Standards) verwenden sollen. Weitere Informationen finden Sie unter Angeben des AWS CodeBuild-Endpunkts.	28. März 2018
AWS CodeBuild verfügbar in Asien-Pazifik (Mumbai), Europa (Paris) und Südamerika (São Paulo)	AWS CodeBuild ist jetzt in den Regionen Asien-Pazifik (Mumbai), Europa (Paris) und Südamerika (São Paulo) verfügbar. Weitere Informationen finden Sie unter AWS CodeBuild im Allgemeine Amazon Web Services-Referenz.	28. März 2018

Änderung	Beschreibung	Datum
GitHub Unterstützung für Unternehmensserver	CodeBuild kann jetzt aus Quellcode bauen, der in einem GitHub Enterprise Server-Repository gespeichert ist. Weitere Informationen finden Sie unter GitHub Beispiel für einen Enterprise Server.	25. Januar 2018
Support von tiefen Git-Klons	CodeBuild unterstützt jetzt die Erstellung eines Shallow-Clones, dessen Historie auf die angegebene Anzahl von Commits gekürzt ist. Weitere Informationen finden Sie unter Erstellen eines Build-Projekts.	25. Januar 2018
VPC-Unterstützung	VPC-fähige Builds können jetzt auf Ressourcen innerhalb Ihrer VPC zugreifen. Weitere Informationen finden Sie unter VPC-Unterstützung.	27. November 2017
Unterstützung des Cachings von Abhängigkeiten	CodeBuild unterstützt jetzt das Zwischenspeichern von Abhängigkeiten. Dies ermöglicht es CodeBuild , bestimmte wiederverwendbare Teile der Build-Umgebung im Cache zu speichern und diese für mehrere Builds zu verwenden.	27. November 2017

Änderung	Beschreibung	Datum
Build Badges-Support	CodeBuild unterstützt jetzt die Verwendung von Build-Badges, die ein integrierbares, dynamisch generiertes Bild (Badge) bereitstellen, das den Status des letzten Builds für ein Projekt anzeigt. Weitere Informationen finden Sie unter Build Badges-Beispiel.	27. November 2017
AWS Config Integration	AWS Config wird jetzt CodeBuild als AWS Ressource unterstützt, was bedeutet, dass der Dienst Ihre CodeBuild Projekte verfolgen kann. Weitere Informationen zu finden AWS Config Sie unterAWS Config Probe.	20. Oktober 2017
Automatischer Neuaufbau des aktualisierten Quellcodes in GitHub Repositorys	Wenn Ihr Quellcode in einem GitHub Repository gespeichert ist, können Sie aktivieren, AWS CodeBuild dass Ihr Quellcode jedes Mal neu erstellt wird, wenn eine Codeänderung in das Repository übertragen wird. Weitere Informationen finden Sie unter GitHub Beispiel für einen Pull-Request und einen Webhook-Filter.	21. September 2017

Änderung	Beschreibung	Datum
Neue Möglichkeiten zum Speichern und Abrufen sensibler oder großer Umgebungsvariablen im Amazon EC2 Systems Manager Parameter Store	Sie können jetzt die AWS CodeBuild Konsole oder die verwenden AWS CLI , um sensible oder große Umgebungsvariablen abzurufen, die im Amazon EC2 Systems Manager Parameter Store gespeicherte sind. Sie können jetzt auch die AWS CodeBuild Konsole verwenden, um diese Typen von Umgebungsvariablen im Amazon EC2 Systems Manager Parameter Store zu speichern. Bisher konnten Sie diese Arten von Umgebungsvariablen nur abrufen, indem Sie sie in einer Build-Spezifikation einschlossen oder Build-Befehle zur Automatisierung der AWS CLI ausführten. Sie konnten diese Typen von Umgebungsvariablen nur mithilfe der Amazon EC2 Systems Manager Parameter Store-Konsole speichern. Weitere Informationen finden Sie unter Erstellen eines Build-ProjektsÄndern der Einstellungen eines Build-Projekts, undAusführen eines Build.	14. September 2017

Änderung	Beschreibung	Datum
Unterstützung der Build-Löschung	Sie können jetzt Builds in AWS CodeBuild löschen. Weitere Informationen finden Sie unter Löschen von Builds.	31. August 2017
Aktualisierte Methode zum Abrufen sensibler oder großer Umgebungsvariablen, die im Amazon EC2 Systems Manager Parameter Store gespeichert sind, mithilfe einer Buildspec	AWS CodeBuild macht es jetzt einfacher, eine Buildspec zu verwenden, um sensible oder große Umgebungsvariablen abzurufen, die im Amazon EC2 Systems Manager Parameter Store gespeichert sind. Bisher konnten Sie diese Arten von Umgebungsvariablen nur abrufen, indem Sie Build-Befehle ausführten, um die AWS CLI zu automatisieren. Weitere Informationen finden Sie in der Zuordnung unter `parameter-store` Syntax der Build-Spezifikation	10. August 2017
AWS CodeBuild unterstützt Bitbucket	CodeBuild kann jetzt aus Quellcode bauen, der in einem Bitbucket-Repository gespeichert ist. Weitere Informationen finden Sie unter Erstellen eines Build-Projekts und Ausführen eines Build.	10. August 2017

Änderung	Beschreibung	Datum
AWS CodeBuild verfügbar in den USA West (Nordkali fornien), Europa (London) und Kanada (Zentral)	AWS CodeBuild ist jetzt in den Regionen USA West (Nordkali fornien), Europa (London) und Kanada (Mitte) verfügbar . Weitere Informationen finden Sie unter AWS CodeBuild im Allgemeine Amazon Web Services-Referenz.	29. Juni 2017
Unterstützung alternativer Namen und Speicherorte für Build-Spezifikationsdateien	Sie können jetzt einen anderen Dateinamen oder Speicherort einer Build-Spe zifikationsdatei für ein Build-Projekt angeben und müssen nicht die Standard-Build-Spezifikationsdatei mit dem Namen `buildspec.yml` im Stammverzeichnis des Quellcodes nutzen. Weitere Informationen finden Sie unter Dateiname der Build-Spe zifikation und Speicherort.	27. Juni 2017
Aktualisiertes Build-Ben achrichtigungsbeispiel	CodeBuild bietet jetzt integrier te Unterstützung für Build-Ben achrichtigungen über Amazon CloudWatch Events und Amazon Simple Notification Service (Amazon SNS). Der vorherige Build-Benachrichti gungsbeispiel wurde aktualisi ert, um das neue Verhalten zu demonstrieren.	22. Juni 2017

Änderung	Beschreibung	Datum
Beispiel für einen Docker im benutzerdefinierten Image hinzugefügt	Es wurde ein Beispiel hinzugefügt, das zeigt, wie ein Docker-Image verwendet wird, CodeBuild und ein benutzerd efiniertes Docker-Build-Image zum Erstellen und Ausführen eines Docker-Images. Weitere Informationen hierzu finden Sie unter Docker im benutzerd efinierten Image – Beispiel.	7. Juni 2017
Holen Sie sich den Quellcode für Pull-Requests GitHub	Wenn Sie einen Build ausführen CodeBuild , der auf dem in einem GitHub Repository gespeicherten Quellcode basiert, können Sie jetzt eine GitHub Pull-Requ est-ID für den Build angeben. Sie können alternativ auch eine Commit-ID, einen Namen einer Verzweigung oder einen Tag-Namen angeben. Weitere Informationen finden Sie unter dem Wert für die Quellversion in Ausführen eines Build (Konsole) oder dem `sourceVersion` Wert inAusführen eines Build (AWS CLI).	6. Juni 2017

Änderung	Beschreibung	Datum
Version der Build-Spezifikation aktualisiert	Eine neue Version des Build-Spezifikationsformat wurde freigegeben. Version 0.2 behebt das Problem, dass jeder Build-Befehl in einer separaten Instanz der Standard-Shell CodeBuild ausgeführt wird. Darüber hinaus wurde in Version 0.2 `environment_variables` in `env` umbenannt und `plaintext` in `variables`. Weitere Informationen finden Sie unter Referenz zur Build-Spezifikation für CodeBuild.	9. Mai 2017
Dockerfiles für Build-Images sind verfügbar in GitHub	Definitionen für viele der von bereitgestellten Build-Images AWS CodeBuild sind als Dockerfiles in GitHub verfügbar. Weitere Informationen finden Sie in der Spalte Definition der Tabelle unter. Docker-Images bereitgestellt von CodeBuild	2. Mai 2017
AWS CodeBuild verfügbar in Europa (Frankfurt), Asien-Pazifik (Singapur), Asien-Pazifik (Sydney) und Asien-Pazifik (Tokio)	AWS CodeBuild ist jetzt in den Regionen Europa (Frankfurt), Asien-Pazifik (Singapur), Asien-Pazifik (Sydney) und Asien-Pazifik (Tokio) verfügbar. Weitere Informationen finden Sie unter AWS CodeBuild im Allgemeine Amazon Web Services-Referenz.	21. März 2017

Änderung	Beschreibung	Datum
CodePipeline Unterstützung von Testaktionen für CodeBuild	Sie können jetzt zu einer Pipeline in CodePipeline einer Testaktion hinzufügen, die verwendet CodeBuild. Weitere Informationen finden Sie unter Hinzufügen einer CodeBuild -Testaktion zu einer Pipeline (CodePipeline-Konsole).	8. März 2017
Build-Spezifikationsdateien unterstützen den Abruf der Build-Ausgabe in ausgewählten Top-Level-Verzeichnissen	Mit Buildspec-Dateien können Sie jetzt einzelne Verzeichnisse der obersten Ebene angeben, deren Inhalt Sie in Build-Ausgabeartefakte CodeBuild aufnehmen können. Dies können Sie durch Verwenden der Zuweisung `base-directory` tun. Weitere Informationen finden Sie unter Syntax der Build-Spezifikation.	8. Februar 2017

Änderung	Beschreibung	Datum
Integrierte Umgebungsvariablen	AWS CodeBuild stellt zusätzliche integrierte Umgebungsvariablen bereit, die Ihre Builds verwenden können. Hierzu gehören Umgebungsvariablen, die die Entität beschreiben, die den Build gestartet hat, die URL zum Quellcode-Repository, die Versions-ID des Quellcodes und mehr. Weitere Informationen finden Sie unter Umgebungsvariablen in Build-Umgebungen.	30. Januar 2017
AWS CodeBuild verfügbar in USA Ost (Ohio)	AWS CodeBuild ist jetzt in der Region USA Ost (Ohio) verfügbar. Weitere Informationen finden Sie unter AWS CodeBuild im Allgemeine Amazon Web Services-Referenz.	19. Januar 2017

Änderung	Beschreibung	Datum
Informationen zur Shell und zum Verhalten von Befehlen	CodeBuild führt jeden von Ihnen angegebenen Befehl in einer separaten Instanz der Standard-Shell einer Build-Umgebung aus. Dieses Standardverhalten kann einige unerwartete Nebeneffekte für Ihre Befehle hervorrufen. Wir empfehlen Ihnen, gegebenen falls einige Ansätze, um dieses Standardverhalten zu umgehen. Weitere Informati onen finden Sie unter Shells und Befehle in Build-Umg ebungen.	9. Dezember 2016
Informationen zu Umgebungs variablen	CodeBuild stellt mehrere Umgebungsvariablen bereit, die Sie in Ihren Build-Befehlen verwenden können. Sie können auch Ihre eigenen Umgebungs variablen festlegen. Weitere Informationen finden Sie unter Umgebungsvariablen in Build-Umgebungen.	7. Dezember 2016
Thema Fehlerbehebung	Informationen zur Fehlerbeh ebung sind jetzt verfügbar. Weitere Informationen finden Sie unter Fehlerbehebung AWS CodeBuild.	5. Dezember 2016

Änderung	Beschreibung	Datum
Erste Version von Jenkins-Plugin	Dies ist die erste Version des CodeBuild Jenkins-Plugins. Weitere Informationen finden Sie unter Verwenden von AWS CodeBuild mit Jenkins.	5. Dezember 2016
Erstausgabe des User Guide	Dies ist die erste Version des CodeBuild Benutzerhandbuchs.	1. Dezember 2016

AWS-Glossar

Die neueste AWS-Terminologie finden Sie im AWS-Glossar in der AWS-Glossar-Referenz.

Die vorliegende Übersetzung wurde maschinell erstellt. Im Falle eines Konflikts oder eines Widerspruchs zwischen dieser übersetzten Fassung und der englischen Fassung (einschließlich infolge von Verzögerungen bei der Übersetzung) ist die englische Fassung maßgeblich.