ユーザーガイド

AWS CodeStar

AWS CodeStar: ユーザーガイド

Table of Contents

..... viii
AWS CodeStar とは? 1
AWS CodeStar の機能 1
AWS CodeStar の使用を開始するには 2
セットアップ 3
ステップ 1: アカウントを作成する 3
にサインアップする AWS アカウント 3
管理アクセスを持つユーザーを作成する 4
ステップ 2: AWS CodeStar サービスロールを作成する 5
ステップ 3: ユーザーの IAM アクセス許可を設定する 6
ステップ 4: AWS CodeStar プロジェクトの Amazon EC2 キーペアの作成 6
ステップ 5: AWS CodeStar コンソールを開く 7
次のステップ 7
AWS CodeStar の使用の開始 8
ステップ 1: AWS CodeStar プロジェクトを作成する 9
ステップ 2: AWS CodeStar ユーザープロファイルの表示情報を追加する 15
ステップ 3: プロジェクトを表示する 15
ステップ 4: 変更をコミットする 16
ステップ 5: チームメンバーの追加 22
ステップ 6: クリーンアップ 24
ステップ 7: 本番稼働環境のプロジェクトの準備 25
次のステップ 25
サーバーレスプロジェクトのチュートリアル 26
概要 27
ステップ 1: プロジェクトを作成する 28
ステップ 2: プロジェクトリソースを調べる 29
ステップ 3: ウェブサービスをテストする 33
ステップ 4: プロジェクトコードを編集するためのローカルワークステーションの設定 34
ステップ 5: ウェブサービスにロジックを追加する 34
ステップ 6: 拡張ウェブサービスをテストする 37
ステップ 7: ウェブサービスにユニットテストを追加する 38
ステップ 8: ユニットテストの結果を表示する 40
ステップ 9: クリーンアップ 41
次のステップ 42

AWS CLI プロジェクトチュートリアル 42
ステップ 1: サンプルソースコードのダウンロードと確認 43
ステップ 2: サンプルツールチェーンテンプレートのダウンロード 44
ステップ 3:AWS CloudFormation のツールチェーンテンプレートのテスト 45
ステップ 4: ソースコードとツールチェーンテンプレートのアップロード 46
ステップ 5: AWS CodeStar でプロジェクトを作成する 47
Alexa スキルプロジェクトのチュートリアル 50
前提条件 50
ステップ 1: プロジェクトを作成して Amazon 開発者アカウントに接続する 51
ステップ 2: Alexa Simulator でスキルをテストする 52
ステップ 3: プロジェクトリソースを調べる 53
ステップ 4: スキルの応答を変更する 53
ステップ 5: ローカルワークステーションを設定してプロジェクトリポジトリに接続する 54
次のステップ 54
チュートリアル:GitHub ソースリポジトリを使用してプロジェクトを作成する 55
ステップ 1: プロジェクトと GitHub リポジトリの作成 55
ステップ 2: ソースコードの表示 59
ステップ 3: GitHub プルリクエストの作成 59
プロジェクトテンプレート 61
AWS CodeStar プロジェクトファイルとリソース 61
はじめに: プロジェクトテンプレートを選択する 63
テンプレートコンピューティングプラットフォームを選択する 63
テンプレートアプリケーションタイプを選択する 64
テンプレートのプログラミング言語の選択 65
AWS CodeStar プロジェクトを変更する方法 65
アプリケーションソースコードの変更と変更のプッシュ 66
Template.yml ファイルを使用してアプリケーションリソースを変更する 66
..... 67
AWS CodeStar ベストプラクティス 68
AWS CodeStar リソースで使用するセキュリティのベストプラクティス 68
依存関係のバージョンを設定するベストプラクティス 68
AWS CodeStar リソースで使用するモニタリングとログ記録のベストプラクティス 69
プロジェクト作業 70
プロジェクトの作成 71
AWS CodeStar コンソールでプロジェクトを作成する (コンソール) 72
AWS CodeStar でプロジェクトを作成する (AWS CLI) 77

AWS CodeStar で IDE を使用する 84
AWS CodeStar で AWS Cloud9 を使用する 85
AWS CodeStar で Eclipse を使用する 92
AWS CodeStar で Visual Studio を使用する 97
プロジェクトのリソースの変更 99
サポートされているリソースの変更 99
AWS CodePipeline にステージを追加する 101
AWS Elastic Beanstalk 環境設定の変更 102
ソースコードの AWS Lambda 関数を変更する 102
プロジェクトのトレースを有効にする 102
リソースをプロジェクトに追加する 105
IAM ロールをプロジェクトに追加する 111
Prod ステージとエンドポイントをプロジェクトに追加する 112
AWS CodeStar プロジェクトで SSM パラメータを安全に使用する 122
AWS Lambda プロジェクトのトラフィックを移行する 124
AWS CodeStar プロジェクトを本番稼働用に移行する 131
GitHub リポジトリを作成する 132
プロジェクトタグの操作 133
プロジェクトにタグを追加する 133
プロジェクトからタグを削除する 134
プロジェクトのタグのリストを取得します。 134
プロジェクトの削除 134
AWS CodeStar のプロジェクトを削除する (コンソール) 136
AWS CodeStar のプロジェクトを削除する (AWS CLI) 137
チームでの作業 139
プロジェクトにチームメンバーを追加する 141
チームメンバーを追加する (コンソール) 143
チームメンバーを追加および表示する (AWS CLI) 145
チームアクセス許可の管理 146
チームアクセス許可の管理 (コンソール) 147
チームアクセス許可の管理 (AWS CLI) 148
プロジェクトからチームメンバーを削除する 148
チームメンバーを削除する (コンソール) 149
チームメンバーを削除する (AWS CLI) 150
AWS CodeStar ユーザープロファイルの操作方法 151
表示情報の管理 151

ユーザープロファイルの管理 (コンソール) 152
ユーザープロファイルの管理 (AWS CLI) 153
ユーザープロファイルへのパブリックキーの追加 156
ポリシーキーを管理する (コンソール) 156
パブリックキーを管理する (AWS CLI) 157
プライベートキーを使用して Amazon EC2 インスタンスに接続 158
セキュリティ 160
データ保護 161
でのデータ暗号化 AWS CodeStar 162
Identity and Access Management 162
対象者 163
アイデンティティを使用した認証 163
ポリシーを使用したアクセスの管理 167
AWS CodeStar の仕組み IAM 169
AWS CodeStar プロジェクトレベルのポリシーとアクセス許可 181
アイデンティティベースのポリシーの例 187
トラブルシューティング 218
AWS CloudTrail を使用した AWS CodeStar API コールのログ記録 220
AWS CodeStarCloudTrail での 情報 220
AWS CodeStar ログファイルエントリの概要 221
コンプライアンス検証 223
耐障害性 223
インフラストラクチャセキュリティ 223
Limits 225
トラブルシューティング AWS CodeStar 227
プロジェクトの作成の失敗: プロジェクトが作成されませんでした 227
プロジェクトの作成: プロジェクトの作成時に Amazon EC2 の設定を編集しようとすると エラーが発生します 228
プロジェクトの削除: AWS CodeStar プロジェクトは削除されましたが、リソースはまだ存在 します 229
チーム管理の失敗: IAM ユーザーを AWS CodeStar プロジェクトのチームに追加できませんで した 230
アクセス障害: フェデレーティッドユーザーが AWS CodeStar プロジェクトにアクセスできな い 231
アクセス障害: フェデレーティッドユーザーが AWS Cloud9 環境にアクセスしたり、環境を作 成したりできない 231

アクセスの失敗: フェデレーティッドユーザーは AWS CodeStar プロジェクトを作成できますが、プロジェクトリソースを表示できません 232
サービスロールの問題: サービスロールを作成できませんでした 232
サービスロールの問題: サービスロールが有効でないか、または存在しません 232
プロジェクトロールの問題: AWS CodeStar プロジェクト内のインスタンスの AWS Elastic Beanstalk ヘルスステータスチェックが失敗する 233
プロジェクトロールの問題: プロジェクトロールが有効でないか、または存在しません 234
プロジェクトの拡張機能: JIRA に接続できません 234
GitHub: リポジトリのコミット履歴、問題、またはコードにアクセスできない 234
AWS CloudFormation: アクセス許可の不足により、スタックの作成がロールバックされた 235
AWS CloudFormation は iam:PassRole on Lambda 実行ロールを実行する権限がありません .. 235
GitHub リポジトリの接続を作成できません 236
リリースノート 237
AWS 用語集 243

2024 年 7 月 31 日に、Amazon Web Services (AWS) は AWS CodeStar プロジェクトの作成と表示のサポートを終了します。2024 年 7 月 31 日以降、 AWS CodeStar コンソールにアクセスしたり、新しいプロジェクトを作成したりできなくなります。ただし、ソースリポジトリ AWS CodeStar、パイプライン、ビルドなど、 によって作成された AWS リソースは、この変更の影響を受けず、引き続き機能します。 AWS CodeStar 接続と AWS CodeStar 通知は、この中止の影響を受けません。

作業の追跡、コードの開発、アプリケーションの構築、テスト、デプロイを行う場合、Amazon CodeCatalyst は効率的な開始プロセスと、ソフトウェアプロジェクトを管理するための追加機能を提供します。Amazon の機能https://codecatalyst.aws/exploreおよび料金の詳細をご覧ください CodeCatalyst。

翻訳は機械翻訳により提供されています。提供された翻訳内容と英語版の間で齟齬、不一致または矛盾がある場合、英語版が優先します。

AWS CodeStar とは?

AWS CodeStar は、AWS でソフトウェア開発プロジェクトを作成、管理、および操作するクラウドベースのサービスです。AWS CodeStar プロジェクトを使用して、AWS でアプリケーションをすばやく開発、構築、およびデプロイすることができます。AWS CodeStar プロジェクトは、プロジェクトの開発ツールチェーンと AWS サービスを作成し、統合します。AWS CodeStar プロジェクトテンプレートの選択に応じて、そのツールチェーンにはソース管理、ビルド、デプロイ、仮想サーバーまたはサーバーレスリソースなどが含まれます。また、AWS CodeStar は、プロジェクトユーザー (チームメンバーと呼ばれる) に必要なアクセス許可を管理します。AWS CodeStar プロジェクトにチームメンバーとしてユーザーを追加することで、プロジェクトの所有者は各チームメンバーのロールに合ったプロジェクトやリソースへのアクセス権限を迅速かつ簡単に付与できます。

トピック

- AWS CodeStar の機能
- AWS CodeStar の使用を開始するには

AWS CodeStar の機能

AWS CodeStar を使用して、クラウドでのアプリケーション開発をセットアップし、集中管理された単一のダッシュボードから開発を管理できます。具体的な内容は以下のとおりです :

- ウェブアプリケーション、ウェブサービス、その他のプロジェクトテンプレートを使用して数分で AWS で新しいソフトウェアプロジェクトを始められます。AWS CodeStar にはさまざまなプロジェクトタイプとプログラミング言語のプロジェクトテンプレートが含まれています。AWS CodeStar はセットアップを処理するため、プロジェクトリソースがすべて連携するように設定されています。
- チームのプロジェクトアクセスを管理する: AWS CodeStar は、プロジェクトチームメンバーに、ツールやリソースにアクセスするために必要なロールを割り当てることができる一元化されたコンソールを提供します。これらのアクセス許可は、プロジェクトで使用されるすべての AWS サービスに自動的に適用されるため、複雑な IAM ポリシーを作成または管理する必要はありません。
- プロジェクトを 1 か所で視覚化、操作、および共同作業する: AWS CodeStar にはプロジェクトダッシュボードがあり、プロジェクト、ツールチェーン、重要なイベントの全体像を提供します。最新のコードコミットなどの最新のプロジェクトアクティビティをモニタリングし、コード変更のステータス、ビルド結果、およびデプロイをすべて同じウェブページから追跡できます。1 つのダッシュボードからプロジェクトの状況をモニタリングし、問題を精査できます。

- 必要なすべてのツールで反復処理をすばやく実行する: AWS CodeStar には、プロジェクトの統合開発ツールチェーンが含まれています。チームメンバーがコードをプッシュすると、変更が自動的にデプロイされます。課題追跡機能との統合により、チームメンバーは次に何をする必要があるかを把握することができます。チームと連携して、コードの配信のすべてのフェーズでより迅速かつ効率的に作業できます。

AWS CodeStar の使用を開始するには

AWS CodeStar の使用を開始するには

1. 「」のステップに従って、 の使用についてAWS CodeStar準備しますAWS CodeStarのセットアップ。
2. AWS CodeStar の使用の開始 チュートリアルのステップに従って、AWS CodeStar を試します。
3. AWS CodeStar プロジェクトにチームメンバーを追加する のステップに従って、他のデベロッパーとプロジェクトを共有します。
4. AWS CodeStar で IDE を使用する のステップに従って、利用したい IDE を統合します。

AWS CodeStarのセットアップ

の使用を開始する前に AWS CodeStar、次のステップを完了する必要があります。

トピック

- ステップ 1: アカウントを作成する
- ステップ 2: AWS CodeStar サービスロールを作成する
- ステップ 3: ユーザーの IAM アクセス許可を設定する
- ステップ 4: AWS CodeStar プロジェクトの Amazon EC2 キーペアの作成
- ステップ 5: AWS CodeStar コンソールを開く
- 次のステップ

ステップ 1: アカウントを作成する

にサインアップする AWS アカウント

がない場合は AWS アカウント、次の手順を実行して作成します。

にサインアップするには AWS アカウント

1. https://portal.aws.amazon.com/billing/signup を開きます。
2. オンラインの手順に従います。

   サインアップ手順の一環として、通話呼び出しを受け取り、電話キーパッドで検証コードを入力するように求められます。

   にサインアップすると AWS アカウント、 AWS アカウントのルートユーザーが作成されます。ルートユーザーには、アカウントのすべての AWS のサービス とリソースへのアクセス権があります。セキュリティのベストプラクティスとして、ユーザーに管理アクセスを割り当て、ルートユーザーのみを使用してルートユーザーアクセスが必要なタスクを実行してください。

AWS サインアッププロセスが完了すると、 から確認メールが送信されます。https://aws.amazon.com/ の [アカウント] をクリックして、いつでもアカウントの現在のアクティビティを表示し、アカウントを管理することができます。

管理アクセスを持つユーザーを作成する

にサインアップしたら AWS アカウント、 を保護し AWS アカウントのルートユーザー、 を有効にして AWS IAM Identity Center、日常的なタスクにルートユーザーを使用しないように管理ユーザーを作成します。

のセキュリティ保護 AWS アカウントのルートユーザー

1. ルートユーザーを選択し、 AWS アカウント E メールアドレスを入力して、アカウント所有者AWS Management Consoleとして にサインインします。次のページでパスワードを入力します。

   ルートユーザーを使用してサインインする方法については、AWS サインイン ユーザーガイドの「ルートユーザーとしてサインインする」を参照してください。
2. ルートユーザーの多要素認証 (MFA) を有効にします。

   手順については、「IAM ユーザーガイド」の AWS アカウント 「ルートユーザーの仮想 MFA デバイスを有効にする (コンソール)」を参照してください。

管理アクセスを持つユーザーを作成する

1. IAM アイデンティティセンターを有効にします。

   手順については、「AWS IAM Identity Center ユーザーガイド」の「AWS IAM Identity Centerの有効化」を参照してください。
2. IAM アイデンティティセンターで、ユーザーに管理アクセスを付与します。

   を ID ソース IAM アイデンティティセンターディレクトリ として使用する方法のチュートリアルについては、「 ユーザーガイド」の「デフォルト でユーザーアクセス IAM アイデンティティセンターディレクトリを設定するAWS IAM Identity Center 」を参照してください。

管理アクセス権を持つユーザーとしてサインインする

- IAM アイデンティティセンターのユーザーとしてサインインするには、IAM アイデンティティセンターのユーザーの作成時に E メールアドレスに送信されたサインイン URL を使用します。

  IAM Identity Center ユーザーを使用してサインインする方法については、「 AWS サインイン ユーザーガイド」の AWS 「 アクセスポータルへのサインイン」を参照してください。

追加のユーザーにアクセス権を割り当てる

1. IAM アイデンティティセンターで、最小特権のアクセス許可を適用するというベストプラクティスに従ったアクセス許可セットを作成します。

   手順については、「AWS IAM Identity Center ユーザーガイド」の「権限設定を作成する」を参照してください。

2. グループにユーザーを割り当て、そのグループにシングルサインオンアクセス権を割り当てます。

   手順については、「AWS IAM Identity Center ユーザーガイド」の「グループの参加」を参照してください。

ステップ 2: AWS CodeStar サービスロールを作成する

ユーザーに代わって AWS リソースを管理するアクセス AWS CodeStar 許可と IAM アクセス許可を付与するために使用されるサービスロールを作成します。サービスロールは一度作成するだけで済みます。

> **Important**
>
> このサービスロールを作成するには、 管理ユーザー (またはルートアカウント) としてサインインする必要があります。詳細については、「最初の IAM ユーザーとグループの作成」を参照してください。

1. https://console.aws.amazon.com/codestar/ で AWS CodeStar コンソールを開きます。
2. [Start project] (プロジェクトのスタート) を選択します。

   [Start project] (プロジェクトのスタート) が表示されず、プロジェクトリストページに誘導されている場合は、サービスロールが作成されています。

3. [Create service role] (サービスロールの作成) で、[Yes, create role] (はい、ロールを作成します) を選択します。
4. ウィザードを終了します。詳細については後程見ていきます。

ステップ 3: ユーザーの IAM アクセス許可を設定する

管理ユーザーに加えて、IAM ユーザー、フェデレーティッドユーザー、ルートユーザー、または引き受けたロール AWS CodeStar として を使用できます。IAM ユーザーとフェデレーティッドユーザーに対して何が AWS CodeStar できるかについては、「」を参照してくださいAWS CodeStar IAM ロール。

IAM ユーザーを設定していない場合は、「IAM ユーザー」を参照してください。

アクセス権限を付与するには、ユーザー、グループ、またはロールにアクセス許可を追加します。

- のユーザーとグループ AWS IAM Identity Center：

  アクセス許可セットを作成します。「AWS IAM Identity Center ユーザーガイド」の「権限設定を作成する」の手順に従ってください。

- IAM 内で、ID プロバイダーによって管理されているユーザー:

  ID フェデレーションのロールを作成します。詳細については、「IAM ユーザーガイド」の「サードパーティー ID プロバイダー (フェデレーション) 用のロールの作成」を参照してください。

- IAM ユーザー:
  - ユーザーが担当できるロールを作成します。手順については、「IAM ユーザーガイド」の「IAM ユーザー用ロールの作成」を参照してください。
  - (お奨めできない方法) ポリシーをユーザーに直接アタッチするか、ユーザーをユーザーグループに追加する。詳細については、「IAM ユーザーガイド」の「ユーザー (コンソール) へのアクセス権限の追加」を参照してください。

ステップ 4: AWS CodeStar プロジェクトの Amazon EC2 キーペアの作成

多くの AWS CodeStar プロジェクトは、Amazon EC2 インスタンス AWS Elastic Beanstalk にコードをデプロイするために AWS CodeDeploy または を使用します。プロジェクトに関連付けられている Amazon EC2 インスタンスにアクセスするには、IAM ユーザーの Amazon EC2 キーペアを作成します。IAM ユーザーは、Amazon EC2 キーを作成して管理するアクセス許可を持っている必要があります (例：`ec2:CreateKeyPair` アクションと `ec2:ImportKeyPair` アクションのアクセス許可)。詳細については、「Amazon EC2 のキーペア」 を参照してください。

ステップ 5: AWS CodeStar コンソールを開く

にサインインし AWS Management Console、https://console.aws.amazon.com/codestar/ で AWS CodeStar コンソールを開きます。

次のステップ

これで、セットアップが完了しました。の使用を開始するには、 AWS CodeStar「」を参照してくださいAWS CodeStar の使用の開始。

AWS CodeStar の使用の開始

このチュートリアルでは、AWS CodeStar を使用してウェブアプリケーションを作成します。このプロジェクトには、ソースリポジトリのサンプルコード、継続的なデプロイツールチェーン、および、プロジェクトを表示およびモニタリングできるプロジェクトダッシュボードが含まれています。

このステップでは、次のことを行います：

- AWS CodeStar でプロジェクトを作成します。
- プロジェクトを調査します。
- コード変更をコミットします。
- コードの変更が自動的にデプロイされるのを確認します。
- プロジェクトで作業する他の人を追加します。
- 不要になったプロジェクトリソースをクリーンアップします。

> **Note**
>
> まだ行っていない場合は、まず「AWS CodeStarのセットアップ」のステップ (例: ステップ 2: AWS CodeStar サービスロールを作成する) を完了します。IAM の管理者ユーザーであるアカウントを使用してサインインする必要があります。プロジェクトを作成するには、**AWSCodeStarFullAccess** ポリシーを持つ IAM ユーザーを使用して、AWS Management Console にサインインする必要があります。

トピック

- ステップ 1: AWS CodeStar プロジェクトを作成する
- ステップ 2: AWS CodeStar ユーザープロファイルの表示情報を追加する
- ステップ 3: プロジェクトを表示する
- ステップ 4: 変更をコミットする
- ステップ 5: チームメンバーの追加
- ステップ 6: クリーンアップ
- ステップ 7: 本番稼働環境のプロジェクトの準備
- 次のステップ
- チュートリアル: AWS CodeStar でサーバーレスプロジェクトを作成および管理する

- チュートリアル: AWS CLI を使用して AWS CodeStar にプロジェクトを作成する
- チュートリアル: AWS CodeStarで Alexa スキルプロジェクトを作成する
- チュートリアル:GitHub ソースリポジトリを使用してプロジェクトを作成する

ステップ 1: AWS CodeStar プロジェクトを作成する

このステップでは、ウェブアプリケーション用の JavaScript (Node.js) ソフトウェア開発プロジェクトを作成します。AWS CodeStar プロジェクトテンプレートを使用してプロジェクトを作成します。

> **Note**
>
> このチュートリアルで使用する AWS CodeStar プロジェクトテンプレートでは、次のオプションを使用します。
>
> - [Application category] (アプリケーションカテゴリー) : ウェブアプリケーション
> - [Programming language] (プログラム言語) : Node.js
> - [AWS サービス] : Amazon EC2
>
> 他のオプションを選択した場合は、このチュートリアルに記載されている内容と一致しない場合があります。

AWS CodeStar でプロジェクトを作成するには

1. AWS Management Consoleにサインインして、https://console.aws.amazon.com/codestar/ で、AWS CodeStar コンソールを開きます。

   プロジェクトとそのリソースを作成する AWS リージョンにサインインしていることを確認してください。例えば、米国東部 (オハイオ) でプロジェクトを作成するには、AWS リージョンが選択されていることを確認します。AWS CodeStar が利用可能な AWS リージョンの詳細については、「AWS 全般のリファレンス」の「リージョンとエンドポイント」を参照してください。
2. [AWS CodeStar] ページで、[プロジェクトの作成] を選択します。
3. [プロジェクトのテンプレートを選択する] ページで、AWS CodeStar プロジェクトテンプレートのリストからプロジェクトの種類を選択します。フィルタバーを使用して選択を絞り込むことができます。例えば、Amazon EC2 インスタンスにデプロイされる Node.js で記述され

たウェブアプリケーションプロジェクトの場合は、[Web application] (ウェブアプリケーション)、[Node.js] の順に選択し、[Amazon EC2] チェックボックスをオンにします。次に、オプションのセットで使用可能なテンプレートから選択します。

詳細については、「AWS CodeStar プロジェクトテンプレート」を参照してください。

4. [Next] (次へ) を選択します。
5. [プロジェクト名] で、*My First Project* などのプロジェクト名を入力します。[Project ID] (プロジェクト ID) では、プロジェクトの ID はこのプロジェクト名から派生しますが、15 文字の制限があります。

   例えば、*My First Project* という名前のプロジェクトのデフォルト ID は *my-first-projec* です。このプロジェクト ID は、プロジェクトに関連付けられているすべてのリソースの名前のベースです。AWS CodeStar は、このプロジェクト ID をコードリポジトリの URL の一部として使用するほか、IAM の関連するセキュリティアクセスロールとポリシーの名前にも使用します。プロジェクトの作成後、プロジェクト ID は変更できません。プロジェクトを作成する前にプロジェクト ID を編集するには、[Project ID] (プロジェクト ID) で、使用する ID を入力します。

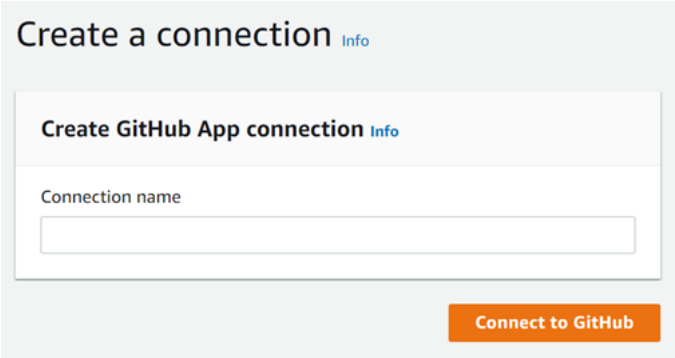
   プロジェクト名とプロジェクト ID の制限の詳細については、AWS CodeStar における制限 を参照してください。

   > **Note**
   >
   > AWS リージョン内の AWS アカウントでは、プロジェクト ID が一意である必要があります。

6. リポジトリプロバイダー、AWS CodeCommit または [GitHub] を選択します。
7. AWS CodeCommit を選択した場合は、[Repository name] (リポジトリ名) で、デフォルトの AWS CodeCommit リポジトリ名を受け入れるか、別の名前を入力します。ステップ 9 に進みます。
8. [GitHub] を選択した場合、接続リソースを選択または作成する必要があります。既存の接続がある場合は、検索フィールドで選択します。それ以外の場合は、ここで新しい接続を作成します。[Connect to GitHub] (GitHub に接続) を選択します。

   [Create a connection] (接続の作成) ページが表示されます。

> **Note**
>
> 接続を作成するには、GitHub アカウントが必要です。組織の接続を作成する場合は、組織の所有者である必要があります。



a. [GitHub App 接続の作成] で、[接続名] テキスト入力フィールドに接続名を入力します。[Connect to GitHub] (GitHub に接続) を選択します。

   [Connect to GitHub] (GitHub に接続) ページが表示され、[GitHub Apps] フィールドが表示されます。

b. [GitHub Apps] で、アプリケーションのインストールを選択するか、[Install a new app] (新しいアプリケーションをインストールする) を選択してアプリケーションを作成します。

   > **Note**
   >
   > 特定のプロバイダーへのすべての接続に対してアプリを 1 つインストールします。GitHub app 用 AWS コネクターをすでにインストールしている場合は、これを選択してこのステップをスキップしてください。

c. [Install AWS Connector for GitHub] ページで、アプリをインストールするアカウントを選択します。

> **Note**
>
> アプリケーションをインストール済みである場合は、[Configure] (設定) を選択してアプリのインストールの変更ページに進むか、戻るボタンでコンソールに戻ることができます。

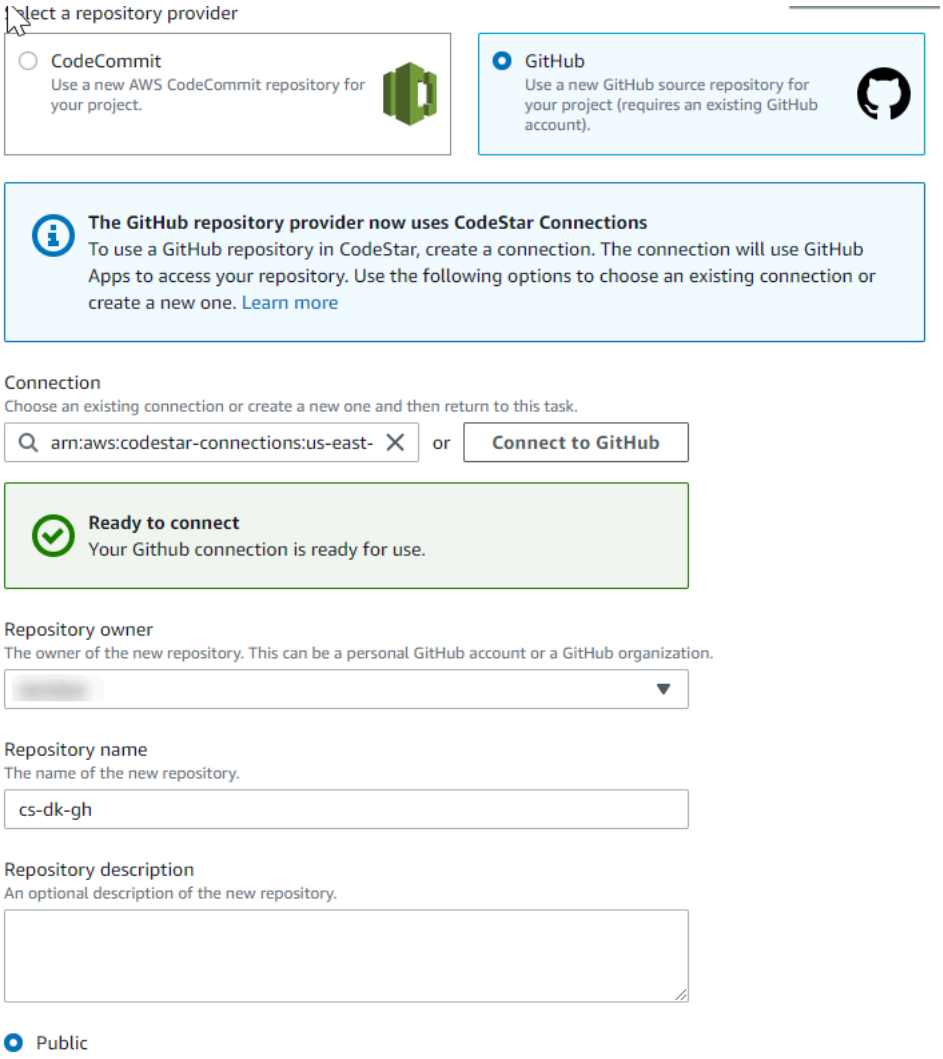
d. [Confirm password to continue] (パスワードを確認して続行) ページが表示される場合、GitHub パスワードを入力し、[Sign in] (サインイン) を選択します。

e. [GitHub 用 AWS コネクターのインストール] ページで、デフォルトのままで、[インストール] を選択します。

f. [GitHub へ接続] ページで、新規インストールのインストール ID が [GitHub Apps] テキスト入力フィールドに表示されます。

   接続が作成された後、CodeStar の [create project] (プロジェクトを作成) ページで、[Ready to connect] (接続準備完了) メッセージが表示されます。

> **Note**
>
> [Developer Tools] (デベロッパーツール) コンソールの[Settings] (設定) で接続を表示できます。詳細については、[Getting started with connections] (接続入門ガイド) を参照してください。



g. [Repository owner] (リポジトリ所有者) で、GitHub 組織または個人用 GitHub アカウントを選択します。

h. [Repository name] (リポジトリ名) で、デフォルトの GitHub リポジトリ名を受け入れるか、別の名前を入力します。

i. [Public] (公開) または[Private] (プライベート) を選択します。

> **Note**
>
> AWS Cloud9 を開発環境として使用する場合は、[Public] (公開) を選択する必要があります。

j. (オプション) [Repository description] (リポジトリの説明) に、GitHub リポジトリの説明を入力します。

> **Note**
>
> Alexa スキルプロジェクトテンプレートを選択する場合は、Amazon 開発者アカウントを接続する必要があります。Alexa スキルプロジェクトの使用方法の詳細については、「チュートリアル: AWS CodeStarで Alexa スキルプロジェクトを作成する」を参照してください。

9. プロジェクトが Amazon EC2 インスタンスにデプロイされ、変更を加える場合は、[Amazon EC2 Configuration] (Amazon EC2 の設定) で Amazon EC2 インスタンスを設定します。例えば、プロジェクトの使用可能なインスタンスタイプから選択できます。

> **Note**
>
> 異なる Amazon EC2 インスタンスタイプは、異なるレベルのコンピューティングパワーを提供し、異なる関連費用が発生する可能性があります。詳細については、[Amazon EC2 Instance Types] (Amazon EC2 インスタンスタイプ) と [Amazon EC2 Pricing] (Amazon EC2 の料金) を参照してください。
> 複数の仮想プライベートクラウド (VPC) または複数のサブネットが Amazon 仮想プライベートクラウド で作成されている場合は、使用する VPC とサブネットを選択することもできます。ただし、ハードウェア専有インスタンスでサポートされていない Amazon EC2 インスタンスタイプを選択した場合は、インスタンスのテナンシーが [Dedicated] (専有) に設定されている VPC を選択することはできません。
> 詳細については、[What Is Amazon VPC?] (Amazon VPC とは) および [Dedicated Instance Basics] (ハードウェア専有インスタンスの基礎) を参照してください。

   [Key pair] (キーペア) で、ステップ 4: AWS CodeStar プロジェクトの Amazon EC2 キーペアの作成 で作成した Amazon EC2 キーペアを選択します。[I acknowledge that I have access to the private key file] (私はプライベートキーファイルへのアクセス権があることを認めます) を選択します。

10. [Next] (次へ) を選択します。
11. リソースと設定の詳細を確認します。
12. [Next] (次へ) または [Create project] (プロジェクトの作成) を選択します。(表示される選択はプロジェクトテンプレートによって異なります。)

プロジェクト (リポジトリを含む) の作成には数分かかる場合があります。

13. プロジェクトのリポジトリの作成後は、[Repository] (リポジトリ) ページを使用して、リポジトリへのアクセス権を設定します。[Next steps] (次のステップ) のリンクを使用して、IDE を設定、課題追跡を設定、チームメンバーをプロジェクトに追加できます。

ステップ 2: AWS CodeStar ユーザープロファイルの表示情報を追加する

プロジェクトを作成すると、所有者としてプロジェクトチームに追加されます。今回、AWS CodeStar を初めて使用する場合は、以下の入力を求められます。

- 他のユーザーに表示する表示名。
- 他のユーザーに表示する E メールアドレス。

この情報は、AWS CodeStar ユーザープロファイルで使用されます。ユーザープロファイルはプロジェクト固有ではありませんが、AWS リージョンに限定されています。ユーザープロファイルは、ユーザーがプロジェクトに属している AWS リージョンごとに作成する必要があります。希望に応じて、プロファイルごとに異なる情報を含めることができます。

ユーザー名と E メールアドレスを入力し、[Next] (次へ) を選択します。

> **Note**
>
> このユーザー名と E メールアドレスは AWS CodeStar ユーザープロファイルで使用されます。プロジェクトで AWS 以外のリソース (例: GitHub リポジトリや Atlassian JIRA の課題) を使用している場合、これらのリソースプロバイダーには、ユーザー名と E メールアドレスが異なる独自のユーザープロファイルが設定されている可能性があります。詳細については、リソースプロバイダのドキュメントを参照してください。

ステップ 3: プロジェクトを表示する

[AWS CodeStar プロジェクト] ページは、プロジェクトへの最新のコミット、継続的な配信パイプラインの状態、インスタンスのパフォーマンスなど、プロジェクトリソースのステータスをチームに表示する場所です。これらのリソースの詳細については、ナビゲーションバーから対応するページを選択してください。

新しいプロジェクトでは、ナビゲーションバーには次のページが表示されます：

- [Overview] (概要) ページには、プロジェクトのアクティビティ、プロジェクトリソース、およびプロジェクトの README コンテンツに関する情報が表示されます。
- [IDE]ページでは、プロジェクトを統合開発環境 (IDE) に接続して、ソースコードの変更を修正、テスト、プッシュします。これには、GitHub と AWS CodeCommit リポジトリの両方に IDE を設定するための手順や AWS Cloud9 環境に関する情報が含まれています。
- [Repository] (リポジトリ) ページには、名前、プロバイダー、最終変更日時、クローン URL など、リポジトリの詳細が表示されます。また、最新のコミットに関する情報を表示し、プルリクエストを作成することもできます。
- [Pipeline] (パイプライン) ページには、パイプラインに関する CI/CD 情報が表示されます。名前、最新のアクション、ステータスなどのパイプラインの詳細を表示できます。パイプラインの履歴を表示し、変更をリリースできます。また、パイプラインの個々のステップのステータスを表示することもできます。
- [Monitoring] (モニタリング) ページには、Amazon EC2 または AWS Lambda プロジェクトの設定に応じたメトリクスが表示されます。たとえば、AWS Elastic Beanstalk またはパイプラインの CodeDeploy リソースによってデプロイされた Amazon EC2 インスタンスの CPU 使用率を表示します。AWS Lambda を使用するプロジェクトでは、 Lambda 関数の呼び出しとエラーのメトリクスが表示されます。この情報は時間単位で表示されます。このチュートリアルで提案された AWS CodeStar プロジェクトテンプレートを使用した場合、アプリケーションが最初にそのインスタンスにデプロイされる際に、アクティビティが著しく増加しているのが分かるはずです。モニタリングを更新してインスタンスヘルスの変化を表示すると、問題やより多くのリソースの必要などを識別するのに役立ちます。
- [課題] ページは、AWS CodeStar プロジェクトと Atlassian JIRA プロジェクトを統合するためのものです。このタイルを設定すると、お客様やプロジェクトチームはプロジェクトダッシュボードから JIRA の課題を追跡できます。

コンソールの左側のナビゲーションペインでは、[Project] (プロジェクト)、[Team] (チーム)、[Settings] (設定) ページを移動できます。

ステップ 4: 変更をコミットする

まず、プロジェクトに含まれていたサンプルアプリケーションを表示します。プロジェクトナビゲーションのどこからでも [View application] (アプリケーションの表示) を選択して、アプリケーションの外観を確認します。新しいウインドウまたはブラウザのタブに、サンプルウェブアプリケーション

が表示されます。AWS CodeStar がビルドおよびデプロイされたプロジェクトの例を以下に示します。

コードを確認するには、ナビゲーションバーで [Repository] (リポジトリ) を選択します。[Repository name] (リポジトリ名) の下にあるリンクを選択すると、プロジェクトのリポジトリが新しいタブまたはウィンドウで開きます。リポジトリの readme ファイル (`README.md`) の内容を読み、それらのファイルの内容を参照します。

このステップでは、コードを変更してその変更をリポジトリにプッシュします。これにはいくつかの方法があります：

- プロジェクトのコードが CodeCommit または GitHub リポジトリに保存されている場合は、AWS Cloud9 を使用してウェブブラウザからコードを直接操作することができます。ツールのインストールは必要ありません。詳細については、「プロジェクトの AWS Cloud9 環境を作成する」を参照してください。
- プロジェクトのコードが CodeCommit リポジトリに保存され、Visual Studio または Eclipse がインストールされている場合は、AWS Toolkit for Visual Studio または AWS Toolkit for Eclipse を使用してより簡単にコードに接続できます。詳細については、「AWS CodeStar で IDE を使用する」を参照してください。Visual Studio または Eclipse がない場合は、Git クライアントをインストールし、このステップの後のステップに従ってください。
- プロジェクトのコードが GitHub リポジトリに保存されている場合は、IDE のツールを使用して GitHub に接続することができます。
  - Visual Studio では、GitHub Extension for Visual Studio などのツールを使用することができます。詳細については、GitHub Extension for Visual Studio ウェブサイトの [Overview] (概要) ページ、および GitHub ウェブサイトの [Getting Started with GitHub for Visual Studio] (GitHub for Visual Studio 入門) を参照してください。
  - Eclipse の場合は、EGit for Eclipse などのツールを使用することができます。詳細については、EGit ウェブサイトの [EGit Documentation] (EGit ドキュメント) を参照してください。
  - その他の IDE については、IDE のドキュメントを参照してください。
- 他の種類のコードリポジトリについては、リポジトリプロバイダのドキュメントを参照してください。

次の手順では、サンプルの基本的な変更を行う方法について説明します。

変更をコミットするようコンピュータを設定するには (IAM ユーザー)

> **Note**
>
> この手順では、プロジェクトのコードが CodeCommit リポジトリに保存されていることを想定しています。他の種類のコードリポジトリについては、リポジトリプロバイダのドキュメントを参照してください。次の手順「プロジェクトリポジトリのクローンを作成して変更するには」に進みます。
> コードが CodeCommit に保存されており、既に CodeCommit を使用しているか、AWS CodeStar コンソールを使用してプロジェクトの AWS Cloud9 開発環境を作成する場合は、これ以上設定は必要ありません。次の手順「プロジェクトリポジトリのクローンを作成して変更するには」に進みます。

1. ローカルコンピュータに Git をインストールします。
2. AWS Management Console にサインインして、https://console.aws.amazon.com/iam/ で、IAM コンソールを開きます。

   CodeCommit にある AWS CodeStar プロジェクトリポジトリへの接続に Git 認証情報を使用する IAM ユーザーとしてサインインします。
3. IAM コンソールのナビゲーションペインで [Users] (ユーザー) を選択し、ユーザーのリストから自分の IAM ユーザーを選択します。
4. [User details] (ユーザーの詳細) ページで、[Security Credentials] (認証情報) タブを選択し、[HTTPS Git credentials for CodeCommit] (CodeCommit の HTTPS Git 認証情報) で、[Generate] (生成) を選択します。

   > **Note**
   >
   > Git 認証情報として自身のサインイン認証情報を選択することはできません。詳細については、[Use Git Credentials and HTTPS with CodeCommit] (CodeCommit で Git 認証情報と HTTPS を使用する) を参照してください。
5. IAM が生成したサインイン認証情報をコピーします。[Show] (表示) を選択しこの情報をローカルコンピュータの安全なファイルにコピーして貼り付ける、または、[Download credentials] (認証情報をダウンロード) を選択してこの情報を .CSV ファイルとしてダウンロードします。CodeCommit に接続するには、この情報が必要です。

   認証情報を保存したら、[Close] を選択します。

> **Important**
>
> これは、サインイン認証情報を保存する唯一の機会です。パスワードを保存しないと、IAM コンソールからユーザー名をコピーすることはできますが、パスワードを参照することはできません。パスワードをリセットして保存する必要があります。

変更をコミットするようコンピュータを設定するには (フェデレーティッドユーザー)

コンソールを使用してリポジトリにファイルをアップロードするか、Git を使用してローカルコンピュータから接続することができます。フェデレーティッドアクセスを使用している場合は、以下のステップに従い、Git を使用して、ローカルコンピュータからリポジトリに接続してクローンを作成します。

> **Note**
>
> この手順では、プロジェクトのコードが CodeCommit リポジトリに保存されていることを想定しています。他の種類のコードリポジトリについては、リポジトリプロバイダのドキュメントを参照してください。次の手順「プロジェクトリポジトリのクローンを作成して変更するには」に進みます。

1. ローカルコンピュータに Git をインストールします。
2. AWS CLIをインストールします。
3. フェデレーティッドユーザー用の一時的セキュリティ認証情報を設定します。詳細については、[Temporary Access to CodeCommit Repositories] (CodeCommit リポジトリへの一時アクセス) を参照してください。一時認証情報は、次で構成されます：

   - AWS アクセスキー
   - AWS シークレットキー
   - セッショントークン

   一時的なセキュリティ認証情報の詳細については、「GetFederationToken のアクセス許可」を参照してください。
4. AWS CLI の認証情報ヘルパーを使用してリポジトリに接続します。詳細については、「AWS CLI 認証情報ヘルパーを使用した、Linux、macOS、または UNIX での AWS CodeCommit リポ

ジトリへの HTTPS 接続のセットアップ手順」または「AWS CLI 認証情報ヘルパーを使用して Windows で AWS CodeCommit リポジトリへの HTTPS 接続をセットアップする手順」を参照してください。

5. 次の例では、CodeCommit リポジトリに接続し、コミットをプッシュする方法を示します。

例: プロジェクトリポジトリのクローンを作成して変更するには

> **Note**
>
> この手順では、プロジェクトのコードリポジトリをコンピュータに複製し、プロジェクトの `index.html` ファイルを変更して、その変更をリモートリポジトリにプッシュする方法を説明します。この手順では、CodeCommit プロジェクトのコードが リポジトリに保存されていることと、コマンドラインから Git クライアントを使用していることを前提としています。他の種類のコードリポジトリまたはツールでリポジトリを複製し、ファイルを変更してコードをプッシュする方法については、プロバイダのドキュメントを参照してください。

1. AWS CodeStar コンソールを使用して、プロジェクトの AWS Cloud9 開発環境を作成している場合は、開発環境を開き、手順のステップ 3 に進みます。開発環境を開くには、「プロジェクトの AWS Cloud9 環境を開く」を参照してください。

   AWS CodeStar コンソールでプロジェクトを開き、ナビゲーションバーで、[リポジトリ] を選択します。[Clone URL] (URLのクローンを作成) で、CodeCommit 用に設定した接続タイプのプロトコルを選択して、リンクをコピーします。例えば、CodeCommit 用の Git 認証情報の設定の手順に従っている場合は、[HTTPS] を選択します。

2. ローカルコンピュータで、端末またはコマンドラインウィンドウを開き、一時ディレクトリにディレクトリを変更します。[git clone] コマンドを実行して、リポジトリのクローンをコンピュータに作成します。コピーしたリンクを貼り付けます。例えば、CodeCommit では HTTPS を使用します :

   ```
   git clone https://git-codecommit.us-east-2.amazonaws.com/v1/repos/my-first-projec
   ```

   初めて接続すると、リポジトリのサインイン認証情報の入力を求められます。CodeCommitでは、前の手順でダウンロードした Git サインイン認証情報を入力します。

3. コンピュータのクローンしたディレクトリに移動し、内容を参照します。

4. index.html ファイル (パブリックフォルダ内) を開き、ファイルを変更します。たとえば、<H2> タグの後に次のような段落を追加します：

```
<P>Hello, world!</P>
```

ファイルを保存します。

5. 端末またはコマンドプロンプトで、変更したファイルを追加し、変更をコミットし、プッシュします。

```
git add index.html
git commit -m "Making my first change to the web app"
git push
```

6. [Repository] (リポジトリ) ページで、進行中の変更を表示します。そのリポジトリのコミット履歴が、コミットメッセージを含め、コミットで更新されているのを確認できます。[Pipeline] (パイプライン) ページでは、パイプラインがリポジトリへの変更を取得し、構築およびデプロイをスタートするのを確認できます。ウェブアプリケーションのデプロイ後、[View application] (アプリケーションの表示) を選択して変更内容を表示します。

> **Note**
>
> いずれかの [Pipeline](パイプライン) ステージで[Failed] (失敗しました) が表示される場合は、以下のトラブルシューティングのヘルプを参照してください：
>
> - [Source] (ソース) ステージの場合は、AWS CodeCommit ユーザーガイドのトラブルシューティング AWS CodeCommitを参照してください。
> - [Build] (ビルド) ステージの場合は、AWS CodeBuild ユーザーガイドのトラブルシューティング AWS CodeBuildを参照してください。
> - [Deploy] (デプロイ) ステージの場合は、AWS CloudFormation ユーザーガイドのトラブルシューティング AWS CloudFormationを参照してください。
> - その他の問題については、「トラブルシューティング AWS CodeStar」を参照してください。

ステップ 5: チームメンバーの追加

AWS CodeStar プロジェクトはいずれも、3 つの AWS CodeStar ロールで既に設定されています。各ロールは、プロジェクトとそのリソースへの独自のレベルのアクセスを提供します。

- [Owner] (所有者) : チームメンバーの追加と削除、プロジェクトダッシュボードの変更、プロジェクトの削除を行うことができます。
- [Contributor] (寄稿者) : コードが CodeCommit に保存されている場合は、プロジェクトダッシュボードを変更してコードを投稿できますが、チームメンバーの追加や削除、プロジェクトの削除を行うことはできません。これは、AWS CodeStar プロジェクトで、ほとんどのチームメンバーに対して選択すべきロールです。
- [Viewer] (閲覧者) : コードが CodeCommit に保存されている場合は、プロジェクトダッシュボード、プロジェクトコード、およびプロジェクトの状態を表示できますが、プロジェクトダッシュボードからタイルを移動、追加、または削除することはできません。

> **Important**
>
> プロジェクトで AWS 以外のリソース (例: GitHub リポジトリ、または Atlassian JIRA の問題) が使用されている場合、それらのリソースへのアクセスは AWS CodeStar ではなくリソースプロバイダーによって制御されます。詳細については、リソースプロバイダのドキュメントを参照してください。
>
> AWS CodeStar プロジェクトにアクセスできるユーザーは、AWS CodeStar コンソールを使用して AWS 以外のリソースにアクセスできますが、そのプロジェクトに関連している可能性があります。
>
> AWS CodeStar では、プロジェクトチームのメンバーが、プロジェクトの関連する AWS Cloud9 開発環境に参加することは許可されていません。チームメンバーによる共有環境への参加を許可するには、「AWS Cloud9 環境をプロジェクトチームメンバーと共有する」を参照してください。

チームとプロジェクトロールの詳細については、「AWS CodeStar のチームで作業する」 を参照してください。

チームメンバーを AWS CodeStar プロジェクトに追加するには (コンソール)

1. https://console.aws.amazon.com/codestar/ で AWS CodeStar コンソールを開きます。
2. ナビゲーションペインから、[Projects] (プロジェクト) を選択し、プロジェクトを選択します。

3. プロジェクトのサイドナビゲーションペインで、[Team] (チーム) を選択します。
4. [Team members] (チームメンバー) ページで、[Add team member] (チームメンバーの追加) を選択します。
5. [Choose user] (ユーザーを選択) で、次のいずれかを実行します：

   - 追加するメンバーの IAM ユーザーが既に存在する場合は、その IAM ユーザーをリストから選択します。

     > **Note**
     >
     > 別の AWS CodeStar プロジェクトに既に追加されているユーザーは、[既存の AWS CodeStar ユーザー] リストに表示されます。

     [プロジェクトロール] で、このユーザーの AWS CodeStar ロール (所有者、寄稿者、閲覧者) を選択します。これは AWS CodeStar プロジェクトレベルのロールで、プロジェクトの所有者によってのみ変更できます。IAM ユーザーにロールが適用されると、AWS CodeStar プロジェクトリソースにアクセスするために必要なアクセス許可がすべて付与されます。コードが IAM の CodeCommit に保存されている場合の Git 認証情報の作成と管理に必要なポリシー、または IAM でユーザーの Amazon EC2 SSH キーをアップロードするのに必要なポリシーが適用されます。

     > **Important**
     >
     > 該当のユーザーとしてコンソールにサインインしていない限り、IAM ユーザーの表示名または E メール情報を入力または変更することはできません。詳細については、「AWS CodeStar ユーザープロファイルの表示情報を管理する 」を参照してください。

     [Add team member] (チームメンバーの追加) を選択します。

   - プロジェクトに追加する人物の IAM ユーザーが存在しない場合は、[Create new IAM user] (新規 IAM ユーザーを作成) を選択します。新規 IAM ユーザーを作成できる IAM コンソールにリダイレクトされます。詳細については「IAM ユーザーガイド」の「AWS アカウントでの IAM ユーザーの作成」を参照してください。IAM ユーザーを作成後、AWS CodeStar コンソールに戻り、ユーザーのリストを更新し、ドロップダウンリストから作成した IAM ユーザーを選

択します。この新しいユーザーに適用する AWS CodeStar 表示名、E メールアドレス、およびプロジェクトロールを入力し、[チームメンバーを追加] を選択します。

> **Note**
>
> 管理しやすいように、少なくとも 1 人のユーザーにプロジェクトの所有者ロールを割り当てます。

6. 新しいチームメンバーに、次の情報を送信します：

   - AWS CodeStar プロジェクトのための接続情報。
   - ソースコードが CodeCommit に保存されている場合、ローカルコンピュータから CodeCommit リポジトリに [instructions for setting up access with Git credentials] (Git 認証情報でアクセスを設定する手順)。
   - AWS CodeStar ユーザープロファイルの操作方法 で説明されているように、ユーザーが表示名、E メールアドレス、公開 Amazon EC2 SSH キーを管理する方法についての情報。
   - ユーザーが AWS を使用するのは初めてで、そのユーザーのために IAM ユーザーを作成した場合のワンタイムパスワードと接続情報。このパスワードはユーザーの初回サインイン時に失効します。ユーザーは新しいパスワードを選択する必要があります。

ステップ 6: クリーンアップ

お疲れ様でした。チュートリアルを完了しました。このプロジェクトとリソースの使用を継続しない場合は、今後お客様の AWS アカウントに課金されないように削除します。

AWS CodeStar でプロジェクトを削除するには

1. https://console.aws.amazon.com/codestar/ で AWS CodeStar コンソールを開きます。
2. ナビゲーションペインで、[Projects] (プロジェクト) を選択します。
3. 削除するプロジェクトを選択して、[Delete] (削除) を選択します。

   または、プロジェクトを開き、コンソールの左側のナビゲーションペインから [Settings] (設定) を選択します。[Project details] (プロジェクトの詳細)ページで、[Delete project] (プロジェクトの削除) を選択します。

4. [Delete confirmation page] (削除の確認ページ) で、[delete] (削除) と入力します。プロジェクトリソースを削除する場合、[Delete resources] (リソースの削除) を選択したままにします。[Delete] (削除) を選択します。

   プロジェクトの削除には数分かかる場合があります。削除された後、プロジェクトは AWS CodeStar コンソールのプロジェクトの一覧に表示されなくなります。

   > **Important**
   >
   > プロジェクトで AWS 以外のリソース (例: GitHub リポジトリ、Atlassian JIRAの課題) を使用している場合は、チェックボックスがオンになっていてもそれらのリソースは削除されません。
   > AWS CodeStar 管理ポリシーが、IAM ユーザーではないロールに手動でアタッチされている場合でも、プロジェクトを削除することはできません。プロジェクトの管理ポリシーをフェデレーティッドユーザーのロールに添付している場合は、プロジェクトを削除する前にポリシーをデタッチする必要があります。詳細については、「???」を参照してください。

ステップ 7: 本番稼働環境のプロジェクトの準備

プロジェクトを作成したら、コードを作成、テスト、およびデプロイすることができます。本番稼働環境でプロジェクトを管理するには、次の考慮事項を確認してください。

- 定期的にパッチを適用し、アプリケーションで使用される依存関係のセキュリティベストプラクティスを確認してください。詳細については、「AWS CodeStar リソースで使用するセキュリティのベストプラクティス」を参照してください。
- プロジェクトのプログラミング言語で提案された環境設定を定期的にモニタリングします。

次のステップ

AWS CodeStar の理解に役立つリソースを以下に示します。

- チュートリアル: AWS CodeStar でサーバーレスプロジェクトを作成および管理する は、AWS Lambda のロジックを使用してウェブサービスを作成、デプロイし、 Amazon API Gateway の API で呼び出すことができるプロジェクトを使用しています。

- AWS CodeStar プロジェクトテンプレート で、作成できる他の種類のプロジェクトについて説明します。
- 他のユーザーによるプロジェクトの参加を許可する方法については、「AWS CodeStar のチームで作業する」を参照してください。

チュートリアル: AWS CodeStar でサーバーレスプロジェクトを作成および管理する

このチュートリアルでは、AWS CodeStar を使用して AWS サーバーレスアプリケーションモデル (AWS SAM) を使用して AWS Lambda でホストされるウェブサービスの AWS リソースを作成および管理するプロジェクトを作成します。

AWS CodeStar は、AWS CloudFormation に依存する AWS SAM を使用して、Amazon API Gateway API、AWS Lambda 関数、Amazon DynamoDB テーブルを含む、サポートされた AWS リソースを簡単に作成および管理する方法を提供します。(このプロジェクトでは Amazon DynamoDB テーブルを使用しません)。

詳細については、GitHub の 「AWS サーバーレスアプリケーションモデル (AWS SAM)」 を参照してください。

前提条件:「AWS CodeStarのセットアップ」の手順を完了すること。

> **Note**
>
> このチュートリアルに関するコスト (例: AWS CodeStar が使用する AWS サービスのコスト) については、AWS アカウントに請求される場合があります。詳細については、「AWS CodeStar の料金」を参照してください。

トピック

- 概要
- ステップ 1: プロジェクトを作成する
- ステップ 2: プロジェクトリソースを調べる
- ステップ 3: ウェブサービスをテストする
- ステップ 4: プロジェクトコードを編集するためのローカルワークステーションの設定
- ステップ 5: ウェブサービスにロジックを追加する

- ステップ 6: 拡張ウェブサービスをテストする
- ステップ 7: ウェブサービスにユニットテストを追加する
- ステップ 8: ユニットテストの結果を表示する
- ステップ 9: クリーンアップ
- 次のステップ

概要

このチュートリアルでは、次の作業を行います：

1. AWS CodeStar を使って、AWS SAM を使用して Python ベースのウェブサービスを構築およびデプロイするプロジェクトを作成します。このウェブサービスは AWS Lambda でホストされており、Amazon API Gateway からアクセスできます。
2. プロジェクトの主なリソースは次のとおりです：
   - AWS CodeCommit リポジトリにはプロジェクトのソースコードが保存されます。このソースコードには、ウェブサービスのロジックが含まれ、関連する AWS リソースが定義されます。
   - AWS CodePipeline パイプラインは、ソースコードの作成を自動化します。このパイプラインは AWS SAM を使用して AWS Lambda に関数を作成しデプロイして、関連する API を Amazon API Gateway に作成し、API を関数に接続します。
   - AWS Lambda にデプロイされる関数。
   - Amazon API Gateway で作成される API。
3. ウェブサービスをテストして、AWS CodeStar が正常にウェブサービスを構築し、デプロイすることを確認します。
4. プロジェクトのソースコードを使用するようにローカルワークステーションを設定します。
5. ローカルワークステーションを使用してプロジェクトのソースコードを変更します。関数をプロジェクトに追加し、変更をソースコードにプッシュすると、AWS CodeStar はウェブサービスの再構築と再デプロイを行います。
6. ウェブサービスを再度テストして、AWS CodeStar が正常に再構築および再デプロイすることを確認します。
7. ローカルワークステーションを使用してユニットテストを作成し、手動テストの一部を自動化されたテストに置き換えます。ユニットテストをプッシュすると、AWS CodeStar は、ウェブサービスの再構築と再デプロイを行い、ユニットテストを実行します。
8. ユニットテストの結果を表示します。

9. プロジェクトをクリーンアップします。このステップは、このチュートリアルに関するコストに関して、AWS アカウントへの不要な請求を防ぐのに役立ちます。

ステップ 1: プロジェクトを作成する

このステップでは、AWS CodeStar コンソールを使用してプロジェクトを作成します。

1. AWS Management Consoleにサインインして、https://console.aws.amazon.com/codestar/ で、AWS CodeStar コンソールを開きます。

   > **Note**
   >
   > 作成した IAM ユーザーに関連付けられた認証情報または AWS CodeStarのセットアップ で識別される認証情報を使用して AWS Management Console にサインインする必要があります。このユーザーには、**AWSCodeStarFullAccess** マネージドポリシーが添付されている必要があります。

2. プロジェクトとそのリソースを作成する AWS リージョンを選択します。

   AWS CodeStar が利用可能な AWS リージョンの詳細については、「AWS 全般のリファレンス」の「リージョンとエンドポイント」を参照してください。

3. [Create project] (プロジェクトの作成) を選択します。
4. [Choose a project template] (プロジェクトのテンプレートを選択する) ページで、以下を選択します：

   - [Application type] (アプリケーションの種類) で、[Web service] (ウェブサービス) を選択します。
   - [Programming language] (プログラミング言語) で、[Python] を選択します。
   - AWS サービスで、AWS Lambdaを選択します。

5. 選択した内容が含まれているボックスを選択します。[Next] (次へ) を選択します。
6. [Project name] (プロジェクト名) に、プロジェクトの名前 (例: **My SAM Project**) を入力します。例とは異なる名前を使用した場合は、必ずこのチュートリアル全体でそれを使用してください。

   [Project ID] で、AWS CodeStar はこのプロジェクトに関連する識別子を選択します (例: [my-sam-project])。別のプロジェクト ID が表示された場合は、このチュートリアル全体でそれを使用してください。

[AWS CodeCommit] は選択されたままにし、[Repository name] (リポジトリ名) の値は変更しないでください。

7. [Next] (次へ) を選択します。
8. 設定を確認し、[Create Project] (プロジェクトの作成) を選択します。

   この AWS リージョンで AWS CodeStar を初めて使用する場合は、[表示名] と [E メール] に、AWS CodeStar で使用する IAM ユーザーの表示名と E メールアドレスを入力します。[Next] (次へ) をクリックします。
9. AWS CodeStar がプロジェクトを作成するまで待ちます。この処理には数分かかることがあります。更新時に[Project provisioned] (プロジェクトのプロビジョニング完了)バナー が表示されるまで次に進まないでください。

ステップ 2: プロジェクトリソースを調べる

このステップでは、プロジェクトの 4 つの AWS リソースを調べて、プロジェクトの仕組みを理解します。

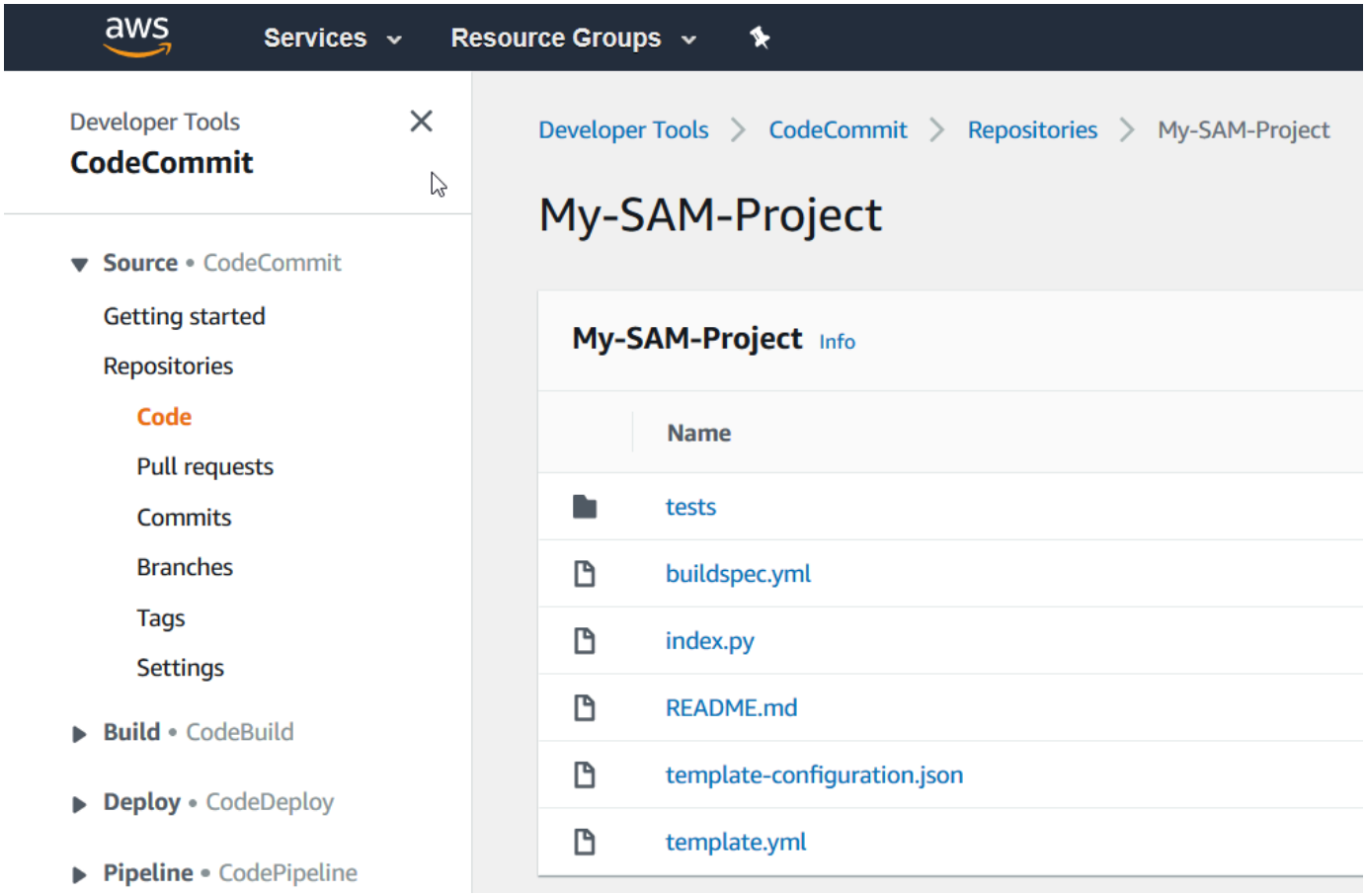
- プロジェクトのソースコードが保存される AWS CodeCommit リポジトリ。AWS CodeStar によって、リポジトリに [my-sam-project] という名前がつけられます。ここで、[my-sam-project] はプロジェクトの名前です。
- CodeBuild および AWS SAM を使用して、ウェブサービスの Lambda 関数およびAPI Gateway の API のビルドおよびデプロイを自動化する AWS CodePipeline パイプライン。AWS CodeStar によって、パイプラインに my-sam-project--Pipeline という名前が付けられます。ここで、my-sam-project はプロジェクトの ID です。
- ウェブサービスのロジックを含む Lambda 関数。AWS CodeStar によって、関数に awscodestar-my-sam-project-lambda-HelloWorld-***RANDOM_ID*** という名前が付けられます。各部分の意味を以下に示します。
  - [my-sam-project] はプロジェクトの ID です。
  - [HelloWorld] は、AWS CodeCommit リポジトリの `template.yaml` ファイルで指定されている関数 ID です。後でこのファイルについて説明します。
  - *[RANDOM_ID]*は、AWS SAM が一意性を保証するために関数に割り当てるランダムな ID です。

- Lambda 関数の呼び出しを簡単にする API Gateway の API。AWS CodeStar によって、API に awscodestar-my-sam-project--lambda という名前が付けられます。ここで、my-sam-project はプロジェクトの ID です。

CodeCommit でソースコードリポジトリを確認するには

1. AWS CodeStar CodeStar コンソールでプロジェクトを開き、ナビゲーションバーで、[リポジトリ] を選択します。
2. [Repository details] (リポジトリの詳細) で、CodeCommit リポジトリ (**My-SAM-Project**) へのリンクを選択します。
3. CodeCommit コンソールの [Code] (コード) ページに、プロジェクトのソースコードファイルが表示されます。

   - `buildspec.yml` では、CodePipeline が CodeBuild に対して、ビルドフェーズで AWS SAM を使用してウェブサービスをパッケージ化するように指示します。
   - `index.py` には、Lambda 関数のロジックが含まれています。この関数は、文字列「`Hello World`」と ISO 形式のタイムスタンプを出力します。
   - `README.md` には、リポジトリに関する一般的な情報が含まれています。
   - `template-configuration.json` には、プロジェクト ID でリソースにタグを付けるために使用されるプレースホルダ付きのプロジェクト ARN が含まれます。
   - `template.yml` は、AWS SAM が、API Gateway でウェブサービスをパッケージ化し API を作成するのに使用します。



ファイルの内容を表示するには、リストから選択します。

CodeCommit コンソール の使用の詳細については、「AWS CodeCommit ユーザーガイド」を参照してください。

CodePipeline でパイプラインを調べるには

1. パイプラインに関する情報を表示するには、AWS CodeStar コンソールでプロジェクトを開き、ナビゲーションバーで [パイプライン] を選択します。パイプラインには以下が含まれています。

   - [Source] (ソース) は、CodeCommit からソースコードを取得するステージです。
   - [Build] (ビルド) は、CodeBuildでソースコードを構築するステージです。
   - [Deploy] (デプロイ) は、ビルドされたソースコードと AWS リソースを AWS SAM でデプロイするステージです。

2. パイプラインの詳細を表示するには、[Pipeline details] (パイプラインの詳細) で、パイプラインを選択して CodePipeline コンソールでパイプラインを開きます。

CodePipeline コンソールの使用の詳細については、「AWS CodePipeline ユーザーガイド」 を参照してください。

[概要] ページでプロジェクトアクティビティと AWS サービスのリソースを調べるには

1. AWS CodeStar コンソールでプロジェクトを開き、ナビゲーションバーから [概要] を選択します。
2. [Project activity] (プロジェクトアクティビティ) リストおよび [Project Resources] (プロジェクトリソース) リストを確認します。

Lambda で関数を調べるには

1. AWS CodeStar コンソールでプロジェクトを開き、サイドナビゲーションバーで、[概要] を選択します。
2. [Project resources] (プロジェクトリソース) の [ARN]列で、Lambda 関数のリンクを選択します。

   関数のコードが Lambda コンソールに表示されます。

Lambda コンソールの使用の詳細については、「AWS Lambda デベロッパーガイド」を参照してください。

API Gateway で API を調べるには

1. AWS CodeStar コンソールでプロジェクトを開き、サイドナビゲーションバーで、[概要] を選択します。
2. [Project resources] (プロジェクトリソース) の [ARN]列で、Amazon API Gateway API のリンクを選択します。

   API Gateway コンソールに API のリソースが表示されます。

API Gateway コンソールの使用については、API Gateway デベロッパーガイド を参照してください。

ステップ 3: ウェブサービスをテストする

このステップでは、AWS CodeStar が構築してデプロイしたばかりのウェブサービスをテストします。

1. 前のステップからのプロジェクトを開いたままで、ナビゲーションバーの [Pipeline] (パイプライン) を選択します。
2. 続行する前に、[Source] (ソース)、[Build] (ビルド)、[Deploy] (デプロイ) ステージで、[Succeeded] (正常に完了) が表示されていることを確認します。この処理には数分かかることがあります。

   > **Note**
   >
   > いずれかのステージで [Failed] (失敗) が表示される場合は、以下のトラブルシューティングのヘルプを参照してください。
   >
   > - [Source] (ソース) ステージの場合は、AWS CodeCommit ユーザーガイドのトラブルシューティング AWS CodeCommitを参照してください。
   > - [Build] (ビルド) ステージの場合は、AWS CodeBuild ユーザーガイドのトラブルシューティング AWS CodeBuildを参照してください。
   > - [Deploy] (デプロイ) ステージの場合は、AWS CloudFormation ユーザーガイドのトラブルシューティング AWS CloudFormationを参照してください。
   > - その他の問題については、「トラブルシューティング AWS CodeStar」を参照してください。

3. [View Application] (アプリケーションの表示) を選択します。

ウェブブラウザで開いている新しいタブで、ウェブサービスは以下のレスポンス出力を表示します：

```
{"output": "Hello World", "timestamp": "2017-08-30T15:53:42.682839"}
```

ステップ 4: プロジェクトコードを編集するためのローカルワークステーションの設定

このステップでは、ローカルワークステーションを設定して、AWS CodeStar プロジェクトのソースコードを編集します。ローカルワークステーションとして、macOS、Windows、または Linux を実行している物理コンピュータまたは仮想コンピュータを利用できます。

1. 前の手順でプロジェクトを開いたままにしておきます。

   - ナビゲーションバーで、[IDE] を選択し、[Access your project code] (プロジェクトコードにアクセス) を展開します。
   - [Command line interface] (コマンドラインインターフェイス) のしたの [View instructions] (手順の表示) を選択します。

     Visual Studio または Eclipse がインストールされている場合は、代わりに [Visual Studio] または [Eclipse] の下の [View instructions] (手順の表示) を選択し、手順に従って ステップ 5: ウェブサービスにロジックを追加する に進んでください。
2. 手順に従って、次のタスクを完了します：

   a. ローカルワークステーションに Git をセットアップします。
   b. IAM コンソールを使用して IAM ユーザーのための Git 認証情報を生成します。
   c. ローカルワークステーションにプロジェクトの CodeCommit リポジトリのクローンを作成します。
3. 左のナビゲーションで、[Project] (プロジェクト) を選択し、プロジェクトの概要に戻ります。

ステップ 5: ウェブサービスにロジックを追加する

このステップでは、ローカルワークステーションを使用してロジックをウェブサービスに追加します。具体的には、Lambda 関数を追加して API Gateway の API に接続します。

1. ローカルワークステーションで、クローンされたソースコードリポジトリが保存されているディレクトリに移動します。
2. そのディレクトリに `hello.py` という名前のファイルを作成します。次のコードを追加し、ファイルを保存します：

```
import json
```

```
def handler(event, context):
  data = {
    'output': 'Hello ' + event["pathParameters"]["name"]
  }
  return {
    'statusCode': 200,
    'body': json.dumps(data),
    'headers': {'Content-Type': 'application/json'}
  }
```

上記のコードは、Hello という文字列と、呼び出し元が関数に送る文字列を出力します。

3. 同じディレクトリで template.yml ファイルを開きます。次のコードをファイルの末尾に追加し、ファイルを保存します。

```
  Hello:
    Type: AWS::Serverless::Function
    Properties:
      FunctionName: !Sub 'awscodestar-${ProjectId}-lambda-Hello'
      Handler: hello.handler
      Runtime: python3.7
      Role:
        Fn::GetAtt:
        - LambdaExecutionRole
        - Arn
      Events:
        GetEvent:
          Type: Api
          Properties:
            Path: /hello/{name}
            Method: get
```

AWSSAM はこのコードを使用して Lambdaに関数を作成し、API Gateway 内の API に新しいメソッドとパスを追加し、このメソッドとパスを新しい関数に接続します。

> **Note**
>
> 前述のコードのインデントは重要です。示されているように、コードを正確に追加しないと、プロジェクトが正しく構築されないことがあります。

4. git add . コマンドを実行して、ファイルの変更を複製されたリポジトリのステージングエリアに追加します。ピリオド (.) を忘れないでください。変更されたすべてのファイルに追加されます。

> **Note**
>
> コマンドラインの代わりに Visual Studio または Eclipse を使用している場合は、Git の使用方法が異なる場合があります。Visual Studio または Eclipse のドキュメントを参照してください。

5. git commit -m "Added hello.py and updated template.yaml." を実行して、クローンされたレポジトリのステージファイルをコミットします。
6. git push を実行してコミットをリモートリポジトリにプッシュします。

> **Note**
>
> 以前に生成したサインイン認証情報の入力を求めるメッセージが表示されることがあります。リモートリポジトリで作業するたびにこれが表示されることを防止するため、Git 認証情報マネージャーをインストールして設定することを考慮してみてください。例えば、macOS または Linux では、ターミナルで git config credential.helper 'cache --timeout 900' を実行して、15 分ごとにプロンプトを表示させることができます。または、git config credential.helper 'store --file ~/.git-credentials' を実行して、プロンプトを再度表示させないようにすることができます。Git は、認証情報をプレーンなファイルのクリアテキストとしてホームディレクトリに保存します。詳細については、Git ウェブサイトの [Git Tools - Credential Storage] (Git Tools - 認証情報ストレージ) を参照してください。

AWS CodeStar はプッシュを検出した後、CodePipeline に対してCodeBuild および AWS SAM を使用してウェブサービスを再構築および再デプロイするよう指示します。[Pipeline] (パイプライン) ページで、デプロイの進行状況を確認できます。

AWS SAM は、新しい関数に [awscodestar-my-sam-project-lambda-Hello-***RANDOM_ID***] という名前を付けます。それぞれの意味は次のとおりです。

- [my-sam-project] はプロジェクトの ID です。
- [Hello] は、`template.yaml` ファイルで指定された関数の ID です。

- *[RANDOM_ID]* は、AWS SAM が一意性を保証するために関数に割り当てるランダムな ID です。

ステップ 6: 拡張ウェブサービスをテストする

このステップでは、このステップでは、前の手順で追加したロジックに基づいて、AWS CodeStar が構築してデプロイした拡張ウェブサービスをテストします。

1. AWS CodeStar コンソールでプロジェクトを開き、ナビゲーションバーで、[パイプライン] を選択します。
2. 続行する前に、パイプラインが再度実行されており、[Source] (ソース)、[Build] (ビルド)、[Deploy] (デプロイ) ステージで、[Succeeded] (正常に完了) が表示されていることを確認します。この処理には数分かかることがあります。

   > **Note**
   >
   > いずれかのステージで [Failed] (失敗) が表示される場合は、以下のトラブルシューティングのヘルプを参照してください。
   >
   > - [Source] (ソース) ステージの場合は、AWS CodeCommit ユーザーガイドのトラブルシューティング AWS CodeCommitを参照してください。
   > - [Build] (ビルド) ステージの場合は、AWS CodeBuild ユーザーガイドのトラブルシューティング AWS CodeBuildを参照してください。
   > - [Deploy] (デプロイ) ステージの場合は、AWS CloudFormation ユーザーガイドのトラブルシューティング AWS CloudFormationを参照してください。
   > - その他の問題については、「トラブルシューティング AWS CodeStar」を参照してください。

3. [View Application] (アプリケーションの表示) を選択します。

   ウェブブラウザで開いている新しいタブで、ウェブサービスは以下のレスポンス出力を表示します：

   ```
   {"output": "Hello World", "timestamp": "2017-08-30T15:53:42.682839"}
   ```

4. タブのアドレスボックスで、パス **/hello/** とファーストネームを URL の最後に追加 (例: https://***API_ID***.execute-api.***REGION_ID***.amazonaws.com/Prod/hello/***YOUR_FIRST_NAME***) し、[Enter] (入力) を押します。

ファーストネームが Mary の場合、ウェブサービスは、次のレスポンス出力を表示します :

```
{"output": "Hello Mary"}
```

ステップ 7: ウェブサービスにユニットテストを追加する

このステップでは、ローカルワークステーションを使用して、ウェブサービスで AWS CodeStar が実行されるテストを追加します。このテストは、以前に行った手動テストに代わるものです。

1. ローカルワークステーションで、クローンされたソースコードリポジトリが保存されているディレクトリに移動します。
2. そのディレクトリに hello_test.py という名前のファイルを作成します。次のコードを追加し、ファイルを保存します。

```
from hello import handler

def test_hello_handler():

  event = {
    'pathParameters': {
      'name': 'testname'
    }
  }

  context = {}

  expected = {
    'body': '{"output": "Hello testname"}',
    'headers': {
      'Content-Type': 'application/json'
    },
    'statusCode': 200
  }

  assert handler(event, context) == expected
```

このテストは、Lambda 関数の出力が予想通りの形式であるかどうかをチェックします。予想通りの形式の場合、テストは成功です。そうでない場合は失敗です。

3. 同じディレクトリで buildspec.yml ファイルを開きます。ファイルの内容を次のコードに置き換えて、ファイルを保存します。

```
version: 0.2

phases:
   install:
      runtime-versions:
         python: 3.7

      commands:
         - pip install pytest
         # Upgrade AWS CLI to the latest version
         - pip install --upgrade awscli


   pre_build:
      commands:
         - pytest

   build:
      commands:
         # Use AWS SAM to package the application by using AWS CloudFormation
         - aws cloudformation package --template template.yml --s3-bucket
 $S3_BUCKET --output-template template-export.yml

         # Do not remove this statement. This command is required for AWS CodeStar
 projects.
         # Update the AWS Partition, AWS Region, account ID and project ID in the
 project ARN on template-configuration.json file so AWS CloudFormation can tag
 project resources.
         - sed -i.bak 's/\$PARTITION\$/'${PARTITION}'/g;s/\$AWS_REGION
\$/'${AWS_REGION}'/g;s/\$ACCOUNT_ID\$/'${ACCOUNT_ID}'/g;s/\$PROJECT_ID\
$/'${PROJECT_ID}'/g' template-configuration.json

artifacts:
   type: zip
   files:
      - template-export.yml
```

```
      - template-configuration.json
```

このビルド仕様では、CodeBuild に対して、ビルド環境に Python テストフレームワーク pytest をインストールするように指示します。CodeBuild は pytest を使用してユニットテストを実行します。これ以外のビルド仕様は、前に作成したものと同じです。

4. Git を使用して、これらの変更内容をリモートリポジトリにプッシュします。

```
git add .

git commit -m "Added hello_test.py and updated buildspec.yml."

git push
```

ステップ 8: ユニットテストの結果を表示する

このステップでは、ユニットテストが成功したか失敗したかを確認します。

1. AWS CodeStar コンソールでプロジェクトを開き、ナビゲーションバーで、[パイプライン] を選択します。
2. 続行する前に、パイプラインが再度実行されたことを確認します。この処理には数分かかることがあります。

   ユニットテストが成功した場合は、[Build] (ビルド) ステージに [Succeeded] (正常に終了) が表示されます。
3. ユニットテスト結果の詳細を表示するには、[Build] (ビルド) ステージで、 [CodeBuild] リンクを選択します。
4. CodeBuild コンソールの [Build Project: my-sam-project] ページの [Build history] (ビルド履歴) で、テーブルの [Build run] (ビルド実行) 列のリンクを選択します。
5. my-sam-project:***BUILD_ID*** ページの [Build logs] (ビルドログ) で、[View entire log] (全てのログを表示) リンクを選択します。
6. Amazon CloudWatch Logs コンソールに、次の例のようなテスト結果のログ出力が表示されます。次のテスト結果では、テストは成功しています。

```
...
============================= test session starts ==============================
platform linux2 -- Python 2.7.12, pytest-3.2.1, py-1.4.34, pluggy-0.4.0
rootdir: /codebuild/output/src123456789/src, inifile:
```

```
collected 1 item

hello_test.py .

============================ 1 passed in 0.01 seconds ============================
...
```

テストが失敗した場合は、ログ出力に詳細が表示され、障害のトラブルシューティングに役立ちます。

ステップ 9: クリーンアップ

このステップでは、プロジェクトをクリーンアップして、このプロジェクトに継続的な料金が発生するのを回避します。

このプロジェクトを引き続き使用したい場合は、この手順をスキップできますが、AWS アカウントに引き続き請求が行われる可能性があります。

1. AWS CodeStar コンソールでプロジェクトを開き、ナビゲーションバーで、[設定] を選択します。
2. [Project details] (プロジェクトの詳細) で、[Delete project] (プロジェクトの削除) を選択します。
3. **`delete`** を入力し、[Delete resources] (リソースの削除) ボックスをオンのまま、[Delete] (削除) を選択します。

> **Important**
>
> このチェックボックスをオフにすると、プロジェクトレコードは AWS CodeStar から削除されますが、プロジェクトの AWS リソースの多くは保持されます。お客様の AWS アカウントに引き続き請求される可能性があります。

このプロジェクト向けに AWS CodeStar で作成された Amazon S3 バケットがある場合は、このステップに従って削除します。

1. https://console.aws.amazon.com/s3/ で Amazon S3 コンソールを開きます。
2. バケットのリストで、[aws-codestar-***REGION_ID***-***ACCOUNT_ID***-my-sam-project--pipe] の横にあるアイコンを選択します。それぞれの種類の意味は次の通りです。

- *[REGION_ID]* は、今削除したプロジェクトの AWS リージョンの ID です。
- *[ACCOUNT_ID]* はお客様の AWS アカウント ID です。
- [my-sam-project] は、今削除したプロジェクトの ID です。

3. [Empty Bucket] (バケットを空にする) を選択します。バケットの名前を入力し、[Confirm] (確認) を選択します。
4. [Delete Bucket] (バケットの削除) を選択します。バケットの名前を入力し、[Confirm] (確認) を選択します。

次のステップ

このチュートリアルを完了したら、次のリソースを確認することをお勧めします。

- AWS CodeStar の使用の開始 チュートリアルでは、Amazon EC2 インスタンス上で実行されている Node.js ベースのウェブアプリケーションを作成しデプロイするプロジェクトを使用します。
- AWS CodeStar プロジェクトテンプレート で、作成できる他の種類のプロジェクトについて説明します。
- AWS CodeStar のチームで作業する では、他の人がどのようにプロジェクトに協力できるかを説明しています。

チュートリアル: AWS CLI を使用して AWS CodeStar にプロジェクトを作成する

このチュートリアルでは、AWS CLI を使用してサンプルソースコードとサンプルのツールチェーンテンプレートを含む AWS CodeStar プロジェクトを作成する方法を示します。AWS CodeStar は、AWS CloudFormation ツールチェーンテンプレートで指定された AWS インフラストラクチャと IAM リソースをプロビジョニングします。このプロジェクトでは、ツールチェーンのリソースを管理して、ソースコードの構築とデプロイを行います。

AWS CodeStar は AWS CloudFormation を使用してサンプルコードの構築とデプロイを行います。このサンプルコードでは、AWS Lambda をホストとし、Amazon API Gateway からアクセスすることができるウェブサービスが作成されます。

前提条件:

- AWS CodeStarのセットアップ のステップを完了します。

- Amazon S3 ストレージバケットを作成済みである必要があります。このチュートリアルでは、サンプルのソースコードとツールチェーンテンプレートをこの場所にアップロードします。

> **Note**
>
> このチュートリアルに関するコスト (例: AWS CodeStar が使用する AWS サービス) については、AWS アカウントに請求される場合があります。詳細については、「AWS CodeStar の料金」を参照してください。

トピック

- ステップ 1: サンプルソースコードのダウンロードと確認
- ステップ 2: サンプルツールチェーンテンプレートのダウンロード
- ステップ 3:AWS CloudFormation のツールチェーンテンプレートのテスト
- ステップ 4: ソースコードとツールチェーンテンプレートのアップロード
- ステップ 5: AWS CodeStar でプロジェクトを作成する

ステップ 1: サンプルソースコードのダウンロードと確認

このチュートリアルでは、ダウンロード可能な zip ファイルがあります。Lambda コンピューティングプラットフォームの Node.js サンプルアプリケーションのサンプルソースコードが含まれます。ソースコードをリポジトリに配置したら、フォルダとファイルは次のように表示されます：

```
tests/
app.js
buildspec.yml
index.js
package.json
README.md
template.yml
```

以下のプロジェクト要素はサンプルソースコードで表されます。

- `tests/`: このプロジェクトの CodeBuild プロジェクト用にユニットテストの設定。このフォルダには、サンプルコードが含まれていますが、プロジェクトの作成には必要ありません。
- `app.js`: プロジェクトのアプリケーションソースコード。

- `buildspec.yml`: CodeBuild リソース構築ステージの構築手順。このファイルは、CodeBuild リソースを含むツールチェーンテンプレートで必要です。
- `package.json`: アプリケーションソースコードの依存関係情報。
- `README.md`: AWS CodeStar のすべてのプロジェクトに含まれるプロジェクトの readme ファイル。このファイルはサンプルコードに含まれていますが、プロジェクトの作成には必要ありません。
- `template.yml`: AWS CodeStar のすべてのプロジェクトに含まれるインフラストラクチャテンプレートファイル、または SAM テンプレートファイル。これは、このチュートリアルで後にアップロードするツールチェーン template.yml とは異なります。このファイルはサンプルコードに含まれていますが、プロジェクトの作成には必要ありません。

ステップ 2: サンプルツールチェーンテンプレートのダウンロード

このチュートリアルのサンプルツールチェーンテンプレートでは、リポジトリ (CodeCommit)、パイプライン (CodePipeline)、ビルドコンテナ (CodeBuild) を作成し、AWS CloudFormation を使用してソースコードを Lambda プラットフォームにデプロイします。これらのリソースに加えて、IAM ロール (ランタイム環境のアクセス許可の絞り込みに使用) や、Amazon S3 バケット (CodePipeline によりデプロイメントアーティファクトの保存に使用)、CloudWatch Events ルール (コードをリポジトリにプッシュする際にパイプラインのデプロイのトリガーに使用) もあります。AWS IAM ベストプラクティスに合わせるには、この例で定義されているツールチェーンロールのポリシーを絞り込みます。

サンプルAWS CloudFormation テンプレートをYAMLの形式でダウンロードして解凍します。

チュートリアルの後半で create-project コマンドを実行すると、このテンプレートによって、次のカスタマイズされたツールチェーンリソースが AWS CloudFormation で作成されます。このチュートリアルで作成されるリソースの詳細については、『AWS CloudFormation ユーザーガイド』の次のトピックを参照してください。

- AWS::CodeCommit::Repository AWS CloudFormation リソースで CodeCommit リポジトリを作成します。
- AWS::CodeBuild::Project AWS CloudFormation リソースで CodeBuild 構築プロジェクトを作成します。
- AWS::CodeDeploy::Application AWS CloudFormation リソースで CodeDeploy アプリケーションを作成します。

- AWS::CodePipeline::Pipeline AWS CloudFormation リソースで CodePipeline パイプラインを作成します。
- AWS::S3::Bucket AWS CloudFormation リソースでパイプラインのアーティファクトバケットを作成します。
- AWS::S3::BucketPolicy AWS CloudFormation リソースで、パイプラインのアーティファクトバケットのアーティファクトバケットポリシーを作成します。
- AWS::IAM::Role AWS CloudFormation リソースは、AWS CodeStar に CodeBuild ビルドプロジェクトを管理する権限を与える CodeBuild IAM ワーカーロールを作成します。
- AWS::IAM::Role AWS CloudFormation リソースは、AWS CodeStar にパイプラインを作成するためのアクセス許可を与える CodePipeline IAM ワーカーロールを作成します。
- AWS::IAM::Role AWS CloudFormation リソースで、リソーススタックを作成する AWS CodeStar アクセス許可を付与する AWS CloudFormation IAM ワーカーのロールを作成します。
- AWS::IAM::Role AWS CloudFormation リソースで、リソーススタックを作成する AWS CodeStar アクセス許可を付与する AWS CloudFormation IAM ワーカーのロールを作成します。
- AWS::IAM::Role AWS CloudFormation リソースで、リソーススタックを作成する AWS CodeStar アクセス許可を付与する AWS CloudFormation IAM ワーカーのロールを作成します。
- AWS::Events::Rule AWS CloudFormation リソースは、リポジトリのプッシュイベントを監視する CloudWatch Events ルールを作成します。
- AWS::IAM::Role AWS CloudFormationリソースは、CloudWatch Events IAM ロールを作成します。

ステップ 3:AWS CloudFormation のツールチェーンテンプレートのテスト

ツールチェーンテンプレートをアップロードする前に、AWS CloudFormation のツールチェーンテンプレートをテストし、エラーがある場合にはトラブルシューティングすることができます。

1. 更新したテンプレートをローカルコンピュータに保存し、AWS CloudFormation コンソールを開きます。[Create Stack] (スタックの作成) を選択します。新しいリソースがリストに表示されています。
2. スタックの作成エラーがないかどうかスタックを確認します。
3. テストが完了したら、スタックを削除します。

> **Note**
>
> AWS CloudFormation に作成したスタックとすべてのリソースを削除します。削除しない場合は、プロジェクトを作成すると、リソース名が既に使用されているというエラーが表示される場合があります。

ステップ 4: ソースコードとツールチェーンテンプレートのアップロード

AWS CodeStar プロジェクトを作成するには、まずソースコードを .zip ファイルにし、Amazon S3 に置きます。AWS CodeStar はこの内容でリポジトリを初期化します。AWS CLI にプロジェクトを作成するコマンドを実行する際、入力ファイルでこの場所を指定します。

さらに、`toolchain.yml` ファイルをアップロードして、Amazon S3 に置きます。AWS CLI にプロジェクトを作成するコマンドを実行する際、入力ファイルでこの場所を指定します。

ソースコードとツールチェーンテンプレートをアップロードするには

1. 以下のサンプルファイル構造は、圧縮およびアップロードされるソースファイルとツールチェーンテンプレートを示します。サンプルコードには、`template.yml` ファイルを含みます。このファイルは、`toolchain.yml` ファイルとは異なる点にご注意ください。

   ```
   ls
   src toolchain.yml

   ls src/
   README.md    app.js        buildspec.yml    index.js    package.json
    template.yml    tests
   ```

2. ソースコードファイルの .zip ファイルを作成します。

   ```
   cd src; zip -r "../src.zip" *; cd ../
   ```

3. cp コマンドを使用して、パラメータとしてファイルを含めます。

   以下のコマンドにより、.zip ファイルおよび `toolchain.yml` が Amazon S3 にアップロードされます。

   ```
   aws s3 cp src.zip s3://MyBucket/src.zip
   ```

```
aws s3 cp toolchain.yml s3://MyBucket/toolchain.yml
```

Amazon S3 バケットを設定してソースコードを共有するには

- ソースコードとツールチェーンを Amazon S3 に保存するため、Amazon S3 バケットポリシーとオブジェクト ACL を使用して、他の IAM ユーザーや AWS アカウントが、サンプルからプロジェクトを作成できるようにします。AWS CodeStar では、カスタムプロジェクトを作成するすべてのユーザーが、使用するツールチェーンとソースにアクセスすることができます。

  すべてのユーザーがサンプルを使用できるようにするためには、以下のコマンドを実行します:

  ```
  aws s3api put-object-acl --bucket MyBucket --key toolchain.yml --acl public-read
  aws s3api put-object-acl --bucket MyBucket --key src.zip --acl public-read
  ```

ステップ 5: AWS CodeStar でプロジェクトを作成する

プロジェクトを作成するには、以下の手順に従います。

> **Important**
>
> AWS CLI の任意の AWS リージョンを設定していることを確認します。プロジェクトは、AWS CLI に設定されている AWS リージョンに作成されます。

1. create-project コマンドを実行し、`--generate-cli-skeleton` パラメータを含めます。

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

   JSON 形式のデータが出力に表示されます。ローカルコンピュータ上の場所にあるファイル (例:*`input.json`*)または AWS CLI がインストールされているインスタンス にデータをコピーします。コピーされたデータを次のように変更して、結果を保存します。この入力ファイルは、`MyProject` という名前のプロジェクトに、`myBucket` という名前のバケットで設定されています。

   - `roleArn` パラメータを指定していることを確認します。カスタムテンプレートの場合は、このチュートリアルのサンプルテンプレートのように、ロールを指定する必要があります。この

ロールは、「ステップ 2: サンプルツールチェーンテンプレートのダウンロード」で指定されたすべてのリソースを作成するためのアクセス許可を持っている必要があります。

- stackParameters に ProjectId パラメータを指定していることを確認します。このチュートリアルのサンプルテンプレートでは、このパラメータを使用する必要があります。

```
{
    "name": "MyProject",
    "id": "myproject",
    "description": "Sample project created with the CLI",
    "sourceCode": [
        {
            "source": {
                "s3": {
                    "bucketName": "MyBucket",
                    "bucketKey": "src.zip"
                }
            },
            "destination": {
                "codeCommit": {
                    "name": "myproject"

                }
            }
        }
    ],
    "toolchain": {
        "source": {
            "s3": {
                "bucketName": "MyBucket",
                "bucketKey": "toolchain.yml"
            }
        },
        "roleArn": "role_ARN",
        "stackParameters": {
            "ProjectId": "myproject"
        }
    }
}
```

2. 保存したばかりのファイルがあるディレクトリに移動し、create-project コマンドをもう一度実行します。--cli-input-json パラメータを指定します。

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

3. 成功すると、次のようなデータが出力に表示されます：

```
{
    "id": "project-ID",
    "arn": "arn"
}
```

- この出力には、新しいプロジェクトに関する情報が含まれています：
  - id 値はプロジェクト ID を表します。
  - arn 値は、プロジェクトの ARN を表します。

4. プロジェクトの作成ステータスを確認するには、describe-project コマンドを使用します。--id パラメータを指定します。

```
aws codestar describe-project --id <project_ID>
```

次のようなデータが出力に表示されます。

```
{
    "name": "MyProject",
    "id": "myproject",
    "arn": "arn:aws:codestar:us-east-1:account_ID:project/myproject",
    "description": "",
    "createdTimeStamp": 1539700079.472,
    "stackId": "arn:aws:cloudformation:us-east-1:account_ID:stack/awscodestar-
myproject/stack-ID",
    "status": {
        "state": "CreateInProgress"
    }
}
```

- この出力には、新しいプロジェクトに関する情報が含まれています：
  - id 値は一意のプロジェクト ID を表します。
  - state 値は、プロジェクトの作成ステータス (例: CreateInProgress または CreateComplete) を表します。

プロジェクトが作成される間、コマンドライン またはお好みの IDE からプロジェクトリポジトリに [add team members] (チームメンバーの追加)、または [configure access] (アクセスの設定) を行うことができます。

チュートリアル: AWS CodeStarで Alexa スキルプロジェクトを作成する

AWS CodeStar は、AWS のアプリケーションを迅速に開発、構築、およびデプロイするために必要なツールを備えた、AWS 上のクラウドベースの開発サービスです。AWS CodeStar では、継続的デリバリーのツールチェーン全体を数分で設定でき、コードのリリースをすばやく開始できます。AWS CodeStar の Alexa スキルプロジェクトテンプレートを使用すると、AWS アカウントから、簡単な Hello World Alexa スキルを数クリックのみで作成できます。テンプレートでは、スキル開発用の継続的インテグレーション (CI) ワークフローの使用を開始できる基本的なデプロイパイプラインも作成されます。

AWS CodeStar から Alexa スキルを作成する主な利点は、AWS 内でスキル開発を開始し、Amazon 開発者アカウントをプロジェクトに接続して、スキルを AWS から直接開発ステージにデプロイできることです。また、プロジェクトのすべてのソースコードに関するリポジトリを備えたデプロイ (CI) パイプラインを用意できます。任意の IDE を使用してこのリポジトリを設定し、使い慣れているツールを使用してスキルを作成できます。

前提条件

- https://developer.amazon.com にアクセスして Amazon 開発者アカウントを作成します。 サインアップは無料です。このアカウントが Alexa スキルを所有します。
- AWS アカウントをお持ちでない場合は、次に説明する手順に従ってアカウントを作成してください。

  AWS にサインアップするには

  1. https://aws.amazon.com/ を開いて、[AWS アカウントの作成] を選択します。

     > **Note**
     >
     > AWS アカウントのルートユーザー 認証情報を使用して、すでに AWS Management Console にサインインしている場合は、[Sign in to a different account (別のアカウントにサインインする)] を選択します。IAM 認証情報を使用してすでにコンソールにサイ

ンインしている場合は、[AWS アカウントのルートユーザーの認証情報を使ってサインイン] を選択します。[新しい AWS アカウントの作成] を選択します。

2. オンラインの手順に従います。

> **Important**
>
> Alexa スキルプロジェクトの作成後、編集はすべてプロジェクトリポジトリ内でのみ行います。このスキルを、ASK CLI や ASK 開発者コンソールなど他の Alexa Skills Kit ツールを使用して直接編集しないことをお勧めします。これらのツールは、プロジェクトのリポジトリと統合されていません。これらを使用すると、スキルおよびリポジトリコードが同期されません。

ステップ 1: プロジェクトを作成して Amazon 開発者アカウントに接続する

このチュートリアルでは、AWS Lambda で実行されている Node.js を使用してスキルを作成します。他の言語でもほとんどの手順は同じです。ただし、スキル名は異なります。選択したプロジェクトテンプレート固有の詳細については、プロジェクトリポジトリ内の README.md ファイルを参照してください。

1. AWS Management Consoleにサインインして、https://console.aws.amazon.com/codestar/ で、AWS CodeStar コンソールを開きます。
2. プロジェクトとそのリソースを作成する AWS リージョンを選択します。Alexa スキルランタイムは、以下の AWS リージョンでのみ利用できます：
   - アジアパシフィック (東京)
   - 欧州 (アイルランド)
   - 米国東部 (バージニア北部)
   - 米国西部 (オレゴン)
3. [Create project] (プロジェクトの作成) を選択します。
4. [Choose a project template] (プロジェクトのテンプレートを選択する) ページで、以下を選択します：
   a. [Application type] (アプリケーションの種類) には、[Alexa Skill] (Alexa スキル) を選択します。
   b. [Programming language] (プログラミング言語) には、[Node.js] を選択します。
5. 選択した内容が含まれているボックスを選択します。

6. [Project name] (プロジェクト名) に、プロジェクトの名前 (例: **`My Alexa Skill`**) を入力します。別の名前を使用する場合は、このチュートリアル全体でそれを使用してください。AWS CodeStar によって、[プロジェクト ID] に、このプロジェクトに関連付けられた識別子が選択されます (例: my-alexa-skill)。別のプロジェクト ID が表示された場合は、このチュートリアル全体でそれを使用してください。
7. このチュートリアルではリポジトリに [AWS CodeCommit] を選択し、[リポジトリ名] の値は変更しないでください。
8. [Connect Amazon developer account] (Amazon デベロッパーアカウントを接続) を選択して、スキルをホストする Amazon 開発者アカウントにリンクします。Amazon 開発者アカウントをお持ちでない場合は、まず Amazon Developers からアカウントを作成し、登録を完了してください。
9. Amazon 開発者認証情報を使用してサインインします。[許可] を選択し、[確認] を選択して接続を完了します。
10. Amazon 開発者アカウントに関連付けられた複数のベンダー ID がある場合は、このプロジェクト用に使用するものを選択します。管理者または開発者ロールが割り当てられたアカウントを使用してください。
11. [Next] (次へ) を選択します。
12. (オプション) この AWS リージョンで AWS CodeStar を初めて使用する場合は、AWS CodeStar で使用する IAM ユーザーの表示名と E メールアドレスを入力します。[Next] (次へ) をクリックします。
13. AWS CodeStar がプロジェクトを作成するまで待ちます。この処理には数分かかることがあります。[Project provisioned] (プロジェクトプロビジョン) バナーが表示されるまで次に進まないでください。

ステップ 2: Alexa Simulator でスキルをテストする

最初のステップで、AWS CodeStar によってスキルが作成され Alexa スキル開発ステージにデプロイされました。次は、Alexa Simulator でスキルをテストします。

1. AWS CodeStar コンソールのプロジェクトで、[アプリケーションの表示] を選択します。Alexa Simulator で新しいタブが開きます。
2. ステップ 1 でプロジェクトに接続したアカウントの Amazon 開発者認証情報を使用してサインインします。
3. [Test] (テスト) の下で [Development] (開発) を選択してテストを有効にします。
4. `ask hello node hello` と入力します。スキルのデフォルトの呼び出し名は `hello node` です。

5. スキルは `Hello World!` と応答するはずです。

Alexa Simulator でスキルが有効になると、Amazon 開発者アカウントに登録された Alexa 搭載デバイスで呼び出すこともできます。デバイスでスキルをテストするには、「アレクサ、hello node にあいさつするように頼んで」と言います。

Alexa Simulator の詳細については、「開発者コンソールでスキルをテストする」を参照してください。

ステップ 3: プロジェクトリソースを調べる

プロジェクト作成の一環として、AWS CodeStar はユーザーに代わって AWS リソースも作成します。これらのリソースには、CodeCommit を使用するプロジェクトリポジトリ、 CodePipeline を使用するデプロイパイプライン、AWS Lambda 関数などがあります。 これらのリソースにはナビゲーションバーからアクセスできます。例えば、[Repository] (リポジトリ) を選択すると、CodeCommit リポジトリの詳細を表示します。パイプラインのデプロイのステータスは、[Pipeline] (パイプライン) ページに表示されます。プロジェクトの一部として作成されたすべての AWS リソースの一覧は、ナビゲーションバーの [Overview] (概要) を選択すると表示されます。このリストには、各リソースへのリンクが含まれています。

ステップ 4: スキルの応答を変更する

このステップでは、スキルの応答を少し変更して、イテレーションサイクルを理解します。

1. ナビゲーションバーで [Repository] (リポジトリ) を選択します。[Repository name] (リポジトリ名) の下にあるリンクを選択すると、プロジェクトのリポジトリが新しいタブまたはウィンドウで開きます。このリポジトリには、ビルド仕様 (buildspec.yml)、AWS CloudFormation アプリケーションスタック (template.yml)、readme ファイル、およびスキルパッケージ形式 (プロジェクト構造) のスキルのソースコードが含まれています。
2. [file lambda] > [custom] > [index.js] (Node.js の場合) の順に移動します。このファイルには、リクエスト処理コードが含まれています。これは ASK SDK を使用します。
3. [Edit] (編集) を選択します。
4. 24 行目の文字列 `Hello World!` を、文字列 `Hello. How are you?` に置き換えます。
5. ファイルの末尾までスクロールします。作成者名と、E メールアドレス、およびオプションでコミットメッセージを入力します。
6. [Commit changes] (変更のコミット) を選択してリポジトリに対する変更をコミットします。

7. AWS CodeStar でプロジェクトに戻り、[パイプライン] ページで確認します。パイプラインがデプロイ中であることが表示されます。
8. パイプラインでデプロイが完了したら、もう一度 Alexa Simulator でスキルをテストします。今度はスキルは `Hello. How are you?` と応答するはずです。

ステップ 5: ローカルワークステーションを設定してプロジェクトリポジトリに接続する

以前は CodeCommit コンソールから直接ソース コードに小さな変更を加えました。このステップでは、ローカルワークステーションを使用してプロジェクトリポジトリを設定し、コマンドラインやお好みの IDE からコードを編集および管理できるようにします。以下の手順は、コマンドラインツールをセットアップする方法について説明します。

1. 必要に応じて、AWS CodeStar のプロジェクトダッシュボードに移動します。
2. ナビゲーションバーで [IDE]を選択します。
3. [Access your project code] (プロジェクトコードにアクセスする) から[Command line interface] (コマンドラインインターフェイス) の下の[View instructions] (手順の表示) を選択します。
4. 手順に従って、次のタスクを完了します：
   a. Git Downloads などのウェブサイトからローカルワークステーションに Git をインストールします。
   b. AWS CLI をインストールします。詳細については、「AWS Command Line Interface のインストール」を参照してください。
   c. IAM ユーザーのアクセスキーとシークレットキーを使用して AWS CLI を設定します。詳細については、「AWS CLI を設定する」を参照してください。
   d. ローカルワークステーションにプロジェクトの CodeCommit リポジトリのクローンを作成します。詳細については、[Connect to a CodeCommit Repository] (CodeCommit リポジトリに接続する) を参照してください。

次のステップ

このチュートリアルでは、基本的なスキルの開始方法を説明しました。スキル開発の取り組みを継続するには、以下のリソースを参照してください。

- スキルの基礎の理解のために、Alexa 開発者 YouTube チャンネルの「Alexa スキルのしくみ」や他のビデオをご覧ください。

- スキルのさまざまなコンポーネントの理解には、[skill package format] (スキルパッケージの形式)、[skill manifest schemas] (スキルマニフェストのスキーマ)、[interaction model schemas] (対話モデルのスキーマ) のドキュメントをお読みください。
- アイデアをスキルとするために、[Alexa Skills Kit]および[ASK SDK]のドキュメントをお読みください。

チュートリアル:GitHub ソースリポジトリを使用してプロジェクトを作成する

AWS CodeStar を使用すると、リポジトリを設定して、プロジェクトチームでプルリクエストを作成、レビュー、マージを行うことができます。

このチュートリアルでは、GitHub リポジトリにサンプルウェブアプリケーションのソースコードを格納したプロジェクト、変更をデプロイするパイプライン、アプリケーションがクラウドでホストされている EC2 インスタンスを作成します。プロジェクトを作成した後、このチュートリアルでは、ウェブアプリケーションのホームページに変更を加える GitHub プルリクエストを作成してマージする方法を説明します。

トピック

- ステップ 1: プロジェクトと GitHub リポジトリの作成
- ステップ 2: ソースコードの表示
- ステップ 3: GitHub プルリクエストの作成

ステップ 1: プロジェクトと GitHub リポジトリの作成

このステップでは、コンソールを使用してプロジェクトを作成し、新しい GitHub リポジトリへの接続を作成します。GitHub リポジトリにアクセスするには、AWS CodeStar が GitHub での認可を管理するために使用する接続リソースを作成します。プロジェクトが作成されると、その追加のリソースがプロビジョンされます。

1. AWS Management Consoleにサインインして、https://console.aws.amazon.com/codestar/ で、AWS CodeStar コンソールを開きます。
2. プロジェクトとそのリソースを作成する AWS リージョンを選択します。
3. [AWS CodeStar] ページで、[プロジェクトの作成] を選択します。

4. [Choose a project template] (プロジェクトテンプレートを選択する) ページで [Web application] (ウェブアプリケーション)、[Node.js]、および[Amazon EC2] チェックボックスを選択します。次に、オプションのセットで使用可能なテンプレートから選択します。

   詳細については、「AWS CodeStar プロジェクトテンプレート」を参照してください。

5. [Next] (次へ) を選択します。

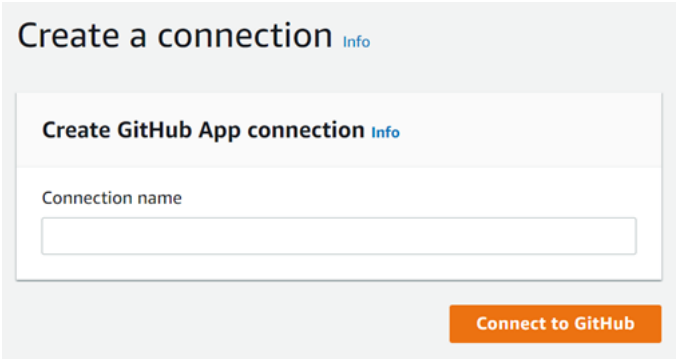
6. [Project name] (プロジェクト名) に、プロジェクトの名前 (例: **MyTeamProject**) を入力します。別の名前を選択した場合は、このチュートリアル全体でその名前を使用してください。

7. [Project repository] (プロジェクトリポジトリ) で、[GitHub] を選択してください。

8. [GitHub] を選択した場合、接続リソースを選択または作成する必要があります。既存の接続がある場合は、検索フィールドで選択します。それ以外の場合は、ここで新しい接続を作成します。[Connect to GitHub] (GitHub に接続) を選択します。

   [Create a connection] (接続の作成) ページが表示されます。

   > **Note**
   >
   > 接続を作成するには、GitHub アカウントが必要です。組織の接続を作成する場合は、組織の所有者である必要があります。



   a. [Create GitHub App connection] (GitHub App 接続の作成) で、[Connection name] (接続名) に接続名を入力します。[Connect to GitHub] (GitHub に接続) を選択します。

      [Connect to GitHub] (GitHub に接続) ページが表示され、[GitHub Apps] フィールドが表示されます。

   b. [GitHub Apps] で、アプリケーションのインストールを選択するか、[Install a new app] (新しいアプリケーションをインストールする) を選択してアプリケーションを作成します。

> **Note**
>
> 特定のプロバイダーへのすべての接続に対してアプリを 1 つインストールします。GitHub app 用 AWS コネクターをすでにインストールしている場合は、これを選択してこのステップをスキップしてください。

c. [Install AWS Connector for GitHub] ページで、アプリをインストールするアカウントを選択します。

> **Note**
>
> アプリケーションをインストール済みである場合は、[Configure] (設定) を選択してアプリのインストールの変更ページに進むか、戻るボタンでコンソールに戻ることができます。

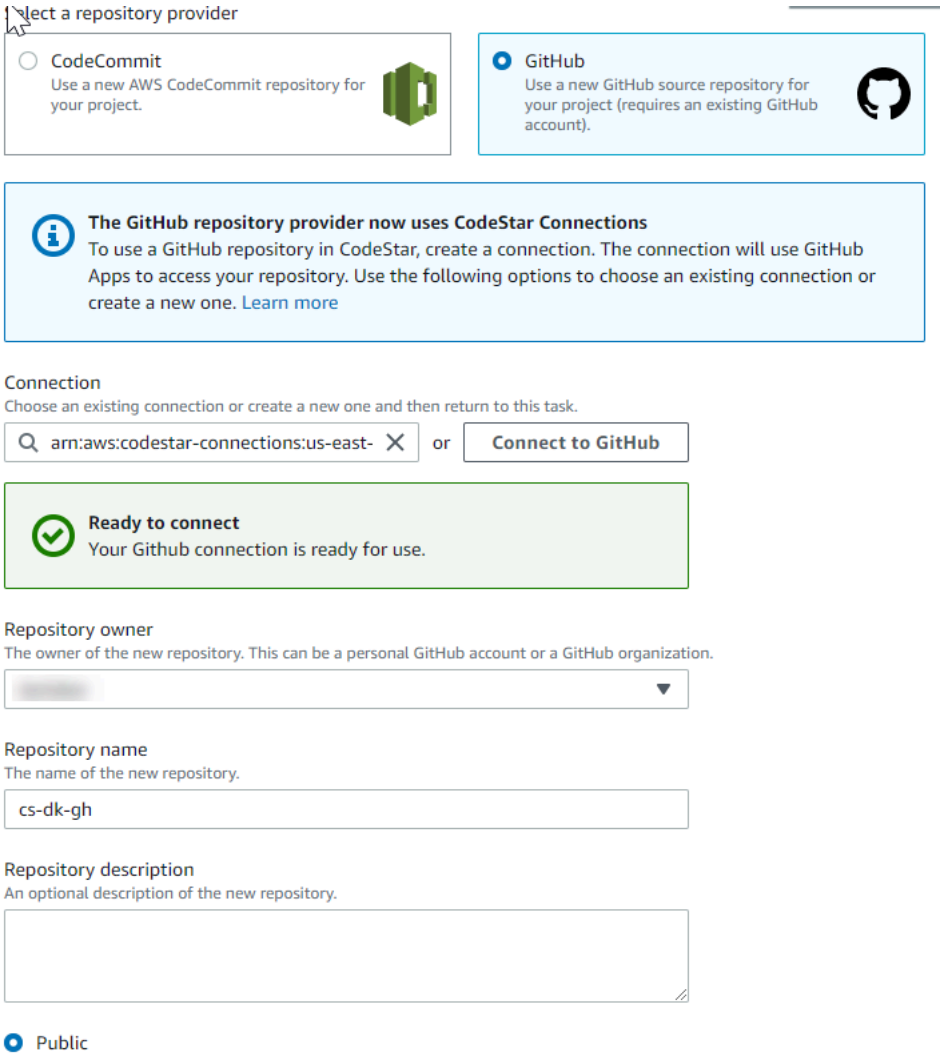
d. [Confirm password to continue] (パスワードを確認して続行) ページが表示される場合、GitHub パスワードを入力し、[Sign in] (サインイン) を選択します。

e. [Install AWS Connector for GitHub] ページで、デフォルトはそのままで [Install] を選択します。

f. [Connect to GitHub] (GitHub へ接続) ページで、新規インストールのインストール ID が [GitHub Apps] に表示されます。

接続が正常に作成された後、CodeStar の [create project] (プロジェクトを作成) ページで、[Ready to connect] (接続準備完了) メッセージが表示されます。

> **Note**
>
> [Developer Tools] (デベロッパーツール) コンソールの [Settings] (設定) で接続を表示できます。詳細については、[Getting started with connections] (接続入門ガイド) を参照してください。



g. [Repository owner] (リポジトリ所有者) で、GitHub 組織または個人用 GitHub アカウントを選択します。

h. [Repository name] (リポジトリ名) で、デフォルトの GitHub リポジトリ名を受け入れるか、別の名前を入力します。

i. [Public] (公開) または[Private] (プライベート) を選択します。

> **Note**
>
> AWS Cloud9 を開発環境として使用する場合は、[Public repository] (公開リポジトリ)を選択する必要があります。

j. (オプション) [Repository description] (リポジトリの説明) に、GitHub リポジトリの説明を入力します。

9. プロジェクトが Amazon EC2 インスタンスにデプロイされ、変更する場合、[Amazon EC2 Configuration](Amazon EC2 の設定)で Amazon EC2 インスタンスを設定します。例えば、プロジェクトの使用可能なインスタンスタイプから選択できます。

   [key pair] (キーペア) で、ステップ 4: AWS CodeStar プロジェクトの Amazon EC2 キーペアの作成 で作成した Amazon EC2 キーペアを選択します。[I acknowledge that I have access to the private key file] (私はプライベートキーファイルへのアクセス権があることを認めます) を選択します。

10. [Next] (次へ) を選択します。
11. リソースと設定の詳細を確認します。
12. [Next] (次へ) または [Create project] (プロジェクトの作成) を選択します。(表示される選択はプロジェクトテンプレートによって異なります。)

    プロジェクトの作成中は、数分間待ってください。

13. プロジェクトが作成された後、[アプリケーションの表示] を選択して、ウェブアプリケーションを表示します。

ステップ 2: ソースコードの表示

このステップでは、ソースコードとソースリポジトリに使用できるツールを表示します。

1. プロジェクトのナビゲーションバーで、[Repository] (リポジトリ) を選択します。

   GitHub でコミットのリストを表示するには、[View commits] (コミットの表示) を選択します。これにより GitHub でコミット履歴が開きます。

   課題を表示するには、プロジェクトの [Issues] (課題) タブを選択します。GitHub で新しい課題を作成するには、[Create GitHub issue] (GitHub の課題を作成する) を選択します。これにより、GitHub でリポジトリの課題フォームが開きます。

2. [Repository] (リポジトリ) タブで、[Repository name] (リポジトリ名) の下にあるリンクを選択すると、プロジェクトのリポジトリが新しいタブまたはウィンドウで開きます。このリポジトリには、プロジェクトのソースコードが含まれています。

ステップ 3: GitHub プルリクエストの作成

このステップでは、ソースコードにわずかな変更を加え、プルリクエストを作成します。

1. GitHub で、リポジトリに新しい機能ブランチを作成します。[main branch] (メインブランチ) ドロップダウンフィールドを選択し、`feature-branch` という名前のフィールドに新しいブランチを入力します。[Create new branch] (新規ブランチを作成) を選択します。ブランチが作成され、チェックアウトされます。
2. GitHub で、`feature-branch`ブランチに変更を加えます。公開フォルダを開き、`index.html` ファイルを開きます。
3. AWS CodeStar コンソールの [プルリクエスト] で、GitHub のプルリクエストを作成するには、[プルリクエストの作成] を選択します。これにより、GitHub でリポジトリのプルリクエストフォームが開きます。GitHub で、鉛筆アイコンを選択してファイルを編集します。

   `Congratulations!` の後、文字列 `Well done, <name>!` を追加してユーザーの名前で `<name>` を置き換えます。[Commit changes] (変更のコミット) を選択します。変更は機能ブランチにコミットされます。
4. AWS CodeStar コンソールで、プロジェクトを選択します。[Repository] (リポジトリ) タブを選択します。[Pull requests] (プルリクエスト) で、[Create pull request] (プルリクエストの作成) を選択します。

   GitHub でフォームが開きます。メインブランチはベースブランチに残します。[Compare to] (比較する) で、機能ブランチを選択します。差分を表示します。
5. GitHub で、[Create pull request] (プルリクエストの作成) を選択します。[Update index.html] (index.htmlの更新) という名前のプルリクエストが作成されます。
6. AWS CodeStar コンソールで、新しいプルリクエストを表示します。[Merge changes] (マージ変更) を選択して、変更をリポジトリにコミットし、プルリクエストをリポジトリのメインブランチとマージします。
7. AWS CodeStar でプロジェクトに戻り、[パイプライン] ページでチェックします。パイプラインがデプロイ中であることが表示されます。
8. プロジェクトが作成された後、[アプリケーションの表示] を選択して、ウェブアプリケーションを表示します。

AWS CodeStar プロジェクトテンプレート

AWS CodeStar プロジェクトテンプレートを使用すると、サンプルアプリケーションから開始し、開発プロジェクトをサポートするために作成された AWS リソースを使用してアプリケーションをデプロイすることができます。AWS CodeStar プロジェクトテンプレートを選択すると、アプリケーションタイプ、プログラミング言語、コンピューティングプラットフォームがプロビジョニングされます。ウェブアプリケーション、ウェブサービス、Alexa スキル、および静的ウェブページを使用してプロジェクトを作成したら、サンプルアプリケーションを独自のものに置き換えることができます。

AWS CodeStar でプロジェクトを作成したら、アプリケーションの配信をサポートする AWS リソースを変更できます。AWS CodeStar は AWS CloudFormation と提携して、コードを使用し、クラウド上のサポートサービスおよびサーバー/サーバーレスプラットフォームを作成できます。AWS CloudFormation では、テキストファイルでインフラストラクチャ全体をモデリングできます。

トピック

- AWS CodeStar プロジェクトファイルとリソース
- はじめに: プロジェクトテンプレートを選択する
- AWS CodeStar プロジェクトを変更する方法

AWS CodeStar プロジェクトファイルとリソース

AWS CodeStar プロジェクトは、コードをデプロイするために作成されたソースコードとリソースの組み合わせです。コードのビルド、リリース、デプロイに役立つリソースのコレクションは、[Toolchain resources] (ツールチェーンリソース) と呼ばれます。プロジェクト作成時、AWS CloudFormation テンプレートを使用して、継続的な統合/継続的デプロイメント (CI/CD) パイプラインのツールチェーンリソースをプロビジョンします。

プロジェクトは、AWS CodeStar を使用して、AWS のリソース作成の経験に応じて、2 種類の方法で作成できます。

- コンソールを使用してプロジェクトを作成すると、AWS CodeStar はリポジトリを含むツールチェーンリソースを作成し、サンプルアプリケーションコードやプロジェクトファイルをリポジトリに取り込みます。コンソールを使用して、事前設定済みのプロジェクトオプションのセットに基づき、サンプルプロジェクトをすばやくセットアップできます。

- CLI を使用してプロジェクトを作成する場合は、ツールチェーンリソースを作成する AWS CloudFormation テンプレートとアプリケーションソースコードを指定します。CLI を使用して、AWS CodeStar がテンプレートからプロジェクトを作成し、サンプルコードでリポジトリを作成できるようにします。

AWS CodeStar プロジェクトは一元管理できます。コンソールで [Create project] (プロジェクトの作成) ウィザードを使用して、サンプルプロジェクトをセットアップします。チームのアクセス許可とリソースを管理するコラボレーションプラットフォームとして使用できます。詳細については、「AWS CodeStar とは?」を参照してください。コンソールを使用してプロジェクトを作成すると、ソースコードがサンプルコードとして提供され、CI/CD ツールチェーンリソースが作成されます。

コンソールでプロジェクトを作成すると、AWS CodeStar によって以下のリソースがプロビジョニングされます。

- GitHub または CodeCommit のソースコードリポジトリ。
- ファイルとディレクトリの詳細を提供する `README.md` ファイル (プロジェクトリポジトリ内)。
- アプリケーションのランタイムスタックの定義を保存する `template.yml` ファイル (プロジェクトリポジトリ内)。このファイルを使用して、ツールチェーンリソースではないプロジェクトリソース (例：通知、データベースサポート、モニタリング、およびトレースに使用される AWS リソース) を追加または変更します。
- パイプラインに関連して作成された AWS サービスおよびリソース (例: Amazon S3 アーティファクトバケット、Amazon CloudWatch Events、および関連するサービスロール)。
- 完全なソースコードとパブリック HTTP エンドポイントを含む実動するサンプルアプリケーション。
- AWS CodeStar プロジェクトテンプレートタイプに基づく AWS コンピューティングリソース:
  - Lambda 関数。
  - Amazon EC2 インスタンス。
  - AWS Elastic Beanstalk 環境。
- 2018 年 12 月 6 日 (PDT) スタート:
  - アクセス許可の境界 (プロジェクトリソースへのアクセスを管理するための特殊な IAM ポリシー)。アクセス許可の境界は、デフォルトでサンプルプロジェクトのロールに添付されています。詳細については、「ワーカーロールの IAM アクセス許可の境界」を参照してください。

  - AWS CloudFormation を使用してプロジェクトリソースを作成するための AWS CloudFormation IAM ロール (IAM ロールなど、AWS CloudFormation でサポートされているすべてのリソースのアクセス許可を含む)。
  - ツールチェーンの IAM ロール。
  - アプリケーションスタックで定義された Lambda の実行ロール (変更可能)。
- 2018 年 12 月 6 日 (PDT) 以前:
  - 限定された一部の AWS CloudFormation リソースをサポートする、プロジェクトリソースを作成するための AWS CloudFormation IAM ロール。
  - CodePipeline リソースを作成するための IAM ロール。
  - CodeBuild リソースを作成するための IAM ロール。
  - プロジェクトタイプで該当する場合、CodeDeploy リソースを作成するための IAM ロール。
  - プロジェクトタイプで該当する場合、Amazon EC2 ウェブアプリケーションを作成するための IAM ロール。
  - CloudWatch Events リソースを作成するための IAM ロール。
  - 動的に変更され、リソースの一部を含めるための Lambda の実行ロール。

このプロジェクトには、ステータスを示す詳細ページがあり、チーム管理へのリンク、IDE またはリポジトリの設定手順へのリンク、リポジトリ内のソースコード変更のコミット履歴が含まれています。また、Jira などの外部の課題追跡ツールに接続するためのツールを選択することもできます。

はじめに: プロジェクトテンプレートを選択する

コンソールで AWS CodeStar プロジェクトを選択すると、事前に設定されたオプションのセットからサンプルコードとリソースを選択して、すぐに始めることができます。これらのオプションは、[Project templates] (プロジェクトテンプレート) と呼ばれます。AWS CodeStar プロジェクトテンプレートはそれぞれ、プログラミング言語、アプリケーションタイプ、およびコンピューティングプラットフォームで構成されています。プロジェクトテンプレートは、選択した組み合わせによって決まります。

テンプレートコンピューティングプラットフォームを選択する

テンプレートごとに、次のコンピューティングプラットフォームタイプのいずれかを設定します。

- AWS Elastic Beanstalk プロジェクトを選択すると、クラウド内の Amazon Elastic Compute Cloud インスタンスの AWS Elastic Beanstalk 環境にデプロイします。

- Amazon EC2 プロジェクトを選択すると、AWS CodeStar によって、クラウド上のアプリケーションをホストするための Linux EC2 インスタンスが作成されます。プロジェクトチームメンバーはインスタンスにアクセスでき、チームは指定されたキーペアを使用して Amazon EC2 インスタンスに SSH 接続します。AWS CodeStar には、チームメンバーのアクセス許可を使用してキーペア接続を管理するマネージド SSH もあります。
- AWS Lambda を選択すると、AWS CodeStar は、管理するためのインスタンスやサーバーなしで、Amazon API Gateway 経由でアクセスするサーバーレス環境を作成します。

テンプレートアプリケーションタイプを選択する

テンプレートごとに、次のアプリケーションタイプのいずれかを設定します：

- ウェブサービス

  ウェブサービスは、API を呼び出すなど、バックグラウンドで実行されるタスクに使用されます。AWS CodeStar がサンプルウェブサービスプロジェクトを作成したら、エンドポイント URL を選択して Hello World 出力を表示できますが、このアプリケーションタイプの主な用途はユーザーインターフェイス (UI) ではありません。このカテゴリの AWS CodeStar プロジェクトテンプレートは、Ruby、Java、ASP.NET、PHP、Node.js などの開発をサポートしています。
- ウェブアプリケーション

  ウェブアプリケーションには、UI が搭載されています。AWS CodeStar がサンプルウェブアプリケーションプロジェクトを作成したら、エンドポイント URL を選択してインタラクティブウェブアプリケーションを表示できます。このカテゴリの AWS CodeStar プロジェクトテンプレートは、Ruby、Java、ASP.NET、PHP、Node.js などの開発をサポートしています。
- 静的ウェブページ

  HTML ウェブサイトのプロジェクトが必要な場合はこのテンプレートを選択します。このカテゴリの AWS CodeStar プロジェクトテンプレートは、HTML5 の開発をサポートします。
- Alexa スキル

  このテンプレートを選択するのは、Alexa スキルのプロジェクトで AWS Lambda 関数を使用する場合のみです。スキルプロジェクトを作成すると、AWS CodeStar によってサービスエンドポイントとして使用できる Amazon リソースネーム (ARN) が返されます。詳細については、「カスタムスキルを AWS Lambda 関数としてホストする」を参照してください。

> **Note**
>
> Alexa スキルの Lambda 関数は、米国東部 (バージニア北部)、米国西部 (オレゴン)、欧州 (アイルランド)、およびアジアパシフィック (東京) の各リージョンでのみサポートされています。

- Config ルール

  アカウントの AWS リソース全体でルールを自動化する AWS Config ルールのプロジェクトが必要な場合は、このテンプレートを選択します。この関数は、ルールのサービスエンドポイントとして使用できる ARN を返します。

テンプレートのプログラミング言語の選択

プロジェクトテンプレートを選択する際、Ruby、Java、ASP.NET、PHP、Node.js などのプログラミング言語を選択します。

AWS CodeStar プロジェクトを変更する方法

プロジェクトを更新するには、以下を更新します：

- アプリケーションのサンプルコードおよびプログラミング言語リソース。
- アプリケーションが保存およびデプロイされるインフラストラクチャを構成するリソース (オペレーティングシステム、サポートアプリケーションとサービス、デプロイパラメータ、クラウドコンピューティングプラットフォーム)。アプリケーションリソースは、`template.yml` ファイルで変更します。これは、アプリケーションのランタイム環境をモデリングする AWS CloudFormation ファイルです。

> **Note**
>
> Alexa スキル AWS CodeStar プロジェクトを使用する場合は、AWS CodeStar ソースリポジトリ (CodeCommit または GitHub) の外部でスキルに変更を加えることはできません。Alexa デベロッパーポータルでスキルを編集すると、その変更がソースリポジトリに表示されない場合があり、2 つのバージョンは同期しません。

アプリケーションソースコードの変更と変更のプッシュ

サンプルソースコード、スクリプト、および他のアプリケーションソースファイルを変更するには、ソースリポジトリのファイルを次のように編集します：

- CodeCommit または GitHub の編集モードを使用する。
- IDE で、プロジェクトを開く (例：AWS Cloud9)。
- リポジトリのクローンをローカルに作成後、変更をコミットおよびプッシュします。詳細については、ステップ 4: 変更をコミットするを参照してください。

Template.yml ファイルを使用してアプリケーションリソースを変更する

インフラストラクチャリソースを手動で変更するのではなく、AWS CloudFormation を使用して、アプリケーションのランタイムリソースをモデリングおよびデプロイします。

ランタイムスタックのアプリケーションリソース (例: Lambda 関数) を変更または追加するには、プロジェクトリポジトリの `template.yml` ファイルを編集します。AWS CloudFormation リソースとして利用可能なリソースを追加することができます。

AWS Lambda 関数のコードまたは設定を変更するには、「リソースをプロジェクトに追加する」を参照してください。

アプリケーションリソースである AWS CloudFormation リソースのタイプを追加するには、プロジェクトのリポジトリの `template.yml` ファイルを変更します。アプリケーションリソースを `template.yml` ファイルの `Resources` セクションに追加すると、AWS CloudFormation および AWS CodeStar によってリソースが作成されます。AWS CloudFormation リソースとその必須プロパティのリストについては、「AWS リソースタイプのリファレンス」を参照してください。詳細については、「ステップ 1: IAM で CloudFormation ワーカーロールを編集する」のこの例を参照してください。

AWS CodeStar では、アプリケーションのランタイム環境を構成およびモデリングすることによって、ベストプラクティスを実装できます。

アプリケーションリソースを変更するアクセス許可を管理する方法

AWS CloudFormation を使用してランタイムアプリケーションリソース (例: Lambda 関数) を追加すると、すでに付与されているアクセス許可を AWS CloudFormation ワーカーロールで使用でき

ます。一部のランタイムアプリケーションリソースでは、`template.yml` ファイルを編集する前に、AWS CloudFormation ワーカーロールのアクセス許可を手動で調整する必要があります。

AWS CloudFormation ワーカーロールのアクセス許可の変更例については、「ステップ 5: インラインポリシーでリソースのアクセス許可を追加する」を参照してください。

AWS CodeStar ベストプラクティス

AWS CodeStar は多数の製品およびサービスと統合されています。以下のセクションでは、AWS CodeStar のベストプラクティス、およびこれらの関連製品とサービスについて説明します。

トピック

- AWS CodeStar リソースで使用するセキュリティのベストプラクティス
- 依存関係のバージョンを設定するベストプラクティス
- AWS CodeStar リソースで使用するモニタリングと ログ記録のベストプラクティス

AWS CodeStar リソースで使用するセキュリティのベストプラクティス

定期的にパッチを適用し、アプリケーションで使用される依存関係のセキュリティベストプラクティスを確認する必要があります。これらのセキュリティのベストプラクティスを使用して、サンプルコードを更新し、本番環境でプロジェクトを維持します。

- 進行中のセキュリティに関する発表と、フレームワークの更新を追跡します。
- プロジェクトをデプロイする前に、フレームワークで開発されたベストプラクティスに従います。
- フレームワークの依存関係を定期的に確認し、必要に応じて更新します。
- 各 AWS CodeStar テンプレートには、プログラミング言語の設定手順が含まれます。プロジェクトのソースリポジトリの `README.md` ファイルを参照してください。
- プロジェクトリソースを分離するためのベストプラクティスとして、AWS CodeStar のセキュリティ に導入された [multi-account strategy] (マルチアカウント戦略) を使用してAWS リソースへの最小特権アクセスを管理します。

依存関係のバージョンを設定するベストプラクティス

AWS CodeStar プロジェクトのサンプルソースコードでは、ソースリポジトリの `package.json` ファイルに表示されている依存関係を使用します。ベストプラクティスとして、常に特定のバージョンを指すように依存関係を設定します。これは、バージョンピンニングとも呼ばれます。予告なしにアプリケーションが中断する可能性があるため、バージョンを `latest` に設定することはお勧めしません。

AWS CodeStar リソースで使用するモニタリングとログ記録のベストプラクティス

AWS のロギング機能を使用すると、ユーザーがアカウントで実行したアクションや使用されたリソースを確認できます。ログファイルは次の情報を表示します：

- アクションが実行された日時。
- アクションのソース IP アドレス。
- 不適切なアクセス権限が理由で失敗したアクション。

AWS CloudTrail は AWS アカウントによって、またはそれに代わるアカウントによって行われた AWS API コールとそれに関連するイベントをログするために使用できます。詳細については、「AWS CloudTrail を使用した AWS CodeStar API コールのログ記録」を参照してください。

AWS CodeStar でのプロジェクトの使用

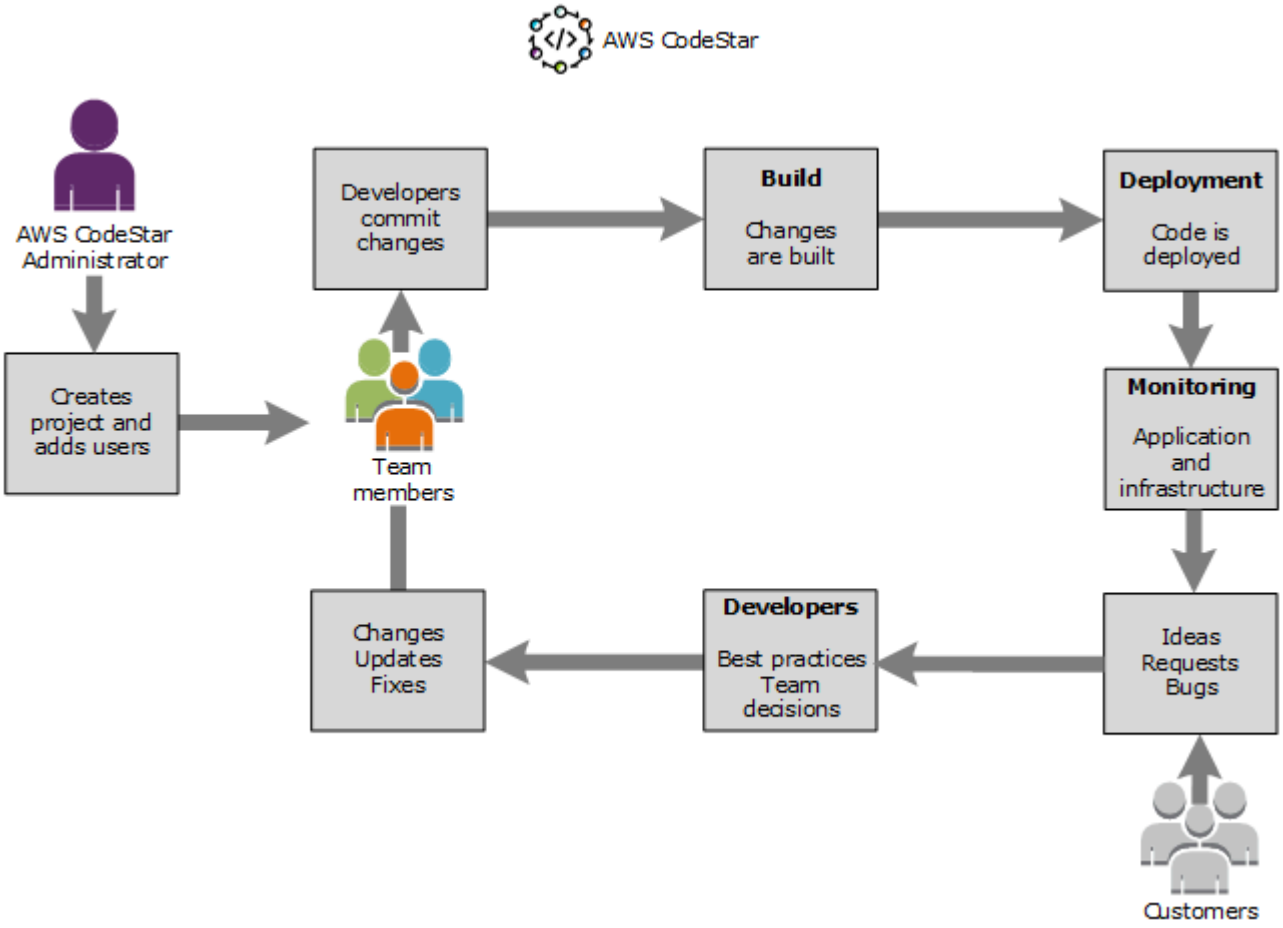
AWS CodeStar プロジェクトテンプレートを使用すると、以下のような必要なリソースが既に設定されているプロジェクトをすばやく作成できます。

- ソースリポジトリ
- 構築環境
- デプロイとホスティングリソース
- プログラム言語

テンプレートにはサンプルソースコードも含まれているため、すぐにプロジェクトで作業を開始できます。

プロジェクトを作成すると、リソースの追加や削除、プロジェクトダッシュボードのカスタマイズ、進行状況のモニタリングを行うことができます。

次の図は、AWS CodeStar プロジェクトの基本的なワークフローを示しています。



図の基本のワークフローでは、AWSCodeStarFullAccess ポリシーを適用したデベロッパーがプロジェクトを作成し、チームメンバーを追加します。一緒にコードを書き込み、ビルド、テスト、およびデプロイします。プロジェクトダッシュボードには、アプリケーションアクティビティの表示とビルドのモニタリング、デプロイメントパイプラインを介したコードの流れの表示などをリアルタイムで使用できるツールが提供されます。チームは、チーム wiki タイルを使用して、情報、ベストプラクティス、リンクを共有します。問題追跡ソフトウェアを統合して、進行状況やタスクを追跡するのに役立ちます。お客様からリクエストやフィードバックを受け取ると、チームはこの情報をプロジェクトに追加し、プロジェクト計画および開発に統合します。プロジェクトが成長するにつれて、チームはそのコードベースをサポートするために、より多くのチームメンバーを追加します。

AWS CodeStar でプロジェクトを作成する

AWS CodeStar コンソールを使用してプロジェクトを作成します。プロジェクトテンプレートを使用する場合は、お客様に必要なリソースが設定されます。このテンプレートには、コーディングを開始するために使用するサンプルコードも含まれています。

プロジェクトを作成するには、AWSCodeStarFullAccess ポリシーまたは同等のアクセス許可を持つ IAM ユーザーとして、AWS Management Console にサインインする必要があります。詳細については、「AWS CodeStarのセットアップ」を参照してください。

> **Note**
>
> このトピックで手順を完了する前に、AWS CodeStarのセットアップ のステップを完了する必要があります。

トピック

- AWS CodeStar コンソールでプロジェクトを作成する (コンソール)
- AWS CodeStar でプロジェクトを作成する (AWS CLI)

AWS CodeStar コンソールでプロジェクトを作成する (コンソール)

AWS CodeStar コンソールを使用してプロジェクトを作成します。

AWS CodeStar でプロジェクトを作成するには

1. AWS Management Consoleにサインインして、https://console.aws.amazon.com/codestar/ で、AWS CodeStar コンソールを開きます。

   プロジェクトとそのリソースを作成する AWS リージョンにサインインしていることを確認してください。例えば、米国東部 (オハイオ) でプロジェクトを作成するには、AWS リージョンが選択されていることを確認します。AWS CodeStar が利用可能な AWS リージョンの詳細については、「AWS 全般のリファレンス」の「リージョンとエンドポイント」を参照してください。
2. [AWS CodeStar] ページで、[プロジェクトの作成] を選択します。
3. [プロジェクトのテンプレートを選択する] ページで、AWS CodeStar プロジェクトテンプレートのリストからプロジェクトの種類を選択します。フィルタバーを使用して選択を絞り込むことができます。例えば、Amazon EC2 インスタンスにデプロイされる Node.js で記述されたウェブアプリケーションプロジェクトの場合は、[Web application] (ウェブアプリケーション)、[Node.js] の順に選択し、[Amazon EC2] チェックボックスをオンにします。次に、オプションのセットで使用可能なテンプレートから選択します。

   詳細については、「AWS CodeStar プロジェクトテンプレート」を参照してください。
4. [Next] (次へ) を選択します。

5. [プロジェクト名] で、*My First Project* などのプロジェクト名を入力します。[Project ID] (プロジェクト ID) では、プロジェクトの ID はこのプロジェクト名から派生しますが、15 文字の制限があります。

   例えば、*My First Project* という名前のプロジェクトのデフォルト ID は *my-first-projec* です。このプロジェクト ID は、プロジェクトに関連付けられているすべてのリソースの名前のベースです。AWS CodeStar は、このプロジェクト ID をコードリポジトリの URL の一部として使用するほか、IAM の関連するセキュリティアクセスロールとポリシーの名前にも使用します。プロジェクトの作成後、プロジェクト ID は変更できません。プロジェクトを作成する前にプロジェクト ID を編集するには、[Project ID] (プロジェクト ID) で、使用する ID を入力します。

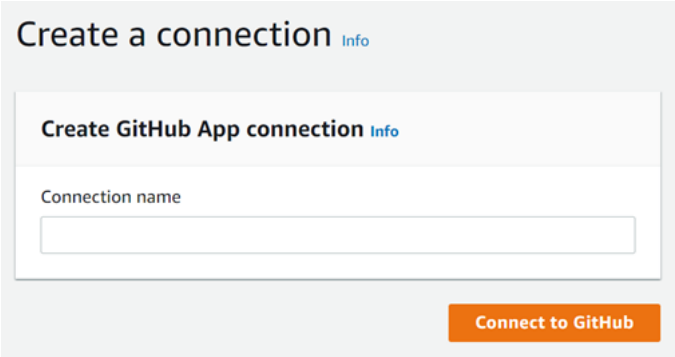
   プロジェクト名とプロジェクト ID の制限の詳細については、AWS CodeStar における制限 を参照してください。

   > **Note**
   >
   > AWS リージョン内の AWS アカウントでは、プロジェクト ID が一意である必要があります。

6. リポジトリプロバイダー、AWS CodeCommit または [GitHub] を選択します。
7. AWS CodeCommit を選択した場合は、[Repository name] (リポジトリ名) で、デフォルトの AWS CodeCommit リポジトリ名を受け入れるか、別の名前を入力します。ステップ 9 に進みます。
8. [GitHub] を選択した場合、接続リソースを選択または作成する必要があります。既存の接続がある場合は、検索フィールドで選択します。それ以外の場合は、ここで新しい接続を作成します。[Connect to GitHub] (GitHub に接続) を選択します。

   [Create a connection] (接続の作成) ページが表示されます。

   > **Note**
   >
   > 接続を作成するには、GitHub アカウントが必要です。組織の接続を作成する場合は、組織の所有者である必要があります。



a. [GitHub App 接続の作成] で、[接続名] テキスト入力フィールドに接続名を入力します。[Connect to GitHub] (GitHub に接続) を選択します。

   [Connect to GitHub] (GitHub に接続) ページが表示され、[GitHub Apps] フィールドが表示されます。

b. [GitHub Apps] で、アプリケーションのインストールを選択するか、[Install a new app] (新しいアプリケーションをインストールする) を選択してアプリケーションを作成します。

> **Note**
>
> 特定のプロバイダーへのすべての接続に対してアプリを 1 つインストールします。GitHub app 用 AWS コネクターをすでにインストールしている場合は、これを選択してこのステップをスキップしてください。

c. [Install AWS Connector for GitHub] ページで、アプリをインストールするアカウントを選択します。

> **Note**
>
> アプリケーションをインストール済みである場合は、[Configure] (設定) を選択してアプリのインストールの変更ページに進むか、戻るボタンでコンソールに戻ることができます。

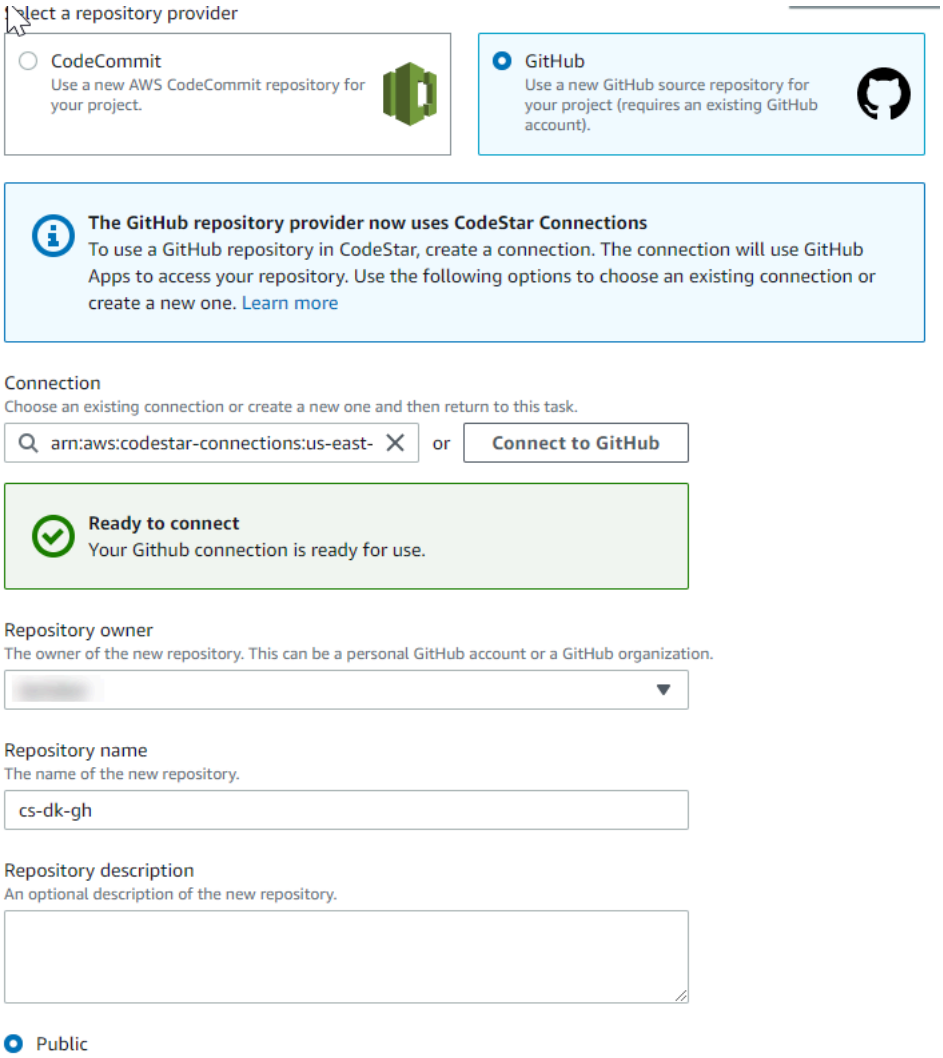
d. [Confirm password to continue] (パスワードを確認して続行) ページが表示される場合、GitHub パスワードを入力し、[Sign in] (サインイン) を選択します。

e. [GitHub 用 AWS コネクターのインストール] ページで、デフォルトのままで、[インストール] を選択します。

f. [GitHub へ接続] ページで、新規インストールのインストール ID が [GitHub Apps] テキスト入力フィールドに表示されます。

接続が作成された後、CodeStar の [create project] (プロジェクトを作成) ページで、[Ready to connect] (接続準備完了) メッセージが表示されます。

> **Note**
>
> [Developer Tools] (デベロッパーツール) コンソールの[Settings] (設定) で接続を表示できます。詳細については、[Getting started with connections] (接続入門ガイド) を参照してください。



g. [Repository owner] (リポジトリ所有者) で、GitHub 組織または個人用 GitHub アカウントを選択します。

h. [Repository name] (リポジトリ名) で、デフォルトの GitHub リポジトリ名を受け入れるか、別の名前を入力します。

i. [Public] (公開) または[Private] (プライベート) を選択します。

> **Note**
>
> AWS Cloud9 を開発環境として使用する場合は、[Public] (公開) を選択する必要があります。

j. (オプション) [Repository description] (リポジトリの説明) に、GitHub リポジトリの説明を入力します。

> **Note**
>
> Alexa スキルプロジェクトテンプレートを選択する場合は、Amazon 開発者アカウントを接続する必要があります。Alexa スキルプロジェクトの操作の詳細については、「チュートリアル: AWS CodeStarで Alexa スキルプロジェクトを作成する」を参照してください。

9. プロジェクトが Amazon EC2 インスタンスにデプロイされ、変更を加える場合は、[Amazon EC2 Configuration] (Amazon EC2 の設定) で Amazon EC2 インスタンスを設定します。例えば、プロジェクトの使用可能なインスタンスタイプから選択できます。

> **Note**
>
> 異なる Amazon EC2 インスタンスタイプは、異なるレベルのコンピューティングパワーを提供し、異なる関連費用が発生する可能性があります。詳細については、[Amazon EC2 Instance Types] (Amazon EC2 インスタンスタイプ) と [Amazon EC2 Pricing] (Amazon EC2 の料金) を参照してください。
> 複数の仮想プライベートクラウド (VPC) または複数のサブネットが Amazon 仮想プライベートクラウド で作成されている場合は、使用する VPC とサブネットを選択することもできます。ただし、ハードウェア専有インスタンスでサポートされていない Amazon EC2 インスタンスタイプを選択した場合は、インスタンスのテナンシーが [Dedicated] (専有) に設定されている VPC を選択することはできません。

詳細については、[What Is Amazon VPC?] (Amazon VPC とは) および [Dedicated Instance Basics] (ハードウェア専有インスタンスの基礎) を参照してください。

[Key pair] (キーペア) で、ステップ 4: AWS CodeStar プロジェクトの Amazon EC2 キーペアの作成 で作成した Amazon EC2 キーペアを選択します。[I acknowledge that I have access to the private key file] (私はプライベートキーファイルへのアクセス権があることを認めます) を選択します。

10. [Next] (次へ) を選択します。
11. リソースと設定の詳細を確認します。
12. [Next] (次へ) または [Create project] (プロジェクトの作成) を選択します。(表示される選択はプロジェクトテンプレートによって異なります。)

    プロジェクト (リポジトリを含む) の作成には数分かかる場合があります。
13. プロジェクトのリポジトリの作成後は、[Repository] (リポジトリ) ページを使用して、リポジトリへのアクセス権を設定します。[Next steps] (次のステップ) のリンクを使用して、IDE を設定、課題追跡を設定、チームメンバーをプロジェクトに追加できます。

プロジェクトが作成される間、コマンドライン またはお好みの IDE からプロジェクトリポジトリに [add team members] (チームメンバーの追加)、または [configure access] (アクセスの設定) を行うことができます。

AWS CodeStar でプロジェクトを作成する (AWS CLI)

AWS CodeStar プロジェクトは、コードをデプロイするために作成されたソースコードとリソースの組み合わせです。コードのビルド、リリース、デプロイに役立つリソースのコレクションは、ツールチェーンリソースと呼ばれます。プロジェクト作成時、AWS CloudFormation テンプレートを使用して、継続的な統合/継続的デプロイメント (CI/CD) パイプラインのツールチェーンリソースをプロビジョンします。

コンソールでプロジェクトを作成すると、ツールチェーンテンプレートが作成されます。AWS CLI を使用してプロジェクトを作成する際、ツールチェーンリソースを作成するツールチェーンテンプレートを作成します。

完全なツールチェーンを作成するには、次の推奨リソースが必要です：

1. ソースコードを保存する CodeCommit または GitHub リポジトリ。

2. リポジトリへの変更をリッスンするよう設定されている CodePipeline パイプライン。

   a. CodeBuild を使用してユニットテストまたは統合テストを使用する場合は、ビルドステージをパイプラインに追加してビルドアーティファクトを追加することをお勧めします。

   b. CodeDeploy または AWS CloudFormation を使用してビルドアーティファクトとソースコードをランタイムインフラストラクチャにデプロイするデプロイステージを、パイプラインに追加することをお勧めします。

   > **Note**
   >
   > CodePipeline では、2 つ以上のステージがパイプラインに必要であり、最初のステージはソースステージにする必要があるため、ビルドステージまたはデプロイステージを 2 番目のステージとして追加します。

AWS CodeStar ツールチェーンは、CloudFormation テンプレートとして定義されています。

このタスクで説明しているチュートリアルおよびサンプルリソースの設定については、「チュートリアル: AWS CLI を使用して AWS CodeStar にプロジェクトを作成する」を参照してください。

前提条件:

プロジェクトを作成する際、入力ファイルで次のパラメータを指定します。以下が指定されていないと、AWS CodeStar で空のプロジェクトが作成されます。

- ソースコード。このパラメータがリクエストに含まれている場合は、ツールチェーンテンプレートも含む必要があります。
  - ソースコードには、プロジェクトの実行に必要なアプリケーションコードを含む必要があります。
  - ソースコードには、必要な設定ファイル (例: CodeBuild プロジェクトの場合は buildspec.yml、CodeDeploy のデプロイの場合は appspec.yml) を含む必要があります。
  - ソースコードには、オプションの項目 (例: README、またはツールチェーン以外の AWS リソースの場合は template.yml) を含めることもできます。
- ツールチェーンテンプレート。ツールチェーンテンプレートは、プロジェクトで管理される AWS リソースおよび IAM ロールをプロビジョンします。
- ソースの場所。プロジェクトのソースコードおよびツールチェーンテンプレートを指定する場合は、場所を指定する必要があります。ソースファイルおよびツールチェーンテンプレートを

Amazon S3 バケットにアップロードします。AWS CodeStar はファイルを取得後、それを使用してプロジェクトを作成します。

> **Important**
>
> AWS CLI の任意の AWSリージョンを設定していることを確認します。プロジェクトは、AWS CLI に設定されている AWS リージョンに作成されます。

1. create-project コマンドを実行し、`--generate-cli-skeleton` パラメータを含めます。

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

JSON 形式のデータが出力に表示されます。ローカルコンピュータ上の場所にあるファイル (例:*input.json*)または AWS CLI がインストールされているインスタンス にデータをコピーします。コピーされたデータを次のように変更して、結果を保存します。

```
{
    "name": "project-name",
    "id": "project-id",
    "description": "description",
    "sourceCode": [
        {
            "source": {
                "s3": {
                    "bucketName": "s3-bucket-name",
                    "bucketKey": "s3-bucket-object-key"
                }
            },
            "destination": {
                "codeCommit": {
                    "name": "codecommit-repository-name"
                },
                "gitHub": {
                    "name": "github-repository-name",
                    "description": "github-repository-description",
                    "type": "github-repository-type",
                    "owner": "github-repository-owner",
                    "privateRepository": true,
                    "issuesEnabled": true,
```

```
                    "token": "github-personal-access-token"
                }
            }
        }
    ],
    "toolchain": {
        "source": {
            "s3": {
                "bucketName": "s3-bucket-name",
                "bucketKey": "s3-bucket-object-key"
            }
        },
        "roleArn": "service-role-arn",
        "stackParameters": {
            "KeyName": "key-name"
        }
    },
    "tags": {
        "KeyName": "key-name"
    }
}
```

以下に置き換えます:

- *project-name*: 必須。この AWS CodeStar プロジェクトの分かりやすい名前。
- *project-id*: 必須。この AWS CodeStar プロジェクトのプロジェクト ID。

> **Note**
>
> プロジェクト作成時、一意のプロジェクト ID が必要です。既に存在するプロジェクト ID を使用して入力ファイルを送信すると、エラーが表示されます。

- *##*: オプション。この AWS CodeStar プロジェクトの説明。
- *sourceCode*: オプション。プロジェクト用に指定されたソースコードの設定情報。現在は、単一の sourceCode オブジェクトのみサポートされています。sourceCode オブジェクトにはそれぞれ、AWS CodeStar によってソースコードが取得される場所と、ソースコードが追加される送信先に関する情報が含まれます。
  - *source*: 必須。これにより、ソースコードをアップロードした場所が定義されます。サポートされるソースは Amazon S3 のみです。AWS CodeStar はソースコードを取得して、プロジェクト作成後にリポジトリにそれを含めます。

- *S3*: オプション。ソースコードの Amazon S3 の場所。
  - *bucket-name*: ソースコードが含まれるバケット。
  - *bucket-key*: ソースコードを含む .zip ファイル (例: `src.zip`) を指すバケットのプリフィックスとオブジェクトキー。
- *###*: オプション。プロジェクト作成時にソースコードが追加される送信先の場所。ソースコードの送信先として、CodeCommit および GitHub がサポートされています。

  これらの 2 つのオプションのうちいずれかのみ指定することができます :

  - *codeCommit*: 必要な属性は、ソースコードを含む必要のある CodeCommit リポジトリの名前のみです。このリポジトリは、ツールチェーンテンプレートに含まれている必要があります。

    > **Note**
    >
    > CodeCommit では、ツールチェーンスタックで定義されているリポジトリの名前を指定する必要があります。AWS CodeStar は、Amazon S3 で指定したソースコードを使用してこのリポジトリを初期化します。

  - *gitHub*: このオブジェクトは、GitHub リポジトリを作成し、ソースコードと連携するために必要な情報を表します。GitHub リポジトリを選択した場合は、以下の値が必要になります。

    > **Note**
    >
    > GitHub の場合は、既存の GitHub リポジトリを指定することはできません。AWS CodeStar によって、リポジトリが作成され、Amazon S3 にアップロードしたソースコードが追加されます。AWS CodeStar は次の情報を使用して、リポジトリを GitHub に作成します。

    - *name*: 必須。GitHub リポジトリの名前。
    - *description*: 必須。GitHub リポジトリの説明。
    - *type*: 必須。GitHub リポジトリのタイプ。有効な値は、[User] (ユーザー)または [Organization] (組織)です。
    - *owner*: 必須。リポジトリの所有者を表す GitHub ユーザー の名前。リポジトリの所有者が GitHub 組織の場合は、組織名を入力します。

- *privateRepository*: 必須。このリポジトリをプライベートにするか公開にするか。有効な値は、true または false です。
- *issuesEnabled*: 必須。このリポジトリで、GitHub の課題を有効にするかどうか。有効な値は、true または false です。
- *[token]* (トークン) : オプション。これは、AWS CodeStar から GitHub アカウントにアクセスするために使用される個人用アクセストークンです。このトークンには、スコープ repo、user、admin:repo_hook を含む必要があります。GitHub からの個人用アクセストークンを取得するには、GitHub ウェブサイトの[Creating a Personal Access Token for the Command Line] (コマンドライン用のパーソナルアクセストークンの作成) を参照してください。

> **Note**
>
> CLI を使用して GitHub ソースリポジトリでプロジェクトを作成する場合、AWS CodeStar はトークンを使用して OAuth アプリを介してリポジトリにアクセスします。コンソールを使用して GitHub ソースリポジトリでプロジェクトを作成する場合、AWS CodeStar は接続リソースを使用し、GitHub アプリケーションでリポジトリにアクセスします。

- *toolchain*: プロジェクトの作成時に設定される CI/CD ツールチェーンに関する情報。ツールチェーンテンプレートをアップロードした場所が含まれています。テンプレートによって、ツールチェーンリソースが含まれる AWS CloudFormation スタックが作成されます。 また、参照する AWS CloudFormation のパラメータの上書きとスタックの作成に使用されるロールも含まれます。AWS CodeStar は、テンプレートを取得後、AWS CloudFormation を使用してテンプレートを実行します。
  - *source*: 必須。ツールチェーンテンプレートの場所です。Amazon S3 のみ、ソースの場所としてサポートされています。
    - *S3*: オプション。ツールチェーンテンプレートをアップロードした Amazon S3 の場所。
      - *bucket-name* : Amazon S3 バケットの名前。
      - *bucket-key*: ツールチェーンテンプレートを含む .yml ファイルまたは .json ファイル (例: `files/toolchain.yml`) を指すバケットのプリフィックスとオブジェクトキー。

- *stackParameters*: オプション。AWS CloudFormation に渡されるキーと値のペアが含まれます。これらはパラメータです (ある場合)。ツールチェーンテンプレートは参照用にセットアップされます。
- *role*: オプション。アカウントでツールチェーンリソースを作成するために使用されるロール。このロールは次を満たしている必要があります：
  - ロールが指定されていない場合、ツールチェーンが AWS CodeStar クイックスタートテンプレートである場合は、AWS CodeStar はアカウント用に作成されたデフォルトのサービスロールを使用します。サービスロールがアカウントに存在しない場合は、新たに作成することができます。詳細については、ステップ 2: AWS CodeStar サービスロールを作成するを参照してください。
  - カスタムツールチェーンテンプレートをアップロードして使用している場合は、ロールを指定する必要があります。AWS CodeStar のサービスロールとポリシーステートメントに基づいてロールを作成できます。 このポリシーステートメントの例については、「 AWSCodeStarServiceRole ポリシー」を参照してください。
- *tags*: オプション。AWS CodeStar プロジェクトにアタッチされているタグ。

> **Note**
>
> これらのタグは、プロジェクトに含まれるリソースに添付されていません。

2. 保存したばかりのファイルがあるディレクトリに移動し、create-project コマンドをもう一度実行します。`--cli-input-json` パラメータを指定します。

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

3. 成功すると、次のようなデータが出力に表示されます：

```
{
    "id": "project-ID",
    "arn": "arn"
}
```

- この出力には、新しいプロジェクトに関する情報が含まれています：
  - `id` 値はプロジェクト ID を表します。
  - `arn` 値は、プロジェクトの ARN を表します。

4. プロジェクトの作成ステータスを確認するには、describe-project コマンドを使用します。--id パラメータを指定します。

```
aws codestar describe-project --id <project_ID>
```

次のようなデータが出力に表示されます。

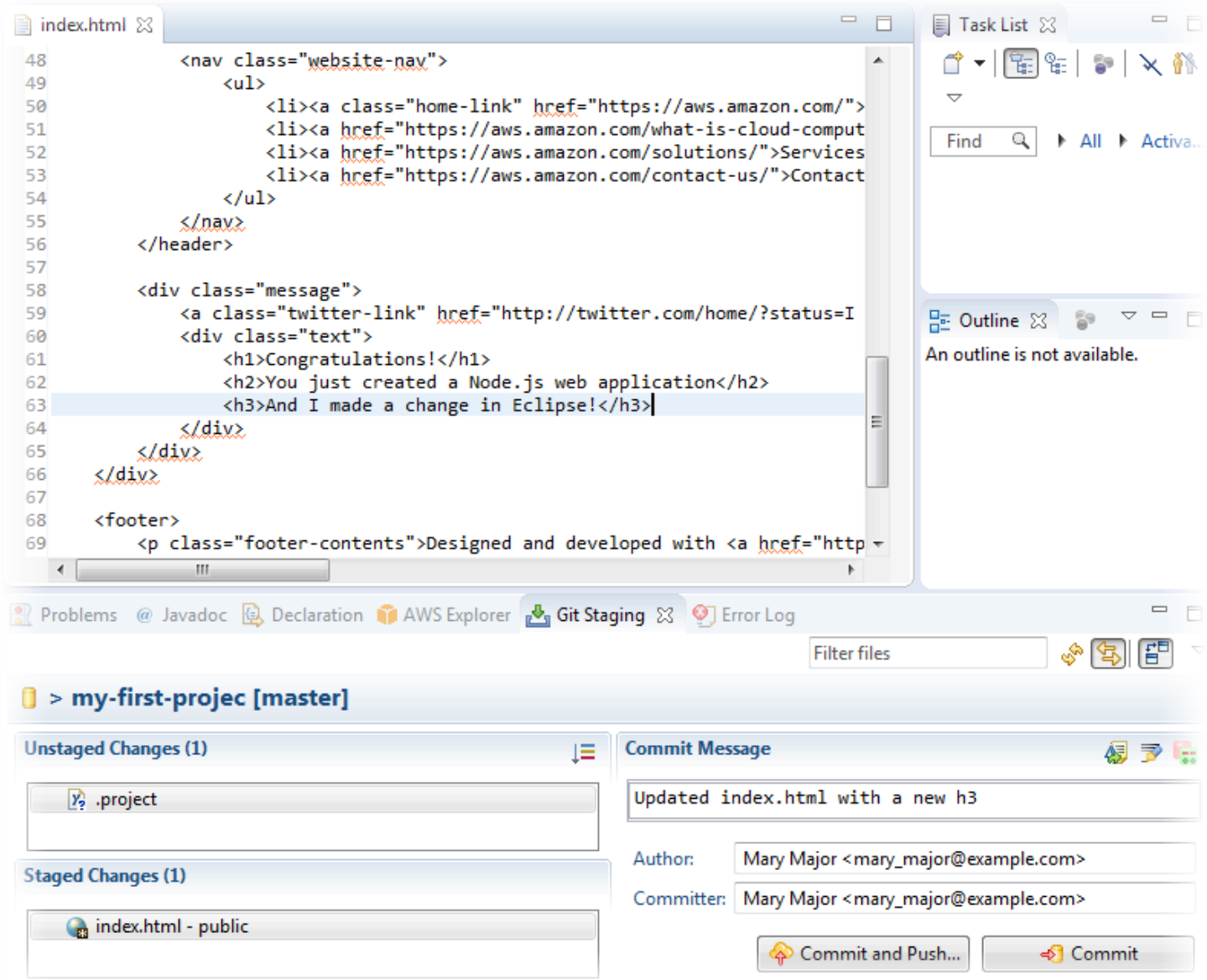
```
{
    "name": "MyProject",
    "id": "myproject",
    "arn": "arn:aws:codestar:us-east-1:account_ID:project/myproject",
    "description": "",
    "createdTimeStamp": 1539700079.472,
    "stackId": "arn:aws:cloudformation:us-east-1:account_ID:stack/awscodestar-
myproject/stack-ID",
    "status": {
        "state": "CreateInProgress"
    }
}
```

- この出力には、新しいプロジェクトに関する情報が含まれています :
  - state 値は、プロジェクトの作成ステータス (例: CreateInProgress または CreateComplete) を表します。

プロジェクトが作成される間、コマンドライン またはお好みの IDE からプロジェクトリポジトリに [add team members] (チームメンバーの追加)、または [configure access] (アクセスの設定) を行うことができます。

AWS CodeStar で IDE を使用する

IDE を AWS CodeStar と統合すると、任意の環境でコードの作成と開発を続けることができます。変更は、コードをコミットしてプッシュするたびに AWS CodeStar プロジェクトに追加されます。



トピック

- AWS CodeStar で AWS Cloud9 を使用する
- AWS CodeStar で Eclipse を使用する
- AWS CodeStar で Visual Studio を使用する

AWS CodeStar で AWS Cloud9 を使用する

AWS Cloud9 を使用してコード変更を行い、AWS CodeStar プロジェクトでソフトウェアを開発することができます。AWS Cloud9 は、ウェブブラウザからアクセスできるオンライン IDE です。この IDE では、リッチなコード編集エクスペリエンスを実現しており、複数のプログラミング言語、

ランタイムデバッガ、組み込みターミナルがサポートされています。Amazon EC2 インスタンスは、バックグラウンドで AWS Cloud9 開発環境をホストしています。この環境では、AWS Cloud9 IDE を使用して、AWS CodeStar プロジェクトコードファイルにアクセスすることができます。詳細については、AWS Cloud9ユーザーガイドを参照してください。

AWS CodeStar コンソールまたは AWS Cloud9 コンソールを使用して、コードを保存するプロジェクトの AWS Cloud9 開発環境を CodeCommit で作成できます。GitHub にコードを保存する AWS CodeStar プロジェクトでは、AWS Cloud9 コンソールのみ使用することができます。このトピックでは、両方のコンソールの使用方法について説明します。

AWS Cloud9 を使用するには、以下が必要です：

- チームメンバーとして AWS CodeStar プロジェクトに追加された IAM ユーザー。
- IAM ユーザーの AWS 認証情報 (AWS CodeStar プロジェクトで CodeCommit にソースコードを保存している場合)。

トピック

- プロジェクトの AWS Cloud9 環境を作成する
- プロジェクトの AWS Cloud9 環境を開く
- AWS Cloud9 環境をプロジェクトチームメンバーと共有する
- プロジェクトから AWS Cloud9 環境を削除する
- AWS Cloud9 での GitHub の使用
- その他のリソース

プロジェクトの AWS Cloud9 環境を作成する

次のステップに従い、AWS CodeStar プロジェクトの AWS Cloud9 開発環境を作成します。

1. 新しいプロジェクトを作成する場合、プロジェクトの作成 の手順を行います。
2. AWS CodeStar コンソールでプロジェクトを開きます。ナビゲーションバーで、[IDE] を選択します。[Create environment] (環境の作成) を選択し、次のステップを実行します。

> **Important**
>
> プロジェクトが、AWS Cloud9 がサポート対象外の AWS リージョンにある場合は、ナビゲーションバーの [IDE] タブに AWS Cloud9 オプションは表示されません。ただし、

開発環境を作成するために AWS Cloud9 コンソールを使用して、新しい環境を開いてから、プロジェクトの AWS CodeCommit リポジトリに接続することができます。次の手順をスキップして、「AWS Cloud9 ユーザーガイド」 の「環境を作成する」、「環境を開く」、「AWS CodeCommit サンプル」を参照してください。サポートされている AWS リージョンのリストについては、「Amazon Web Services 全般のリファレンス」の「AWS Cloud9」を参照してください。

[AWS Cloud9 環境の作成] で、プロジェクトのデフォルトをカスタマイズします。

1. 環境をホストするように Amazon EC2 インスタンスのデフォルトのタイプを変更するには、[Instance type] (インスタンスタイプ)で、インスタンスタイプを選択します。
2. AWS Cloud9 は、インスタンスと通信するための AWS アカウント で、Amazon Virtual Private Cloud (Amazon VPC) を使用しています。AWS アカウント内の Amazon VPC の設定方法によって、次のいずれかの操作を実行します。

アカウントに VPC があり、その VPC に 1 つ以上のサブネットがある	AWS Cloud9 が使用する VPC が、アカウントのデフォルト VPC である	VPC のサブネットが 1 つのみである	この操作を行います
No	—	—	VPC が存在しない場合は、作成してください。[Network settings] (ネットワーク設定) を展開します。[Network(VPC)] (ネットワーク (VPC)) で、[Create VPC] (VPC の作成) を選択し、そのページの手順に従います。詳細については、「AWS Cloud9 ユーザーガイド」 の「AWS Cloud9 用の Amazon VPC の作成」を参照してください。 VPC が存在するがサブネットがない場合は、作成します。[Network settings] (ネットワーク設定) を展開します。[Network (VPC)] (ネットワーク (VPC))

アカウントに VPC があり、その VPC に 1 つ以上のサブネットがある	AWS Cloud9 が使用する VPC が、アカウントのデフォルト VPC である	VPC のサブネットが 1 つのみである	この操作を行います
			で、[Create subnet] (サブネットの作成) を選択し、手順に従います。詳細については、AWS Cloud9 ユーザーガイド の 「AWS Cloud9 用のサブネットの作成」 を参照してください。
はい	Yes	Yes	この手順のステップ 4 に進みます。(AWS Cloud9 ではデフォルト VPC とその単一のサブネットを使用。)
はい	Yes	No	[Subnet] (サブネット) で、AWS Cloud9 で使用する、事前に選択されたデフォルト VPC のサブネットを選択します。
Yes	No	はい/いいえ	[Network (VPC)] (ネットワーク (VPC))で、AWS Cloud9 が使用する VPC を選択します。[Subnet] (サブネット) で、その VPC において AWS Cloud9 が使用するサブネットを選択します。

詳細については、「AWS Cloud9ユーザーガイド」の AWS Cloud9 開発環境用 Amazon VPC の設定 を参照してください。。

3. [Environment name] (環境名) を入力し、オプションで [Environment description] (環境の説明) を追加します。

> **Note**
>
> 環境名はユーザーごとに一意である必要があります。

4. AWS Cloud9 が使用されていないときに環境をシャットダウンするデフォルトの時間を変更するには、[Cost-saving settings] (コスト削減設定) を展開して設定を変更します。
5. [Create environment] (環境の作成) を選択します。

環境を開くには、「プロジェクトの AWS Cloud9 環境を開く」を参照してください。

以下のステップに従い、プロジェクトの環境を 2 つ以上作成します。例えば、ある環境を使用してコードの一部を処理し、別の環境を使用して異なる設定でコードの同じ部分を処理することができます。

プロジェクトの AWS Cloud9 環境を開く

次のステップに従い、AWS CodeStar プロジェクトで作成した AWS Cloud9 開発環境を開きます。

1. AWS CodeStar コンソールでプロジェクトを開き、サイドナビゲーションバーで、[IDE] を選択します。

> **Important**
>
> プロジェクトのソースコードが GitHub に保存されている場合は、ナビゲーションバーに [IDE] は表示されません。ただし、AWS Cloud9 コンソールを使用して、既存の環境を開くことができます。この手順の残りの手順をスキップして、「AWS Cloud9 ユーザーガイド」 の 「環境を開く」 およびAWS Cloud9 での GitHub の使用 を参照してください。

2. [AWS Cloud9 環境] または [共有された AWS Cloud9 環境] で、開く環境の [IDE を開く] を選択します。

AWS Cloud9 IDE を使用して、プロジェクトの AWS CodeCommit リポジトリでコードの操作をすぐにスタートすることができます。詳細については、「AWS Cloud9ユーザーガイド」 の 環境ウィンドウ」、「エディター、タブ、ペイン」、および「ターミナル」、「AWS CodeCommit ユーザーガイド」 の 「基本的な Git コマンド」 を参照してください。

AWS Cloud9 環境をプロジェクトチームメンバーと共有する

AWS CodeStar プロジェクトの AWS Cloud9 開発環境を作成したら、AWS アカウント間で他のユーザー (例: プロジェクトチームメンバー) を招待して、同じ環境にアクセスすることができます。これは、2 人のプログラマーが交互にコーディングし、画面共有しながら同じコードに関するアドバイス

を行うか、同じワークステーションに座ってペアプログラミングを行う場合に特に便利です。環境メンバーは、共有された AWS Cloud9 IDE を使用して、コードエディタでハイライトされた各メンバーのコードの変更を確認し、コーディング中に他のメンバーとテキストチャットすることができます。

プロジェクトにチームメンバーを追加しても、プロジェクトの関連する AWS Cloud9 開発環境へのそのメンバーの参加は自動的に許可されません。プロジェクトの環境にアクセスできるようにプロジェクトのチームメンバーを招待するには、適切な環境のメンバーアクセスロールを確認し、AWS 管理ポリシーをユーザーに適用して、ユーザーを環境に招待する必要があります。詳細については、「AWS Cloud9 ユーザーガイド」の「環境メンバーのアクセスロールについて」および「環境に IAM ユーザーを招待する」を参照してください。

プロジェクトの環境にアクセスできるようにプロジェクトチームメンバーを招待すると、AWS CodeStar コンソールにそのチームメンバーへの環境が表示されます。環境はプロジェクトの AWS CodeStar コンソールの [IDE] タブにある [共有環境] リストに表示されます。このリストを表示するには、チームメンバーがコンソールでプロジェクトを開き、ナビゲーションバーの [IDE] を選択します。

> **Important**
>
> プロジェクトのソースコードが GitHub に保存されている場合は、ナビゲーションバーに [IDE] は表示されません。ただし、AWS Cloud9 コンソールを使用して、プロジェクトチームメンバーを含む、AWS アカウント間で他のユーザーを招待して環境にアクセスすることができます。これを行うには、このガイドの「AWS Cloud9 での GitHub の使用」と「AWS Cloud9 ユーザーガイド」の「環境メンバーのアクセスロールについて」および「環境に IAM ユーザーを招待する」を参照してください。

また、環境にアクセスできるように、プロジェクトチームメンバーではないユーザーを招待することもできます。例えば、ユーザーはプロジェクトのコードを操作するが、プロジェクトの他の部分へアクセスできないようにすることができます。このタイプのユーザーを招待するには、「AWS Cloud9 ユーザーガイド」の「環境メンバーのアクセスロールについて」および「環境にIAM ユーザーを招待する」を参照してください。プロジェクトの環境にアクセスできるようにプロジェクトチームメンバーではないユーザーを招待すると、そのユーザーは、AWS Cloud9 コンソールを使用して、環境にアクセスすることができます。詳細については、「AWS Cloud9 ユーザーガイド」 の「環境を開く」 を参照してください。

プロジェクトから AWS Cloud9 環境を削除する

プロジェクトとそのすべての AWS リソースを AWS CodeStar から削除すると、AWS CodeStar コンソールで作成された関連するすべての AWS Cloud9 開発環境も削除され、復元もできません。開発環境は、プロジェクトを削除する必要なく、プロジェクトから削除することができます。

1. AWS CodeStar コンソールでプロジェクトを開き、サイドナビゲーションバーで、[IDE] を選択します。

   > **Important**
   >
   > プロジェクトのソースコードが GitHub に保存されている場合は、ナビゲーションバーに [IDE] は表示されません。ただし、AWS Cloud9 コンソールを使用して開発環境を削除することができます。この手順の残りの手順をスキップして、「AWS Cloud9ユーザーガイド」 の 「環境を削除する」 を参照してください。

2. [Cloud9 environments] (Cloud9 環境) で削除する環境を選択し、[Delete] (削除) を選択します。
3. 開発環境の削除を確認するため **`delete`** を入力し、[Delete] (削除) を選択します。

   > **Warning**
   >
   > 削除した開発環境を復元することはできません。この環境でコミットされていないコードの変更はすべて、失われます。

AWS Cloud9 での GitHub の使用

GitHub にソースコードが格納されている AWS CodeStar プロジェクトの場合、AWS CodeStar コンソールは AWS Cloud9 開発環境での直接操作をサポートしていません。ただし、AWS Cloud9 コンソールを使用して、GitHub リポジトリのソースコードを操作できます。

1. AWS Cloud9 コンソールを使用して、AWS Cloud9 開発環境を作成します。詳細については、「AWS Cloud9 ユーザーガイド」 の 「環境の作成」 を参照してください。
2. AWS Cloud9 コンソールを使用して、開発環境を開きます。詳細については、「AWS Cloud9 ユーザーガイド」 の 「環境を開く」 を参照してください。
3. IDE で、ターミナルセッションを使用して GitHub リポジトリに接続します (クローニングと呼ばれるプロセス)。ターミナルセッションが実行されていない場合は、IDE のメニューバーで [Window, New Terminal](ウインドウ、新しいターミナル) を選択します。GitHub リポジト

リのクローン作成に使用するコマンドについては、GitHub のヘルプウェブサイトの [Cloning a Repository] (リポジトリのクローン作成) を参照してください。

GitHub リポジトリのメインページに移動するには、AWS CodeStar コンソールでプロジェクトを開き、サイドナビゲーションバーで [コード] を選択します。

4. IDE の [Environment] (環境) ウィンドウとエディタタブを使用して、コードを表示、変更、保存します。詳細については、「AWS Cloud9ユーザーガイド」 の 「環境」ウィンドウ および エディター、タブ、ペイン を参照してください。
5. IDE のターミナルセッションの Git を使用して、コードの変更をリポジトリにプッシュし、リポジトリの他のコードの変更を定期的にリポジトリからプルできます。詳細については、GitHub ヘルプウェブサイトの [Pushing to a Remote Repository] (リモートリポジトリへのプッシュ) および [Fetching a Remote Repository] (リモートリポジトリの取得) を参照してください。Git コマンドについては、GitHub のヘルプウェブサイトの 「Git Cheatsheet」 を参照してください。

> **Note**
>
> リポジトリからコードをプッシュまたはプルするたびに GitHub のサインイン認証情報の入力を求めるように Git に指示する場合は、認証情報ヘルパーを使用できます。詳細については、GitHub Help ウェブサイトの [Caching Your GitHub Password in Git] (Git に GitHub パスワードをキャッシュする) を参照してください。

その他のリソース

AWS Cloud9 の使用に関する詳細は、「AWS Cloud9 ユーザーガイド」の以下を参照してください：

- チュートリアル
- 環境を使用する
- IDE を操作する
- サンプル

AWS CodeStar で Eclipse を使用する

Eclipse を使用してコード変更を行い、AWS CodeStar プロジェクトでソフトウェアを開発することができます。Eclipse で AWS CodeStar プロジェクトコードを編集し、AWS CodeStar プロジェクトのソースリポジトリに変更をコミットしてプッシュできます。

> **Note**
>
> このトピックの情報は、ソースコードを CodeCommit に格納する AWS CodeStar プロジェクトにのみ適用されます。AWS CodeStar プロジェクトでソースコードが GitHub に保存されている場合は、EGit for Eclipse などのツールを使用できます。詳細については、EGit ウェブサイトの [EGit Documentation] (EGit ドキュメント) を参照してください。

AWS CodeStar プロジェクトがソースコードを CodeCommit に保存する場合は、AWS CodeStar をサポートする AWS Toolkit for Eclipse のバージョンをインストールする必要があります。また、所有者または共同編集者のロールを持つ AWS CodeStar プロジェクトチームのメンバーでなければなりません。

Eclipse を使用するには、以下の条件を満たす必要があります：

- チームメンバーとして AWS CodeStar プロジェクトに追加された IAM ユーザー。
- IAM ユーザーの Git 認証情報 (サインイン認証情報)(AWS CodeStar プロジェクトで CodeCommit にソースコードを保存している場合)。
- ローカルコンピュータに Eclipse と AWS Toolkit for Eclipse をインストールするための十分な権限。

トピック

- 手順 1: AWS Toolkit for Eclipse をインストールする
- ステップ 2: AWS CodeStar プロジェクトを Eclipse にインポートする
- ステップ 3: Eclipse で AWS CodeStar プロジェクトコードを編集する

手順 1: AWS Toolkit for Eclipse をインストールする

Toolkit for Eclipse は、Eclipse に追加できるソフトウェアパッケージです。Eclipse の他のソフトウェアパッケージと同じ方法でインストールおよび管理されます。AWS CodeStar ツールキットは Toolkit for Eclipse の一部として含まれています。

AWS CodeStar モジュールを含む Toolkit for Eclipse をインストールするには

1. Eclipse をローカルコンピュータにインストールします。サポートされている Eclipse のバージョンには、Luna、Mars、Neon があります。

2. Toolkit for Eclipse をダウンロードしてインストールします。詳細については、「AWS Toolkit for Eclipse 入門ガイド」 を参照してください。
3. Eclipse で [Help] (ヘルプ) を選択してから、[Install New Software] (新しいソフトウェアのインストール) を選択します。
4. [Available Software] (利用可能なソフトウェア) で、[Add] (追加) を選択します。
5. [Add Repository] (リポジトリの追加) で、[Archive] (アーカイブ) を選択し、.zip ファイルを保存した場所を参照して、ファイルを開きます。[Name] (名前) を空欄にし、[OK] を選択します。
6. [利用可能なソフトウェア] で、[すべて選択] を選択して、[AWS コア管理ツール] と [開発者用ツール] の両方を選択し、[次へ] を選択します。
7. [Install Details] (インストールの詳細) で、[Next] (次へ) を選択します。
8. [Review Licenses] (ライセンスの確認) で、ライセンス契約を確認します。[I accept the terms of the license agreement] (ライセンス契約の条項に同意します) を選択し、[Finish] (完了) を選択します。Eclipse を再起動します。

ステップ 2: AWS CodeStar プロジェクトを Eclipse にインポートする

Toolkit for Eclipse をインストールしたら、AWS CodeStar プロジェクトをインポートし、IDE からコードを編集、コミット、プッシュすることができます。

> **Note**
>
> Eclipse の単一のワークスペースに複数の AWS CodeStar プロジェクトを追加することができますが、あるプロジェクトから別のプロジェクトに変更するときにプロジェクトの認証情報を更新する必要があります。

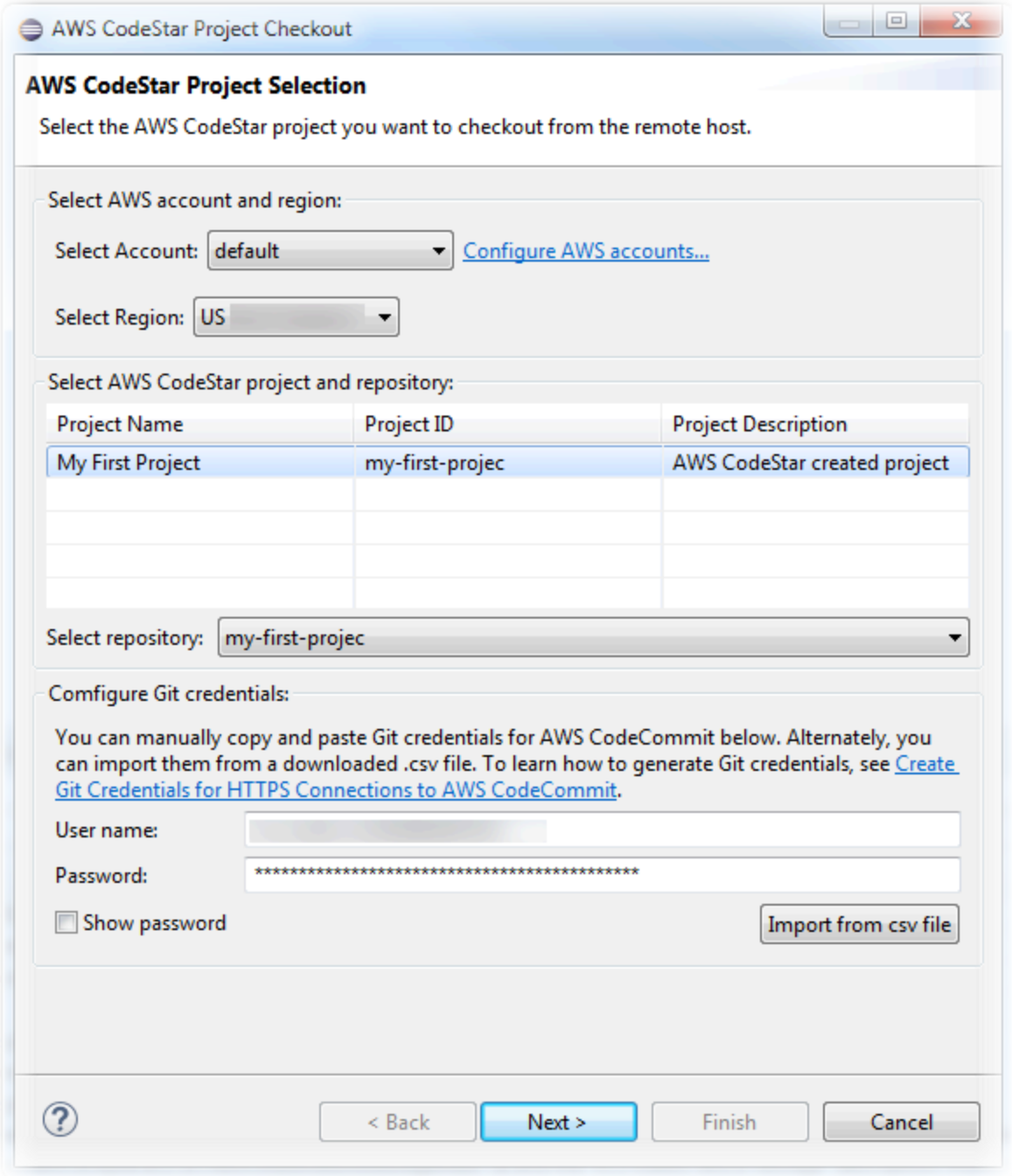
AWS CodeStar プロジェクトをインポートするには

1. AWS メニューから [AWS CodeStar プロジェクトのインポート] を選択します。または、[File] (ファイル)、[Import] (インポート) の順に選択します。[Select] で、[AWS] を展開して、[AWS CodeStar プロジェクト] を選択します。

   [Next] (次へ) をクリックします。
2. [AWS CodeStar プロジェクトの選択] で、AWS プロファイルと AWS CodeStar プロジェクトがホストされている AWS リージョンを選択します。お使いのコンピュータにアクセスキーとシー

クレットキーが設定された AWS プロファイルがない場合は、[AWS アカウントの設定] を選択し、指示に従います。

[AWS CodeStar プロジェクトとリポジトリの選択] で、リストから AWS CodeStar プロジェクトを選択します。[Git 認証情報の設定] に、プロジェクトのリポジトリにアクセスするために生成したユーザー名とパスワードを入力します。(git 認証情報がない場合は、「ご利用開始にあたって」を参照してください。) [Next] (次へ) を選択します。



3. プロジェクトのリポジトリのすべてのブランチがデフォルトで選択されます。1 つまたは複数のブランチをインポートしたくない場合は、ボックスをクリアして、[Next] (次へ) を選択します。
4. [Local Destination] (ローカルの作成先) で、インポートウィザードがコンピュータ上でローカルリポジトリを作成する場所を選択し、[Finish] (完了) を選択します。
5. [Project Explorer] で、プロジェクトツリーを展開して、AWS CodeStar プロジェクトのファイルを参照します。

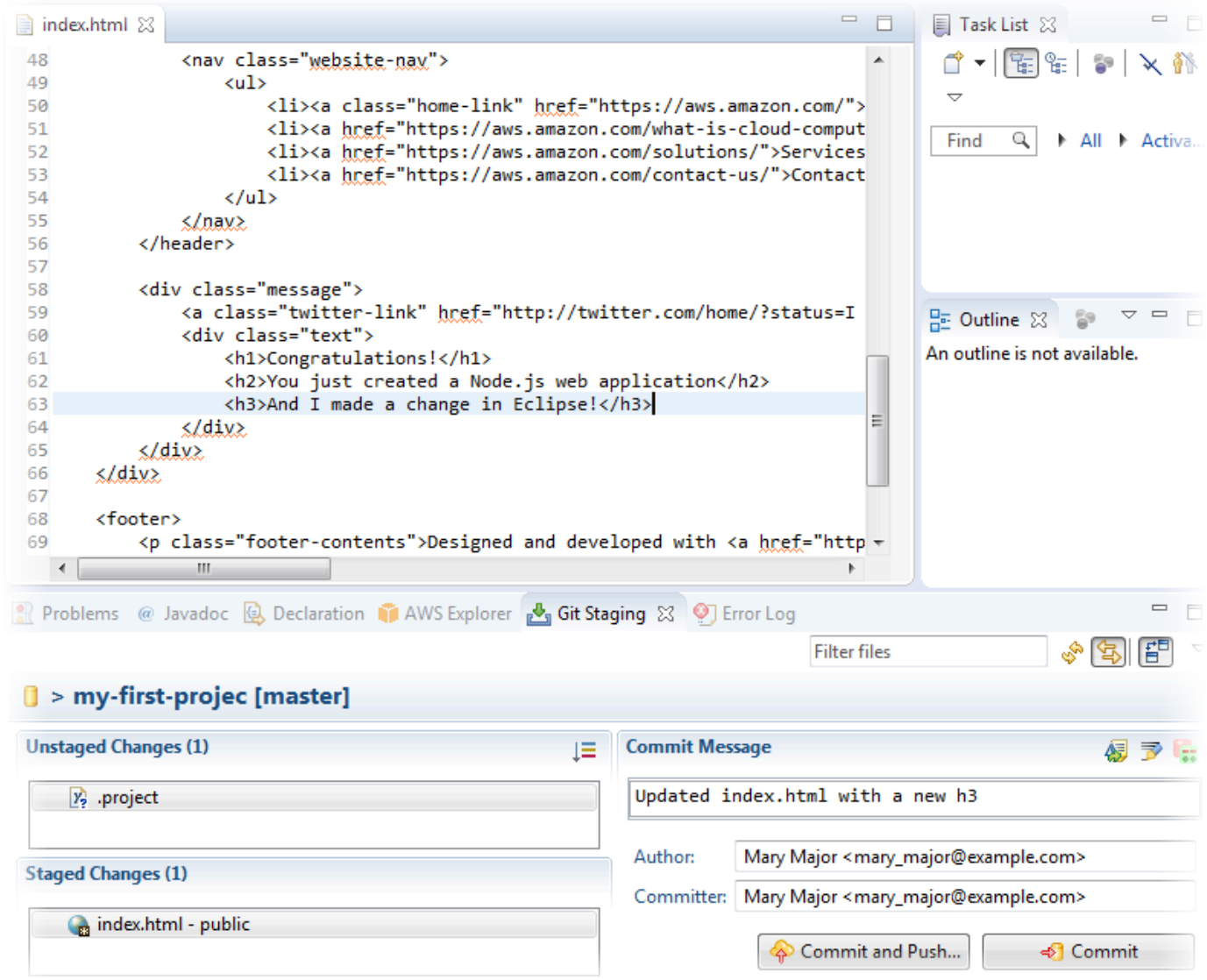
ステップ 3: Eclipse で AWS CodeStar プロジェクトコードを編集する

AWS CodeStar プロジェクトを Eclipse ワークスペースにインポートしたら、プロジェクトのコードを編集して変更を保存し、コードをコミットしてプロジェクトのソースリポジトリにプッシュできます。これは Eclipse 用の EGit プラグインを使用する Git リポジトリと同じプロセスです。詳細については、Eclipse ウェブサイトの 「EGit ユーザーガイド」 を参照してください。

プロジェクトコードを編集し、AWS CodeStar プロジェクトのソースリポジトリに最初にコミットするには

1. [Project Explorer] で、プロジェクトツリーを展開して、AWS CodeStar プロジェクトのファイルを参照します。
2. 1 つまたは複数のコードファイルを編集し、変更を保存します。
3. 変更をコミットする準備ができたら、そのファイルのコンテキストメニューを開き、[Team] (チーム)、[Commit] (コミット) の順に選択します。

   プロジェクトビューで [Git Staging] (Git をステージ) ウィンドウが既に開いている場合は、このステップをスキップします。
4. [Git Staging] (Git をステージ) で、変更されたファイルを [Staged Changes] (ステージングされた変更) に移動して変更をステージします。[Commit Message] (コミットメッセージ) にコミットメッセージを入力し、[Commit and Push] (コミットとプッシュ) を選択します。



コード変更のデプロイを表示するには、プロジェクトのダッシュボードに戻ります。詳細については、「ステップ 3: プロジェクトを表示する」を参照してください。

AWS CodeStar で Visual Studio を使用する

Visual Studio を使用してコード変更を行い、AWS CodeStar プロジェクトでソフトウェアを開発することができます。

> **Note**
>
> Visual Studio for Macでは AWS ツールキット がサポートされていないため、AWS CodeStar で使用できません。

このトピックの情報は、ソースコードを CodeCommit に格納する AWS CodeStar プロジェクトにのみ適用されます。AWS CodeStar プロジェクトでソースコードが GitHub に保存されている場合は、GitHub Extension for Visual Studio などのツールを使用できます。詳細については、GitHub Extension for Visual Studio ウェブサイトの [Overview] (概要) ページ、および GitHub ウェブサイトの [Getting Started with GitHub for Visual Studio] (GitHub for Visual Studio 入門) を参照してください。

Visual Studio を使用して AWS CodeStar プロジェクトのソースリポジトリ内のコードを編集するには、AWS CodeStar をサポートする AWS Toolkit for Visual Studio のバージョンをインストールする必要があります。所有者または寄稿者のロールを持つ AWS CodeStar プロジェクトチームのメンバーでなければなりません。

Visual Studio を使用するには、以下の条件を満たす必要があります:

- チームメンバーとして AWS CodeStar プロジェクトに追加された IAM ユーザー。
- IAM ユーザーの AWS 認証情報 (例：アクセスキーやシークレットキー)。
- ローカルコンピュータに Visual Studio と AWS Toolkit for Visual Studio をインストールするための十分な権限。

Toolkit for Visual Studio は、Visual Studio に追加できるソフトウェアパッケージです。Visual Studio の他のソフトウェアパッケージと同じ方法でインストールおよび管理されます。

AWS CodeStar モジュールを使用して Toolkit for Visual Studio をインストールし、プロジェクトリポジトリへのアクセスを設定するには

1. ローカルコンピュータに Visual Studio をインストールします。
2. Toolkit for Visual Studio をダウンロードしてインストールし、.zip ファイルをローカルフォルダまたはディレクトリに保存します。[Getting Started with the AWS Toolkit for Visual Studio] ページに、AWS 認証情報を入力またはインポートし、[Save and Close] を選択します。
3. Visual Studio で、[Team Explorer] (チームエクスプローラー) を開きます。[Hosted Service Providers] (ホストされているサービスプロバイダー) で、[CodeCommit] を検索し、[Connect] (接続) を選択します。
4. [Manage Connections] (接続を管理)で、[Clone] (クローン) を選択します。プロジェクトのリポジトリとそのリポジトリのクローン作成先のローカルコンピュータのフォルダを選択し、[OK] を選択します。

5. Git 認証情報を作成するように求められたら、[Yes] (はい) を選択します。ツールキットは、ユーザーに代わって認証情報を作成しようとします。安全な場所に認証情報ファイルを保存します。これは、これらの認証情報を保存する必要がある唯一の機会です。ツールキットがユーザーの代わりに認証情報を作成できない場合、または [No] を選択した場合は、独自の Git 認証情報を作成して提供する必要があります。詳細については、「変更をコミットするようコンピュータを設定するには (IAM ユーザー)」を参照するか、オンラインの指示に従ってください。

   プロジェクトのクローン作成が完了すると、Visual Studio でコードを編集し、プロジェクトのリポジトリへの変更を CodeCommit でコミットしてプッシュする準備が整います。

AWS CodeStar プロジェクトで AWS リソースを変更する

AWS CodeStar でプロジェクトを作成したら、AWS CodeStar がプロジェクトに追加する AWS リソースのデフォルトセットを変更することができます。

サポートされているリソースの変更

次の表には、AWS CodeStar プロジェクトのデフォルトの AWS リソースへのサポート対象の変更を示します。

変更	メモ
AWS CodePipeline にステージを追加します。	「AWS CodePipeline にステージを追加する」を参照してください。
Elastic Beanstalk 環境設定を変更します。	「AWS Elastic Beanstalk 環境設定の変更」を参照してください。
AWS Lambda 関数のコードまたは設定、IAM ロール、Amazon API Gateway の API を変更します。	「ソースコードの AWS Lambda 関数を変更する」を参照してください。
AWS Lambda プロジェクトにリソースを追加し、新しいリソースを作成してアクセスするためのアクセス許可を展開します。	「リソースをプロジェクトに追加する」を参照してください。
AWS Lambda 関数用の CodeDeploy を使用したトラフィック移行を追加します。	「AWS Lambda プロジェクトのトラフィックを移行する」を参照してください。

変更	メモ
AWS X-Ray サポートの追加	「プロジェクトのトレースを有効にする」を参照してください。
AWS CodeBuild で実行されるユニットテストのビルドフェーズを追加するには、プロジェクトの buildspec.yml ファイルを編集します。	サーバーレスプロジェクトのチュートリアルの「ステップ 7: ウェブサービスにユニットテストを追加する」を参照してください。
独自の IAM ロールをプロジェクトに追加します。	「IAM ロールをプロジェクトに追加する」を参照してください。
IAM ロールの定義の変更	アプリケーションスタックで定義されているロールの場合。ツールチェーンまたは AWS CloudFormation スタックで定義されているロールを変更することはできません。
Lambda プロジェクトを変更してエンドポイントを追加します。	
EC2 プロジェクトを変更してエンドポイントを追加します。	
Elastic Beanstalk プロジェクトを変更してエンドポイントを追加します。	
プロジェクトを編集し、Prod ステージとエンドポイントを追加します。	「Prod ステージとエンドポイントをプロジェクトに追加する」を参照してください。
SSM パラメータを AWS CodeStar プロジェクトで安全に使用します。	「the section called “AWS CodeStar プロジェクトで SSM パラメータを安全に使用する”」を参照してください。

以下の変更はサポートされていません。

- 別のデプロイターゲット (たとえば、AWS CodeDeploy ではなく AWS Elastic Beanstalk へデプロイ) に切り替えます。
- フレンドリウェブエンドポイント名を追加します。

- CodeCommit リポジトリ名を変更します (CodeCommit に接続された AWS CodeStar プロジェクトの場合)。
- GitHub リポジトリに接続された AWS CodeStar プロジェクトの場合、GitHub レポジトリを切断してから、そのプロジェクトにリポジトリを再接続するか、そのプロジェクトに他のリポジトリを接続します。パイプラインの [ソース] ステージで、CodePipeline コンソール (AWS CodeStar コンソールではない) を使用して、GitHub を切断して再接続することができます。ただし、[ソース] ステージを別の GitHub リポジトリに再接続すると、プロジェクトの AWS CodeStar ダッシュボードの [リポジトリ] および [問題] タイルの情報が間違っているか古くなっている場合があります。GitHub リポジトリを切断しても、そのリポジトリの情報はコミット履歴から削除されず、GitHub はプロジェクトの AWS CodeStar プロジェクトダッシュボードでタイルを発行します。この情報を削除するには、GitHub ウェブサイトを使用して AWS CodeStar プロジェクトから GitHub へのアクセスを無効にします。アクセスを取り消すには、GitHub ウェブサイトで、GitHub アカウントプロファイルの設定ページの [Authorized OAuth Apps] (許可されたOAuth Apps) セクションを使用します。
- CodeCommit リポジトリ (CodeCommit に接続された AWS CodeStar プロジェクト用) を切断してから、そのプロジェクトにリポジトリを再接続するか、そのプロジェクトに他のリポジトリを接続します。

AWS CodePipeline にステージを追加する

AWS CodeStar がプロジェクトで作成するパイプラインに新しいステージを追加できます。詳細については、「AWS CodePipeline ユーザーガイド」の「AWS CodePipeline でパイプラインを編集する」を参照してください。

> **Note**
>
> 新しいステージが、AWS CodeStar で作成されなかった AWS リソースに依存する場合は、パイプラインが破損する可能性があります。これは、AWS CodeStar が AWS CodePipeline 用に作成した IAM ロールは、デフォルトではこれらのリソースにアクセスできない可能性があるためです。
>
> AWS CodeStar によって作成されたものではない AWS リソースへのアクセスを AWS CodePipeline に許可するには、AWS CodeStar によって作成された IAM ロールを変更する必要があります。AWS CodeStar がプロジェクトの定期的な更新チェックを実行するたびに IAM ロールの変更を削除する可能性があるため、これはサポートされていません。

AWS Elastic Beanstalk 環境設定の変更

AWS CodeStar がプロジェクトで作成する Elastic Beanstalk 環境の設定を変更することができます。例えば、AWS CodeStar プロジェクトのデフォルト Elastic Beanstalk 環境を単一インスタンスからロードバランスに変更できます。これを行うには、プロジェクトのリポジトリ内の `template.yml` ファイルを編集します。プロジェクトのワーカーロールのアクセス許可を変更することが必要になる場合もあります。テンプレートの変更をプッシュした後、AWS CodeStar および AWS CloudFormation はリソースをプロビジョニングします。

`template.yml` ファイルの編集方法については、「Template.yml ファイルを使用してアプリケーションリソースを変更する」を参照してください。Elastic Beanstalk 環境の詳細については、「AWS Elastic Beanstalk デベロッパーガイド」の「AWS Elastic Beanstalk 環境マネジメントコンソール」を参照してください。

ソースコードの AWS Lambda 関数を変更する

AWS CodeStar がプロジェクトで作成する Lambda 関数のコードや設定、関連する IAM ロール、API Gateway API を変更できます。これを行うには、AWS サーバーレスアプリケーションモデル (AWS SAM) をプロジェクトの CodeCommit リポジトリ内の `template.yaml` ファイルと共に使用することをお勧めします。この `template.yaml` ファイルは、関数の名前、ハンドラ、ランタイム、IAM ロール、および API Gateway の API を定義します。詳細については、GitHub ウェブサイトで「AWS SAM を使用してサーバーレスアプリケーションを作成する方法」を参照してください。

プロジェクトのトレースを有効にする

AWS X-Ray にはトレース機能が搭載されています。この機能は、分散アプリケーションのパフォーマンス動作 (例: 応答時間のレイテンシー) の分析に使用できます。トレースを AWS CodeStar プロジェクトに追加したら、AWS X-Ray コンソールを使用して、アプリケーションビューおよび応答時間を表示することができます。

> **Note**
>
> 以下のプロジェクトサポートの変更により作成された以下のプロジェクトでは、これらのステップを使用できます。
>
> - 任意の Lambda プロジェクト。

- 2018 年 8 月 3 日以降に作成された Amazon EC2 または Elastic Beanstalk プロジェクトの場合、AWS CodeStar により、プロジェクトリポジトリに `/template.yml` ファイルがプロビジョニングされています。

AWS CodeStar のテンプレートにはそれぞれ、アプリケーションの AWS ランタイム依存関係 (データベーステーブルと Lambda 関数) をモデリングする AWS CloudFormation ファイルが含まれています。このファイルは、ファイル `/template.yml` のソースリポジトリに保存されています。

このファイルを変更してトレースに追加するには、AWS X-Ray リソースを `Resources` セクションに追加します。また、AWS CloudFormation でリソースを作成できるように、プロジェクトの IAM アクセス許可を変更することができます。テンプレート要素およびフォーマットについては、「AWS リソースタイプのリファレンス」を参照してください。

テンプレートをカスタマイズする大まかなステップを以下に示します。

1. ステップ 1: トレースに必要な IAM のワーカーロールを編集する
2. ステップ 2: トレース用に template.yml ファイルを変更する
3. ステップ 3: トレース用にテンプレートの変更をコミットおよびプッシュする
4. ステップ 4: トレース用の AWS CloudFormation スタック更新をモニタリングする

ステップ 1: トレースに必要な IAM のワーカーロールを編集する

ステップ 1 および 4 を実行するには、管理者ユーザーとしてサインインする必要があります。このステップでは、Lambda プロジェクトのアクセス許可を編集する例を示します。

> **Note**
>
> プロジェクトがアクセス許可の境界ポリシーでプロビジョニングされた場合は、このステップをスキップできます。
> 2018 年 12 月 6 日 (PDT) 以降に作成されたプロジェクトの場合、AWS CodeStar により、アクセス許可の境界ポリシーを使用してプロジェクトがプロビジョニングされています。

1. AWS Management Console にサインインし、AWS CodeStar コンソール (https://console.aws.amazon.com/codestar/) を開きます。

2. プロジェクトを作成するか、`template.yml file` を含む既存のプロジェクトを選択して、[Project resources] (プロジェクトリソース) ページを開きます。
3. [Project Resources] (プロジェクトリソース) のリソースリストで、CodeStarWorker/Lambda ロール用に作成した IAM ロールを検索します。ルール名の形式は次のとおりです: `role/CodeStarWorker-`*`Project_name`*`-lambda-`*`Function_name`*。ロールの ARN を選択します。
4. ロールが IAM コンソールで開きます。[Attach policies] (ポリシーの添付) を選択します。`AWSXrayWriteOnlyAccess` ポリシーを検索し、その横にあるボックスを選択して、[Attach Policy] (ポリシーの添付) を選択します。

ステップ 2: トレース用に template.yml ファイルを変更する

1. https://console.aws.amazon.com/codestar/ で AWS CodeStar コンソールを開きます。
2. サーバーレスプロジェクトを選択し、[Code] (コード) ページを開きます。リポジトリの最上位で、`template.yml` ファイルを検索して編集します。`Resources` で、リソースを `Properties` セクションに貼り付けます。

```
      Tracing: Active
```

この例では、変更後のテンプレートを示します。

```
Resources:
  GetHelloWorld:
    Type: AWS::Serverless::Function
    Properties:
      Handler: index.get
      Runtime: nodejs4.3
      Tracing: Active # Enable X-Ray tracing for the function
      Role:
        Fn::ImportValue:
          !Join ['-', [!Ref 'ProjectId', !Ref 'AWS::Region', 'LambdaTrustRole']]
      Events:
        GetEvent:
          Type: Api
          Properties:
            Path: /
            Method: get
```

ステップ 3: トレース用にテンプレートの変更をコミットおよびプッシュする

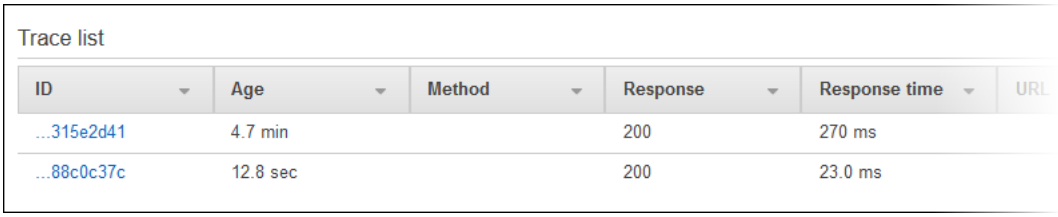
- `template.yml` ファイルの変更をコミットおよびプッシュします。

> **Note**
>
> これにより、パイプラインが開始されます。IAM アクセス許可を更新する前に変更をコミットすると、パイプラインスタート後、AWS CloudFormation スタック更新でエラーが発生し、このスタック更新はロールバックされます。この問題が発生した場合は、アクセス許可を修正し、パイプランを再起動します。

ステップ 4: トレース用の AWS CloudFormation スタック更新をモニタリングする

1. AWS CloudFormation スタック更新は、プロジェクトのパイプランでデプロイステージのスタート時にスタートします。スタック更新のステータスを確認するには、AWS CodeStar ダッシュボードで、パイプラインの AWS CloudFormation ステージを選択します。

   AWS CloudFormation のスタック更新でエラーが表示される場合は、「AWS CloudFormation: アクセス許可の不足により、スタックの作成がロールバックされた」のトラブルシューティングガイドラインを参照してください。ワーカーロールのアクセス許可が不足している場合は、プロジェクトの Lambda ワーカーロールに添付されているポリシーを編集します。「ステップ 1: トレースに必要な IAM のワーカーロールを編集する」を参照してください。
2. パイプラインが正常に完了したことを確認するには、ダッシュボードを使用します。アプリケーションでトレースが有効になりました。
3. トレースが有効になったことを確認するには、Lambda コンソールで関数の詳細を表示します。
4. プロジェクトのアプリケーションエンドポイントを選択します。このアプリケーションとのやり取りがトレースされます。トレースの情報は、AWS X-Ray コンソールで確認できます。



リソースをプロジェクトに追加する

すべてのプロジェクトの AWS CodeStar テンプレートにはそれぞれ、アプリケーションの AWS ランタイム依存関係 (データベーステーブルや Lambda 関数など) をモデリングする AWS CloudFormation ファイルが含まれています。これは、ファイル `/template.yml` のソースリポジトリに保存されています。

> **Note**
>
> 以下のプロジェクトサポートの変更により作成された以下のプロジェクトでは、これらのステップを使用できます。
>
> - 任意の Lambda プロジェクト。
> - 2018 年 8 月 3 日以降に作成された Amazon EC2 または Elastic Beanstalk プロジェクトの場合、AWS CodeStar により、プロジェクトリポジトリに `/template.yml` ファイルがプロビジョニングされています。

このファイルを変更するには、AWS CloudFormation リソースを `Resources` セクションに追加します。`template.yml` ファイルを変更することで、AWS CodeStar および AWS CloudFormation で、新しいリソースをプロジェクトに追加できるようになります。一部のリソースでは、他のアクセス許可をプロジェクトの CloudFormation ワーカーロールのポリシーに追加する必要があります。テンプレート要素およびフォーマットについては、「AWS リソースタイプのリファレンス」を参照してください。

プロジェクトに追加する必要のあるリソースを決定したら、テンプレートをカスタマイズするための大まかな手順に従って実行します。AWS CloudFormation リソースとその必要なプロパティのリストについては、「AWS リソースタイプのリファレンス」を参照してください。

1. ステップ 1: IAM で CloudFormation ワーカーロールを編集する (任意)
2. ステップ 2: template.yml ファイルを変更する
3. ステップ 3: テンプレートの変更をコミットおよびプッシュする
4. ステップ 4: AWS CloudFormation スタック更新をモニタリングする
5. ステップ 5: インラインポリシーでリソースのアクセス許可を追加する

このセクションのステップに従って、AWS CodeStar プロジェクトテンプレートの変更とリソースの追加を行い、プロジェクトの CloudFormation ワーカーロールのアクセス許可を IAM で展開します。この例では、AWS::SQS::Queue リソースは、`template.yml` ファイルに追加されます。この変更によって、AWS CloudFormation の自動応答が開始され、Amazon Simple キューサービスキューがプロジェクトに追加されます。

ステップ 1: IAM で CloudFormation ワーカーロールを編集する

ステップ 1 および 5 を実行するには、管理者ユーザーとしてサインインする必要があります。

> **Note**
>
> プロジェクトがアクセス許可の境界ポリシーでプロビジョニングされた場合は、このステップをスキップできます。
> 2018 年 12 月 6 日 (PDT) 以降に作成されたプロジェクトの場合、AWS CodeStar により、アクセス許可の境界ポリシーを使用してプロジェクトがプロビジョニングされています。

1. AWS Management Consoleにサインインして、https://console.aws.amazon.com/codestar/ で、AWS CodeStar コンソールを開きます。
2. プロジェクトを作成するか、`template.yml file` を含む既存のプロジェクトを選択して、[Project resources] (プロジェクトリソース) ページを開きます。
3. [Project Resources] (プロジェクトリソース) のリソースリストで、CodeStarWorker/AWS CloudFormation ロール用に作成した IAM ロールを検索します。ルール名の形式は次のとおりです: `role/CodeStarWorker-`*`Project_name`*`-CloudFormation`。
4. ロールが IAM コンソールで開きます。[Permissions] (アクセス許可) タブの [Inline Policies] (インラインポリシー) で、サービスロールポリシーの列を開き、[Edit Policy] (ポリシーの編集) を選択します。
5. [JSON] タブを選択してポリシーを編集します。

   > **Note**
   >
   > ワーカーロールに添付されるポリシーは `CodeStarWorkerCloudFormationRolePolicy` です。

6. [JSON] フィールドで、`Statement` 要素に次のポリシーステートメントを追加します。

```
{
  "Action": [
    "sqs:CreateQueue",
    "sqs:DeleteQueue",
    "sqs:GetQueueAttributes",
    "sqs:SetQueueAttributes",
    "sqs:ListQueues",
```

```
    "sqs:GetQueueUrl"
  ],
  "Resource": [
    "*"
  ],
  "Effect": "Allow"
}
```

7. [Review policy] (ポリシーの確認) を選択して、ポリシーにエラーがないことを確認し、[Save changes] (変更の保存) を選択します。

ステップ 2: template.yml ファイルを変更する

1. https://console.aws.amazon.com/codestar/ で AWS CodeStar コンソールを開きます。
2. サーバーレスプロジェクトを選択し、[Code] (コード) ページを開きます。リポジトリの最上位にある `template.yml` の場所を書き留めます。
3. リポジトリの `template.yml` ファイルを編集するには、IDE、コンソール、またはコマンドラインをローカルリポジトリで使用します。リソースを `Resources` セクションに貼り付けます。この例では、以下のテキストをコピーすると、`Resources` セクションが追加されます。

```
Resources:
  TestQueue:
    Type: AWS::SQS::Queue
```

この例では、変更後のテンプレートを示します。

```
Resources:
  HelloWorld:
    Type: AWS::Serverless::Function
    Properties:
      Handler: index.handler
      Runtime: python3.6
      Role:
        Fn::ImportValue:
          !Join ['-', [!Ref 'ProjectId', !Ref 'AWS::Region', 'LambdaTrustRole']]
      Events:
        GetEvent:
          Type: Api
          Properties:
            Path: /
            Method: get
        PostEvent:
          Type: Api
          Properties:
            Path: /
            Method: post

  TestQueue:
    Type: AWS::SQS::Queue
```

ステップ 3: テンプレートの変更をコミットおよびプッシュする

- ステップ 2 で保存した `template.yml` ファイルの変更をコミットおよびプッシュします。

> **Note**
>
> これにより、パイプラインが開始されます。IAM アクセス許可を更新する前に変更をコミットすると、パイプライン開始後、AWS CloudFormation スタック更新でエラーが発生します。その結果、このスタック更新はロールバックされます。この問題が発生した場合は、アクセス許可を修正し、パイプランを再起動します。

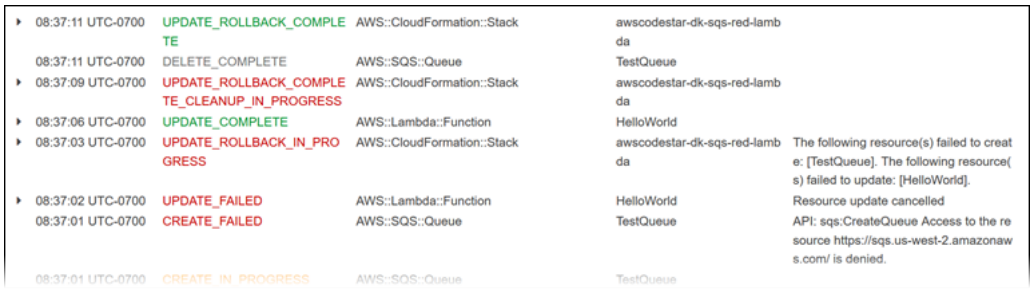
ステップ 4: AWS CloudFormation スタック更新をモニタリングする

1. プロジェクトのパイプランでデプロイステージがスタートされると、AWS CloudFormation スタック更新がスタートします。パイプラインの AWS CloudFormation ステージを AWS CodeStar ダッシュボードで選択し、スタック更新を表示します。

   トラブルシューティング：

   必要なリソースのアクセス許可が不足している場合、このスタック更新は失敗します。AWS CodeStar ダッシュボードビューのエラーステータスを表示して、プロジェクトのパイプラインを確認します。

   パイプラインのデプロイステージの [CloudFormation] リンクを選択して、AWS CloudFormation コンソールで障害をトラブルシューティングします。コンソールの [Events] (イベント) リストで、プロジェクトを選択して、スタック作成の詳細を表示します。障害の詳細が記載されたメッセージが表示されます。この例では、`sqs:CreateQueue` のアクセス許可が不足しています。



08:37:11 UTC-0700	UPDATE_ROLLBACK_COMPLETE	AWS::CloudFormation::Stack	awscodestar-dk-sqs-red-lambda	
08:37:11 UTC-0700	DELETE_COMPLETE	AWS::SQS::Queue	TestQueue	
08:37:09 UTC-0700	UPDATE_ROLLBACK_COMPLETE_CLEANUP_IN_PROGRESS	AWS::CloudFormation::Stack	awscodestar-dk-sqs-red-lambda	
08:37:06 UTC-0700	UPDATE_COMPLETE	AWS::Lambda::Function	HelloWorld	
08:37:03 UTC-0700	UPDATE_ROLLBACK_IN_PROGRESS	AWS::CloudFormation::Stack	awscodestar-dk-sqs-red-lambda	The following resource(s) failed to create: [TestQueue]. The following resource(s) failed to update: [HelloWorld].
08:37:02 UTC-0700	UPDATE_FAILED	AWS::Lambda::Function	HelloWorld	Resource update cancelled
08:37:01 UTC-0700	CREATE_FAILED	AWS::SQS::Queue	TestQueue	API: sqs:CreateQueue Access to the resource https://sqs.us-west-2.amazonaws.com/ is denied.
08:37:01 UTC-0700	CREATE_IN_PROGRESS	AWS::SQS::Queue	TestQueue	

   必要なアクセス許可を追加するには、プロジェクトの AWS CloudFormation ワーカーロールに添付されているポリシーを編集します。「ステップ 1: IAM で CloudFormation ワーカーロールを編集する」を参照してください。

2. パイプラインが正常に実行されると、AWS CloudFormation スタックでリソースが作成されます。AWS CloudFormation リストの [Resources] (リソース) で、プロジェクト用に作成されたリソースを表示します。この例では、TestQueue キューが [Resources] (リソース) セクションに一覧表示されています。

   キュー URL は、AWS CloudFormation で利用できます。キュー URL はこの形式に従います。

   ```
   https://{REGION_ENDPOINT}/queue.|api-domain|/{YOUR_ACCOUNT_NUMBER}/
   {YOUR_QUEUE_NAME}
   ```

   詳細については、[Send an Amazon SQS Message] (Amazon SQS メッセージの送信)、[Receive a Message from an Amazon SQS Queue] (Amazon SQS キューからのメッセージの受信)、および [Delete a Message from an Amazon SQS Queue] (Amazon SQS キューからのメッセージの削除) を参照してください。

ステップ 5: インラインポリシーでリソースのアクセス許可を追加する

新しいリソースへのアクセス権をチームメンバーに付与するには、適切なインラインポリシーをユーザーのロールに追加します。必ずしもすべてのリソースでアクセス許可を追加する必要があるわけではありません。以下のステップを実行するには、ルートユーザー、アカウントの管理者ユーザー、または `AdministratorAccess` 管理ポリシーか同等のポリシーが添付されている IAM ユーザーかフェデレーティッドユーザーとして、コンソールにサインイン済みである必要があります。

JSON ポリシーエディタを使用してポリシーを作成するには

1. AWS Management Console にサインインして、IAM コンソール (https://console.aws.amazon.com/iam/) を開きます。
2. 左側のナビゲーションペインで、[ポリシー] を選択します。

   初めて [ポリシー] を選択する場合には、[管理ポリシーにようこそ] ページが表示されます。[Get Started] (今すぐ始める) を選択します。
3. ページの上部で、[ポリシーの作成] を選択します。
4. [ポリシーエディタ] セクションで、[JSON] オプションを選択します。
5. 次の JSON ポリシードキュメントを入力します。

   ```
   {
    "Action": [
      "sqs:CreateQueue",
   ```

```
      "sqs:DeleteQueue",
      "sqs:GetQueueAttributes",
      "sqs:SetQueueAttributes",
      "sqs:ListQueues",
      "sqs:GetQueueUrl"
    ],
    "Resource": [
      "*"
    ],
    "Effect": "Allow"
}
```

6. [Next] (次へ) をクリックします。

> **Note**
>
> いつでも [Visual] と [JSON] エディタオプションを切り替えることができます。ただし、[Visual] エディタで [次] に変更または選択した場合、IAM はポリシーを再構成して visual エディタに合わせて最適化することがあります。詳細については、IAM ユーザーガイドの「ポリシーの再構成」を参照してください。

7. [確認と作成] ページで、作成するポリシーの [ポリシー名] と [説明] (オプション) を入力します。[このポリシーで定義されているアクセス許可] を確認して、ポリシーによって付与されたアクセス許可を確認します。
8. [Create Policy] (ポリシーの作成) をクリックして、新しいポリシーを保存します。

IAM ロールをプロジェクトに追加する

2018 年 12 月 6 日 (PDT) 時点では、アプリケーションスタック (template.yml) で独自のロールとポリシーを定義できます。特権のエスカレーションと破壊的なアクションのリスクを軽減するために、作成する IAM エンティティごとに、プロジェクト固有のアクセス許可の境界を設定する必要があります。複数の関数を含む Lambda プロジェクトがある場合は、関数ごとに IAM ロールを作成するのがベストプラクティスです。

IAM ロールをプロジェクトに追加するには

1. プロジェクトの `template.yml` ファイルを編集します。
2. `Resources:` セクションで、次の例の形式を使用して IAM リソースを追加します：

```
  SampleRole:
  Description: Sample Lambda role
  Type: AWS::IAM::Role
  Properties:
    AssumeRolePolicyDocument:
      Statement:
      - Effect: Allow
        Principal:
          Service: [lambda.amazonaws.com]
        Action: sts:AssumeRole
    ManagedPolicyArns:
      -  arn:aws:iam::aws:policy/service-role/AWSLambdaBasicExecutionRole
    PermissionsBoundary: !Sub 'arn:${AWS::Partition}:iam::${AWS::AccountId}:policy/
CodeStar_${ProjectId}_PermissionsBoundary'
```

3. パイプラインを通して変更をリリースし、成功を確認します。

Prod ステージとエンドポイントをプロジェクトに追加する

このセクションの手順を使用して、新しい本番稼働 (Prod) ステージをパイプラインに追加し、パイプラインの Deploy および Prod ステージ間に手動の承認ステージを追加します。これにより、プロジェクトのパイプラインが実行されるときに、追加のリソーススタックが作成されます。

> **Note**
>
> 次の場合は、以下の手順を使用できます：
>
> - 2018 年 8 月 3 日以降に作成されたプロジェクトの場合、AWS CodeStar により、`/template.yml` ファイルとともに、Amazon EC2、Elastic Beanstalk、および Lambda プロジェクトがプロジェクトレポジトリにプロビジョニングされています。
> - 2018 年 12 月 6 日 (PDT) 以降に作成されたプロジェクトの場合、AWS CodeStar により、アクセス許可の境界ポリシーを使用してプロジェクトがプロビジョニングされています。

すべての AWS CodeStar プロジェクトは、アプリケーションの AWS ランタイム依存関係 (Linux インスタンスや Lambda 関数など) をモデリングする AWS CloudFormation テンプレートファイルが含まれています。この /template.yml ファイルは、ソースリポジトリに保存されています。

/template.yml ファイルで、Stage パラメータを使用して、プロジェクトパイプラインの新しいステージのリソーススタックを追加します。

```
 Stage:
   Type: String
   Description: The name for a project pipeline stage, such as Staging or Prod, for
which resources are provisioned and deployed.
   Default: ''
```

Stage パラメータは、リソースでプロジェクト ID が参照されているすべての名前付きリソースに適用されます。たとえば、次のロール名は、テンプレート内の名前付きリソースです：

```
RoleName: !Sub 'CodeStar-${ProjectId}-WebApp${Stage}'
```

前提条件

AWS CodeStar コンソールのテンプレートオプションを使用して、プロジェクトを作成します。

IAM ユーザーに次のアクセス許可があることを確認します：

- プロジェクト AWS CloudFormation ロールの iam:PassRole。
- プロジェクトツールチェーンロールの iam:PassRole。
- cloudformation:DescribeStacks
- cloudformation:ListChangeSets

Elastic Beanstalk または Amazon EC2 プロジェクトの場合のみ:

- codedeploy:CreateApplication
- codedeploy:CreateDeploymentGroup
- codedeploy:GetApplication
- codedeploy:GetDeploymentConfig
- codedeploy:GetDeploymentGroup
- elasticloadbalancing:DescribeTargetGroups

トピック

- ステップ 1: CodeDeploy で新しいデプロイグループを作成する (Amazon EC2 プロジェクトのみ)
- ステップ 2: Prod ステージの新しいパイプラインステージを追加する
- ステップ 3: 手動承認ステージを追加する
- ステップ 4: 変更をプッシュし、AWS CloudFormation スタックの更新をモニタリングする

ステップ 1: CodeDeploy で新しいデプロイグループを作成する (Amazon EC2 プロジェクトのみ)

CodeDeploy アプリケーションを選択し、新しいインスタンスに関連付けられた新しいデプロイグループを追加します。

> **Note**
>
> プロジェクトが Lambda または Elastic Beanstalk プロジェクトである場合は、このステップはスキップできます。

1. https://console.aws.amazon.com/codedeploy で、CodeDeploy コンソールを開きます。
2. AWS CodeStar で作成されたときにプロジェクト用に生成された CodeDeploy アプリケーションを選択します。
3. [Deployment groups] (デプロイグループ) で、[Create deployment group] (デプロイグループの作成) を選択します。
4. [Deployment group name] (デプロイグループ名) に「***<project-id>-prod-Env***」と入力します。
5. [サービスロール] で、AWS CodeStar プロジェクトのツールチェーンワーカーロールを選択します。
6. [Deployment type] (デプロイタイプ) で、[In-place] (インプレース) を選択します。
7. [Environment configuration] (環境設定) で、[Amazon EC2 Instances] (Amazon EC2 インスタンス) タブを選択します。
8. [Tag](タグ)グループで、[Key] (キー) の [`aws:cloudformation:stack-name`] を選択します。[Value] (値) で、[`awscodestar-<projectid>-infrastructure-prod`] (GenerateChangeSet アクション用に作成されるスタック) を選択します。

9. [Deployment settings] (デプロイ設定) で [CodeDeployDefault.AllAtOnce] を選択します。
10. [Choose a load balancer] (ロードバランサーを選択) をクリアします。
11. [Create deployment group] (デプロイグループの作成) を選択します。

    これで、2 番目のデプロイグループが作成されました。

ステップ 2: Prod ステージの新しいパイプラインステージを追加する

プロジェクトの Deploy ステージと同じ一連のデプロイアクションを使用してステージを追加します。たとえば、Amazon EC2 プロジェクトの新しい Prod ステージには、プロジェクト用に作成された [Deploy] (デプロイ) ステージと同じアクションが必要です。

[Deploy] (デプロイ) ステージからパラメータおよびフィールドをコピーするには

1. AWS CodeStar プロジェクトダッシュボードから [パイプラインの詳細] を選択し、 コンソールでパイプラインを開きます。
2. [Edit] (編集) を選択します。
3. [Deploy] (デプロイ) ステージで、[Edit stage] (ステージを編集) を選択します。
4. [GenerateChangeSet] アクションの編集アイコンを選択します。次のフィールドの値を書き留めておきます。下記の値は、新しいアクションの作成時に使用します。

   - スタックの名前
   - 変更セット名
   - テンプレート
   - テンプレート構成
   - 入力アーティファクト
5. [Advanced] (詳細) を展開し、[Parameters] (パラメータ) で、プロジェクトのパラメータをコピーします。これらのパラメータを新しいアクションに貼り付けます。例えば、ここに表示されるパラメータを JSON 形式でコピーします。

   - Lambda プロジェクト:

     ```
     {
       "ProjectId":"MyProject"
     }
     ```

   - Amazon EC2 プロジェクト:

```
{

  "ProjectId":"MyProject",
  "InstanceType":"t2.micro",
  "WebAppInstanceProfile":"awscodestar-MyProject-WebAppInstanceProfile-
EXAMPLEY5VSFS",
  "ImageId":"ami-EXAMPLE1",
  "KeyPairName":"my-keypair",
  "SubnetId":"subnet-EXAMPLE",
  "VpcId":"vpc-EXAMPLE1"
}
```

- Elastic Beanstalk プロジェクト：

```
{
  "ProjectId":"MyProject",
  "InstanceType":"t2.micro",
  "KeyPairName":"my-keypair",
  "SubnetId":"subnet-EXAMPLE",
  "VpcId":"vpc-EXAMPLE",
  "SolutionStackName":"64bit Amazon Linux 2018.03 v3.0.5 running Tomcat 8 Java
 8",
  "EBTrustRole":"CodeStarWorker-myproject-EBService",
  "EBInstanceProfile":"awscodestar-myproject-EBInstanceProfile-11111EXAMPLE"
}
```

6. ステージの編集ペインで、[Cancel] (キャンセル) を選択します。

新しい Prod ステージで GenerateChangeSet アクションを作成するには

> **Note**
>
> 新しいアクションを追加した後編集モードのまま、編集のために新しいアクションを再度開くと、一部のフィールドが表示されない場合があります。また、以下が表示される場合があります：スタック stack-name は存在しません
> このエラーが出てもパイプラインを保存することはできます。ただし、表示されないフィールドを復元するには、新しいアクションを削除してから再び追加する必要があります。パイプラインを保存して実行した後、スタックが認識されエラーが表示されなくなります。

1. まだパイプラインが表示されていない場合は、AWS CodeStar プロジェクトダッシュボードから [Pipeline Details (パイプラインの詳細)] を選択して、コンソールでパイプラインを開きます。
2. [Edit] (編集) を選択します。
3. 図の最下部で [+ Add stage] (+ ステージの追加) を選択します。
4. ステージ名 (例：**Prod**) を入力し、[+ Add action group] (+ アクショングループの追加) を選択します。
5. [Action name] (アクション名) に、名前を入力します (例：**GenerateChangeSet**)。
6. [Action provider] (アクションプロバイダ) で、[AWS CloudFormation] を選択します。
7. [Action mode] (アクションモード) で [Create or replace a change set] (変更セットの作成または置換) を選択します。
8. [Stack name] (スタック名) で、このアクションによって作成される AWS CloudFormation スタックの新しい名前を入力します。デプロイスタック名と同じ名前でスタートし、**-prod** を追加します:

   - Lambda プロジェクト:`awscodestar-<project_name>-lambda-prod`
   - Amazon EC2 および Elastic Beanstalk プロジェクト:`awscodestar-<project_name>-infrastructure-prod`

   > **Note**
   >
   > スタックは正確に **awscodestar-<project_name>-** で始まる必要があり、それ以外の場合はスタックの作成は失敗します。

9. [Change set name] (変更セット名) で、既存の [Deploy] (デプロイ) ステージ (例：**pipeline-changeset**) で指定されたのと同じ変更セット名を入力します。
10. [Input artifacts] (入力アーティファクト) で、[Build artifact] (ビルドアーティファクト) を選択します。
11. [Template] (テンプレート) で、既存の [Deploy] (デプロイ) ステージ (例：**<project-ID>-BuildArtifact::template.yml**) で指定されたのと同じ変更テンプレート名を入力します。
12. [Template configuration] (テンプレート設定) で、デプロイステージ (例：**<project-ID>-BuildArtifact::template-configuration.json**) で指定されたのと同じ変更テンプレート設定ファイル名を入力します。
13. [Capabilities] (機能) で、 CAPABILITY_NAMED_IAM を選択します。

14. [Role name](ロール名) で、プロジェクトの AWS CloudFormation ワーカーロールの名前を選択します。
15. [Advanced] (詳細) を展開し、[Parameters] (パラメータ) で、プロジェクトのパラメータを貼り付けます。Amazon EC2 プロジェクト用に、ここに示すように JSON 形式でStage パラメータを含めます：

```
{

  "ProjectId":"MyProject",
  "InstanceType":"t2.micro",
  "WebAppInstanceProfile":"awscodestar-MyProject-WebAppInstanceProfile-
EXAMPLEY5VSFS",
  "ImageId":"ami-EXAMPLE1",
  "KeyPairName":"my-keypair",
  "SubnetId":"subnet-EXAMPLE",
  "VpcId":"vpc-EXAMPLE1",
  "Stage":"Prod"
}
```

> **Note**
>
> 変更するパラメータだけではなく、プロジェクトのすべてのパラメータを貼り付けます。

16. [Save] (保存) を選択します。
17. AWS CodePipeline のペインで、[Save pipeline change] (パイプラインの変更を保存)、[Save change] (変更の保存) の順に選択します。

> **Note**
>
> 変更検出リソースの削除および追加を通知するメッセージが表示されることがあります。メッセージを確認して、このチュートリアルの次のステップに進みます。

更新されたパイプラインを表示します。

新しい Prod ステージで ExecuteChangeSet アクションを作成するには

1. まだパイプラインを表示していない場合は、AWS CodeStar プロジェクトダッシュボードから [Pipeline Details (パイプラインの詳細)] を選択して、コンソールでパイプラインを開きます。
2. [Edit] (編集) を選択します。
3. 新しい Prod ステージで、新しい GenerateChangeSet アクションの後で、[+ Add action group] (+ アクショングループの追加) を選択します。
4. [Action name] (アクション名) に、名前を入力します (例：**ExecuteChangeSet**)。
5. [Action provider] (アクションプロバイダ) で、[AWS CloudFormation] を選択します。
6. [Action mode] (アクションモード) で、 [Execute a change set] (変更セットの実行) を選択します。
7. [Stack name] (スタック名) で、GenerateChangeSet アクションで入力した AWS CloudFormation スタックの新しい名前 (例：**awscodestar-<project-ID>-infrastructure-prod**) を入力します。
8. [Change set name] (変更セット名) で、[Deploy] (デプロイ) ステージで使用したのと同じ変更セット名 (例：**pipeline-changeset**) を入力します。
9. [Done] (完了) を選択します。
10. AWS CodePipeline のペインで、[Save pipeline change] (パイプラインの変更を保存)、[Save change] (変更の保存) の順に選択します。

> **Note**
>
> 変更検出リソースの削除および追加を通知するメッセージが表示されることがあります。メッセージを確認して、このチュートリアルの次のステップに進みます。

更新されたパイプラインを表示します。

新しい Prod ステージで CodeDeploy デプロイアクションを作成するには (Amazon EC2 プロジェクトのみ)

1. Prod ステージの新しいアクションの後、[+ Action] (+ アクション) を選択します。
2. [Action name] (アクション名) に、名前を入力します (例：**Deploy**)。
3. [Action provider] (アクションプロバイダ) で、[AWS CodeDeploy] を選択します。

4. [Application name] (アプリケーション名) で、プロジェクトの CodeDeploy アプリケーションの名前を選択します。
5. [Deployment group] (デプロイグループ) で、ステップ 2 で作成した新しい CodeDeploy デプロイグループの名前を選択します。
6. [Input artifacts] (入力アーティファクト) で、既存のステージで使用されたのと同じビルドアーティファクトを選択します。
7. [Done] (完了) を選択します。
8. AWS CodePipeline のペインで、[Save pipeline change] (パイプラインの変更を保存)、[Save change] (変更の保存) の順に選択します。更新されたパイプラインを表示します。

ステップ 3: 手動承認ステージを追加する

ベストプラクティスとして、新しい本番稼働ステージの前に手動承認ステージを追加します。

1. 左上の [Edit] (編集) を選択します。
2. パイプラインの図で、[Deploy] (デプロイ) と [Prod deployment] (Prod デプロイ)ステージの間で、[+ Add stage] (+ ステージの追加) を選択します。
3. [Edit stage] (ステージを編集) で、ステージ名 (例：**`Approval`**) を入力し、[+ Add action group] (+ アクショングループの追加) を選択します。
4. [Action name] (アクション名) に、名前を入力します (例：**`Approval`**)。
5. [Approval type] (承認の種類) で、[Manual approval] (手動承認) を選択します。
6. (オプション) [Configuration] (設定) の [SNS Topic ARN] (SNS トピック ARN) で、作成してサブスクライブした SNS トピックを選択します。
7. [Add Action] (アクションの追加) を選択します。
8. AWS CodePipeline のペインで、[Save pipeline change] (パイプラインの変更を保存)、[Save change] (変更の保存) の順に選択します。更新されたパイプラインを表示します。
9. 変更を送信してパイプラインの構築をスタートするには、[Release change] (変更のリリース)、[Release] (リリース) の順に選択します。

ステップ 4: 変更をプッシュし、AWS CloudFormation スタックの更新をモニタリングする

1. パイプラインの実行中に、次の手順を使用して、新しいステージのスタックとエンドポイントの作成をフォローすることができます。

2. パイプラインで[Deploy] (デプロイ)ステージがスタートすると、AWS CloudFormation スタックの更新がスタートします。パイプラインの AWS CloudFormation ステージを AWS CodeStar ダッシュボードで選択し、スタック更新通知を表示します。スタック作成の詳細を表示するには、コンソールで、[Events] (イベント) リストからプロジェクトを選択します。
3. パイプラインが正常に完了すると、AWS CloudFormation スタックでリソースが作成されます。AWS CloudFormation コンソールで、プロジェクトのインフラストラクチャスタックを選択します。スタック名は次の形式に従います：

   - Lambda プロジェクト:`awscodestar-<project_name>-lambda-prod`
   - Amazon EC2 および Elastic Beanstalk プロジェクト:`awscodestar-<project_name>-infrastructure-prod`

   AWS CloudFormationコンソールの[Resources] (リソース) リストで、プロジェクト用に作成されたリソースを表示します。この例では、新しい Amazon EC2 インスタンスが [Resources] (リソース) セクションに表示されます。
4. 本番稼働ステージのエンドポイントにアクセスします：

   - Elastic Beanstalk プロジェクトの場合、AWS CloudFormation コンソールで新しいスタックを開き、[Resources] (リソース) を展開します。Elastic Beanstalk アプリケーションを選択します。Elastic Beanstalk コンソールでリンクが開きます。[Environments] (環境) を選択します。[URL] で URL を選択し、ブラウザでエンドポイントを開きます。
   - Lambda プロジェクトの場合、AWS CloudFormation コンソールで新しいスタックを開き、[Resources] (リソース) を展開します。[API Gateway resource](API Gateway リソース) を選択します。リンクが API Gateway コンソールで開きます。[Stages] (ステージ) を選択します。[Invoke URL] (呼び出し URL) で URL を選択し、ブラウザでエンドポイントを開きます。
   - Amazon EC2 プロジェクトの場合、AWS CodeStar コンソールでプロジェクトリソースリストから新しい Amazon EC2 インスタンスを選択します。Amazon EC2 コンソールの[Instance] (インスタンス) ページでリンクが開きます。[Description] (説明) タブを選択し、[Public DNS (IPv4)] (パブリック DNS (IPv4)) の URL をコピーして、ブラウザでその URL を開きます。
5. 変更がデプロイされていることを確認します。

AWS CodeStar プロジェクトで SSM パラメータを安全に使用する

多くのお客様は認証情報などの機密情報を [Systems Manager Parameter Store] (システムマネージャパラメータストア) のパラメータに保存します。 AWS CodeStar プロジェクトでこれらのパラメータを安全に使用できるようになりました。例えば、ツールチェーンスタック (template.yml) でアプリケーションリソースを定義するときに、 CodeBuild または のビルド仕様で SSM パラメータを使用できます。

AWS CodeStar プロジェクトで SSM パラメータを使用するには、パラメータに AWS CodeStar プロジェクト ARN を手動でタグ付けする必要があります。また、タグ付けしたパラメータにアクセスするには、AWS CodeStar ツールチェーンワーカーロールに適切なアクセス許可を付与する必要があります。

開始する前に

- [Create a new] (新規作成) または、アクセスする情報を含む既存の Systems Manager パラメータを特定します。
- 使用する AWS CodeStar プロジェクトを特定するか、新しいプロジェクト を作成します。
- CodeStar プロジェクト ARN を書き留めます。以下のような形式です：`arn:aws:codestar:region-id:account-id:project/project-id`

AWS CodeStar プロジェクト ARN を使用してパラメータにタグを付ける

ステップバイステップの手順については、[Tagging Systems Manager Parameters] (システムマネージャパラメータへのタグ付け) を参照してください。

1. [Key] (キー) に、「`awscodestar:projectArn`」と入力します。
2. 値 に、 からプロジェクト ARN を入力します CodeStar。 `arn:aws:codestar:region-id:account-id:project/project-id`
3. [Save] (保存) を選択します。

これで、template.yml ファイルで SSM パラメータをリファレンスできるようになります。ツールチェーンのワーカーロールで使用する場合は、追加のアクセス許可を付与する必要があります。

AWS CodeStar プロジェクトツールチェーンでタグ付けされたパラメータを使用するアクセス許可を付与する

> **Note**
>
> 以下のステップは、2018 年 12 月 6 日 (PDT) 以降に作成されたプロジェクトにのみ適用されます。

1. 使用する CodeStar プロジェクトの AWS プロジェクトダッシュボードを開きます。
2. [Project] (プロジェクト) をクリックして作成済みリソースのリストを表示し、ツールチェーンのワーカーロールを見つけます。`role/CodeStarWorker-`*`project-id`*`-ToolChain` という形式の名前の IAM リソースです。
3. ARN をクリックして、IAM コンソールで開きます。
4. を見つけ ToolChainWorkerPolicy 、必要に応じて展開します。
5. [Edit Policy] (ポリシーの編集) をクリックします。
6. `Action:` の下に次の行を追加します：

   ```
   ssm:GetParameter*
   ```

7. [Review policy] (ポリシーの確認) をクリックしてから、[Save changes] (変更の保存) をクリックします。

2018 年 12 月 6 日 (PDT) 以前に作成されたプロジェクトの場合は、次のアクセス許可を各サービスのワーカーロールに追加する必要があります。

```
        {
            "Action": [
                "ssm:GetParameter*"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Condition": {
                "StringEquals": {
                    "ssm:ResourceTag/awscodestar:projectArn": "arn:aws:codestar:region-
id:account-id:project/project-id"
                }
```

```
            }
        }
```

AWS Lambda プロジェクトのトラフィックを移行する

AWS CodeDeploy では、AWS CodeStar サーバーレスプロジェクトの AWS Lambda 関数の関数バージョンのデプロイをサポートしています。AWS Lambda のデプロイは、既存の Lambda 関数からの着信トラフィックを、更新された Lambda 関数バージョンに移行します。更新された Lambda 関数をテストするには、別のバージョンをデプロイし、必要に応じて、デプロイを最初のバージョンにロールバックします。

このセクションのステップに従い、AWS CodeStar プロジェクトテンプレートを変更し、CodeStarWorker ロールの IAM アクセス許可を更新します。このタスクでは、エイリアスの AWS Lambda 関数を作成する自動応答を AWS CloudFormation でスタートし、更新された環境にトラフィックを移行するように AWS CodeDeploy に指示します。

> **Note**
>
> 2018 年 12 月 12 日以前に AWS CodeStar プロジェクトを作成した場合にのみ、これらのステップを完了します。

AWS CodeDeploy には、アプリケーションの AWS Lambda 関数のバージョンにトラフィックを移行することを許可するデプロイオプションが 3 種類あります。

- Canary: トラフィックは 2 つの増分で移行されます。残りのトラフィックが 2 回目の増分で移行される前に、最初の増分および間隔で更新された Lambda 関数のバージョンに移行されるトラフィックの割合 (%) を分単位で指定する、事前定義された Canary オプションから選択できます。
- Linear: トラフィックは等しい増分で移行され、増分間の間隔 (分) も同じです。増分ごとに移行するトラフィックの割合 (%) と、増分間の間隔 (分) を指定する、事前定義済み線形オプションから選択できます。トラフィックは毎回同じ間隔 (分) の等しい増分で移行されます。増分ごとに移行するトラフィックの割合 (%) と、増分間の間隔 (分) を指定する、事前定義済み線形オプションから選択できます。
- All-at-once: すべてのトラフィックは元の Lambda 関数から最新バージョンの Lambda 関数に一度に移行されます。

デプロイプリファレンスのタイプ
Canary10Percent30Minutes
Canary10Percent5Minutes
Canary10Percent10Minutes
Canary10Percent15Minutes
Linear10PercentEvery10Minutes
Linear10PercentEvery1Minute
Linear10PercentEvery2Minutes
Linear10PercentEvery3Minutes
AllAtOnce

AWS Lambda コンピューティングプラットフォームの AWS CodeDeploy デプロイの詳細については、AWS Lambda コンピューティングプラットフォームでのデプロイ を参照してください。

AWS SAM の詳細については、GitHub の AWSサーバーレスアプリケーションモデル (AWS SAM) を参照してください。

前提条件:

サーバーレスプロジェクトを作成する場合、Lambda コンピューティングプラットフォームで任意のテンプレートを選択します。ステップ 4～6 を実行するには、管理者ユーザーとしてサインインする必要があります。

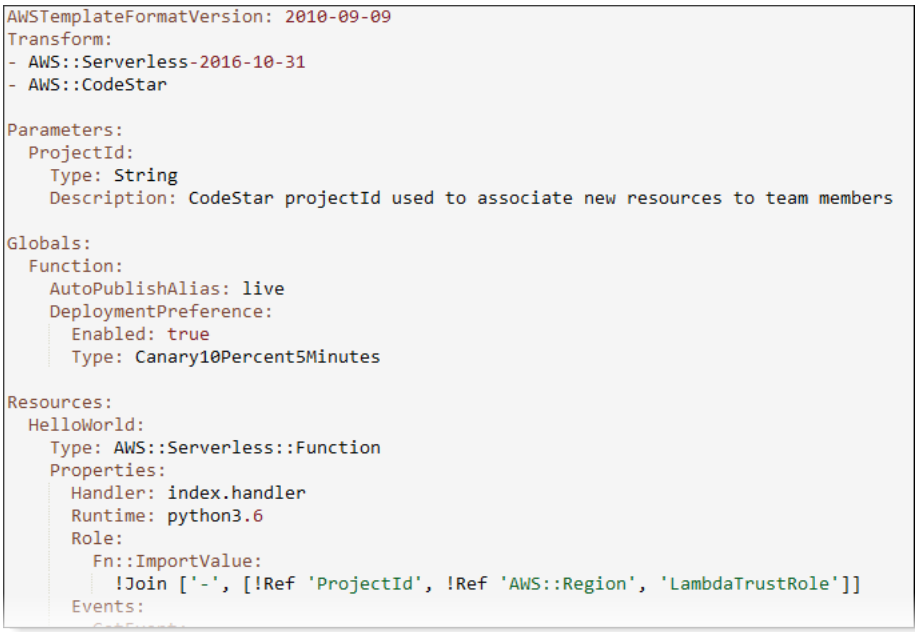
ステップ 1: SAM テンプレートを変更して AWS Lambda バージョンのデプロイパラメータを追加する

1. https://console.aws.amazon.com/codestar/ で AWS CodeStar コンソールを開きます。
2. プロジェクトを作成するか、`template.yml` ファイルを含む既存のプロジェクトを選択して、[Code] (コード) ページを開きます。リポジトリの最上位で、変更する `template.yml` という名前の SAM テンプレートの場所を書き留めます。

3. template.yml ファイルを IDE またはローカルリポジトリで開きます。以下のテキストをコピーして、Globals セクションをファイルに追加します。このチュートリアルのサンプルテキストでは、Canary10Percent5Minutes オプションを選択します。

```
Globals:
  Function:
    AutoPublishAlias: live
    DeploymentPreference:
      Enabled: true
      Type: Canary10Percent5Minutes
```

この例では、Globals セクション追加後に変更されたテンプレートを示します。



```
AWSTemplateFormatVersion: 2010-09-09
Transform:
- AWS::Serverless-2016-10-31
- AWS::CodeStar

Parameters:
  ProjectId:
    Type: String
    Description: CodeStar projectId used to associate new resources to team members

Globals:
  Function:
    AutoPublishAlias: live
    DeploymentPreference:
      Enabled: true
      Type: Canary10Percent5Minutes

Resources:
  HelloWorld:
    Type: AWS::Serverless::Function
    Properties:
      Handler: index.handler
      Runtime: python3.6
      Role:
        Fn::ImportValue:
          !Join ['-', [!Ref 'ProjectId', !Ref 'AWS::Region', 'LambdaTrustRole']]
      Events:
```

詳細については、SAM テンプレートの グローバルセクション リファレンスガイドを参照してください。

ステップ 2: AWS CloudFormation ロールを編集してアクセス許可を追加する

1. AWS Management Console にサインインし、AWS CodeStar コンソール (https://console.aws.amazon.com/codestar/) を開きます。

> **Note**
>
> 作成した IAM ユーザーに関連付けられた認証情報または AWS CodeStarのセットアップ で識別される認証情報を使用して AWS Management Console にサインインする必要があります。このユーザーには、[**AWSCodeStarFullAccess**] という名前の AWS 管理ポリシーが添付されている必要があります。

2. 既存のサーバーレスプロジェクトを選択し、[Project resources] (プロジェクトリソース) ページを開きます。
3. [Resources] (リソース) で、CodeStarWorker/AWS CloudFormation ロール用に作成した IAM ロールを選択します。ロールが IAM コンソールで開きます。
4. [Permissions] (アクセス許可) タブの [Inline Policies] (インラインポリシー) で、サービスロールポリシーの列の [Edit Policy] (ポリシーの編集) を選択します。[JSON] タブを選択して、JSON 形式でポリシーを編集します。

> **Note**
>
> サービスロールは `CodeStarWorkerCloudFormationRolePolicy` という名前になります。

5. [JSON] フィールドで、`Statement` 要素内に次のポリシーステートメントを追加します。*`region`* と *`id`* のプレースホルダーは、お客様のリージョンとアカウント ID に置き換えてください。

```
{
  "Action": [
    "s3:GetObject",
    "s3:GetObjectVersion",
    "s3:GetBucketVersioning"
  ],
  "Resource": "*",
  "Effect": "Allow"
},
{
  "Action": [
    "s3:PutObject"
  ],
  "Resource": [
```

```
      "arn:aws:s3:::codepipeline*"
    ],
    "Effect": "Allow"
  },
  {
    "Action": [
      "lambda:*"
    ],
    "Resource": [
      "arn:aws:lambda:region:id:function:*"
    ],
    "Effect": "Allow"
  },
  {
    "Action": [
      "apigateway:*"
    ],
    "Resource": [
      "arn:aws:apigateway:region::*"
    ],
    "Effect": "Allow"
  },
  {
    "Action": [
      "iam:GetRole",
      "iam:CreateRole",
      "iam:DeleteRole",
      "iam:PutRolePolicy"
    ],
    "Resource": [
      "arn:aws:iam::id:role/*"
    ],
    "Effect": "Allow"
  },
  {
    "Action": [
      "iam:AttachRolePolicy",
      "iam:DeleteRolePolicy",
      "iam:DetachRolePolicy"
    ],
    "Resource": [
      "arn:aws:iam::id:role/*"
    ],
    "Effect": "Allow"
```

```
},
{
  "Action": [
    "iam:PassRole"
  ],
  "Resource": [
    "*"
  ],
  "Effect": "Allow"
},
{
  "Action": [
    "codedeploy:CreateApplication",
    "codedeploy:DeleteApplication",
    "codedeploy:RegisterApplicationRevision"
  ],
  "Resource": [
    "arn:aws:codedeploy:region:id:application:*"
  ],
  "Effect": "Allow"
},
{
  "Action": [
    "codedeploy:CreateDeploymentGroup",
    "codedeploy:CreateDeployment",
    "codedeploy:DeleteDeploymentGroup",
    "codedeploy:GetDeployment"
  ],
  "Resource": [
    "arn:aws:codedeploy:region:id:deploymentgroup:*"
  ],
  "Effect": "Allow"
},
{
  "Action": [
    "codedeploy:GetDeploymentConfig"
  ],
  "Resource": [
    "arn:aws:codedeploy:region:id:deploymentconfig:*"
  ],
  "Effect": "Allow"
}
```

6. [Review policy] (ポリシーの確認) を選択して、ポリシーにエラーがないことを確認します。ポリシーにエラーがなければ、[Save changes] (変更の保存) を選択します。

ステップ 3: テンプレートの変更をコミットおよびプッシュして、AWS Lambda バージョンの移行を開始する

1. ステップ 1 で保存した `template.yml` ファイルの変更をコミットおよびプッシュします。

   > **Note**
   >
   > これにより、パイプラインが開始されます。IAM アクセス許可を更新する前に変更をコミットすると、パイプライン開始後、AWS CloudFormation スタック更新でエラーが発生し、このスタック更新はロールバックされます。この問題が発生した場合は、アクセス許可を修正してから、パイプランを再起動します。

2. AWS CloudFormation スタック更新は、プロジェクトのパイプランで[Deploy] (デプロイ) ステージのスタート時にスタートします。デプロイ開始時のスタック更新の通知を確認するには、AWS CodeStar ダッシュボードで、パイプラインの AWS CloudFormation ステージを選択します。

   AWS CloudFormation は、スタック更新中、次のようにプロジェクトリソースを自動的に更新します:

   - AWS CloudFormation では、エイリアスの Lambda 関数、イベントフック、およびリソースを作成して、`template.yml` ファイルを処理します。
   - AWS CloudFormation は、Lambda を呼び出して、新しい関数のバージョンを作成します。
   - AWS CloudFormation は AppSpec ファイルを作成し、AWS CodeDeploy を呼び出してトラフィックを移行します。

   SAM でのエイリアスの Lambda 関数の発行の詳細については、AWS サーバーレスアプリケーションモデル(SAM) テンプレートリファレンスを参照してください。AWS CodeDeploy AppSpec ファイルのイベントフックおよびリソースの詳細については、AppSpec の「resources」セクション (AWS Lambda デプロイのみ) および AWS Lambda デプロイの AppSpec の「hooks」セクション を参照してください。

3. パイプラインが正常に完了すると、AWS CloudFormation スタックでリソースが作成されます。[Project] (プロジェクト) ページの [Project Resources] (プロジェクトリソース) リスト

で、AWS CodeDeploy アプリケーション、AWS CodeDeploy デプロイグループ、およびプロジェクト用に作成した AWS CodeDeploy サービスロールのリソースを表示します。

4. 新しいバージョンを作成するには、リポジトリで Lambda 関数を変更します。新しいデプロイが開始し、トラフィックは、SAM テンプレートで示されるデプロイタイプに応じて、移行されます。新しいバージョンに移行されるトラフィックのステータスを表示するには、[Project] (プロジェクト) ページの [Project Resources] (プロジェクトリソース) リストで、AWS CodeDeploy デプロイへのリンクを選択します。
5. 各バージョンに関する詳細を表示するには、[Revisions] (リビジョン) で、AWS CodeDeploy デプロイグループへのリンクを選択します。
6. ローカル作業ディレクトリで、AWS Lambda 関数を変更し、その変更をプロジェクトリポジトリにコミットすることができます。AWS CloudFormation では、同様に次のリビジョンを管理する上で、AWS CodeDeploy をサポートしています。Lambda デプロイの再デプロイ、停止、ロールバックの詳細については、AWS Lambda コンピューティングプラットフォームでのデプロイ を参照してください。

AWS CodeStar プロジェクトを本番稼働用に移行する

AWS CodeStar プロジェクトを使用してアプリケーションを作成し、AWS CodeStar が提供する内容を確認したら、プロジェクトを本番稼働に移行することができます。これを行う 1 つの方法は、AWS CodeStar の外部にアプリケーションの AWS リソースを複製することです。リポジトリ、ビルドプロジェクト、パイプライン、デプロイは引き続き必要ですが、それらを AWS CodeStar で作成するのではなく、AWS CloudFormation を使用して再作成します。

> **Note**
>
> 最初に AWS CodeStar クイックスタートの 1 つを使用して類似したプロジェクトを作成または表示し、それを独自のプロジェクトのテンプレートとして使用して、必要なリソースとポリシーを確実に含めるようにすると便利です。

AWS CodeStar プロジェクトは、コードをデプロイするために作成されたソースコードとリソースの組み合わせです。コードのビルド、リリース、デプロイに役立つリソースのコレクションは、ツールチェーンリソースと呼ばれます。プロジェクト作成時、AWS CloudFormation テンプレートを使用して、継続的な統合/継続的デプロイメント (CI/CD) パイプラインのツールチェーンリソースをプロビジョンします。

コンソールでプロジェクトを作成すると、ツールチェーンテンプレートが作成されます。AWS CLI を使用してプロジェクトを作成する際、ツールチェーンリソースを作成するツールチェーンテンプレートを作成します。

完全なツールチェーンを作成するには、次の推奨リソースが必要です：

1. ソースコードを保存する CodeCommit または GitHub リポジトリ。
2. リポジトリへの変更をリッスンするよう設定されている CodePipeline パイプライン。
   a. AWS CodeBuild を使用してユニットテストまたは統合テストを使用する場合は、ビルドステージをパイプラインに追加してビルドアーティファクトを作成することをお勧めします。
   b. CodeDeploy または AWS CloudFormation を使用してビルドアーティファクトとソースコードをランタイムインフラストラクチャにデプロイするデプロイステージを、パイプラインに追加することをお勧めします。

> **Note**
>
> CodePipeline では、2 つ以上のステージがパイプラインに必要であり、最初のステージはソースステージにする必要があるため、ビルドステージまたはデプロイステージを 2 番目のステージとして追加します。

トピック

- GitHub リポジトリを作成する

GitHub リポジトリを作成する

ツールチェーンテンプレートで定義して、GitHub リポジトリを作成します。コードをリポジトリにアップロードできるように、ソースコードを含む ZIP ファイルの場所がすでに作成されていることを確認します。また、AWS がユーザーに代わって GitHub に接続できるように、GitHub に個人用アクセストークンを作成しておく必要があります。GitHub の個人用アクセストークンに加えて、渡す Code オブジェクトに対する `s3.GetObject` アクセス許可も必要です。

パブリック GitHub リポジトリを指定するには、AWS CloudFormation のツールチェーンテンプレートに次のようなコードを追加します。

```
    GitHubRepo:
```

```
    Condition: CreateGitHubRepo
    Description: GitHub repository for application source code
    Properties:
      Code:
        S3:
          Bucket: MyCodeS3Bucket
          Key: MyCodeS3BucketKey
      EnableIssues: true
      IsPrivate: false
      RepositoryAccessToken: MyGitHubPersonalAccessToken
      RepositoryDescription: MyAppCodeRepository
      RepositoryName: MyAppSource
      RepositoryOwner: MyGitHubUserName
    Type: AWS::CodeStar::GitHubRepository
```

このコードは次の情報を指定します。

- 組み込みたいコードの場所、Amazon S3 バケットである必要があります。
- GitHub リポジトリで課題を有効にするかどうか。
- GitHub リポジトリがプライベートであるかどうか。
- 作成した GitHub 個人用アクセストークン。
- 作成するリポジトリの説明、名前、所有者。

指定する情報の詳細については、AWS CloudFormationユーザーガイドの AWS::CodeStar::GitHubRepository を参照してください。

AWS CodeStar でのプロジェクトタグの操作

AWS CodeStar でタグをプロジェクトに関連付けることができます。タグは、プロジェクトを管理するのに役立ちます。例えば、ベータ版リリースで組織が進めているプロジェクトに `Release` キーと `Beta` の値を持つタグを追加できます。

プロジェクトにタグを追加する

1. AWS CodeStar コンソールでプロジェクトを開き、そのサイドナビゲーションペインで、[設定] を選択します。
2. [Tags] (タグ) で、[Edit] (編集) を選択します。

3. [Key] (キー) に、タグの名前を入力します。[Value] (値) に、タグの値を入力します。
4. オプション：[Add tag] (タグの追加) を選択して、複数のタグを追加します。
5. タグの追加が終了したら、[Save] (保存) を選択します。

プロジェクトからタグを削除する

1. AWS CodeStar コンソールでプロジェクトを開き、そのサイドナビゲーションペインで、[設定] を選択します。
2. [Tags] (タグ) で、[Edit] (編集) を選択します。
3. [Tags] (タグ) で、削除するタグを見つけ、[Remove tag] (タグの削除) を選択します。
4. [Save] (保存) を選択します。

プロジェクトのタグのリストを取得します。

AWS CLI を使用して AWS CodeStar list-tags-for-project コマンドを実行し、プロジェクトの名前を指定します。

```
aws codestar list-tags-for-project --id my-first-projec
```

成功すると、次のようなタグのリストが出力に表示されます。

```
{
  "tags": {
    "Release": "Beta"
  }
}
```

AWS CodeStar プロジェクトの削除

プロジェクトが不要になった場合は、プロジェクトとそのリソースを削除して、AWS で追加料金が発生しないようにします。プロジェクトを削除すると、すべてのチームメンバーはそのプロジェクトから削除されます。プロジェクトロールは、IAM ユーザーから削除されますが、AWS CodeStar でのユーザープロファイルは変更されません。AWS CodeStar コンソールまたは AWS CLI を使用して、プロジェクトを削除できます。プロジェクトを削除するには、AWS CodeStar サービスロールである `aws-codestar-service-role` が必要です。このサービスロールは変更されずに AWS CodeStar によって引き受け可能である必要があります。

Important

AWS CodeStar でプロジェクトを削除すると、元に戻すことができません。デフォルトでは、プロジェクトのすべての AWS リソースは、以下を含む AWS アカウントで削除されます。

- プロジェクトの CodeCommit リポジトリと、そのリポジトリに保存されているすべてのもの。
- プロジェクトとそのリソース用に設定された AWS CodeStar プロジェクトロールと関連付けられた IAM ポリシー。
- プロジェクト用に作成されたすべての Amazon EC2 インスタンス。
- 次のような、デプロイアプリケーションと関連リソース。
  - CodeDeploy アプリケーションおよび関連するデプロイグループ。
  - AWS Lambda 関数と関連付けられた API Gateway API。
  - AWS Elastic Beanstalk アプリケーションと関連付けられた環境。
- CodePipeline でのプロジェクトの継続的なデプロイパイプライン。
- プロジェクトに関連付けられた AWS CloudFormation スタック。
- AWS CodeStar コンソールを使用して作成された AWS Cloud9 開発環境。この環境でコミットされていないコードの変更はすべて、失われます。

プロジェクトと一緒に、プロジェクトリソースをすべて削除するには、[Delete resources] (リソースを削除) チェックボックスをオンにします。このオプションをオフにした場合、AWS CodeStar のプロジェクトは削除され、これらのリソースへのアクセスを有効にしたプロジェクトロールは IAM から削除されますが、他のすべてのリソースは保持されます。AWS のこうしたリソースに対して引き続き課金が発生する可能性があります。1 つ以上のリソースが不要になると判断した場合は、それらを手動で削除する必要があります。詳細については、「プロジェクトの削除: AWS CodeStar プロジェクトは削除されましたが、リソースはまだ存在します」を参照してください。
プロジェクトを削除する際にリソースを保持することにした場合は、ベストプラクティスとして、[Project details] (プロジェクトの詳細) ページからリソースのリストをコピーします。このように、プロジェクトがなくなっても、保有しているすべてのリソースのレコードは削除されません。

トピック

- AWS CodeStar のプロジェクトを削除する (コンソール)
- AWS CodeStar のプロジェクトを削除する (AWS CLI)

AWS CodeStar のプロジェクトを削除する (コンソール)

プロジェクトを削除するには、AWS CodeStar コンソールを使用します。

AWS CodeStar でプロジェクトを削除するには

1. https://console.aws.amazon.com/codestar/ で AWS CodeStar コンソールを開きます。
2. ナビゲーションペインで、[Projects] (プロジェクト) を選択します。
3. 削除するプロジェクトを選択して、[Delete] (削除) を選択します。

   または、プロジェクトを開き、コンソールの左側のナビゲーションペインから [Settings] (設定) を選択します。[Project details] (プロジェクトの詳細)ページで、[Delete project] (プロジェクトの削除) を選択します。
4. [Delete confirmation page] (削除の確認ページ) で、[delete] (削除) と入力します。プロジェクトリソースを削除する場合、[Delete resources] (リソースの削除) を選択したままにします。[Delete] (削除) を選択します。

   プロジェクトの削除には数分かかる場合があります。削除された後、プロジェクトは AWS CodeStar コンソールのプロジェクトの一覧に表示されなくなります。

> **Important**
>
> プロジェクトで AWS 以外のリソース (例: GitHub リポジトリ、Atlassian JIRAの課題) を使用している場合は、チェックボックスがオンになっていてもそれらのリソースは削除されません。
> AWS CodeStar 管理ポリシーが、IAM ユーザーではないロールに手動でアタッチされている場合でも、プロジェクトを削除することはできません。プロジェクトの管理ポリシーをフェデレーティッドユーザーのロールに添付している場合は、プロジェクトを削除する前にポリシーをデタッチする必要があります。詳細については、「???」を参照してください。

AWS CodeStar のプロジェクトを削除する (AWS CLI)

AWS CLI を使用してプロジェクトを削除できます。

AWS CodeStar でプロジェクトを削除するには

1. ターミナル (Linux、macOS、または Unix) またはコマンドプロンプト (Windows) で、プロジェクト名を含む delete-project コマンドを実行します。たとえば、ID *my-2nd-project* のプロジェクトを削除するには、次のように入力します：

   ```
   aws codestar delete-project --id my-2nd-project
   ```

   このコマンドは、次のような出力を返します：

   ```
   {
       "projectArn":"arn:aws:codestar:us-east-2:111111111111:project/my-2nd-project"
   }
   ```

   プロジェクトはすぐには削除されません。
2. プロジェクトの名前を含めて、describe-project コマンドを実行します。たとえば、ID が *my-2nd-project* のプロジェクトのステータスを確認するには、次のようなコマンドを実行します。

   ```
   
       aws codestar describe-project --id my-2nd-project
   
   ```

   プロジェクトがまだ削除されていない場合、このコマンドは以下のような出力を返します：

   ```
   {
       "name": "my project",
       "id": "my-2nd-project",
       "arn": "arn:aws:codestar:us-west-2:123456789012:project/my-2nd-project",
       "description": "My second CodeStar project.",
       "createdTimeStamp": 1572547510.128,
       "status": {
           "state": "CreateComplete"
       }
   }
   ```

プロジェクトが削除されている場合、このコマンドは以下のような出力を返します：

```
   An error occurred (ProjectNotFoundException) when calling the DescribeProject
operation: The project ID was not found: my-2nd-project. Make sure that the
project ID is correct and then try again.
```

3. list-projects コマンドを実行し、削除されたプロジェクトが AWS アカウントに関連付けられたプロジェクトのリストに表示されないことを確認します。

```
aws codestar list-projects
```

AWS CodeStar のチームで作業する

開発プロジェクトを作成したら、一緒に作業できるように他のユーザーにアクセス権を付与します。AWS CodeStar では、各プロジェクトにプロジェクトチームがあります。ユーザーは、複数の AWS CodeStar プロジェクトに所属し、それぞれに異なる AWS CodeStar ロール (つまり、アクセス許可が異なる) を持つことができます。AWS CodeStar コンソールで、ユーザーは AWS アカウントに関連付けられているすべてのプロジェクトを表示できますが、それらのプロジェクトのチームメンバーである場合にのみプロジェクトの表示および作業ができます。

チームメンバーは自分のわかりやすい名前を選択できます。また、他のチームメンバーと連絡できるように E メールアドレスを追加できます。所有者でないチームメンバーがプロジェクトの AWS CodeStar ロールを変更することはできません。

AWS CodeStar の各プロジェクトに 3 つのロールがあります。

AWS CodeStar プロジェクトのロールとアクセス権限

ロール名	プロジェクトダッシュボードとステータスの表示	プロジェクトリソースの追加、削除、アクセス	チームメンバーの追加と削除	プロジェクトの削除
所有者	x	x	x	x
[Contributor] (寄稿者)	x	x		
閲覧者	x			

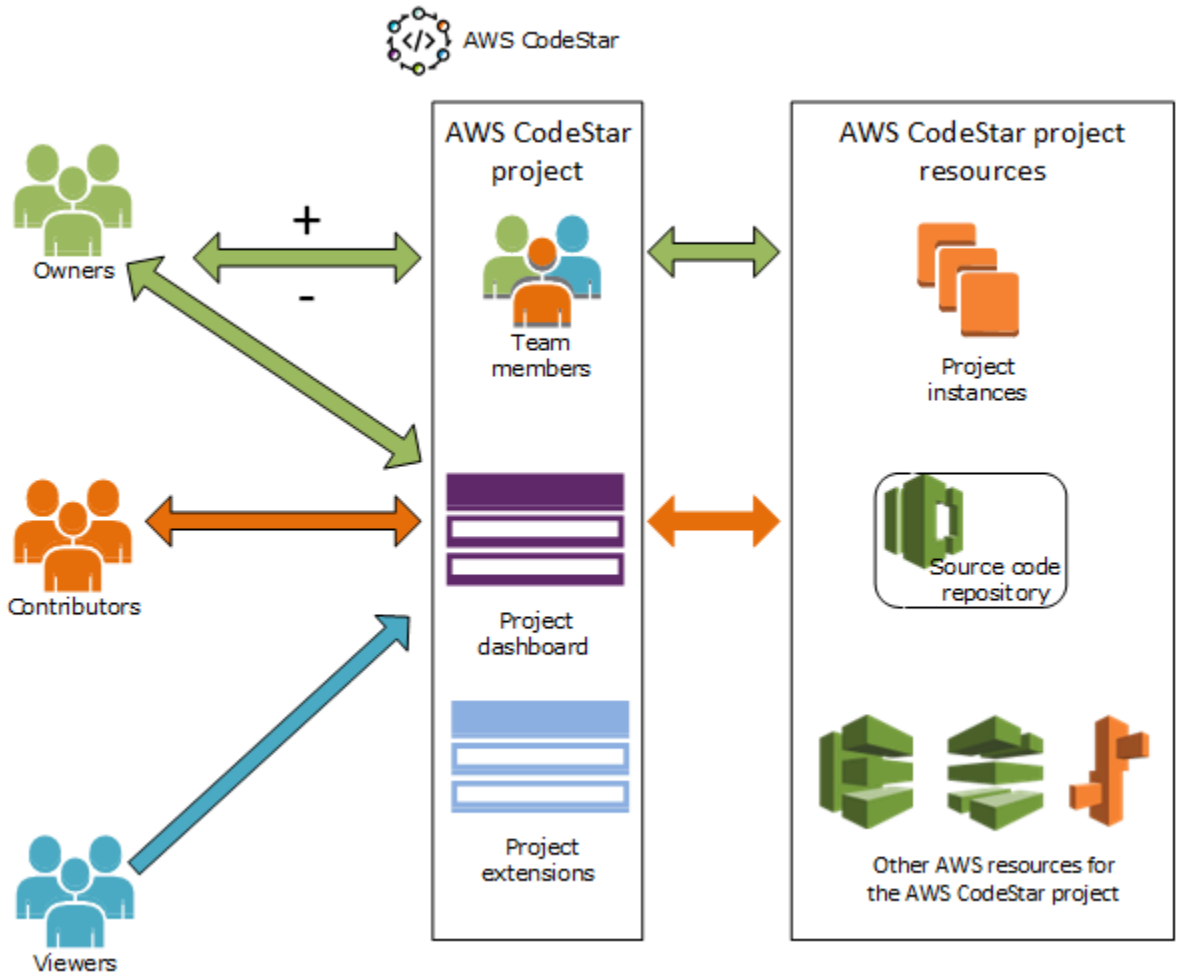
- 所有者: コードが CodeCommit に保存されている場合に、他のチームメンバーの追加および削除、プロジェクトリポジトリへのコードの投稿、プロジェクトに関連した Linux で実行されている Amazon EC2 インスタンスへの他のチームメンバーのリモートアクセスの許可および拒否、プロジェクトダッシュボードの設定、および、プロジェクトの削除ができます。
- 寄稿者: コードが CodeCommit に保存されている場合に、JIRA タイルなどのダッシュボードリソースの追加および削除、プロジェクトリポジトリへのコードの投稿、およびダッシュボードの十分な操作ができます。チームメンバーの追加または削除、リソースへのリモートアクセスの許可または拒否、および、プロジェクトの削除はできません。これは、ほとんどのチームメンバーに対して選択すべきロールです。

- 閲覧者: コードが CodeCommit に保存されている場合に、プロジェクトダッシュボード、コードの表示、およびダッシュボードタイルへのプロジェクトとリソースの状態の表示ができます。

> **Important**
>
> プロジェクトで AWS 以外のリソース (例: GitHub リポジトリ、または Atlassian JIRA の問題) が使用されている場合、それらのリソースへのアクセスは AWS CodeStar ではなくリソースプロバイダーによって制御されます。詳細については、リソースプロバイダのドキュメントを参照してください。
>
> AWS CodeStar プロジェクトにアクセスできるユーザーは、AWS CodeStar コンソールを使用して AWS 以外のリソースにアクセスできますが、そのプロジェクトに関連している可能性があります。
>
> AWS CodeStar では、プロジェクトチームのメンバーが、プロジェクトの関連する AWS Cloud9 開発環境に参加することは自動で許可されません。チームメンバーによる共有環境への参加を許可するには、「AWS Cloud9 環境をプロジェクトチームメンバーと共有する」を参照してください。

IAM ポリシーは、各プロジェクトロールに関連付けられています。このポリシーは、リソースを反映してプロジェクト用にカスタマイズされています。これらのポリシーの詳細については、「AWS CodeStar ID ベースのポリシーの例」を参照してください。

以下の図は、各ロールと AWS CodeStar プロジェクトの関係を示しています。



トピック

- AWS CodeStar プロジェクトにチームメンバーを追加する
- AWS CodeStar チームメンバーのアクセス許可の管理
- AWS CodeStar プロジェクトからチームメンバーを削除する

AWS CodeStar プロジェクトにチームメンバーを追加する

AWS CodeStar プロジェクトで所有者ロールがある場合、または IAM ユーザーに `AWSCodeStarFullAccess` ポリシーが適用されている場合は、他の IAM ユーザーをプロジェクトチームに追加できます。これは、AWS CodeStar ロール (所有者、寄稿者、および表示者) をユーザーに適用する簡単なプロセスです。これらのロールはプロジェクトごとでカスタマイズされています。例えば、Project A の寄稿者チームメンバーは、Project B の寄稿者チームメンバーのリソースとは異なるリソースへのアクセス許可を持っている可能性があります。チームメンバーは、プロジェク

トで 1 つのロールのみを持つことができます。チームメンバーを追加すると、ロールによって定義されたレベルでプロジェクトに直ちにかかわることができます。

AWS CodeStar ロールとチームメンバーシップの利点は、次のとおりです。

- チームメンバーの IAM でアクセス許可を手動で設定する必要はありません。
- チームメンバーのプロジェクトへのアクセスレベルを簡単に変更できます。
- ユーザーは、チームメンバーである場合にのみ、AWS CodeStar コンソールでプロジェクトにアクセスできます。
- プロジェクトへのユーザーアクセスは、ロールによって定義されます。

チームと AWS CodeStar ロールの詳細については、「AWS CodeStar のチームで作業する」と「AWS CodeStar ユーザープロファイルの操作方法 」を参照してください。

チームメンバーをプロジェクトに追加するには、そのプロジェクトの AWS CodeStar 所有者のロールか、または `AWSCodeStarFullAccess` ポリシーが必要です。

> **Important**
>
> チームメンバーを追加しても、AWS 以外のリソース (例: GitHub リポジトリや Atlassian JIRA の課題) へのメンバーのアクセスには影響しません。これらのアクセス権限可は AWS CodeStar ではなく、リソースプロバイダによって制御されます。詳細については、リソースプロバイダのドキュメントを参照してください。
>
> AWS CodeStar プロジェクトにアクセスできるユーザーは、AWS CodeStar コンソールを使用して AWS 以外のリソースで、そのプロジェクトに関連しているリソースにアクセスできます。
>
> プロジェクトにチームメンバーを追加しても、プロジェクトの関連する AWS Cloud9 開発環境へのそのメンバーの参加は自動的に許可されません。チームメンバーによる共有環境への参加を許可するには、「AWS Cloud9 環境をプロジェクトチームメンバーと共有する」を参照してください。
>
> プロジェクトへのアクセス権をフェデレーティッドユーザーに付与するには、AWS CodeStar 所有者、寄稿者、または表示者の管理ポリシーをフェデレーティッドユーザーによって引き受けられたロールに手動でアタッチする必要があります。詳細については、「へのフェデレーティッドユーザーアクセス AWS CodeStar」を参照してください。

トピック

- チームメンバーを追加する (コンソール)
- チームメンバーを追加および表示する (AWS CLI)

チームメンバーを追加する (コンソール)

チームメンバーをプロジェクトに追加するには、AWS CodeStar コンソールを使用します。追加するユーザーの IAM ユーザーがすでに存在する場合は、その IAM ユーザーを追加することができます。存在しない場合は、プロジェクトに追加する際、そのユーザーの IAM ユーザーを作成することができます。

チームメンバーを AWS CodeStar プロジェクトに追加するには (コンソール)

1. https://console.aws.amazon.com/codestar/ で AWS CodeStar コンソールを開きます。
2. ナビゲーションペインから、[Projects] (プロジェクト) を選択し、プロジェクトを選択します。
3. プロジェクトのサイドナビゲーションペインで、[Team] (チーム) を選択します。
4. [Team members] (チームメンバー) ページで、[Add team member] (チームメンバーの追加) を選択します。
5. [Choose user] (ユーザーを選択) で、次のいずれかを実行します：

   - 追加する人物の IAM ユーザーがすでに存在する場合は、その IAM ユーザーをリストから選択します。

     > **Note**
     >
     > 別の AWS CodeStar プロジェクトにすでに追加されているユーザーは、[既存の AWS CodeStar ユーザー] リストに表示されます。

     [プロジェクトロール] で、このユーザーの AWS CodeStar ロール (所有者、寄稿者、閲覧者) を選択します。これは AWS CodeStar プロジェクトレベルのロールで、プロジェクトの所有者によってのみ変更できます。IAM ユーザーにロールが適用されると、AWS CodeStar プロジェクトリソースにアクセスするために必要なアクセス許可がすべて付与されます。コードが IAM の CodeCommit に保存されている場合の Git 認証情報の作成と管理に必要なポリシー、または IAM でユーザーの Amazon EC2 SSH キーをアップロードするのに必要なポリシーが適用されます。

> **Important**
>
> 該当のユーザーとしてコンソールにサインインしていない限り、IAM ユーザーの表示名または E メール情報を入力または変更することはできません。詳細については、「AWS CodeStar ユーザープロファイルの表示情報を管理する」を参照してください。

[Add team member] (チームメンバーの追加) を選択します。

- プロジェクトに追加する人物の IAM ユーザーが存在しない場合は、[Create new IAM user] (新規 IAM ユーザーを作成) を選択します。新規 IAM ユーザーを作成できる IAM コンソールにリダイレクトされます。詳細については「IAM ユーザーガイド」の「AWS アカウントでの IAM ユーザーの作成」を参照してください。IAM ユーザーを作成後、AWS CodeStar コンソールに戻り、ユーザーのリストを更新し、ドロップダウンリストから作成した IAM ユーザーを選択します。この新しいユーザーに適用する AWS CodeStar 表示名、E メールアドレス、およびプロジェクトロールを入力し、[チームメンバーを追加] を選択します。

> **Note**
>
> 管理しやすいように、少なくとも 1 人のユーザーにプロジェクトの所有者ロールを割り当てます。

6. 新しいチームメンバーに、次の情報を送信します：

   - AWS CodeStar プロジェクトのための接続情報。
   - ソースコードが CodeCommit に保存されている場合、ローカルコンピュータから CodeCommit リポジトリに [instructions for setting up access with Git credentials] (Git 認証情報でアクセスを設定する手順)。
   - AWS CodeStar ユーザープロファイルの操作方法 で説明されているように、ユーザーが表示名、E メールアドレス、公開 Amazon EC2 SSH キーを管理する方法についての情報。
   - ユーザーが AWS を使用するのは初めてで、そのユーザーのために IAM ユーザーを作成した場合のワンタイムパスワードと接続情報。このパスワードはユーザーの初回サインイン時に失効します。ユーザーは新しいパスワードを選択する必要があります。

チームメンバーを追加および表示する (AWS CLI)

チームメンバーをプロジェクトチームに追加するには、AWS CLI を使用します。また、プロジェクト内のすべてのチームメンバーに関する情報を表示することもできます。

チームメンバーを追加するには

1. ターミナルまたはコマンドウィンドウを開きます。
2. `--project-id`、`-user-arn`、`--project-role` パラメータを指定して、associate-team-member コマンドを実行します。`--remote-access-allowed` または `--no-remote-access-allowed` パラメータを含めることによって、ユーザーがプロジェクトインスタンスにリモートアクセスできるかどうかを指定することもできます。例:

```
aws codestar associate-team-member --project-id my-first-projec --user-arn
 arn:aws:iam:111111111111:user/Jane_Doe --project-role Contributor --remote-access-
allowed
```

このコマンドは出力なしを返します。

すべてのチームメンバーを表示するには (AWS CLI)

1. ターミナルまたはコマンドウィンドウを開きます。
2. `--project-id` パラメータを指定して、list-team-members コマンドを実行します。例:

```
aws codestar list-team-members --project-id my-first-projec
```

このコマンドは、次のような出力を返します :

```
{
    "teamMembers":[

 {"projectRole":"Owner","remoteAccessAllowed":true,"userArn":"arn:aws:iam::111111111111:use
Mary_Major"},

 {"projectRole":"Contributor","remoteAccessAllowed":true,"userArn":"arn:aws:iam::1111111111
Jane_Doe"},

 {"projectRole":"Contributor","remoteAccessAllowed":true,"userArn":"arn:aws:iam::1111111111
John_Doe"},
```

```
 {"projectRole":"Viewer","remoteAccessAllowed":false,"userArn":"arn:aws:iam::111111111111:u
John_Stiles"}
 ]
}
```

AWS CodeStar チームメンバーのアクセス許可の管理

AWS CodeStar ロールを変更することによって、チームメンバーのアクセス許可を変更します。各チームメンバーは、AWS CodeStar プロジェクト内の 1 つのロールにのみ割り当てることができますが、多くのユーザーを同じロールに割り当てることができます。AWS CodeStar コンソールまたは AWS CLI を使用してアクセス許可を管理することはできません。

> **Important**
>
> チームメンバーのロールを変更するには、そのプロジェクトの AWS CodeStar 所有者のロールを持っているか、または `AWSCodeStarFullAccess` ポリシーが適用されている必要があります。
>
> チームメンバーのアクセス許可を変更しても、AWS 以外のリソース (例: GitHub リポジトリや Atlassian JIRA の課題) へのチームメンバーのアクセス権には影響しません。これらのアクセス権限可は AWS CodeStar ではなく、リソースプロバイダによって制御されます。詳細については、リソースプロバイダのドキュメントを参照してください。
>
> AWS CodeStar プロジェクトにアクセスできるユーザーは、AWS CodeStar コンソールを使用して AWS 以外のリソースにアクセスできますが、そのプロジェクトに関連している可能性があります。
>
> プロジェクトのチームメンバーのロールを変更しても、プロジェクトの AWS Cloud9 開発環境へのそのメンバーの参加は自動的に許可または禁止されません。チームメンバーによる共有環境への参加を許可または拒否するには、「AWS Cloud9 環境をプロジェクトチームメンバーと共有する」を参照してください。

プロジェクトに関連付けられた Amazon EC2 Linux インスタンスにリモートアクセスするアクセス許可をユーザーに付与することもできます。このアクセス許可を付与すると、ユーザーは AWS CodeStar ユーザープロファイルに関連付けられた SSH パブリックキーをすべてのチームプロジェクトにアップロードする必要があります。Linux インスタンスに正常に接続するには、ユーザーは、SSH を設定し、ローカルコンピュータ上にプライベートキーを設定する必要があります。

トピック

- チームアクセス許可の管理 (コンソール)
- チームアクセス許可の管理 (AWS CLI)

チームアクセス許可の管理 (コンソール)

AWS CodeStar コンソールでチームメンバーのロールを管理することができます。チームメンバーがプロジェクトに関連付けられた Amazon EC2 インスタンスにリモートアクセスできるかどうかを管理することもできます。

チームメンバーのロールを変更するには

1. https://console.aws.amazon.com/codestar/ で AWS CodeStar コンソールを開きます。
2. ナビゲーションペインから、[Projects] (プロジェクト) を選択し、プロジェクトを選択します。
3. プロジェクトのサイドナビゲーションペインで、[Team] (チーム) を選択します。
4. [Team members] (チームメンバー) ページで、チームメンバーを選択し、[Edit] (編集) を選択します。
5. [プロジェクトロール] で、このユーザーに付与する AWS CodeStar ロール (所有者、寄稿者、表示者) を選択します。

   AWS CodeStar ロールとアクセス許可の詳細については、「AWS CodeStar のチームで作業する」を参照してください。

   [Edit team member] (チームメンバーを編集) を選択します。

チームメンバーに Amazon EC2 インスタンスへのリモートアクセスのアクセス許可を付与するには

1. https://console.aws.amazon.com/codestar/ で AWS CodeStar コンソールを開きます。
2. ナビゲーションペインから、[Projects] (プロジェクト) を選択し、プロジェクトを選択します。
3. プロジェクトのサイドナビゲーションペインで、[Team] (チーム) を選択します。
4. [Team members] (チームメンバー) ページで、チームメンバーを選択し、[Edit] (編集) を選択します。
5. [Allow SSH access to project instances] (プロジェクトインスタンスへの SSH アクセスを許可する) を選択して、[Edit team member] (チームメンバーの編集) を選択します。

6. (オプション) まだ行っていない場合、AWS CodeStar ユーザーの SSH パブリックキーをアップロードする必要があることをチームメンバーに通知します。詳細については、「AWS CodeStar ユーザープロファイルにパブリックキーを追加する」を参照してください。

チームアクセス許可の管理 (AWS CLI)

AWS CLI を使用して、チームメンバーに割り当てられたプロジェクトロールを管理します。同じ AWS CLI コマンドを使用して、チームメンバーがプロジェクトに関連付けられた Amazon EC2 インスタンスにリモートアクセスできるかどうかを管理することができます。

チームメンバーのアクセス許可を管理するには

1. ターミナルまたはコマンドウィンドウを開きます。
2. `--project-id`、`-user-arn`、`--project-role` パラメータを指定して、update-team-member コマンドを実行します。`--remote-access-allowed` または `--no-remote-access-allowed` パラメータを含めることによって、ユーザーがプロジェクトインスタンスにリモートアクセスできるかどうかを指定することもできます。たとえば、John_Doe という名前の IAM ユーザーのプロジェクトロールを更新し、プロジェクト Amazon EC2 インスタンスへのリモートアクセスなしで閲覧者のアクセス許可に変更するには、次のようにします：

```
aws codestar update-team-member --project-id my-first-projec --user-arn
 arn:aws:iam:111111111111:user/John_Doe --project-role Viewer --no-remote-access-
allowed
```

このコマンドは、次のような出力を返します：

```
{
 "projectRole":"Viewer",
 "remoteAccessAllowed":false,
 "userArn":"arn:aws:iam::111111111111:user/John_Doe"
}
```

AWS CodeStar プロジェクトからチームメンバーを削除する

AWS CodeStar プロジェクトから削除したユーザーはプロジェクトリポジトリのコミット履歴に表示されますが、CodeCommit リポジトリや、プロジェクトパイプラインなどの他のプロジェクトリソースにはアクセスできなくなります。(このルールの例外は、これらのリソースへのアクセスを許

可する他のポリシーが適用されている IAM ユーザーです。) ユーザーはプロジェクトダッシュボードにアクセスすることができないため、今後プロジェクトは、ユーザーが AWS CodeStar ダッシュボードで確認するプロジェクトのリストに表示されません。プロジェクトチームからチームメンバーを削除するには、AWS CodeStar コンソール、または AWS CLI を使用します。

> **Important**
>
> プロジェクトからチームメンバーを削除すると、プロジェクト Amazon EC2 インスタンスへのリモートアクセスは拒否されますが、ユーザーのアクティブな SSH セッションは閉じられません。
> チームメンバーを削除しても、AWS 以外のリソース (例: GitHub リポジトリや Atlassian JIRA の課題) へのチームメンバーのアクセスには影響しません。これらのアクセス権限可は AWS CodeStar ではなく、リソースプロバイダによって制御されます。詳細については、リソースプロバイダのドキュメントを参照してください。
> プロジェクトからチームメンバーを削除しても、自動的にそのチームメンバーの関連する AWS Cloud9 開発環境が削除されたり、招待された関連する AWS Cloud9 開発環境へのそのメンバーの参加が禁止されたりすることはありません。開発環境を削除するには、「プロジェクトから AWS Cloud9 環境を削除する」を参照してください。チームメンバーによる共有環境への参加を拒否するには、「AWS Cloud9 環境をプロジェクトチームメンバーと共有する」を参照してください。

プロジェクトからチームメンバを削除するには、そのプロジェクトの AWS CodeStar 所有者のロールを持っているか、`AWSCodeStarFullAccess` ポリシーがアカウントに適用されている必要があります。

トピック

- チームメンバーを削除する (コンソール)
- チームメンバーを削除する (AWS CLI)

チームメンバーを削除する (コンソール)

プロジェクトチームからチームメンバーを削除するには、AWS CodeStar コンソールを使用します。

プロジェクトからチームメンバーを削除するには

1. https://console.aws.amazon.com/codestar/ で AWS CodeStar コンソールを開きます。
2. ナビゲーションペインから、[Projects] (プロジェクト) を選択し、プロジェクトを選択します。
3. プロジェクトのサイドナビゲーションペインで、[Team] (チーム) を選択します。
4. リポジトリの [Team members] (チームメンバー) ページで、チームメンバーを選択し、[Remove] (削除) を選択します。

チームメンバーを削除する (AWS CLI)

プロジェクトチームからチームメンバーを削除するには、AWS CLI を使用します。

チームメンバーを削除するには

1. ターミナルまたはコマンドウィンドウを開きます。
2. disassociate-team-member および `--project-id` を指定して、`-user-arn` コマンドを実行します。例:

```
aws codestar disassociate-team-member --project-id my-first-projec --user-arn
 arn:aws:iam:111111111111:user/John_Doe
```

このコマンドは、次のような出力を返します :

```
{
    "projectId": "my-first-projec",
    "userArn": "arn:aws:iam::111111111111:user/John_Doe"
}
```

AWS CodeStar ユーザープロファイルの操作方法

AWS CodeStar ユーザープロファイルは、IAM ユーザーに関連付けられています。このプロファイルには、所属するすべての AWS CodeStar プロジェクトで使用する表示名と E メールアドレスが含まれます。プロファイルに関連付けられる SSH パブリックキーをアップロードできます。このパブリックキーは、所属する AWS CodeStar プロジェクトに関連付けられた Amazon EC2 インスタンスに接続するときに使用する SSH パブリック/プライベートキーのペアの一部です。

> **Note**
>
> これらのトピックの情報は、AWS CodeStar ユーザープロファイルのみを対象としています。プロジェクトで AWS 以外のリソース (例: GitHub リポジトリや Atlassian JIRA の課題) を使用している場合、これらのリソースプロバイダーには、設定が異なる独自のユーザープロファイルを使用してる可能性があります。詳細については、リソースプロバイダのドキュメントを参照してください。

トピック

- AWS CodeStar ユーザープロファイルの表示情報を管理する
- AWS CodeStar ユーザープロファイルにパブリックキーを追加する

AWS CodeStar ユーザープロファイルの表示情報を管理する

ユーザプロファイルの表示名および E メールアドレスを変更するには、AWS CodeStar コンソールまたは AWS CLI を使用します。ユーザープロファイルはプロジェクト固有ではありません。このプロファイルは、IAM ユーザーに関連付けられており、AWS リージョン内で属している AWS CodeStar プロジェクト全体で適用されます。複数の AWS リージョンでプロジェクトに属している場合は、個別のユーザープロファイルがあります。

AWS CodeStar コンソールでのみ自分のユーザープロファイルを管理できます。`AWSCodeStarFullAccess` ポリシーがある場合は、AWS CLI を使用して他のプロファイルを表示および管理できます。

> **Note**
>
> このトピックの情報は、AWS CodeStar ユーザープロファイルのみを対象としています。プロジェクトで AWS 以外のリソース (例: GitHub リポジトリや Atlassian JIRA の課題) を使用

している場合、これらのリソースプロバイダーには、設定が異なる独自のユーザープロファイルを使用してる可能性があります。詳細については、リソースプロバイダのドキュメントを参照してください。

トピック

- ユーザープロファイルの管理 (コンソール)
- ユーザープロファイルの管理 (AWS CLI)

ユーザープロファイルの管理 (コンソール)

自分がチームメンバーであるプロジェクトに移動してプロフィール情報を変更することで、AWS CodeStar コンソールでユーザープロファイルを管理できます。ユーザープロファイルはプロジェクト固有ではなく、ユーザー固有であるため、ユーザープロファイルの変更は、AWS リージョン内で自分がチームメンバーであるすべてのプロジェクトに表示されます。

> **Important**
>
> コンソールでユーザーの表示情報を変更するには、その IAM ユーザーとしてサインインしている必要があります。他のユーザーは、プロジェクトの AWS CodeStar 所有者ロールがあっても、または、`AWSCodeStarFullAccess` ポリシーが適用されていても、表示情報を変更することはできません。

AWS リージョン内のすべてのプロジェクトの表示情報を変更するには

1. https://console.aws.amazon.com/codestar/ で AWS CodeStar コンソールを開きます。
2. ナビゲーションペインで [Projects] (プロジェクト) を選択し、自分がチームメンバーであるプロジェクトを選択します。
3. プロジェクトのサイドナビゲーションペインで、[Team] (チーム) を選択します。
4. [Team members] (チームメンバー) ページで、 IAM ユーザーを選択し、[Edit] (編集) を選択します。
5. 表示名か E メールアドレス、またはその両方を編集して、[Edit team member] (チームメンバーの編集) を選択します。

> **Note**
>
> 表示名と E メールアドレスが必須です。詳細については、「AWS CodeStar における制限」を参照してください。

ユーザープロファイルの管理 (AWS CLI)

AWS CLI を使用して、AWS CodeStar のユーザープロファイルを作成および管理できます。また、AWS CLI を使用して、ユーザープロファイル情報を表示し、AWS リージョンで自分の AWS アカウント用に設定されたすべてのユーザープロファイルを表示できます。

ユーザープロファイルを作成、管理、または表示するリージョンで設定された AWS プロファイルであることを確認してください。

ユーザープロファイルを作成するには

1. ターミナルまたはコマンドウィンドウを開きます。
2. `user-arn`、`display-name`、`email-address` パラメータを指定して、create-user-profile コマンドを実行します。例:

```
aws codestar create-user-profile --user-arn arn:aws:iam:111111111111:user/
John_Stiles --display-name "John Stiles" --email-address "john_stiles@example.com"
```

このコマンドは、次のような出力を返します :

```
{
 "createdTimestamp":1.491439687681E9,"
 displayName":"John Stiles",
 "emailAddress":"john.stiles@example.com",
 "lastModifiedTimestamp":1.491439687681E9,
 "userArn":"arn:aws:iam::111111111111:user/Jane_Doe"
}
```

表示情報を確認するには

1. ターミナルまたはコマンドウィンドウを開きます。
2. `user-arn` パラメータを指定して、describe-user-profile コマンドを実行します。例:

```
aws codestar describe-user-profile --user-arn arn:aws:iam:111111111111:user/
Mary_Major
```

このコマンドは、次のような出力を返します：

```
{
 "createdTimestamp":1.490634364532E9,
 "displayName":"Mary Major",
 "emailAddress":"mary.major@example.com",
 "lastModifiedTimestamp":1.491001935261E9,
 "sshPublicKey":"EXAMPLE=",
 "userArn":"arn:aws:iam::111111111111:user/Mary_Major"
}
```

表示情報を変更するには

1. ターミナルまたはコマンドウィンドウを開きます。
2. user-arn パラメータ、および display-name や email-address パラメータなどの変更するパラメータを指定して、update-user-profile コマンドを実行します。たとえば、「Jane Doe」という表示名のユーザーが表示名を「Jane Mary Doe」に変更する場合は次のようにします：

```
aws codestar update-user-profile --user-arn arn:aws:iam:111111111111:user/Jane_Doe
 --display-name "Jane Mary Doe"
```

このコマンドは、次のような出力を返します：

```
{
 "createdTimestamp":1.491439687681E9,
 "displayName":"Jane Mary Doe",
 "emailAddress":"jane.doe@example.com",
 "lastModifiedTimestamp":1.491442730598E9,
 "sshPublicKey":"EXAMPLE1",
 "userArn":"arn:aws:iam::111111111111:user/Jane_Doe"
}
```

AWS アカウントの AWS リージョンですべてのユーザープロファイルを一覧表示するには

1. ターミナルまたはコマンドウィンドウを開きます。
2. aws codestar list-user-profiles コマンドを実行します。例:

```
aws codestar list-user-profiles
```

このコマンドは、次のような出力を返します：

```
{
  "userProfiles":[
 {
  "displayName":"Jane Doe",
  "emailAddress":"jane.doe@example.com",
  "sshPublicKey":"EXAMPLE1",
  "userArn":"arn:aws:iam::111111111111:user/Jane_Doe"
 },
 {
  "displayName":"John Doe",
  "emailAddress":"john.doe@example.com",
  "sshPublicKey":"EXAMPLE2",
  "userArn":"arn:aws:iam::111111111111:user/John_Doe"
 },
 {
  "displayName":"Mary Major",
  "emailAddress":"mary.major@example.com",
  "sshPublicKey":"EXAMPLE=",
  "userArn":"arn:aws:iam::111111111111:user/Mary_Major"
 },
 {
  "displayName":"John Stiles",
  "emailAddress":"john.stiles@example.com",
  "sshPublicKey":"",
  "userArn":"arn:aws:iam::111111111111:user/John_Stiles"
 }
  ]
}
```

AWS CodeStar ユーザープロファイルにパブリックキーを追加する

パブリック SSH キーは、作成および管理するパブリックキー/プライベートキーのペアの一部としてアップロードできます。この SSH パブリックキー/プライベートキーのペアを使用して、Linux を実行する Amazon EC2 インスタンスにアクセスします。プロジェクト所有者によってリモートアクセスのアクセス許可が付与されている場合は、プロジェクトに関連付けられているインスタンスにのみアクセスできます。 AWS CodeStar コンソールまたは を使用して AWS CLI 、パブリックキーを管理できます。

> **Important**
>
> AWS CodeStar プロジェクト所有者は、プロジェクト所有者、寄稿者、およびビューワーにプロジェクトの Amazon EC2 インスタンスへの SSH アクセスを許可できますが、個々の (所有者、寄稿者、またはビューワー) のみが SSH キーを設定できます。そのためには、ユーザーが、所有者、寄稿者、または閲覧者としてサインインしている必要があります。AWS CodeStar は AWS Cloud9 環境の SSH キーを管理しません。

トピック

- ポリシーキーを管理する (コンソール)
- パブリックキーを管理する (AWS CLI)
- プライベートキーを使用して Amazon EC2 インスタンスに接続

ポリシーキーを管理する (コンソール)

コンソールでパブリックキーとプライベートキーのペアを生成することはできませんが、ローカルで作成し、 AWS CodeStar コンソールからユーザープロファイルの一部として追加または管理できます。

パブリック SSH キーを管理するには

1. 端末または Bash エミュレータのウィンドウから、ssh-keygen コマンドを実行して、ローカルコンピュータに SSH パブリックキーとプライベートキーのペアを生成します。Amazon EC2 で許可されている形式でキーを生成できます。受け入れ可能な形式については、[Importing Your Own Public Key to Amazon EC2] (独自のパブリックキーを Amazon EC2 にインポートする)

を参照してください。理想的には、SSH-2 RSA であるキーを OpenSSH 形式で生成し、2048 ビットを含めます。パブリックキーは、.pub 拡張子の付いたファイルに保存されます。

2. https://console.aws.amazon.com/codestar/ で AWS CodeStar コンソールを開きます。

   自分がチームメンバーであるプロジェクトを選択します。

3. ナビゲーションペインで、[Team] (チーム) を選択します。
4. [Team members] (チームメンバー) ページで、IAM ユーザーの名前を探して、[Edit] (編集) を選択します。
5. [Edit team member] (チームメンバーの編集) ページの[Remote access] (リモートアクセス) で、[Allow SSH access to project instances] (プロジェクトインスタンスへの SSH アクセスを許可する) を有効にします。
6. [SSH Public Key] (SSH パブリックキー) ボックスで、パブリックキーを貼り付け、[Edit team member] (チームメンバーの編集) を選択します。

> **Note**
>
> このフィールドの古いキーを削除して新しいキーを貼り付けることで、パブリックキーを変更できます。パブリックキーを削除するには、このフィールドの内容を削除し、[Edit team member] (チームメンバーの編集) を選択します。

パブリックキーを変更または削除すると、ユーザープロファイルが変更されます。プロジェクトの変更はありません。キーはプロファイルに関連付けられているため、リモートアクセスが許可されているすべてのプロジェクトで変更または削除されます。

パブリックキーを削除すると、リモートアクセスが許可されたすべてのプロジェクトで Linux を実行するAmazon EC2 インスタンスへのアクセスが削除されます。ただし、そのキーを使用して開いている SSH セッションが閉じられることはありません。開いているセッションを閉じたことを確認します。

パブリックキーを管理する (AWS CLI)

を使用して AWS CLI 、ユーザープロファイルの一部として SSH パブリックキーを管理できます。

パブリックキーを管理するには

1. 端末または Bash エミュレータのウィンドウから、ssh-keygen コマンドを実行して、ローカルコンピュータに SSH パブリックキーとプライベートキーのペアを生成します。Amazon EC2 で許可されている形式でキーを生成できます。受け入れ可能な形式については、[Importing Your Own Public Key to Amazon EC2] (独自のパブリックキーを Amazon EC2 にインポートする) を参照してください。理想的には、SSH-2 RSA であるキーを OpenSSH 形式で生成し、2048 ビットを含めます。パブリックキーは、.pub 拡張子の付いたファイルに保存されます。
2. AWS CodeStar ユーザープロファイルで SSH パブリックキーを追加または変更するには、 `--ssh-public-key`パラメータを指定して update-user-profile コマンドを実行します。例:

```
aws codestar update-user-profile --user-arn arn:aws:iam:111111111111:user/Jane_Doe
 --ssh-key-id EXAMPLE1
```

このコマンドは、次のような出力を返します :

```
{
 "createdTimestamp":1.491439687681E9,
 "displayName":"Jane Doe",
 "emailAddress":"jane.doe@example.com",
 "lastModifiedTimestamp":1.491442730598E9,
 "sshPublicKey":"EXAMPLE1",
 "userArn":"arn:aws:iam::111111111111:user/Jane_Doe"
}
```

プライベートキーを使用して Amazon EC2 インスタンスに接続

Amazon EC2 キーペアをすでに作成していることを確認します。のユーザープロファイルにパブリックキーを追加します AWS CodeStar。キーペアを作成するには、「ステップ 4: AWS CodeStar プロジェクトの Amazon EC2 キーペアの作成」を参照してください。パブリックキーをユーザープロファイルに追加するには、このトピックの前半にある手順を参照してください。

プライベートキーを使用して Amazon EC2 Linux インスタンスに接続するには

1. AWS CodeStar コンソールでプロジェクトを開いた状態で、ナビゲーションペインでプロジェクト を選択します。
2. [Project Resources] (プロジェクトリソース) で、[Type] (種類) が [Amazon EC2] で [Name] (名前) が [instance] (インスタンス) で始まる行で、[ARN] リンクを選択します。

3. Amazon EC2 コンソールで、[Connect] (接続) を選択します。
4. [Connect To Your Instance] (インスタンスへ接続) ダイアログボックスの指示に従います。

   ユーザー名については、`ec2-user` を使用します。ユーザー名に誤りがある場合は、インスタンスに接続することはできません。

詳細については、Amazon EC2 ユーザーガイド」の以下のリソースを参照してください。

- SSH を使用した Linux インスタンスへの接続
- PuTTY を使用した Windows から Linux インスタンスへの接続
- を使用した Linux インスタンスへの接続 MindTerm

AWS CodeStar のセキュリティ

AWS では、クラウドのセキュリティが最優先事項です。AWS の顧客は、セキュリティを最も重視する組織の要件を満たすように構築されたデータセンターとネットワークアーキテクチャから利点を得られます。

セキュリティは、AWS とお客様の間の責任共有です。責任共有モデルでは、これをクラウドのセキュリティおよびクラウド内のセキュリティと説明しています。

- クラウドのセキュリティ - AWS は、AWS クラウドで AWS のサービスを実行するインフラストラクチャを保護する責任を負います。また、AWS は、安全に使用できるサービスを提供します。AWS コンプライアンスプログラム の一環として、サードパーティーの監査員が定期的にセキュリティの有効性をテストおよび検証しています 。AWS CodeStar に適用されるコンプライアンスプログラムの詳細については、「コンプライアンスプログラムによる対象範囲内の AWS のサービス」を参照してください。
- クラウド内のセキュリティ - ユーザーの責任は、使用する AWS のサービスに応じて異なります。また、お客様は、データの機密性、お客様の会社の要件、および適用される法律および規制など、その他の要因についても責任を負います。

このドキュメントは、AWS CodeStar を使用する際に責任共有モデルを適用する方法を理解するのに役立ちます。次のトピックでは、セキュリティおよびコンプライアンスの目的を達成するために AWS CodeStar を設定する方法を示します。また、AWS リソースのモニタリングや保護に役立つ、他の AWS CodeStar のサービスの使用方法についても説明します。

AWS CodeStar でカスタムポリシーを作成し、アクセス許可の境界を使用する場合は、タスクの実行に必要なアクセス許可のみを付与し、アクセス許可を対象リソースに狭めて、最小特権のアクセス権を確保します。他のプロジェクトのメンバーがプロジェクト内のリソースにアクセスできないようにするには、組織メンバーに AWS CodeStar プロジェクトごとに個別のアクセス権限を付与します。ベストプラクティスとして、各メンバーのプロジェクトアカウントを作成し、そのアカウントにロールベースのアクセス権を割り当てます。

例えば、DevOps グループの下の各デベロッパーロールのアカウントをプロビジョニングするAWS 組織のAWS Control Tower のようなサービスを使用できます その後、それらのアカウントにアクセス許可を割り当てることができます。全体的なアクセス許可はアカウントに適用されますが、ユーザーはプロジェクト外のリソースへのアクセス権が制限されています。

マルチアカウント戦略 を使用した AWS リソースへの最小特権アクセス権の管理の詳細については、「AWS Control Tower ユーザーガイド」 の 「ランディングゾーン の AWS マルチアカウント戦略」 を参照してください。

トピック

- でのデータ保護 AWS CodeStar
- の Identity and Access Management AWS CodeStar
- AWS CloudTrail を使用した AWS CodeStar API コールのログ記録
- AWS CodeStar のコンプライアンス検証
- AWS CodeStar での耐障害性
- のインフラストラクチャセキュリティ AWS CodeStar

でのデータ保護 AWS CodeStar

責任 AWS 共有モデル、 でのデータ保護に適用されますAWS CodeStar。このモデルで説明されているように、 AWS はすべての を実行するグローバルインフラストラクチャを保護する責任があります AWS クラウド。お客様は、このインフラストラクチャでホストされているコンテンツに対する管理を維持する責任があります。また、使用する AWS のサービス のセキュリティ設定と管理タスクもユーザーの責任となります。データプライバシーの詳細については、「データプライバシー FAQ」を参照してください。欧州におけるデータ保護の詳細については、AWS 「 セキュリティブログ」のAWS 「責任共有モデル」とGDPRブログ記事を参照してください。

データ保護の目的で、認証情報を保護し AWS アカウント 、 AWS IAM Identity Center または AWS Identity and Access Management () を使用して個々のユーザーを設定することをお勧めしますIAM。この方法により、それぞれのジョブを遂行するために必要な権限のみが各ユーザーに付与されます。また、次の方法でデータを保護することもお勧めします:

- 各アカウントで多要素認証 (MFA) を使用します。
- SSL/TLS を使用して AWS リソースと通信します。1TLS.2 が必要で、1.3 TLS をお勧めします。
- を使用して APIおよびユーザーアクティビティのログ記録を設定します AWS CloudTrail。
- AWS 暗号化ソリューションと、 内のすべてのデフォルトのセキュリティコントロールを使用します AWS のサービス。
- Amazon Macie などの高度なマネージドセキュリティサービスを使用します。これらは、Amazon S3 に保存されている機密データの検出と保護を支援します。

- コマンドラインインターフェイスまたは AWS を介して にアクセスするときに FIPS 140-3 検証済みの暗号化モジュールが必要な場合はAPI、FIPSエンドポイントを使用します。利用可能なFIPSエンドポイントの詳細については、「連邦情報処理規格 (FIPS) 140-3」を参照してください。

お客様の E メールアドレスなどの極秘または機密情報は、タグ、または名前フィールドなどの自由形式のテキストフィールドに配置しないことを強くお勧めします。これは、コンソール、、CodeStar または を使用して または他の AWS のサービス を操作する場合API AWS CLIも同様ですAWS SDKs。名前に使用する自由記述のテキストフィールドやタグに入力したデータは、課金や診断ログに使用される場合があります。URL を外部サーバーに提供する場合は、そのサーバーへのリクエストを検証URLするために認証情報を に含めないことを強くお勧めします。

でのデータ暗号化 AWS CodeStar

デフォルトでは、 はプロジェクトに関して保存する情報を AWS CodeStar 暗号化します。プロジェクト名、説明、ユーザーのメールなど、プロジェクト ID 以外はすべて保管時に暗号化されます。プロジェクトに個人情報を入れないでくださいIDs。 は、デフォルトで転送中の情報 AWS CodeStar も暗号化します。保管時の暗号化または転送中の暗号化について、お客様による対応は必要ありません。

の Identity and Access Management AWS CodeStar

AWS Identity and Access Management （ IAM) は、管理者が AWS リソースへのアクセスを安全に制御 AWS のサービス するのに役立つ です。IAM 管理者は、誰を認証 (サインイン) し、誰に AWS CodeStar リソースの使用を承認する (アクセス許可を付与する) かを制御します。IAM は追加料金なしで AWS のサービス 使用できる です。

トピック

- 対象者
- アイデンティティを使用した認証
- ポリシーを使用したアクセスの管理
- AWS CodeStar の仕組み IAM
- AWS CodeStar プロジェクトレベルのポリシーとアクセス許可
- AWS CodeStar ID ベースのポリシーの例
- AWS CodeStar Identity and Access のトラブルシューティング

対象者

AWS Identity and Access Management （ IAM) の使用方法は、 で行う作業によって異なりますAWS CodeStar。

サービスユーザー – AWS CodeStar サービスを使用してジョブを実行する場合、管理者から必要な認証情報とアクセス許可が与えられます。さらに多くのAWS CodeStar 機能を使用して作業を行う場合は、追加のアクセス許可が必要になることがあります。アクセスの管理方法を理解しておくと、管理者に適切な許可をリクエストするうえで役立ちます。の機能にアクセスできない場合は、AWS CodeStar「」を参照してくださいAWS CodeStar Identity and Access のトラブルシューティング。

サービス管理者 – 社内のAWS CodeStar リソースを担当している場合は、通常、 へのフルアクセスがありますAWS CodeStar。サービスユーザーがどのAWS CodeStar 機能やリソースにアクセスするかを決めるのは管理者の仕事です。次に、サービスユーザーのアクセス許可を変更するリクエストをIAM管理者に送信する必要があります。このページの情報を確認して、 の基本概念を理解してくださいIAM。会社で を使用する方法の詳細については、IAMAWS CodeStar「」を参照してくださいAWS CodeStar の仕組み IAM。

IAM 管理者 - IAM管理者は、 へのアクセスを管理するポリシーの作成方法の詳細について確認する場合がありますAWS CodeStar。で使用できるAWS CodeStar アイデンティティベースのポリシーの例を表示するにはIAM、「」を参照してくださいAWS CodeStar ID ベースのポリシーの例。

アイデンティティを使用した認証

認証とは、ID 認証情報 AWS を使用して にサインインする方法です。として、IAMユーザーとしてAWS アカウントのルートユーザー、または IAMロールを引き受けることによって認証 （ にサインイン AWS) される必要があります。

ID ソースを介して提供された認証情報を使用して、フェデレーティッド ID AWS として にサインインできます。 AWS IAM Identity Center （ IAM Identity Center) ユーザー、会社のシングルサインオン認証、Google または Facebook の認証情報は、フェデレーティッド ID の例です。フェデレーティッド ID としてサインインすると、管理者は以前に IAMロールを使用して ID フェデレーションをセットアップしていました。フェデレーション AWS を使用して にアクセスすると、間接的にロールを引き受けることになります。

ユーザーのタイプに応じて、 AWS Management Console または AWS アクセスポータルにサインインできます。へのサインインの詳細については AWS、「 ユーザーガイド」の「 へのサインイン AWS アカウント方法AWS サインイン 」を参照してください。

AWS プログラムで にアクセスする場合、 はソフトウェア開発キット (SDK) とコマンドラインインターフェイス (CLI) AWS を提供し、 認証情報を使用してリクエストに暗号で署名します。 AWS ツールを使用しない場合は、リクエストに自分で署名する必要があります。推奨される方法を使用してリクエストを自分で署名する方法の詳細については、「 IAMユーザーガイド」のAWS API「リクエストの署名」を参照してください。

使用する認証方法を問わず、追加セキュリティ情報の提供をリクエストされる場合もあります。例えば、 AWS では、アカウントのセキュリティを高めるために多要素認証 (MFA) を使用することをお勧めします。詳細については、「 ユーザーガイド」の「多要素認証」および「 ユーザーガイド」の「 での多要素認証 (MFA) AWS IAM の使用」を参照してください。 AWS IAM Identity Center

AWS アカウント ルートユーザー

を作成するときは AWS アカウント、アカウント内のすべての AWS のサービス およびリソースへの完全なアクセス権を持つ 1 つのサインインアイデンティティから始めます。この ID は AWS アカウント ルートユーザーと呼ばれ、アカウントの作成に使用した E メールアドレスとパスワードでサインインすることでアクセスできます。日常的なタスクには、ルートユーザーを使用しないことを強くお勧めします。ルートユーザーの認証情報は保護し、ルートユーザーでしか実行できないタスクを実行するときに使用します。ルートユーザーとしてサインインする必要があるタスクの完全なリストについては、「 IAMユーザーガイド」の「ルートユーザーの認証情報を必要とするタスク」を参照してください。

IAM ユーザーとグループ

IAM ユーザーは、単一のユーザーまたはアプリケーションに対して特定のアクセス許可 AWS アカウント を持つ 内のアイデンティティです。可能であれば、パスワードやアクセスキーなどの長期的な認証情報を持つIAMユーザーを作成する代わりに、一時的な認証情報を使用することをお勧めします。ただし、IAMユーザーとの長期的な認証情報を必要とする特定のユースケースがある場合は、アクセスキーをローテーションすることをお勧めします。詳細については、「 ユーザーガイド」の「長期的な認証情報を必要とするユースケースでアクセスキーを定期的にローテーションするIAM」を参照してください。

IAM グループは、IAMユーザーのコレクションを指定するアイデンティティです。グループとしてサインインすることはできません。グループを使用して、複数のユーザーに対して一度に権限を指定できます。多数のユーザーグループがある場合、グループを使用することで権限の管理が容易になります。例えば、 という名前のグループIAMAdminsを作成し、そのグループにIAMリソースを管理するアクセス許可を付与できます。

ユーザーは、ロールとは異なります。ユーザーは 1 人の人または 1 つのアプリケーションに一意に関連付けられますが、ロールはそれを必要とする任意の人が引き受けるようになっています。ユーザーには永続的な長期の認証情報がありますが、ロールでは一時認証情報が提供されます。詳細については、「 ユーザーガイド」のIAM「 (ロールの代わりに) ユーザーを作成する場合IAM」を参照してください。

IAM ロール

IAM ロールは、特定のアクセス許可 AWS アカウント を持つ 内のアイデンティティです。これは IAM ユーザーと似ていますが、特定のユーザーに関連付けられていません。IAM ロール を切り替える AWS Management Console ことで、 で ロールを一時的に引き受けることができます。ロールを引き受けるには、 または AWS API オペレーションを AWS CLI 呼び出すか、カスタム を使用しますURL。ロールの使用方法の詳細については、「 ユーザーガイド」のIAM「ロールの使用IAM」を参照してください。

IAM 一時的な認証情報を持つ ロールは、以下の状況で役立ちます。

- フェデレーションユーザーアクセス – フェデレーティッド ID に許可を割り当てるには、ロールを作成してそのロールの許可を定義します。フェデレーティッド ID が認証されると、その ID はロールに関連付けられ、ロールで定義されている許可が付与されます。フェデレーションのロールの詳細については、「 ユーザーガイド」の「サードパーティー ID プロバイダーのロールの作成IAM」を参照してください。IAM Identity Center を使用する場合は、アクセス許可セットを設定します。ID が認証後にアクセスできる内容を制御するために、IAMIdentity Center はアクセス許可セットを のロールに関連付けますIAM。アクセス許可セットの詳細については、「AWS IAM Identity Center ユーザーガイド」の「アクセス許可セット」を参照してください。
- 一時的なIAMユーザーアクセス許可 – IAM ユーザーまたはロールは、 IAMロールを引き受けて、特定のタスクに対して異なるアクセス許可を一時的に引き受けることができます。
- クロスアカウントアクセス – IAMロールを使用して、別のアカウントのユーザー (信頼されたプリンシパル) がアカウントのリソースにアクセスすることを許可できます。クロスアカウントアクセスを許可する主な方法は、ロールを使用することです。ただし、一部の では AWS のサービス、(ロールをプロキシとして使用する代わりに) ポリシーをリソースに直接アタッチできます。クロスアカウントアクセスのロールとリソースベースのポリシーの違いについては、「 ユーザーガイド」の「 でのクロスアカウントリソースアクセスIAMIAM」を参照してください。
- クロスサービスアクセス — 一部の は、他の の機能 AWS のサービス を使用します AWS のサービス。例えば、サービスで呼び出しを行うと、そのサービスが Amazon でアプリケーションを実行EC2したり、Amazon S3 にオブジェクトを保存したりするのが一般的です。サービスでは、呼

び出し元プリンシパルの許可、サービスロール、またはサービスリンクロールを使用してこれを行う場合があります。

- 転送アクセスセッション (FAS) – IAM ユーザーまたはロールを使用して でアクションを実行すると AWS、プリンシパルと見なされます。一部のサービスを使用する際に、アクションを実行することで、別のサービスの別のアクションがトリガーされることがあります。FAS は、 を呼び出すプリンシパルのアクセス許可を AWS のサービス、ダウンストリームサービス AWS のサービス へのリクエストのリクエストと組み合わせて使用します。FAS リクエストは、サービスが他の AWS のサービス またはリソースとのやり取りを完了する必要があるリクエストを受け取った場合にのみ行われます。この場合、両方のアクションを実行するための権限が必要です。FAS リクエストを行う際のポリシーの詳細については、「転送アクセスセッション」を参照してください。
- サービスロール – サービスロールは、ユーザーに代わってアクションを実行するためにサービスが引き受ける IAMロールです。IAM 管理者は、 内からサービスロールを作成、変更、削除できますIAM。詳細については、「 ユーザーガイド」の「 にアクセス許可を委任するロールの作成 AWS のサービスIAM」を参照してください。
- サービスにリンクされたロール – サービスにリンクされたロールは、 にリンクされたサービスロールの一種です AWS のサービス。サービスは、ユーザーに代わってアクションを実行するロールを引き受けることができます。サービスにリンクされたロールは に表示され AWS アカウント 、サービスによって所有されます。IAM 管理者は、サービスにリンクされたロールのアクセス許可を表示できますが、編集することはできません。

- Amazon で実行されているアプリケーション EC2 – IAMロールを使用して、EC2インスタンスで実行され、 AWS CLI または AWS API リクエストを行うアプリケーションの一時的な認証情報を管理できます。これは、EC2インスタンス内にアクセスキーを保存するよりも望ましいです。 AWS ロールをEC2インスタンスに割り当て、そのすべてのアプリケーションで使用できるようにするには、インスタンスにアタッチされたインスタンスプロファイルを作成します。インスタンスプロファイルには ロールが含まれており、EC2インスタンスで実行されているプログラムが一時的な認証情報を取得できるようにします。詳細については、「 ユーザーガイド」の「 IAMロールを使用して Amazon EC2インスタンスで実行されているアプリケーションにアクセス許可を付与するIAM」を参照してください。

IAM ロールとIAMユーザーのどちらを使用するかについては、「 ユーザーガイド」の「 IAMロールを作成するタイミング (ユーザーではなく) IAM」を参照してください。

ポリシーを使用したアクセスの管理

でアクセスを制御する AWS には、ポリシーを作成し、 AWS ID またはリソースにアタッチします。ポリシーは AWS 、アイデンティティまたはリソースに関連付けられているときにアクセス許可を定義する のオブジェクトです。 は、プリンシパル (ユーザー、ルートユーザー、またはロールセッション) がリクエストを行うときに、これらのポリシー AWS を評価します。ポリシーでの権限により、リクエストが許可されるか拒否されるかが決まります。ほとんどのポリシーはJSONドキュメント AWS として に保存されます。JSON ポリシードキュメントの構造と内容の詳細については、「 ユーザーガイド」のJSON「ポリシーの概要IAM」を参照してください。

管理者はポリシーを使用して AWS JSON、誰が何にアクセスできるかを指定できます。つまり、どのプリンシパルがどのリソースに対してどのような条件下でアクションを実行できるかということです。

デフォルトでは、ユーザーやロールに権限はありません。必要なリソースに対してアクションを実行するアクセス許可をユーザーに付与するために、IAM管理者はIAMポリシーを作成できます。その後、管理者はIAMポリシーをロールに追加し、ユーザーはロールを引き受けることができます。

IAM ポリシーは、オペレーションの実行に使用するメソッドに関係なく、アクションのアクセス許可を定義します。例えば、`iam:GetRole`アクションを許可するポリシーがあるとします。そのポリシーを持つユーザーは、 AWS Management Console、、 AWS CLIまたは AWS からロール情報を取得できますAPI。

アイデンティティベースのポリシー

ID ベースのポリシーは、IAMユーザー、ユーザーのグループ、ロールなどの ID にアタッチできるJSONアクセス許可ポリシードキュメントです。これらのポリシーは、ユーザーとロールが実行できるアクション、リソース、および条件をコントロールします。アイデンティティベースのポリシーを作成する方法については、「 ユーザーガイド」のIAM「ポリシーの作成IAM」を参照してください。

アイデンティティベースのポリシーは、さらにインラインポリシーまたはマネージドポリシーに分類できます。インラインポリシーは、単一のユーザー、グループ、またはロールに直接埋め込まれています。管理ポリシーは、 内の複数のユーザー、グループ、ロールにアタッチできるスタンドアロンポリシーです AWS アカウント。管理ポリシーには、 AWS 管理ポリシーとカスタマー管理ポリシーが含まれます。管理ポリシーとインラインポリシーのどちらかを選択する方法については、IAM ユーザーガイドの「管理ポリシーとインラインポリシーの選択」を参照してください。

リソースベースのポリシー

リソースベースのポリシーは、リソースにアタッチするJSONポリシードキュメントです。リソースベースのポリシーの例としては、IAMロールの信頼ポリシー や Amazon S3 バケットポリシー などがあります。リソースベースのポリシーをサポートするサービスでは、サービス管理者はポリシーを使用して特定のリソースへのアクセスをコントロールできます。ポリシーがアタッチされているリソースの場合、指定されたプリンシパルがそのリソースに対して実行できるアクションと条件は、ポリシーによって定義されます。リソースベースのポリシーでは、プリンシパルを指定する必要があります。プリンシパルには、アカウント、ユーザー、ロール、フェデレーティッドユーザー、または を含めることができます AWS のサービス。

リソースベースのポリシーは、そのサービス内にあるインラインポリシーです。リソースベースのポリシーIAMでは、 の AWS 管理ポリシーを使用できません。

アクセスコントロールリスト (ACLs)

アクセスコントロールリスト (ACLs) は、リソースへのアクセス許可を持つプリンシパル (アカウントメンバー、ユーザー、またはロール) を制御します。ACLs はリソースベースのポリシーに似ていますが、JSONポリシードキュメント形式を使用しません。

Amazon S3、 AWS WAF、および Amazon VPCは、 をサポートするサービスの例ですACLs。の詳細についてはACLs、Amazon Simple Storage Service デベロッパーガイドの「アクセスコントロールリスト (ACL) の概要」を参照してください。

その他のポリシータイプ

AWS は、一般的ではない追加のポリシータイプをサポートします。これらのポリシータイプでは、より一般的なポリシータイプで付与された最大の権限を設定できます。

- アクセス許可の境界 – アクセス許可の境界は、アイデンティティベースのポリシーがIAMエンティティ (IAMユーザーまたはロール) に付与できるアクセス許可の上限を設定する高度な機能です。エンティティにアクセス許可の境界を設定できます。結果として得られる権限は、エンティティのアイデンティティベースポリシーとそのアクセス許可の境界の共通部分になります。`Principal` フィールドでユーザーまたはロールを指定するリソースベースのポリシーでは、アクセス許可の境界は制限されません。これらのポリシーのいずれかを明示的に拒否した場合、権限は無効になります。アクセス許可の境界の詳細については、「 IAMユーザーガイド」の「 IAMエンティティのアクセス許可の境界」を参照してください。
- サービスコントロールポリシー (SCPs) – SCPsは、 の組織または組織単位 (OU) に対する最大アクセス許可を指定するJSONポリシーです AWS Organizations。 AWS Organizations は、 AWS ア

カウント ビジネスが所有する複数の をグループ化して一元管理するためのサービスです。組織内のすべての機能を有効にすると、サービスコントロールポリシー (SCPs) をアカウントの一部またはすべてに適用できます。は、各 を含むメンバーアカウントのエンティティのアクセス許可SCP を制限します AWS アカウントのルートユーザー。Organizations と の詳細についてはSCPs、「 AWS Organizations ユーザーガイド」の「サービスコントロールポリシー」を参照してください。

- セッションポリシー - セッションポリシーは、ロールまたはフェデレーションユーザーの一時的なセッションをプログラムで作成する際にパラメータとして渡す高度なポリシーです。結果としてセッションの権限は、ユーザーまたはロールのアイデンティティベースポリシーとセッションポリシーの共通部分になります。また、リソースベースのポリシーから権限が派生する場合もあります。これらのポリシーのいずれかを明示的に拒否した場合、権限は無効になります。詳細については、「 ユーザーガイド」の「セッションポリシーIAM」を参照してください。

複数のポリシータイプ

1 つのリクエストに複数のタイプのポリシーが適用されると、結果として作成される権限を理解するのがさらに難しくなります。複数のポリシータイプが関係する場合にリクエストを許可するかどうか AWS を決定する方法については、「 ユーザーガイド」の「ポリシー評価ロジックIAM」を参照してください。

AWS CodeStar の仕組み IAM

IAM を使用して へのアクセスを管理する前にAWS CodeStar、 で使用できるIAM機能を理解しておく必要がありますAWS CodeStar。AWS CodeStar およびその他の AWS のサービスが と連携する方法の概要を把握するにはIAM、「 IAMユーザーガイド」のAWS 「 と連携IAMする のサービス」を参照してください。

トピック

- AWS CodeStar ID ベースのポリシー
- AWS CodeStar リソースベースのポリシー
- AWS CodeStar タグに基づいた承認
- AWS CodeStar IAM ロール
- IAM へのユーザーアクセス AWS CodeStar
- へのフェデレーティッドユーザーアクセス AWS CodeStar
- での一時的な認証情報の使用 AWS CodeStar
- サービスリンクロール

- サービスロール

AWS CodeStar ID ベースのポリシー

IAM アイデンティティベースのポリシーでは、許可または拒否されたアクションとリソース、およびアクションが許可または拒否される条件を指定できます。 AWS CodeStar は、ユーザーに代わって複数のアイデンティティベースのポリシーを作成します。これにより、 は AWS CodeStar プロジェクトの範囲内 AWS CodeStar でリソースを作成および管理できます。AWS CodeStar は、特定のアクション、リソース、および条件キーをサポートします。JSON ポリシーで使用するすべての要素については、 ユーザーガイドのIAMJSON「 ポリシー要素のリファレンスIAM」を参照してください。

アクション

管理者はポリシーを使用して AWS JSON、誰が何にアクセスできるかを指定できます。つまり、どのプリンシパルがどのリソースに対してどのような条件下でアクションを実行できるかということです。

JSON ポリシーの `Action`要素は、ポリシーでアクセスを許可または拒否するために使用できるアクションを記述します。ポリシーアクションの名前は通常、関連する AWS APIオペレーションと同じです。一致するAPIオペレーションを持たないアクセス許可のみのアクションなど、いくつかの例外があります。また、ポリシーに複数のアクションが必要なオペレーションもあります。これらの追加アクションは、依存アクションと呼ばれます。

このアクションは、関連付けられたオペレーションを実行するための権限を付与するポリシーで使用されます。

のポリシーアクションは、アクションの前にプレフィックス AWS CodeStar を使用しますcodestar:。例えば、指定されたIAMユーザーが AWS CodeStar プロジェクトの説明などのプロジェクトの属性を編集できるようにするには、次のポリシーステートメントを使用できます。

```
{
  "Version": "2012-10-17",
  "Statement" : [
    {
      "Effect" : "Allow",
      "Action" : [
        "codestar:UpdateProject"
      ],
      "Resource" : "arn:aws:codestar:us-east-2:project/my-first-projec"
```

```
    }
  ]
}
```

ポリシーステートメントには、Action または NotAction 要素を含める必要があります。AWS CodeStar は、このサービスで実行できるタスクを記述する独自のアクションのセットを定義します。

単一ステートメントに複数アクションを指定するには、次のようにカンマで区切ります:

```
"Action": [
      "codestar:action1",
      "codestar:action2"
```

ワイルドカード (*) を使用して複数アクションを指定できます。例えば、List という単語で始まるすべてのアクションを指定するには、次のアクションを含めます。

```
"Action": "codestar:List*"
```

AWS CodeStar アクションのリストを確認するには、「 ユーザーガイド」の「 で定義されるアクションAWS CodeStarIAM」を参照してください。

リソース

管理者はポリシーを使用して AWS JSON、誰が何にアクセスできるかを指定できます。つまり、どのプリンシパルがどのリソースに対してどのような条件下でアクションを実行できるかということです。

Policy ResourceJSON要素は、アクションが適用されるオブジェクトを指定します。ステートメントには、Resource または NotResource 要素を含める必要があります。ベストプラクティスとして、Amazon リソースネーム (ARN) を使用してリソースを指定します。これは、リソースレベルの許可と呼ばれる特定のリソースタイプをサポートするアクションに対して実行できます。

オペレーションのリスト化など、リソースレベルの権限をサポートしないアクションの場合は、ステートメントがすべてのリソースに適用されることを示すために、ワイルドカード (*) を使用します。

```
"Resource": "*"
```

AWS CodeStar プロジェクトリソースには次の がありますARN。

```
arn:aws:codestar:region:account:project/resource-specifier
```

の形式の詳細についてはARNs、「Amazon リソースネーム (ARNs) 」および AWS 「サービス名前空間」を参照してください。

例えば、次の例では、 AWS という名前の AWS CodeStar プロジェクトをリージョン 111111111111の AWS アカウント*my-first-projec*に登録しますus-east-2。

```
arn:aws:codestar:us-east-2:111111111111:project/my-first-projec
```

以下に、リージョン の AWS アカウントmy-projに登録された名前で始まる AWS CodeStar プロジェクトを指定します111111111111 AWS us-east-2。

```
arn:aws:codestar:us-east-2:111111111111:project/my-proj*
```

プロジェクトの一覧表示など、一部のAWS CodeStar アクションはリソースで実行できません。このような場合は、ワイルドカード *を使用する必要があります。

```
"LisProjects": "*"
```

AWS CodeStar リソースタイプとその のリストを確認するにはARNs、「 IAMユーザーガイド」の「 で定義されるリソースAWS CodeStar」を参照してください。各リソースARNの を指定できるアクションについては、「 で定義されるアクションAWS CodeStar」を参照してください。

条件キー

AWS CodeStar はサービス固有の条件キーを提供しませんが、一部のグローバル条件キーの使用をサポートしています。すべての AWS グローバル条件キーを確認するには、「 IAMユーザーガイドAWS 」の「グローバル条件コンテキストキー」を参照してください。

例

AWS CodeStar アイデンティティベースのポリシーの例を表示するには、「」を参照してくださいAWS CodeStar ID ベースのポリシーの例。

AWS CodeStar リソースベースのポリシー

AWS CodeStar は、リソースベースのポリシーをサポートしていません。

AWS CodeStar タグに基づいた承認

タグをAWS CodeStar プロジェクトにアタッチしたり、 へのリクエストでタグを渡すことができますAWS CodeStar。タグに基づいてアクセスを管理するには、`codestar:ResourceTag/key-name`、`aws:RequestTag/key-name`、または `aws:TagKeys` の条件キーを使用して、ポリシーの条件要素でタグ情報を提供します。AWS CodeStar リソースのタグ付けの詳細については、「」を参照してくださいthe section called “プロジェクトタグの操作”。

プロジェクトのタグに基づいて AWS CodeStar プロジェクトへのアクセスを制限するためのアイデンティティベースのポリシーの例を表示するには、「」を参照してくださいタグに基づく AWS CodeStar プロジェクトの表示。

AWS CodeStar IAM ロール

IAM ロールは、特定のアクセス許可を持つ AWS アカウント内のエンティティです。

ユーザー 、IAMフェデレーティッドユーザー、ルートユーザー、または引き受けたロール AWS CodeStar として を使用できます。適切なアクセス許可を持つすべてのユーザータイプは、 リソースへのプロジェクトアクセス許可を管理できます AWS が、IAMユーザーのプロジェクトアクセス許可は自動的に AWS CodeStar 管理されます。IAM ポリシーとロールは、プロジェクトロールに基づいて、そのユーザーにアクセス許可とアクセス許可を付与します。IAM コンソールを使用して、 およびその他のアクセス許可をIAMユーザーに割り当てる AWS CodeStar 他のポリシーを作成できます。

たとえば、ユーザーに AWS CodeStar プロジェクトの表示を許可するが、変更は許可しないとします。この場合、ビューワーロールを持つ AWS CodeStar プロジェクトにIAMユーザーを追加します。すべての AWS CodeStar プロジェクトには、プロジェクトへのアクセスを制御するのに役立つ一連のポリシーがあります。さらに、どのユーザーが にアクセスできるかを制御できます AWS CodeStar。

AWS CodeStar アクセスは、IAMユーザーとフェデレーティッドユーザーでは異なる方法で処理されます。チームに追加できるのはIAMユーザーのみです。プロジェクトへのアクセス許可をIAMユーザーに付与するには、プロジェクトチームにユーザーを追加し、ユーザーにロールを割り当てます。フェデレーティッドユーザーにプロジェクトへのアクセス許可を付与するには、 AWS CodeStar プロジェクトロールの 管理ポリシーをフェデレーティッドユーザーのロールに手動でアタッチします。

この表は、各タイプのアクセス権で利用できるツールについて説明しています。

アクセス許可の機能	IAM ユーザー	フェデレーティッドユーザー	ルートユーザー
SSH Amazon EC2および Elastic Beanstalk プロジェクトのリモートアクセス用の キー管理	✓		
AWS CodeCommit SSH アクセス	✓		
IAM によって管理されるユーザーアクセス許可 AWS CodeStar	✓		
手動で管理されるプロジェクトのアクセス許可		✓	✓
ユーザーはチームメンバーとしてプロジェクトに追加することができます	✓		

IAM へのユーザーアクセス AWS CodeStar

プロジェクトにIAMユーザーを追加し、そのユーザーのロールを選択すると、 AWS CodeStar は適切なポリシーを自動的にIAMユーザーに適用します。IAM ユーザーの場合、 でポリシーやアクセス許可を直接アタッチまたは管理する必要はありませんIAM。 AWS CodeStar プロジェクトにIAMユーザーを追加する方法については、「」を参照してくださいAWS CodeStar プロジェクトにチームメンバーを追加する 。 AWS CodeStar プロジェクトからIAMユーザーを削除する方法については、「」を参照してくださいAWS CodeStar プロジェクトからチームメンバーを削除する 。

インラインポリシーをIAMユーザーにアタッチする

プロジェクトにユーザーを追加すると、 はユーザーのロールに一致するプロジェクトの 管理ポリシー AWS CodeStar を自動的にアタッチします。プロジェクトの AWS CodeStar 管理ポリシーを IAM ユーザーに手動でアタッチしないでください。を除き`AWSCodeStarFullAccess`、 AWS CodeStar プロジェクトのIAMユーザーのアクセス許可を変更するポリシーをアタッチすることはお勧めしません。独自のポリシーを作成してアタッチする場合は、「 IAMユーザーガイド」のIAM「IDアクセス許可の追加と削除」を参照してください。

へのフェデレーティッドユーザーアクセス AWS CodeStar

IAM ユーザーを作成するか、ルートユーザーを使用する代わりに、、エンタープライズユーザーディレクトリ AWS Directory Service、ウェブ ID プロバイダー、またはロールを引き受けるIAMユーザーからユーザー ID を使用できます。このようなユーザーはフェデレーションユーザーと呼ばれます。

AWS CodeStar プロジェクトAWS CodeStar レベルのポリシーとアクセス許可で説明されている管理ポリシーをユーザーのロールに手動でアタッチして、フェデレーティッドユーザーにプロジェクトへのアクセスを許可します。 IAMがプロジェクトリソースとIAMロール AWS CodeStar を作成した後、所有者、寄稿者、またはビューワーポリシーをアタッチします。

前提条件:

- ID プロバイダーを設定する必要があります。例えば、SAMLID プロバイダーをセットアップし、プロバイダーを介して AWS 認証を設定できます。ID プロバイダーの設定の詳細については、IAM「ID プロバイダーの作成」を参照してください。SAML フェデレーションの詳細については、「2.0 SAML ベースのフェデレーションについて」を参照してください。
- ID プロバイダーを通じてアクセスをリクエストする際に引き受けるフェデレーティッドユーザーのロールを作成済みである必要があります。フェデレーティッドユーザーがロールを引き受けることを許可するSTS信頼ポリシーをロールにアタッチする必要があります。詳細については、「 IAM ユーザーガイド」の「フェデレーティッドユーザーとロール」を参照してください。
- AWS CodeStar プロジェクトを作成し、プロジェクト ID を知っている必要があります。

ID プロバイダーのロールの作成方法については、「サードパーティーの ID プロバイダー (フェデレーション) 用のロールの作成」を参照してください。

フェデレーティッドユーザーロールに AWSCodeStarFullAccess 管理ポリシーをアタッチする

プロジェクトを作成するためのアクセス許可をフェデレーティッドユーザーに付与するには、`AWSCodeStarFullAccess` 管理ポリシーを添付します。これらのステップを実行するには、ルートユーザー、アカウントの管理者ユーザー、または関連する`AdministratorAccess`管理ポリシーまたは同等のものを持つIAMユーザーまたはフェデレーティッドユーザーとしてコンソールにサインインしている必要があります。

> **Note**
>
> プロジェクト作成後、プロジェクト所有者のアクセス許可は自動的に適用されません。「プロジェクトの AWS CodeStar ビューワー/コントリビューター/所有者管理ポリシーをフェデレーティッドユーザーロールにアタッチする」に示されているように、アカウントの管理者のアクセス許可を持つロールを使用して、所有者の管理ポリシーをアタッチします。

1. IAM コンソールを開きます。ナビゲーションペインで、[Policies] (ポリシー) を選択します。
2. 検索フィールドに「`AWSCodeStarFullAccess`」と入力します。ポリシータイプが [AWS managed] (マネージド) のポリシー名が表示されます。このポリシーを展開して、ポリシーステートメントのアクセス許可を参照することができます。
3. ポリシーの横にある円形を選択して、[Policy Actions] (ポリシーアクション) の [Attach] (添付) を選択します。
4. [Summary] (概要) ページで、 [Attached entities] (添付されたエンティティ) タブを選択します。[Attach] (添付) を選択します。
5. [Attach Policy] (ポリシーの添付) ページの検索フィールドで、フェデレーティッドユーザーのロールをフィルタリングします。ロールの名前の横にあるボックスを選択し、[Attach policy] (ポリシーの添付) を選択します。[Attached entities] (添付されたエンティティ) タブに、新しいアタッチメントが表示されます。

プロジェクトの AWS CodeStar ビューワー/コントリビューター/所有者管理ポリシーをフェデレーティッドユーザーロールにアタッチする

プロジェクトへのアクセス権をフェデレーティッドユーザーに付与するには、適切な所有者、寄稿者、閲覧者の管理ポリシーをユーザーのロールに添付します。管理ポリシーによって、適切なアクセス許可のレベルが指定されます。IAM ユーザーとは異なり、フェデレーティッドユーザーの 管理ポリシーを手動でアタッチおよびデタッチする必要があります。これは、 のチームメンバーにプロジェクト許可を割り当てるのと同じです AWS CodeStar。これらのステップを実行するには、ルートユーザー、アカウントの管理者ユーザー、または関連する`AdministratorAccess`管理ポリシーまたは同等のものを持つIAMユーザーまたはフェデレーティッドユーザーとしてコンソールにサインインしている必要があります。

前提条件:

- フェデレーティッドユーザーが引き受けるロールを作成しているか、既存のロールを設定されている必要があります。

- 付与するアクセス許可のレベルを把握しておく必要があります。所有者、寄稿者、閲覧者のロールに添付されている管理ポリシーは、プロジェクトのロールベースのアクセス許可を提供します。
- AWS CodeStar プロジェクトが作成されている必要があります。管理ポリシーは、プロジェクトが作成されIAMるまで では使用できません。

1. IAM コンソールを開きます。ナビゲーションペインで、[Policies] (ポリシー) を選択します。
2. 検索フィールドにプロジェクト ID を入力します。ポリシータイプが [Customer managed] (カスタマー管理) で、プロジェクトに一致するポリシー名が表示されます。このポリシーを展開して、ポリシーステートメントのアクセス許可を参照することができます。
3. これらのうち、いずれかの管理ポリシーを選択します。ポリシーの横にある円形を選択して、[Policy Actions] (ポリシーアクション) の [Attach] (添付) を選択します。
4. [Summary] (概要) ページで、 [Attached entities] (添付されたエンティティ) タブを選択します。[Attach] (添付) を選択します。
5. [Attach Policy] (ポリシーの添付) ページの検索フィールドで、フェデレーティッドユーザーのロールをフィルタリングします。ロールの名前の横にあるボックスを選択し、[Attach policy] (ポリシーの添付) を選択します。[Attached entities] (添付されたエンティティ) タブに、新しいアタッチメントが表示されます。

フェデレーティッドユーザーロールから AWS CodeStar マネージドポリシーをデタッチする

AWS CodeStar プロジェクトを削除する前に、フェデレーティッドユーザーのロールにアタッチした管理ポリシーを手動でデタッチする必要があります。これらのステップを実行するには、ルートユーザー、アカウントの管理者ユーザー、または関連するAdministratorAccess管理ポリシーまたは同等のものを持つIAMユーザーまたはフェデレーティッドユーザーとしてコンソールにサインインしている必要があります。

1. IAM コンソールを開きます。ナビゲーションペインで、[Policies] (ポリシー) を選択します。
2. 検索フィールドにプロジェクト ID を入力します。
3. ポリシーの横にある円形を選択して、[Policy Actions] (ポリシーアクション) の [Attach] (添付) を選択します。
4. [Summary] (概要) ページで、 [Attached entities] (添付されたエンティティ) タブを選択します。
5. 検索フィールドで、フェデレーティッドユーザーのロールをフィルタリングします。[Detach] (デタッチ) を選択します。

フェデレーティッドユーザーロールに AWS Cloud9 管理ポリシーをアタッチする

AWS Cloud9 開発環境を使用している場合は、 AWSCloud9User管理ポリシーをユーザーのロールにアタッチして、フェデレーティッドユーザーにその環境へのアクセスを許可します。IAM ユーザーとは異なり、フェデレーティッドユーザーの管理ポリシーを手動でアタッチおよびデタッチする必要があります。これらのステップを実行するには、ルートユーザー、アカウントの管理者ユーザー、または関連する AdministratorAccess管理ポリシーまたは同等のポリシーを持つIAMユーザーまたはフェデレーティッドユーザーとしてコンソールにサインインしている必要があります。

前提条件:

- フェデレーティッドユーザーが引き受けるロールを作成しているか、既存のロールを設定されている必要があります。
- 付与するアクセス許可のレベルを把握しておく必要があります。
  - AWSCloud9User 管理ポリシーでは、ユーザーによる次の操作を許可します :
    - 独自の AWS Cloud9 開発環境を作成します。
    - 環境に関する情報を取得する。
    - 環境の設定を変更する。
  - AWSCloud9Administrator 管理ポリシーでは、自分自身または他のユーザーに対する次の操作をユーザーに許可します :
    - 環境を作成する。
    - 環境に関する情報を取得する。
    - 環境を削除する。
    - 環境の設定を変更する。

1. IAM コンソールを開きます。ナビゲーションペインで、[Policies] (ポリシー) を選択します。
2. 検索フィールドにポリシー名を入力します。ポリシータイプが AWS マネージドの管理ポリシーが表示されます。このポリシーを展開して、ポリシーステートメントのアクセス許可を参照することができます。
3. これらのうち、いずれかの管理ポリシーを選択します。ポリシーの横にある円形を選択して、[Policy Actions] (ポリシーアクション) の [Attach] (添付) を選択します。
4. [Summary] (概要) ページで、 [Attached entities] (添付されたエンティティ) タブを選択します。[Attach] (添付) を選択します。
5. [Attach Policy] (ポリシーの添付) ページの検索フィールドで、フェデレーティッドユーザーのロールをフィルタリングします。ロールの名前の横にあるボックスを選択し、[Attach Policy] (ポ

リシーの添付) を選択します。[Attached entities] (添付されたエンティティ) タブに、新しいアタッチメントが表示されます。

フェデレーティッドユーザーロールから AWS Cloud9 マネージドポリシーをデタッチする

AWS Cloud9 開発環境を使用している場合は、アクセスを許可するポリシーをデタッチすることで、フェデレーティッドユーザーのアクセスを削除できます。これらのステップを実行するには、ルートユーザー、アカウントの管理者ユーザー、または関連するAdministratorAccess管理ポリシーまたは同等のものを持つIAMユーザーまたはフェデレーティッドユーザーとしてコンソールにサインインしている必要があります。

1. IAM コンソールを開きます。ナビゲーションペインで、[Policies] (ポリシー) を選択します。
2. 検索フィールドにプロジェクト名を入力します。
3. ポリシーの横にある円形を選択して、[Policy Actions] (ポリシーアクション) の [Attach] (添付) を選択します。
4. [Summary] (概要) ページで、 [Attached entities] (添付されたエンティティ) タブを選択します。
5. 検索フィールドで、フェデレーティッドユーザーのロールをフィルタリングします。[Detach] (デタッチ) を選択します。

での一時的な認証情報の使用 AWS CodeStar

一時的な認証情報を使用して、フェデレーションでサインインしたり、 IAMロールを引き受けたり、クロスアカウントロールを引き受けたりすることができます。一時的なセキュリティ認証情報を取得するには、 AssumeRoleや などのオペレーションを呼び出し AWS STS APIますGetFederationToken。

AWS CodeStar は一時的な認証情報の使用をサポートしますが、 AWS CodeStar チームメンバーの機能はフェデレーティッドアクセスでは機能しません。 AWS CodeStar チームメンバー機能は、チームメンバーとしてのIAMユーザーの追加のみをサポートします。

サービスリンクロール

サービスにリンクされたロールを使用すると、 AWS サービスは他の サービスのリソースにアクセスして、ユーザーに代わってアクションを実行できます。サービスにリンクされたロールはIAMアカウントに表示され、サービスによって所有されます。 管理者は、サービスにリンクされたロールのアクセス許可を表示できますが、編集することはできません。

AWS CodeStar は、サービスにリンクされたロールをサポートしていません。

サービスロール

この機能により、ユーザーに代わってサービスがサービスロールを引き受けることが許可されます。このロールにより、サービスがお客様に代わって他のサービスのリソースにアクセスし、アクションを完了することが許可されます。サービスロールはIAMアカウントに表示され、アカウントによって所有されます。つまり、 管理者は、このロールのアクセス許可を変更できます。ただし、それにより、サービスの機能が損なわれる場合があります。

AWS CodeStar は、サービスロールをサポートします。 は、プロジェクトのリソースを作成および管理するときに aws-codestar-service-role、サービスロール AWS CodeStar を使用します。詳細については、「 ユーザーガイド」の「ロールの用語と概念」を参照してください。 IAM

> **Important**
>
> このサービスロールを作成するには、 管理ユーザーまたはルートアカウントとしてサインインする必要があります。詳細については、 IAMユーザーガイドの「初回アクセスのみ: ルートユーザーの認証情報」および「最初の管理者ユーザーとグループの作成」を参照してください。

このロールは、 で初めてプロジェクトを作成するときに自動的に作成されます AWS CodeStar。サービスロールは、ユーザーに代わって以下のことを行います :

- プロジェクト作成時に選択したリソースを作成する。
- これらのリソースに関する情報を AWS CodeStar プロジェクトダッシュボードに表示します。

また、プロジェクトのリソースを管理するときにも、ユーザーの代わりに機能します。このポリシーステートメントの例については、「 AWSCodeStarServiceRole ポリシー」を参照してください。

さらに、 はプロジェクトタイプに応じて、プロジェクト固有のサービスロールを複数 AWS CodeStar 作成 AWS CloudFormation します。また、ツールチェーンロールはプロジェクトタイプごとに作成されます。

- AWS CloudFormation ロールを使用すると AWS CodeStar 、 AWS CloudFormation にアクセスして AWS CodeStar 、プロジェクトのスタックを作成および変更できます。
- ツールチェーンロールを使用すると AWS CodeStar 、 は他の AWS のサービスにアクセスして、AWS CodeStar プロジェクトのリソースを作成および変更できます。

AWS CodeStar プロジェクトレベルのポリシーとアクセス許可

プロジェクトを作成すると、 はプロジェクトリソースの管理に必要なIAMロールとポリシー AWS CodeStar を作成します。ポリシーは 3 つのカテゴリに分類されます：

- IAM プロジェクトチームメンバー向けの ポリシー。
- IAM ワーカーロールの ポリシー。
- IAM ランタイム実行ロールの ポリシー。

IAM チームメンバーのポリシー

プロジェクトを作成すると、 は、プロジェクトへの所有者、寄稿者、閲覧者アクセス用に 3 つのカスタマー管理ポリシー AWS CodeStar を作成します。すべての AWS CodeStar プロジェクトには、これら 3 つのアクセスレベルのIAMポリシーが含まれています。これらのアクセスレベルはプロジェクト固有であり、標準名を持つ IAMマネージドポリシーで定義されます。ここで、*project-id* は AWS CodeStar プロジェクトの ID (例：*my-first-projec*):

- `CodeStar_`*project-id*`_Owner`
- `CodeStar_`*project-id*`_Contributor`
- `CodeStar_`*project-id*`_Viewer`

> **Important**
>
> これらのポリシーは、 によって変更される可能性があります AWS CodeStar。手動では編集しないでください。アクセス許可を追加または変更する場合は、IAMユーザーに追加のポリシーをアタッチします。

チームメンバー (IAM ユーザー) をプロジェクトに追加してアクセスレベルを選択すると、対応するポリシーがIAMユーザーにアタッチされ、プロジェクトリソースを操作するための適切なアクセス許可のセットがユーザーに付与されます。ほとんどの場合、 でポリシーやアクセス許可を直接アタッチまたは管理する必要はありませんIAM。 AWS CodeStar アクセスレベルポリシーをIAMユーザーに手動でアタッチすることはお勧めしません。どうしても必要な場合は、 AWS CodeStar アクセスレベルポリシーの補足として、独自の管理ポリシーまたはインラインポリシーを作成して、独自のレベルのアクセス許可をIAMユーザーに適用できます。

ポリシーの対象は、厳密にプロジェクトのリソースと特定のアクションになります。インフラストラクチャスタックに新しいリソースが追加されると、サポートされているリソースタイプの 1 AWS CodeStar つである場合、 は、新しいリソースへのアクセス許可を含めるようにチームメンバーポリシーを更新しようとします。

> **Note**
>
> AWS CodeStar プロジェクトのアクセスレベルのポリシーは、そのプロジェクトにのみ適用されます。これにより、ユーザーは自分のロールによって決定されるレベルで、アクセス許可を持つ AWS CodeStar プロジェクトのみを表示して操作できます。 AWS CodeStar プロジェクトを作成するユーザーのみが、プロジェクトに関係なく、すべての AWS CodeStar リソースへのアクセスを許可するポリシーを適用する必要があります。

アクセスレベルポリシーはすべて、 AWS CodeStar アクセスレベルが関連付けられているプロジェクトに関連付けられた AWS リソースによって異なります。他の AWS サービスとは異なり、これらのポリシーは、プロジェクトのリソースの変更に応じてプロジェクトが作成および更新されたときにカスタマイズされます。したがって、正規の所有者、寄稿者、閲覧者の管理ポリシーはありません。

AWS CodeStar 所有者ロールポリシー

`CodeStar_project-id_Owner` カスタマー管理ポリシーでは、ユーザーは制限なしで AWS CodeStar プロジェクト内のすべてのアクションを実行できます。これは、ユーザーがチームメンバーを追加または削除することを許可する唯一のポリシーです。ポリシーの内容は、プロジェクトに関連付けられたリソースによって異なります。例については、「AWS CodeStar 所有者ロールポリシー」を参照してください。

このポリシーを持つIAMユーザーは、プロジェクト内のすべての AWS CodeStar アクションを実行できますが、`AWSCodeStarFullAccess`ポリシーを持つIAMユーザーとは異なり、ユーザーはプロジェクトを作成できません。アクセス`codestar:*`許可の範囲は、特定のリソース (その AWS CodeStar プロジェクト ID に関連付けられたプロジェクト) に限定されます。

AWS CodeStar 寄稿者ロールポリシー

`CodeStar_project-id_Contributor` カスタマー管理ポリシーは、プロジェクトに投稿してプロジェクトダッシュボードを変更することをユーザーに許可しますが、チームメンバーを追加または削除することは許可しません。ポリシーの内容は、プロジェクトに関連付けられたリソースによって異なります。例については、「AWS CodeStar 寄稿者のロールポリシー」を参照してください。

AWS CodeStar ビューワーロールポリシー

CodeStar_*project-id*_Viewer カスタマー管理ポリシーは、 AWS CodeStar内のプロジェクトを表示することをユーザーに許可しますが、リソースを変更したり、チームメンバーを追加または削除することは許可しません。ポリシーの内容は、プロジェクトに関連付けられたリソースによって異なります。例については、「AWS CodeStar ビューワーロールポリシー 」を参照してください。

IAM ワーカーロールのポリシー

2018 年 12 月 6 日以降に AWS CodeStar プロジェクトを作成すると、 AWS CodeStar は PDT2 つのワーカーロール CodeStar-*project-id*-ToolChainと を作成しますCodeStar-*project-id*-CloudFormation。ワーカーロールは、 がサービスに渡すために AWS CodeStar 作成するプロジェクト固有のIAMロールです。これにより、サービスがリソースを作成し、 AWS CodeStar プロジェクトのコンテキストでアクションを実行できるようにアクセス許可が付与されます。ツールチェーンワーカーロールには、、 CodeBuild CodeDeploy、 などのツールチェーンサービスとの信頼関係が確立されています CodePipeline。プロジェクトのチームメンバー (所有者と寄稿者) には、信頼されたダウンストリームサービスにワーカーロールを渡すためのアクセス権が付与されます。このロールのインラインポリシーステートメントの例については、「AWS CodeStar ツールチェーンワーカーロールポリシー (2018 年 12 月 6 日以降PDT) 」を参照してください。

CloudFormation ワーカーロールには AWS CloudFormation、 でサポートされている選択したリソースのアクセス許可と、アプリケーションスタックにIAMユーザー、ロール、ポリシーを作成するアクセス許可が含まれます。また、 との信頼関係も確立されています AWS CloudFormation。権限のエスカレーションと破壊的アクションのリスクを軽減するために、 AWS CloudFormation ロールポリシーには、インフラストラクチャスタックで作成されたすべてのIAMエンティティ (ユーザーまたはロール) にプロジェクト固有のアクセス許可の境界を要求する条件が含まれています。このロールのインラインポリシーステートメントの例については、「AWS CloudFormation ワーカーロールポリシー」を参照してください。

2018 年 12 月 6 日より前に作成されたAWS CodeStar プロジェクトでは、 CodePipeline、CodeBuild、 CloudWatch イベントなどのツールチェーンリソースの個々のワーカーロールPDT AWS CodeStar を作成し、限られたリソースセット AWS CloudFormation をサポートする のワーカーロールも作成します。これらの各ロールには、対応するサービスとの間で確立された信頼関係があります。プロジェクトのチームメンバー (所有者、寄稿者) および他の一部のワーカーロールには、信頼されたダウンストリームサービスにロールを渡すためのアクセス権が付与されます。ワーカーロールのアクセス許可は、一連のプロジェクトリソースでロールが実行できる基本的なアクションのセットまで制限されたインラインポリシーで定義されます。これらのアクセス権限は静的です。作成時にプロジェクトに含まれるリソースへのアクセス許可が含まれますが、プロジェクトに新しい

リソースが追加されるときに更新されません。これらのポリシーステートメントの例については、以下を参照してください：

- AWS CloudFormation ワーカーロールポリシー (2018 年 12 月 6 日より前PDT)
- AWS CodePipeline ワーカーロールポリシー (2018 年 12 月 6 日より前PDT)
- AWS CodeBuild ワーカーロールポリシー (2018 年 12 月 6 日より前PDT)
- Amazon CloudWatch Events ワーカーロールポリシー (2018 年 12 月 6 日より前PDT)

IAM 実行ロールのポリシー

2018 年 12 月 6 日以降に作成されたプロジェクトの場合、 PDTAWS CodeStar はアプリケーションスタックにサンプルプロジェクトの汎用実行ロールを作成します。このロールは、アクセス許可の境界ポリシーを使用してプロジェクトリソースに制限されます。サンプルプロジェクトを拡張すると、追加のIAMロールを作成できます。 AWS CloudFormation ロールポリシーでは、権限のエスカレーションを避けるために、アクセス許可の境界を使用してこれらのロールの範囲を絞り込む必要があります。詳細については、「IAM ロールをプロジェクトに追加する」を参照してください。

2018 年 12 月 6 日より前に作成された Lambda プロジェクトの場合、 はPDT、プロジェクト AWS SAM スタック内のリソースを操作するアクセス許可を持つインラインポリシーがアタッチされた Lambda 実行ロール AWS CodeStar を作成します。新しいリソースがSAMテンプレートに追加されると、サポートされているリソースタイプの 1 つである場合、 は Lambda 実行ロールポリシーを更新 AWS CodeStar して新しいリソースへのアクセス許可を含めます。

IAM アクセス権限の境界

2018 年 12 月 6 日以降、プロジェクトを作成するとPDT、カスタマー管理ポリシーAWS CodeStar が作成され、そのポリシーがプロジェクトのIAMロールにアクセスIAM許可の境界として割り当てられます。AWS CodeStar では、アプリケーションスタックで作成されたすべてのIAMエンティティに許可の境界が必要です。アクセス許可の境界により、ロールが持つことができる最大アクセス許可が管理されますが、アクセス許可を持つロールは提供されません。アクセス許可ポリシーにより、ロールのアクセス許可が定義されます。つまり、どれだけ多くのアクセス許可がロールに追加されても、そのロールを使用するすべてのユーザーは、アクセス許可の境界に含まれている以上のアクションを実行することはできません。アクセス許可ポリシーとアクセス許可の境界がどのように評価されるかについては、「 ユーザーガイド」の「ポリシー評価ロジックIAM」を参照してください。

AWS CodeStar は、プロジェクト固有のアクセス許可の境界を使用して、プロジェクト外のリソースへの権限のエスカレーションを防ぎます。アクセスAWS CodeStar 許可の境界には、プロジェク

トリソースARNsの が含まれます。このポリシーステートメントの例については、「AWS CodeStar アクセス許可の境界ポリシー」を参照してください。

AWS CodeStar 変換は、サポートされているリソースをアプリケーションスタック () を介してプロジェクトに追加または削除するときに、このポリシーを更新しますtemplate.yml。

既存のプロジェクトにアクセスIAM許可の境界を追加する

2018 年 12 月 6 日より前にAWS CodeStar 作成されたプロジェクトPDTがある場合は、プロジェクト内のIAMロールに手動でアクセス許可の境界を追加する必要があります。ベストプラクティスとして、プロジェクトのリソースのみを含むプロジェクト固有の境界を使用し、プロジェクト外部のリソースへの特権のエスカレーションを回避することをお勧めします。プロジェクトの進化に合わせて更新されるAWS CodeStar 管理アクセス許可の境界を使用するには、次の手順に従います。

1. AWS CloudFormation コンソールにサインインし、プロジェクト内のツールチェーンスタックのテンプレートを見つけます。このテンプレートは awscodestar-*project-id* という名前です。
2. このテンプレートを選択し、[Actions] (アクション) を選択して、[View/Edit template in Designer] (デザイナーでテンプレートを表示/編集) を選択します。
3. [Resources] セクションを見つけ、セクションの上部にある次のスニペットを含めます。

```
  PermissionsBoundaryPolicy:
    Description: Creating an IAM managed policy for defining the permissions boundary
 for an AWS CodeStar project
    Type: AWS::IAM::ManagedPolicy
    Properties:
      ManagedPolicyName: !Sub 'CodeStar_${ProjectId }_PermissionsBoundary'
      Description: 'IAM policy to define the permissions boundary for IAM entities
 created in an AWS CodeStar project'
      PolicyDocument:
        Version: '2012-10-17'
        Statement:
        - Sid: '1'
          Effect: Allow
          Action: ['*']
          Resource:
            - !Sub 'arn:${AWS::Partition}:cloudformation:${AWS::Region}:
${AWS::AccountId}:stack/awscodestar-${ProjectId}-*'
```

AWS CloudFormation コンソールからスタックを更新するには、追加のIAMアクセス許可が必要になる場合があります。

4. (オプション) アプリケーション固有のIAMロールを作成する場合は、このステップを完了します。IAM コンソールから、プロジェクトの AWS CloudFormation ロールにアタッチされているインラインポリシーを更新して、次のスニペットを含めます。ポリシーを更新するには、追加のIAMリソースが必要になる場合があります。

```
{
     "Action": [
         "iam:PassRole"
     ],
     "Resource": "arn:aws:iam::{AccountId}:role/CodeStar-{ProjectId}*",
     "Effect": "Allow"
 },
 {
     "Action": [
         "iam:CreateServiceLinkedRole",
         "iam:GetRole",
         "iam:DeleteRole",
         "iam:DeleteUser"
     ],
     "Resource": "*",
     "Effect": "Allow"
 },
 {
     "Action": [
         "iam:AttachRolePolicy",
         "iam:AttachUserPolicy",
         "iam:CreateRole",
         "iam:CreateUser",
         "iam:DeleteRolePolicy",
         "iam:DeleteUserPolicy",
         "iam:DetachUserPolicy",
         "iam:DetachRolePolicy",
         "iam:PutUserPermissionsBoundary",
         "iam:PutRolePermissionsBoundary"
     ],
     "Resource": "*",
     "Condition": {
         "StringEquals": {
```

```
                    "iam:PermissionsBoundary": "arn:aws:iam::{AccountId}:policy/
CodeStar_{ProjectId}_PermissionsBoundary"
                }
            },
            "Effect": "Allow"
        }
```

5. が適切なアクセス許可でアクセス許可の境界AWS CodeStar を更新するように、プロジェクトパイプラインを通じて変更をプッシュします。

詳細については、「IAM ロールをプロジェクトに追加する」を参照してください。

AWS CodeStar ID ベースのポリシーの例

デフォルトでは、IAMユーザーとロールにはAWS CodeStar リソースを作成または変更するアクセス許可はありません。また、、 AWS Management Console AWS CLI、または を使用してタスクを実行することはできません AWS API。管理者は、必要な特定のリソースに対して特定のAPIオペレーションを実行するアクセス許可をユーザーとロールに付与するIAMポリシーを作成する必要があります。その後、管理者は、これらのアクセス許可を必要とするIAMユーザーまたはグループにこれらのポリシーをアタッチする必要があります。

これらのポリシードキュメント例を使用してIAMアイデンティティベースのJSONポリシーを作成する方法については、 ユーザーガイドのJSON「 タブでのポリシーの作成IAM」を参照してください。

トピック

- ポリシーのベストプラクティス
- AWSCodeStarServiceRole ポリシー
- AWSCodeStarFullAccess ポリシー
- AWS CodeStar 所有者ロールポリシー
- AWS CodeStar 寄稿者のロールポリシー
- AWS CodeStar ビューワーロールポリシー
- AWS CodeStar ツールチェーンワーカーロールポリシー (2018 年 12 月 6 日以降PDT)
- AWS CloudFormation ワーカーロールポリシー
- AWS CloudFormation ワーカーロールポリシー (2018 年 12 月 6 日より前PDT)
- AWS CodePipeline ワーカーロールポリシー (2018 年 12 月 6 日より前PDT)

- AWS CodeBuild ワーカーロールポリシー (2018 年 12 月 6 日より前PDT)
- Amazon CloudWatch Events ワーカーロールポリシー (2018 年 12 月 6 日より前PDT)
- AWS CodeStar アクセス許可の境界ポリシー
- プロジェクトのリソースの一覧表示
- AWS CodeStar コンソールの使用
- ユーザーが自分のアクセス許可を表示できるようにする
- AWS CodeStar プロジェクトの更新
- プロジェクトへのチームメンバーの追加
- AWS アカウントに関連付けられたユーザープロファイルの一覧表示
- タグに基づく AWS CodeStar プロジェクトの表示
- AWS CodeStarAWS 管理ポリシーの更新

ポリシーのベストプラクティス

ID ベースのポリシーは、ユーザーのアカウントで誰かがAWS CodeStar リソースを作成、アクセス、または削除できるどうかを決定します。これらのアクションを実行すると、 AWS アカウントに料金が発生する可能性があります。アイデンティティベースポリシーを作成したり編集したりする際には、以下のガイドラインと推奨事項に従ってください:

- AWS 管理ポリシーを開始し、最小特権のアクセス許可に移行する – ユーザーとワークロードにアクセス許可を付与するには、多くの一般的なユースケースにアクセス許可を付与する AWS 管理ポリシーを使用します。これらは で使用できます AWS アカウント。ユースケースに固有の AWS カスタマー管理ポリシーを定義して、アクセス許可をさらに減らすことをお勧めします。詳細については、「 ユーザーガイド」の「 AWS 管理ポリシーAWS 」または「 ジョブ機能の 管理ポリシーIAM」を参照してください。
- 最小特権のアクセス許可を適用する – IAMポリシーでアクセス許可を設定する場合は、タスクの実行に必要なアクセス許可のみを付与します。これを行うには、特定の条件下で特定のリソースに対して実行できるアクションを定義します。これは、最小特権アクセス許可とも呼ばれています。IAM を使用してアクセス許可を適用する方法の詳細については、「 ユーザーガイド」の「 のポリシーとアクセス許可IAMIAM」を参照してください。
- IAM ポリシーの条件を使用してアクセスをさらに制限する – ポリシーに条件を追加して、アクションとリソースへのアクセスを制限できます。例えば、ポリシー条件を記述して、すべてのリクエストを を使用して送信する必要があることを指定できますSSL。条件を使用して、 などの特定の を介してサービスアクションが使用される場合に AWS のサービス、サービスアクションへ

のアクセスを許可することもできます AWS CloudFormation。詳細については、「 ユーザーガイド」のIAMJSON「ポリシー要素: 条件IAM」を参照してください。

- IAM Access Analyzer を使用してIAMポリシーを検証し、安全で機能的なアクセス許可を確保する – IAM Access Analyzer は、ポリシーがポリシー言語 (JSON) とIAMベストプラクティスに準拠するように、新規および既存のIAMポリシーを検証します。IAM Access Analyzer には、安全で機能的なポリシーの作成に役立つ 100 を超えるポリシーチェックと実用的な推奨事項が用意されています。詳細については、「 ユーザーガイド」のIAM「Access Analyzer ポリシーの検証IAM」を参照してください。
- 多要素認証を要求する (MFA) – でIAMユーザーまたはルートユーザーを必要とするシナリオがある場合は AWS アカウント、セキュリティを強化MFAするために をオンにします。API オペレーションが呼び出されるMFAタイミングを要求するには、ポリシーにMFA条件を追加します。詳細については、「 IAMユーザーガイド」のMFA「 で保護されたAPIアクセスの設定」を参照してください。

のベストプラクティスの詳細についてはIAM、「 ユーザーガイド」の「 のセキュリティのベストプラクティスIAMIAM」を参照してください。

AWSCodeStarServiceRole ポリシー

`aws-codestar-service-role` ポリシーは、 が他の サービスでアクションを実行できるようにするサービスロール AWS CodeStar にアタッチされます。に初めてサインインするときは AWS CodeStar、サービスロールを作成します。一度だけ作成する必要があります。ポリシーは、作成後にサービスロールに自動的にアタッチされます。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ProjectEventRules",
            "Effect": "Allow",
            "Action": [
                "events:PutTargets",
                "events:RemoveTargets",
                "events:PutRule",
                "events:DeleteRule",
                "events:DescribeRule"
            ],
            "Resource": [
                "arn:aws:events:*:*:rule/awscodestar-*"
```

```
            ]
        },
        {
            "Sid": "ProjectStack",
            "Effect": "Allow",
            "Action": [
                "cloudformation:*Stack*",
                "cloudformation:CreateChangeSet",
                "cloudformation:ExecuteChangeSet",
                "cloudformation:DeleteChangeSet",
                "cloudformation:GetTemplate"
            ],
            "Resource": [
                "arn:aws:cloudformation:*:*:stack/awscodestar-*",
                "arn:aws:cloudformation:*:*:stack/awseb-*",
                "arn:aws:cloudformation:*:*:stack/aws-cloud9-*",
                "arn:aws:cloudformation:*:aws:transform/CodeStar*"
            ]
        },
        {
            "Sid": "ProjectStackTemplate",
            "Effect": "Allow",
            "Action": [
                "cloudformation:GetTemplateSummary",
                "cloudformation:DescribeChangeSet"
            ],
            "Resource": "*"
        },
        {
            "Sid": "ProjectQuickstarts",
            "Effect": "Allow",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::awscodestar-*/*"
            ]
        },
        {
            "Sid": "ProjectS3Buckets",
            "Effect": "Allow",
            "Action": [
                "s3:*"
            ],
```

```
            "Resource": [
                "arn:aws:s3:::aws-codestar-*",
                "arn:aws:s3:::elasticbeanstalk-*"
            ]
        },
        {
            "Sid": "ProjectServices",
            "Effect": "Allow",
            "Action": [
                "codestar:*",
                "codecommit:*",
                "codepipeline:*",
                "codedeploy:*",
                "codebuild:*",
                "autoscaling:*",
                "cloudwatch:Put*",
                "ec2:*",
                "elasticbeanstalk:*",
                "elasticloadbalancing:*",
                "iam:ListRoles",
                "logs:*",
                "sns:*",
                "cloud9:CreateEnvironmentEC2",
                "cloud9:DeleteEnvironment",
                "cloud9:DescribeEnvironment*",
                "cloud9:ListEnvironments"
            ],
            "Resource": "*"
        },
        {
            "Sid": "ProjectWorkerRoles",
            "Effect": "Allow",
            "Action": [
                "iam:AttachRolePolicy",
                "iam:CreateRole",
                "iam:DeleteRole",
                "iam:DeleteRolePolicy",
                "iam:DetachRolePolicy",
                "iam:GetRole",
                "iam:PassRole",
                "iam:GetRolePolicy",
                "iam:PutRolePolicy",
                "iam:SetDefaultPolicyVersion",
                "iam:CreatePolicy",
```

```
                "iam:DeletePolicy",
                "iam:AddRoleToInstanceProfile",
                "iam:CreateInstanceProfile",
                "iam:DeleteInstanceProfile",
                "iam:RemoveRoleFromInstanceProfile"
            ],
            "Resource": [
                "arn:aws:iam::*:role/CodeStarWorker*",
                "arn:aws:iam::*:policy/CodeStarWorker*",
                "arn:aws:iam::*:instance-profile/awscodestar-*"
            ]
        },
        {
            "Sid": "ProjectTeamMembers",
            "Effect": "Allow",
            "Action": [
                "iam:AttachUserPolicy",
                "iam:DetachUserPolicy"
            ],
            "Resource": "*",
            "Condition": {
                "ArnEquals": {
                    "iam:PolicyArn": [
                        "arn:aws:iam::*:policy/CodeStar_*"
                    ]
                }
            }
        },
        {
            "Sid": "ProjectRoles",
            "Effect": "Allow",
            "Action": [
                "iam:CreatePolicy",
                "iam:DeletePolicy",
                "iam:CreatePolicyVersion",
                "iam:DeletePolicyVersion",
                "iam:ListEntitiesForPolicy",
                "iam:ListPolicyVersions",
                "iam:GetPolicy",
                "iam:GetPolicyVersion"
            ],
            "Resource": [
                "arn:aws:iam::*:policy/CodeStar_*"
            ]
```

```
        },
        {
            "Sid": "InspectServiceRole",
            "Effect": "Allow",
            "Action": [
                "iam:ListAttachedRolePolicies"
            ],
            "Resource": [
                "arn:aws:iam::*:role/aws-codestar-service-role",
                "arn:aws:iam::*:role/service-role/aws-codestar-service-role"
            ]
        },
        {
            "Sid": "IAMLinkRole",
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceLinkedRole"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:AWSServiceName": "cloud9.amazonaws.com"
                }
            }
        },
        {
            "Sid": "DescribeConfigRuleForARN",
            "Effect": "Allow",
            "Action": [
                "config:DescribeConfigRules"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Sid": "ProjectCodeStarConnections",
            "Effect": "Allow",
            "Action": [
                "codestar-connections:UseConnection",
                "codestar-connections:GetConnection"
            ],
            "Resource": "*"
        },
```

```
        {
            "Sid": "ProjectCodeStarConnectionsPassConnections",
            "Effect": "Allow",
            "Action": "codestar-connections:PassConnection",
            "Resource": "*",
            "Condition": {
                "StringEqualsIfExists": {
                    "codestar-connections:PassedToService":
 "codepipeline.amazonaws.com"
                }
            }
        }
    ]
}
```

AWSCodeStarFullAccess ポリシー

AWS CodeStarのセットアップ 手順では、 という名前のポリシーAWSCodeStarFullAccessをIAMユーザーにアタッチしました。このポリシーステートメントにより、ユーザーは、 AWS アカウントに関連付けられたすべての使用可能なリソース AWS CodeStar を使用して、 で使用可能な AWS CodeStar すべてのアクションを実行できます。これには、プロジェクトの作成と削除が含まれます。次の例は、代表的な AWSCodeStarFullAccess ポリシーのスニペットです。実際のポリシーは、新しい AWS CodeStar プロジェクトを開始するときに選択したテンプレートによって異なります。

AWS CloudFormation は、ターゲットスタックcloudformation::DescribeStacksなしで を呼び出すときにアクセスcloudformation::ListStacks許可を必要とします。

許可の詳細

このポリシーには以下を実行するためのアクセス許可が含まれています。

- ec2- EC2インスタンスに関する情報を取得して AWS CodeStar プロジェクトを作成します。
- cloud9- AWS Command Line Interface 環境に関する情報を取得します。
- cloudformation– AWS CodeStar プロジェクトスタックに関する情報を取得します。
- codestar– AWS CodeStar プロジェクト内でアクションを実行します。

```
{
  "Version": "2012-10-17",
  "Statement": [
```

```
    {
      "Sid": "CodeStarEC2",
      "Effect": "Allow",
      "Action": [
        "codestar:*",
        "ec2:DescribeKeyPairs",
        "ec2:DescribeVpcs",
        "ec2:DescribeSubnets",
        "cloud9:DescribeEnvironment*"
      ],
      "Resource": "*"
    },
    {
      "Sid": "CodeStarCF",
      "Effect": "Allow",
      "Action": [
        "cloudformation:DescribeStack*",
        "cloudformation:ListStacks*",
        "cloudformation:GetTemplateSummary"
      ],
      "Resource": [
        "arn:aws:cloudformation:*:*:stack/awscodestar-*"
      ]
    }
  ]
}
```

ほとんどの場合、すべてのユーザーにこのような過剰なアクセス許可を付与することはありません。代わりに、 によって管理されるプロジェクトロールを使用して、プロジェクトレベルのアクセス許可を追加できます AWS CodeStar。ロールは AWS CodeStar 、プロジェクトへの特定のレベルのアクセスを許可し、次のように名前が付けられます。

- [所有者]
- 寄稿者
- 表示者

AWS CodeStar 所有者ロールポリシー

AWS CodeStar 所有者ロールポリシーは、ユーザーが制限なしで AWS CodeStar プロジェクト内のすべてのアクションを実行することを許可します。AWS CodeStar は、所有者アクセスレベルのプロジェクトチームメンバーにCodeStar_*project-id*_Ownerポリシーを適用します。

```
...
{
  "Effect": "Allow",
  "Action": [
    ...
    "codestar:*",
    ...
  ],
  "Resource": [
    "arn:aws:codestar:us-east-2:111111111111:project/project-id",
    "arn:aws:iam::account-id:policy/CodeStar_project-id_Owner"
  ]
},
{
  "Effect": "Allow",
  "Action": [
    "codestar:DescribeUserProfile",
    "codestar:ListProjects",
    "codestar:ListUserProfiles",
    "codestar:VerifyServiceRole",
    ...
  ],
  "Resource": [
    "*"
  ]
},
{
  "Effect": "Allow",
  "Action": [
    "codestar:*UserProfile",
    ...
  ],
  "Resource": [
    "arn:aws:iam::account-id:user/user-name"
  ]
}
...
```

AWS CodeStar 寄稿者のロールポリシー

AWS CodeStar 寄稿者ロールポリシーは、ユーザーがプロジェクトに寄稿し、プロジェクトダッシュボードを変更することを許可します。 は、寄稿者アクセスレベルのプロジェクトチームメン

バーにCodeStar_*project-id*_ContributorポリシーAWS CodeStar を適用します。寄稿者のアクセス権を持っているユーザーは、プロジェクトへの寄稿とダッシュボードの変更はできますが、チームメンバーの追加や削除はできません。

```
...
{
  "Effect": "Allow",
  "Action": [
    ...
    "codestar:Describe*",
    "codestar:Get*",
    "codestar:List*",
    "codestar:PutExtendedAccess",
    ...
  ],
  "Resource": [
    "arn:aws:codestar:us-east-2:111111111111:project/project-id",
    "arn:aws:iam::account-id:policy/CodeStar_project-id_Contributor"
  ]
},
{
  "Effect": "Allow",
  "Action": [
    "codestar:DescribeUserProfile",
    "codestar:ListProjects",
    "codestar:ListUserProfiles",
    "codestar:VerifyServiceRole",
    ...
  ],
  "Resource": [
    "*"
  ]
},
{
  "Effect": "Allow",
  "Action": [
    "codestar:*UserProfile",
    ...
  ],
  "Resource": [
    "arn:aws:iam::account-id:user/user-name"
  ]
}
```

```
...
```

AWS CodeStar ビューワーロールポリシー

AWS CodeStar ビューワーロールポリシーは、ユーザーが でプロジェクトを表示することを許可しますAWS CodeStar。AWS CodeStar は、ビューワーアクセスレベルのプロジェクトチームメンバーにCodeStar_*project-id*_Viewerポリシーを適用します。ビューワーアクセス権を持つユーザーは、 でプロジェクトを表示できますがAWS CodeStar、そのリソースを変更したり、チームメンバーを追加または削除したりすることはできません。

```
...
{
  "Effect": "Allow",
  "Action": [
    ...
    "codestar:Describe*",
    "codestar:Get*",
    "codestar:List*",
    ...
  ],
  "Resource": [
    "arn:aws:codestar:us-east-2:111111111111:project/project-id",
    "arn:aws:iam::account-id:policy/CodeStar_project-id_Viewer"
  ]
},
{
  "Effect": "Allow",
  "Action": [
    "codestar:DescribeUserProfile",
    "codestar:ListProjects",
    "codestar:ListUserProfiles",
    "codestar:VerifyServiceRole",
    ...
  ],
  "Resource": [
    "*"
  ]
},
{
  "Effect": "Allow",
  "Action": [
    "codestar:*UserProfile",
    ...
```

```
  ],
  "Resource": [
    "arn:aws:iam::account-id:user/user-name"
  ]
}
...
```

AWS CodeStar ツールチェーンワーカーロールポリシー (2018 年 12 月 6 日以降 PDT)

2018 年 12 月 6 日以降に作成された AWS CodeStar プロジェクトの場合、 AWS CodeStar はPDT、他の AWS サービスでプロジェクトのリソースを作成するワーカーロールのインラインポリシーを作成します。ポリシーの内容は、作成するプロジェクトのタイプによって異なります。ポリシーの例を次に示します。詳細については、「IAM ワーカーロールのポリシー」を参照してください。

```
{
  "Statement": [
    {
      "Action": [
        "s3:GetObject",
        "s3:GetObjectVersion",
        "s3:GetBucketVersioning",
        "s3:PutObject*",
        "codecommit:CancelUploadArchive",
        "codecommit:GetBranch",
        "codecommit:GetCommit",
        "codecommit:GetUploadArchiveStatus",
        "codecommit:GitPull",
        "codecommit:UploadArchive",
        "codebuild:StartBuild",
        "codebuild:BatchGetBuilds",
        "codebuild:StopBuild",
        "logs:CreateLogGroup",
        "logs:CreateLogStream",
        "logs:PutLogEvents",
        "cloudformation:DescribeStacks",
        "cloudformation:DescribeChangeSet",
        "cloudformation:CreateChangeSet",
        "cloudformation:DeleteChangeSet",
        "cloudformation:ExecuteChangeSet",
        "codepipeline:StartPipelineExecution",
        "lambda:ListFunctions",
```

```
        "lambda:InvokeFunction",
        "sns:Publish"
      ],
      "Resource": [
        "*"
      ],
      "Effect": "Allow"
    },
    {
      "Action": [
        "iam:PassRole"
      ],
      "Resource": [
        "*"
      ],
      "Effect": "Allow"
    },
    {
      "Action": [
        "kms:GenerateDataKey*",
        "kms:Encrypt",
        "kms:Decrypt"
      ],
      "Resource": [
        "*"
      ],
      "Effect": "Allow"
    }
  ]
}
```

AWS CloudFormation ワーカーロールポリシー

2018 年 12 月 6 日以降に作成された AWS CodeStar プロジェクトの場合、 AWS CodeStar は PDT、AWS CodeStar プロジェクトの AWS CloudFormation リソースを作成するワーカーロールのインラインポリシーを作成します。ポリシーの内容は、プロジェクトで必要なリソースのタイプによって異なります。ポリシーの例を次に示します。詳細については、「IAM ワーカーロールのポリシー」を参照してください。

```
{
{
    "Statement": [
```

```
        {
            "Action": [
                "s3:PutObject",
                "s3:GetObject",
                "s3:GetObjectVersion"
            ],
            "Resource": [
                "arn:aws:s3:::aws-codestar-region-id-account-id-project-id",
                "arn:aws:s3:::aws-codestar-region-id-account-id-project-id/*"
            ],
            "Effect": "Allow"
        },
        {
            "Action": [
                "apigateway:DELETE",
                "apigateway:GET",
                "apigateway:PATCH",
                "apigateway:POST",
                "apigateway:PUT",
                "codedeploy:CreateApplication",
                "codedeploy:CreateDeployment",
                "codedeploy:CreateDeploymentConfig",
                "codedeploy:CreateDeploymentGroup",
                "codedeploy:DeleteApplication",
                "codedeploy:DeleteDeployment",
                "codedeploy:DeleteDeploymentConfig",
                "codedeploy:DeleteDeploymentGroup",
                "codedeploy:GetDeployment",
                "codedeploy:GetDeploymentConfig",
                "codedeploy:GetDeploymentGroup",
                "codedeploy:RegisterApplicationRevision",
                "codestar:SyncResources",
                "config:DeleteConfigRule",
                "config:DescribeConfigRules",
                "config:ListTagsForResource",
                "config:PutConfigRule",
                "config:TagResource",
                "config:UntagResource",
                "dynamodb:CreateTable",
                "dynamodb:DeleteTable",
                "dynamodb:DescribeContinuousBackups",
                "dynamodb:DescribeTable",
                "dynamodb:DescribeTimeToLive",
                "dynamodb:ListTagsOfResource",
```

```
                "dynamodb:TagResource",
                "dynamodb:UntagResource",
                "dynamodb:UpdateContinuousBackups",
                "dynamodb:UpdateTable",
                "dynamodb:UpdateTimeToLive",
                "ec2:AssociateIamInstanceProfile",
                "ec2:AttachVolume",
                "ec2:CreateSecurityGroup",
                "ec2:createTags",
                "ec2:DescribeIamInstanceProfileAssociations",
                "ec2:DescribeInstances",
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeSubnets",
                "ec2:DetachVolume",
                "ec2:DisassociateIamInstanceProfile",
                "ec2:ModifyInstanceAttribute",
                "ec2:ModifyInstanceCreditSpecification",
                "ec2:ModifyInstancePlacement",
                "ec2:MonitorInstances",
                "ec2:ReplaceIamInstanceProfileAssociation",
                "ec2:RunInstances",
                "ec2:StartInstances",
                "ec2:StopInstances",
                "ec2:TerminateInstances",
                "events:DeleteRule",
                "events:DescribeRule",
                "events:ListTagsForResource",
                "events:PutRule",
                "events:PutTargets",
                "events:RemoveTargets",
                "events:TagResource",
                "events:UntagResource",
                "kinesis:AddTagsToStream",
                "kinesis:CreateStream",
                "kinesis:DecreaseStreamRetentionPeriod",
                "kinesis:DeleteStream",
                "kinesis:DescribeStream",
                "kinesis:IncreaseStreamRetentionPeriod",
                "kinesis:RemoveTagsFromStream",
                "kinesis:StartStreamEncryption",
                "kinesis:StopStreamEncryption",
                "kinesis:UpdateShardCount",
                "lambda:CreateAlias",
                "lambda:CreateFunction",
```

```
                "lambda:DeleteAlias",
                "lambda:DeleteFunction",
                "lambda:DeleteFunctionConcurrency",
                "lambda:GetFunction",
                "lambda:GetFunctionConfiguration",
                "lambda:ListTags",
                "lambda:ListVersionsByFunction",
                "lambda:PublishVersion",
                "lambda:PutFunctionConcurrency",
                "lambda:TagResource",
                "lambda:UntagResource",
                "lambda:UpdateAlias",
                "lambda:UpdateFunctionCode",
                "lambda:UpdateFunctionConfiguration",
                "s3:CreateBucket",
                "s3:DeleteBucket",
                "s3:DeleteBucketWebsite",
                "s3:PutAccelerateConfiguration",
                "s3:PutAnalyticsConfiguration",
                "s3:PutBucketAcl",
                "s3:PutBucketCORS",
                "s3:PutBucketLogging",
                "s3:PutBucketNotification",
                "s3:PutBucketPublicAccessBlock",
                "s3:PutBucketVersioning",
                "s3:PutBucketWebsite",
                "s3:PutEncryptionConfiguration",
                "s3:PutInventoryConfiguration",
                "s3:PutLifecycleConfiguration",
                "s3:PutMetricsConfiguration",
                "s3:PutReplicationConfiguration",
                "sns:CreateTopic",
                "sns:DeleteTopic",
                "sns:GetTopicAttributes",
                "sns:ListSubscriptionsByTopic",
                "sns:ListTopics",
                "sns:SetSubscriptionAttributes",
                "sns:Subscribe",
                "sns:Unsubscribe",
                "sqs:CreateQueue",
                "sqs:DeleteQueue",
                "sqs:GetQueueAttributes",
                "sqs:GetQueueUrl",
                "sqs:ListQueueTags",
```

```
                "sqs:TagQueue",
                "sqs:UntagQueue"
            ],
            "Resource": "*",
            "Effect": "Allow"
        },
        {
            "Action": [
                "lambda:AddPermission",
                "lambda:RemovePermission"
            ],
            "Resource": [
                "arn:aws:lambda:region-id:account-id:function:awscodestar-*"
            ],
            "Effect": "Allow"
        },
        {
            "Action": [
                "iam:PassRole"
            ],
            "Resource": [
                "arn:aws:iam::account-id:role/CodeStar-project-id*"
            ],
            "Effect": "Allow"
        },
        {
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "codedeploy.amazonaws.com"
                }
            },
            "Action": [
                "iam:PassRole"
            ],
            "Resource": [
                "arn:aws:iam::account-id:role/CodeStarWorker-project-id-CodeDeploy"
            ],
            "Effect": "Allow"
        },
        {
            "Action": [
                "cloudformation:CreateChangeSet"
            ],
            "Resource": [
```

```
                "arn:aws:cloudformation:region-id:aws:transform/Serverless-2016-10-31",
                "arn:aws:cloudformation:region-id:aws:transform/CodeStar"
            ],
            "Effect": "Allow"
        },
        {
            "Action": [
                "iam:CreateServiceLinkedRole",
                "iam:GetRole",
                "iam:DeleteRole",
                "iam:DeleteUser"
            ],
            "Resource": "*",
            "Effect": "Allow"
        },
        {
            "Condition": {
                "StringEquals": {
                    "iam:PermissionsBoundary": "arn:aws:iam::account-id:policy/
CodeStar_project-id_PermissionsBoundary"
                }
            },
            "Action": [
                "iam:AttachRolePolicy",
                "iam:AttachUserPolicy",
                "iam:CreateRole",
                "iam:CreateUser",
                "iam:DeleteRolePolicy",
                "iam:DeleteUserPolicy",
                "iam:DetachUserPolicy",
                "iam:DetachRolePolicy",
                "iam:PutUserPermissionsBoundary",
                "iam:PutRolePermissionsBoundary"
            ],
            "Resource": "*",
            "Effect": "Allow"
        },
        {
            "Action": [
                "kms:CreateKey",
                "kms:CreateAlias",
                "kms:DeleteAlias",
                "kms:DisableKey",
                "kms:EnableKey",
```

```
                "kms:UpdateAlias",
                "kms:TagResource",
                "kms:UntagResource"
            ],
            "Resource": "*",
            "Effect": "Allow"
        },
        {
            "Condition": {
                "StringEquals": {
                    "ssm:ResourceTag/awscodestar:projectArn":
 "arn:aws:codestar:project-id:account-id:project/project-id"
                }
            },
            "Action": [
                "ssm:GetParameter*"
            ],
            "Resource": "*",
            "Effect": "Allow"
        }
    ]
}
```

AWS CloudFormation ワーカーロールポリシー (2018 年 12 月 6 日より前PDT)

AWS CodeStar プロジェクトが 2018 年 12 月 6 日より前に作成された場合、 は AWS CloudFormation ワーカーロールのインラインポリシーPDTAWS CodeStar を作成しました。ポリシーステートメントの例を以下に示します。

```
{
    "Statement": [
        {
            "Action": [
                "s3:PutObject",
                "s3:GetObject",
                "s3:GetObjectVersion"
            ],
            "Resource": [
                "arn:aws:s3:::aws-codestar-us-east-1-account-id-project-id-pipe",
                "arn:aws:s3:::aws-codestar-us-east-1-account-id-project-id-pipe/*"
            ],
            "Effect": "Allow"
```

```
        },
        {
            "Action": [
                "codestar:SyncResources",
                "lambda:CreateFunction",
                "lambda:DeleteFunction",
                "lambda:AddPermission",
                "lambda:UpdateFunction",
                "lambda:UpdateFunctionCode",
                "lambda:GetFunction",
                "lambda:GetFunctionConfiguration",
                "lambda:UpdateFunctionConfiguration",
                "lambda:RemovePermission",
                "lambda:listTags",
                "lambda:TagResource",
                "lambda:UntagResource",
                "apigateway:*",
                "dynamodb:CreateTable",
                "dynamodb:DeleteTable",
                "dynamodb:DescribeTable",
                "kinesis:CreateStream",
                "kinesis:DeleteStream",
                "kinesis:DescribeStream",
                "sns:CreateTopic",
                "sns:DeleteTopic",
                "sns:ListTopics",
                "sns:GetTopicAttributes",
                "sns:SetTopicAttributes",
                "s3:CreateBucket",
                "s3:DeleteBucket",
                "config:DescribeConfigRules",
                "config:PutConfigRule",
                "config:DeleteConfigRule",
                "ec2:*",
                "autoscaling:*",
                "elasticloadbalancing:*",
                "elasticbeanstalk:*"
            ],
            "Resource": "*",
            "Effect": "Allow"
        },
        {
            "Action": [
                "iam:PassRole"
```

```
            ],
            "Resource": [
                "arn:aws:iam::account-id:role/CodeStarWorker-project-id-Lambda"
            ],
            "Effect": "Allow"
        },
        {
            "Action": [
                "cloudformation:CreateChangeSet"
            ],
            "Resource": [
                "arn:aws:cloudformation:us-east-1:aws:transform/Serverless-2016-10-31",
                "arn:aws:cloudformation:us-east-1:aws:transform/CodeStar"
            ],
            "Effect": "Allow"
        }
    ]
}
```

AWS CodePipeline ワーカーロールポリシー (2018 年 12 月 6 日より前PDT)

AWS CodeStar プロジェクトが 2018 年 12 月 6 日より前に作成された場合、 は CodePipeline ワーカーロールのインラインポリシーPDTAWS CodeStar を作成しました。ポリシーステートメントの例を以下に示します。

```
{
    "Statement": [
        {
            "Action": [
                "s3:GetObject",
                "s3:GetObjectVersion",
                "s3:GetBucketVersioning",
                "s3:PutObject"
            ],
            "Resource": [
                "arn:aws:s3:::aws-codestar-us-east-1-account-id-project-id-pipe",
                "arn:aws:s3:::aws-codestar-us-east-1-account-id-project-id-pipe/*"
            ],
            "Effect": "Allow"
        },
        {
            "Action": [
                "codecommit:CancelUploadArchive",
```

```
                "codecommit:GetBranch",
                "codecommit:GetCommit",
                "codecommit:GetUploadArchiveStatus",
                "codecommit:UploadArchive"
            ],
            "Resource": [
                "arn:aws:codecommit:us-east-1:account-id:project-id"
            ],
            "Effect": "Allow"
        },
        {
            "Action": [
                "codebuild:StartBuild",
                "codebuild:BatchGetBuilds",
                "codebuild:StopBuild"
            ],
            "Resource": [
                "arn:aws:codebuild:us-east-1:account-id:project/project-id"
            ],
            "Effect": "Allow"
        },
        {
            "Action": [
                "cloudformation:DescribeStacks",
                "cloudformation:DescribeChangeSet",
                "cloudformation:CreateChangeSet",
                "cloudformation:DeleteChangeSet",
                "cloudformation:ExecuteChangeSet"
            ],
            "Resource": [
                "arn:aws:cloudformation:us-east-1:account-id:stack/awscodestar-project-
id-lambda/*"
            ],
            "Effect": "Allow"
        },
        {
            "Action": [
                "iam:PassRole"
            ],
            "Resource": [
                "arn:aws:iam::account-id:role/CodeStarWorker-project-id-CloudFormation"
            ],
            "Effect": "Allow"
        }
```

```
    ]
}
```

AWS CodeBuild ワーカーロールポリシー (2018 年 12 月 6 日より前PDT)

AWS CodeStar プロジェクトが 2018 年 12 月 6 日より前に作成された場合、 は CodeBuild ワーカーロールのインラインポリシーPDTAWS CodeStar を作成しました。ポリシーステートメントの例を以下に示します。

```
{
    "Statement": [
        {
            "Action": [
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": "*",
            "Effect": "Allow"
        },
        {
            "Action": [
                "s3:PutObject",
                "s3:GetObject",
                "s3:GetObjectVersion"
            ],
            "Resource": [
                "arn:aws:s3:::aws-codestar-us-east-1-account-id-project-id-pipe",
                "arn:aws:s3:::aws-codestar-us-east-1-account-id-project-id-pipe/*",
                "arn:aws:s3:::aws-codestar-us-east-1-account-id-project-id-app",
                "arn:aws:s3:::aws-codestar-us-east-1-account-id-project-id-app/*"
            ],
            "Effect": "Allow"
        },
        {
            "Action": [
                "codecommit:GitPull"
            ],
            "Resource": [
                "arn:aws:codecommit:us-east-1:account-id:project-id"
            ],
            "Effect": "Allow"
        },
```

```
        {
            "Action": [
                "kms:GenerateDataKey*",
                "kms:Encrypt",
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:kms:us-east-1:account-id:alias/aws/s3"
            ],
            "Effect": "Allow"
        }
    ]
}
```

Amazon CloudWatch Events ワーカーロールポリシー (2018 年 12 月 6 日より前 PDT)

AWS CodeStar プロジェクトが 2018 年 12 月 6 日より前に作成された場合、 は CloudWatch Events ワーカーロールのインラインポリシーPDTAWS CodeStar を作成しました。ポリシーステートメントの例を以下に示します。

```
{
    "Statement": [
        {
            "Action": [
                "codepipeline:StartPipelineExecution"
            ],
            "Resource": [
                "arn:aws:codepipeline:us-east-1:account-id:project-id-Pipeline"
            ],
            "Effect": "Allow"
        }
    ]
}
```

AWS CodeStar アクセス許可の境界ポリシー

2018 年 12 月 6 日以降にAWS CodeStar プロジェクトを作成すると、 PDTAWS CodeStar はプロジェクトのアクセス許可の境界ポリシーを作成します。このポリシーでは、プロジェクト外部のリソースへの特権のエスカレーションを防止できます。これは、プロジェクトの進化につれて更新され

る動的なポリシーです。ポリシーの内容は、作成するプロジェクトのタイプによって異なります。ポリシーの例を次に示します。詳細については、「IAM アクセス権限の境界」を参照してください。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "1",
      "Effect": "Allow",
      "Action": [
        "s3:GetObject"
      ],
      "Resource": [
        "arn:aws:s3:::*/AWSLogs/*/Config/*"
      ]
    },
    {
      "Sid": "2",
      "Effect": "Allow",
      "Action": [
        "*"
      ],
      "Resource": [
        "arn:aws:codestar:us-east-1:account-id:project/project-id",
        "arn:aws:cloudformation:us-east-1:account-id:stack/awscodestar-project-id-
lambda/eefbbf20-c1d9-11e8-8a3a-500c28b4e461",
        "arn:aws:cloudformation:us-east-1:account-id:stack/awscodestar-project-
id/4b80b3f0-c1d9-11e8-8517-500c28b236fd",
        "arn:aws:codebuild:us-east-1:account-id:project/project-id",
        "arn:aws:codecommit:us-east-1:account-id:project-id",
        "arn:aws:codepipeline:us-east-1:account-id:project-id-Pipeline",
        "arn:aws:execute-api:us-east-1:account-id:7rlst5mrgi",
        "arn:aws:iam::account-id:role/CodeStarWorker-project-id-CloudFormation",
        "arn:aws:iam::account-id:role/CodeStarWorker-project-id-CloudWatchEventRule",
        "arn:aws:iam::account-id:role/CodeStarWorker-project-id-CodeBuild",
        "arn:aws:iam::account-id:role/CodeStarWorker-project-id-CodePipeline",
        "arn:aws:iam::account-id:role/CodeStarWorker-project-id-Lambda",
        "arn:aws:lambda:us-east-1:account-id:function:awscodestar-project-id-lambda-
GetHelloWorld-KFKTXYNH9573",
        "arn:aws:s3:::aws-codestar-us-east-1-account-id-project-id-app",
        "arn:aws:s3:::aws-codestar-us-east-1-account-id-project-id-pipe"
      ]
    },
```

```
    {
      "Sid": "3",
      "Effect": "Allow",
      "Action": [
        "apigateway:GET",
        "config:Describe*",
        "config:Get*",
        "config:List*",
        "config:Put*",
        "logs:CreateLogGroup",
        "logs:CreateLogStream",
        "logs:DescribeLogGroups",
        "logs:PutLogEvents"
      ],
      "Resource": [
        "*"
      ]
    }
  ]
}
```

プロジェクトのリソースの一覧表示

この例では、 AWS アカウントの指定されたIAMユーザーに、 AWS CodeStar プロジェクトのリソースを一覧表示するためのアクセス権を付与します。

```
{
  "Version": "2012-10-17",
  "Statement" : [
    {
      "Effect" : "Allow",
      "Action" : [
        "codestar:ListResources",
      ],
      "Resource" : "arn:aws:codestar:us-east-2:project/my-first-projec"
    }
  ]
}
```

AWS CodeStar コンソールの使用

AWS CodeStar コンソールにアクセスするための特定のアクセス許可は必要ありません
が、AWSCodeStarFullAccessポリシーまたは AWS CodeStar プロジェクトレベルのロール

の所有者、寄稿者、または閲覧者のいずれかがいない限り、便利な操作を行うことはできません。AWSCodeStarFullAccess の詳細については、「AWSCodeStarFullAccess ポリシー」をご参照ください。プロジェクトレベルのポリシーの詳細については、「IAM チームメンバーのポリシー」を参照してください。

AWS CLI または のみを呼び出すユーザーには、最小限のコンソールアクセス許可を付与する必要はありません AWS API。代わりに、実行しようとしているAPIオペレーションに一致するアクションのみへのアクセスを許可します。

ユーザーが自分のアクセス許可を表示できるようにする

この例では、IAMユーザーがユーザー ID にアタッチされているインラインポリシーと管理ポリシーを表示できるようにするポリシーを作成する方法を示します。このポリシーには、コンソールで、または AWS CLI または を使用してプログラムでこのアクションを実行するアクセス許可が含まれています AWS API。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
```

```
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

AWS CodeStar プロジェクトの更新

この例では、 AWS アカウントの指定されたIAMユーザーに、 AWS CodeStar プロジェクトの説明などのプロジェクトの属性を編集するためのアクセス権を付与します。

```
{
  "Version": "2012-10-17",
  "Statement" : [
    {
      "Effect" : "Allow",
      "Action" : [
        "codestar:UpdateProject"
      ],
      "Resource" : "arn:aws:codestar:us-east-2:project/my-first-projec"
    }
  ]
}
```

プロジェクトへのチームメンバーの追加

この例では、指定されたIAMユーザーにプロジェクト ID を持つ AWS CodeStar プロジェクトにチームメンバーを追加する権限を付与します。*my-first-projec*ただし、そのユーザーがチームメンバーを削除する権限を明示的に拒否するには、次のようにします。

```
{
  "Version": "2012-10-17",
  "Statement" : [
    {
      "Effect" : "Allow",
      "Action" : [
        "codestar:AssociateTeamMember",
      ],
      "Resource" : "arn:aws:codestar:us-east-2:project/my-first-projec"
    },
    {
```

```
      "Effect" : "Deny",
      "Action" : [
        "codestar:DisassociateTeamMember",
      ],
      "Resource" : "arn:aws:codestar:us-east-2:project/my-first-projec"
    }
      ]

  ]
}
```

AWS アカウントに関連付けられたユーザープロファイルの一覧表示

この例では、このポリシーがアタッチされている IAM ユーザーに、 AWS アカウントに関連付けられているすべての AWS CodeStar ユーザープロファイルを一覧表示することを許可します。

```
{
  "Version": "2012-10-17",
  "Statement" : [
    {
      "Effect" : "Allow",
      "Action" : [
        "codestar:ListUserProfiles",
              ],
      "Resource" : "*"
    }
  ]
}
```

タグに基づく AWS CodeStar プロジェクトの表示

アイデンティティベースのポリシーの条件を使用して、タグに基づいてAWS CodeStar プロジェクトへのアクセスを制御できます。この例では、プロジェクトを表示できるポリシーを作成する方法を示します。ただし、プロジェクトタグ Owner にそのユーザーのユーザー名の値がある場合のみ、アクセス許可は付与されます。このポリシーでは、このアクションをコンソールで実行するために必要なアクセス許可も付与します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ListProjectsInConsole",
```

```
            "Effect": "Allow",
            "Action": "codestar:ListProjects",
            "Resource": "*"
        },
        {
            "Sid": "ViewProjectIfOwner",
            "Effect": "Allow",
            "Action": "codestar:GetProject,
            "Resource": "arn:aws:codestar:*:*:project/*",
            "Condition": {
                "StringEquals": {"codestar:ResourceTag/Owner": "${aws:username}"}
            }
        }
    ]
}
```

このポリシーは、アカウントのIAMユーザーにアタッチできます。という名前のユーザーがAWS CodeStar プロジェクトを表示richard-roeしようとする場合は、プロジェクトに Owner=richard-roeまたは のタグを付ける必要がありますowner=richard-roe。それ以外の場合、アクセスは拒否されます。条件キー名では大文字と小文字が区別されないため、条件タグキー Owner は Owner と owner の両方に一致します。詳細については、「 ユーザーガイド」のIAMJSON「ポリシー要素: 条件IAM」を参照してください。

AWS CodeStarAWS 管理ポリシーの更新

このサービスがこれらの変更の追跡を開始したAWS CodeStar 以降の の AWS マネージドポリシーの更新に関する詳細を表示します。このページの変更に関する自動アラートを受け取るには、AWS CodeStar ドキュメント履歴ページのRSSフィードにサブスクライブします。

変更	説明	日付
AWSCodeStarFullAccess ポリシー – AWSCodeStarFullAccess ポリシーの更新	AWS CodeStar アクセスロールポリシーが更新されました。ポリシーの結果は同じですが、クラウドフォーメーションには DescribeStacks ListStacks に加えて が必要です。これは既に必要です。	2023 年 3 月 24 日

変更	説明	日付
AWSCodeStarServiceRole ポリシー — AWSCodeStarServiceRole ポリシーの更新	AWS CodeStar サービスロールのポリシーが更新され、ポリシーステートメントの冗長アクションが修正されました。 サービスロールポリシーは、AWS CodeStar サービスがユーザーに代わってアクションを実行することを許可します。	2021 年 9 月 23 日
AWS CodeStar が変更の追跡を開始しました	AWS CodeStar が AWS マネージドポリシーの変更の追跡を開始しました。	2021 年 9 月 23 日

AWS CodeStar Identity and Access のトラブルシューティング

次の情報は、 および の使用時に発生する可能性がある一般的な問題の診断AWS CodeStar と修正に役立ちますIAM。

トピック

- でアクションを実行する権限がない AWS CodeStar
- iam を実行する権限がありません。PassRole
- 自分の AWS アカウント以外のユーザーに自分のAWS CodeStarリソースへのアクセスを許可したい

でアクションを実行する権限がない AWS CodeStar

から、アクションを実行する権限がないと AWS Management Console 通知された場合は、管理者に連絡してサポートを依頼してください。サインイン認証情報を提供した担当者が管理者です。

次の例のエラーは、mateojacksonIAMユーザーがコンソールを使用して の詳細を表示しようとすると発生します。*widget* ただし、 にはcodestar:*GetWidget*アクセス許可がありません。

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 codestar:GetWidget on resource: my-example-widget
```

この場合、Mateo は、codestar:*GetWidget* アクションを使用して *my-example-widget* リソースへのアクセスが許可されるように、管理者にポリシーの更新を依頼します。

iam を実行する権限がありません。PassRole

iam:PassRole アクションを実行する権限がないというエラーが表示された場合は、ポリシーを更新して にロールを渡すことができるようにする必要がありますAWS CodeStar。

一部の AWS のサービス では、新しいサービスロールまたはサービスにリンクされたロールを作成する代わりに、そのサービスに既存のロールを渡すことができます。そのためには、サービスにロールを渡す権限が必要です。

次の例のエラーは、 というIAMユーザーがコンソールを使用して marymajor でアクションを実行しようする場合に発生しますAWS CodeStar。ただし、このアクションをサービスが実行するには、サービスロールから付与された権限が必要です。メアリーには、ロールをサービスに渡す許可がありません。

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

この場合、Mary のポリシーを更新してメアリーに iam:PassRole アクションの実行を許可する必要があります。

サポートが必要な場合は、 AWS 管理者にお問い合わせください。サインイン認証情報を提供した担当者が管理者です。

自分の AWS アカウント以外のユーザーに自分のAWS CodeStarリソースへのアクセスを許可したい

他のアカウントのユーザーや組織外の人が、リソースにアクセスするために使用できるロールを作成できます。ロールの引き受けを委託するユーザーを指定できます。リソースベースのポリシーまたはアクセスコントロールリスト (ACLs) をサポートするサービスの場合、これらのポリシーを使用して、ユーザーにリソースへのアクセスを許可できます。

詳細については、以下を参照してください。

- がこれらの機能AWS CodeStar をサポートしているかどうかを確認するには、「」を参照してくださいAWS CodeStar の仕組み IAM。
- 所有している のリソースへのアクセスを提供する方法については、 AWS アカウント 「 ユーザーガイド」の「所有 AWS アカウント している別の のIAMユーザーへのアクセスを提供するIAM」を参照してください。
- リソースへのアクセスをサードパーティー に提供する方法については AWS アカウント、 IAM ユーザーガイドの「サードパーティー AWS アカウント が所有する へのアクセスを提供する」を参照してください。
- ID フェデレーションを通じてアクセスを提供する方法については、 IAMユーザーガイドの「外部認証されたユーザーへのアクセスの提供 (ID フェデレーション)」を参照してください。
- クロスアカウントアクセスでのロールとリソースベースのポリシーの使用の違いについては、ユーザーガイドの「 でのクロスアカウントリソースアクセスIAMIAM」を参照してください。

AWS CloudTrail を使用した AWS CodeStar API コールのログ記録

AWS CodeStar は AWS CloudTrail という、AWS CodeStar の ユーザーやロール、または AWS のサービスによって実行されたアクションを記録するサービスと統合しています。CloudTrail は、AWS CodeStar のすべての API コールをイベントとしてキャプチャします。キャプチャされた呼び出しには、AWS CodeStar コンソールの呼び出しと、AWS CodeStar API オペレーションへのコード呼び出しが含まれます。トレイルを作成する場合、AWS CodeStar のイベントを含む S3 バケットに CloudTrail イベントの継続的な配信を有効にできます。追跡を設定しない場合でも、CloudTrail コンソールの [Event history] (イベント履歴) で最新のイベントを表示できます。CloudTrail で収集された情報を使用して、AWS CodeStar に対するリクエスト、リクエスト元の IP アドレス、リクエスト者、リクエスト日時などの詳細を確認できます。

CloudTrail の詳細については、「AWS CloudTrailユーザーガイド」を参照してください。

AWS CodeStarCloudTrail での 情報

CloudTrail は、アカウントを作成すると AWS アカウントで有効になります。AWS CodeStar でアクティビティが発生すると、そのアクティビティは [Event history (イベント履歴)] で AWS のその他のサービスのイベントと共に CloudTrail イベントに記録されます。最近のイベントは、AWS アカウントで表示、検索、ダウンロードできます。詳細については、「CloudTrail イベント履歴でのイベントの表示」を参照してください。

AWS のイベントなど、AWS CodeStar アカウントのイベントの継続的なレコードについては、追跡を作成します。デフォルトでは、コンソールで証跡を作成するときに、証跡がすべての AWS リー

ジョンに適用されます。追跡では、AWS パーティションのすべてのリージョンからのイベントが口グに記録され、指定した S3 バケットにログファイルが配信されます。その他の AWS のサービスを設定して、CloudTrail ログで収集されたデータをより詳細に分析し、それに基づく対応を行うことができます。詳細については、次を参照してください。

- 追跡を作成するための概要
- CloudTrail のサポート対象サービスと統合
- Amazon SNS の CloudTrail の通知の設定
- 複数のリージョンから CloudTrail ログファイルを受け取る、および複数のアカウントから CloudTrail ログファイルを受け取る

すべての AWS CodeStar アクションは CloudTrail によってログに記録され、AWS CodeStarAPI リファレンスに記録されます。例えば、`DescribeProject`、`UpdateProject`、`AssociateTeamMember` の各アクションを呼び出すと、CloudTrail ログファイルにエントリが生成されます。

各イベントまたはログエントリには、リクエストの生成者に関する情報が含まれます。同一性情報は次の判断に役立ちます。

- リクエストが、ルートと IAM ユーザー認証情報のどちらを使用して送信されたか。
- リクエストがロールまたはフェデレーションユーザーの一時的なセキュリティ認証情報を使用して行われたかどうか。
- リクエストが、別の AWS のサービスによって送信されたかどうか。

詳細については、「CloudTrail userIdentity 要素」を参照してください。

AWS CodeStar ログファイルエントリの概要

CloudTrail のログファイルには、単一か複数のログエントリがあります。イベントはあらゆるソースからの単一のリクエストを表し、リクエストされたアクション、アクションの日時、リクエストのパラメータなどの情報が含まれます。CloudTrail ログファイルは、パブリック API コールの順序付けられたスタックトレースではないため、特定の順序では表示されません。

次の例は、AWS CodeStar で `CreateProject` オペレーションが呼び出されたことを示す CloudTrail ログエントリを示しています。

```
{
```

```
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AROAJLIN2OF3UBEXAMPLE:role-name",
    "arn": "arn:aws:sts::account-ID:assumed-role/role-name/role-session-name",
    "accountId": "account-ID",
    "accessKeyId": "ASIAJ44LFQS5XEXAMPLE",
    "sessionContext": {
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2017-06-04T23:56:57Z"
      },
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AROAJLIN2OF3UBEXAMPLE",
        "arn": "arn:aws:iam::account-ID:role/service-role/role-name",
        "accountId": "account-ID",
        "userName": "role-name"
      }
    },
    "invokedBy": "codestar.amazonaws.com"
  },
  "eventTime": "2017-06-04T23:56:57Z",
  "eventSource": "codestar.amazonaws.com",
  "eventName": "CreateProject",
  "awsRegion": "region-ID",
  "sourceIPAddress": "codestar.amazonaws.com",
  "userAgent": "codestar.amazonaws.com",
  "requestParameters": {
    "clientRequestToken": "arn:aws:cloudformation:region-ID:account-ID:stack/stack-
name/additional-ID",
    "id": "project-ID",
    "stackId": "arn:aws:cloudformation:region-ID:account-ID:stack/stack-
name/additional-ID",
    "description": "AWS CodeStar created project",
    "name": "project-name",
    "projectTemplateId": "arn:aws:codestar:region-ID::project-template/project-
template-name"
  },
  "responseElements": {
    "projectTemplateId": "arn:aws:codestar:region-ID::project-template/project-
template-name",
    "arn": "arn:aws:codestar:us-east-1:account-ID:project/project-ID",
```

```
    "clientRequestToken": "arn:aws:cloudformation:region-ID:account-ID:stack/stack-
name/additional-ID",
    "id": "project-ID"
  },
  "requestID": "7d7556d0-4981-11e7-a3bc-dd5daEXAMPLE",
  "eventID": "6b0d6e28-7a1e-4a73-981b-c8fdbEXAMPLE",
  "eventType": "AwsApiCall",
  "recipientAccountId": "account-ID"
}
```

AWS CodeStar のコンプライアンス検証

AWS CodeStar は AWS コンプライアンスプログラムの対象範囲外です。

特定のコンプライアンスプログラムの対象範囲内の AWS のサービスのリストについては、「コンプライアンスプログラムによる対象範囲内の AWS のサービス」を参照してください。一般的な情報については、「AWS コンプライアンスプログラム」を参照してください。

AWS Artifact を使用して、サードパーティーの監査レポートをダウンロードできます。詳細については、「AWS Artifact のレポートのダウンロード」を参照してください。

AWS CodeStar での耐障害性

AWS のグローバルインフラストラクチャは AWS リージョンとアベイラビリティーゾーンを中心に構築されます。AWSリージョンには、低レイテンシー、高いスループット、そして高度の冗長ネットワークで接続されている複数の物理的に独立し隔離されたアベイラビリティーゾーンがあります。アベイラビリティーゾーンでは、アベイラビリティーゾーン間で中断せずに、自動的にフェイルオーバーするアプリケーションとデータベースを設計および運用することができます。アベイラビリティーゾーンは、従来の単一または複数のデータセンターインフラストラクチャよりも可用性、耐障害性、および拡張性が優れています。

AWS リージョンとアベイラビリティーゾーンの詳細については、AWS グローバルインフラストラクチャを参照してください。

のインフラストラクチャセキュリティ AWS CodeStar

マネージドサービスである AWS CodeStar は、 AWS グローバルネットワークセキュリティで保護されています。 AWS セキュリティサービスと がインフラストラクチャ AWS を保護する方法については、AWS 「 クラウドセキュリティ」を参照してください。インフラストラクチャセキュリティの

ベストプラクティスを使用して AWS 環境を設計するには、「 Security Pillar AWS Well-Architected Framework」の「Infrastructure Protection」を参照してください。

が AWS 公開したAPI呼び出しを使用して、ネットワーク CodeStar 経由で にアクセスします。クライアントは以下をサポートする必要があります:

- Transport Layer Security (TLS) 。1TLS.2 が必要で、1.3 TLS をお勧めします。
- (Ephemeral Diffie-HellmanPFS) や DHE (Elliptic Curve Ephemeral Diffie-Hellman) などの完全前方秘匿性 ECDHE () を備えた暗号スイート。これらのモードは、Java 7 以降など、ほとんどの最新システムでサポートされています。

さらに、 リクエストは、 IAMプリンシパルに関連付けられたアクセスキー ID とシークレットアクセスキーを使用して署名する必要があります。または、AWS Security Token Service (AWS STS) を使用して、一時セキュリティ認証情報を生成し、リクエストに署名することもできます。

デフォルトでは、 AWS CodeStar はサービストラフィックを分離しません。を使用して作成されたプロジェクト AWS CodeStar は、Amazon 、APIGateway、または Elastic Beanstalk を介してアクセス設定を手動で変更しない限りEC2、パブリックインターネットで公開されます。これは意図的なものです。Amazon EC2、APIGateway、または Elastic Beanstalk のアクセス設定は、すべてのインターネットアクセスの防止を含め、必要な程度に変更できます。

AWS CodeStar では、デフォルトではVPCエンドポイント (AWS PrivateLink) をサポートしていませんが、プロジェクトリソースでそのサポートを直接設定できます。

AWS CodeStar における制限

次の表は、AWS CodeStar の制限を表します。AWS CodeStar は、プロジェクトリソースの他の AWS のサービスによって異なります。これらのサービスの制限を変更することができます。変更できる制限の詳細については、「AWS サービス制限」を参照してください。

プロジェクト数	AWS アカウントで最大 333 のプロジェクトまでです。実際の上限は、他のサービスの依存性のレベル (例: AWS アカウントで許可される CodePipeline 内のパイプラインの最大数) によって異なります。
IAM ユーザーが所属できる AWS CodeStar プロジェクトの数	個々の IAM ユーザーあたり最大 10 までです。
プロジェクト ID	プロジェクト ID は AWS アカウント内で一意である必要があります。プロジェクト ID は 2 文字以上にする必要があり、15 文字を超えることはできません。使用できる文字は次のとおりです。 a ~ z の文字。 0 ~ 9 の数字。 特殊文字 - (マイナス記号) 。 大文字、スペース、. (ピリオド) 、@ (アットマーク) 、_ (アンダースコア) などのその他の文字は許可されていません。
プロジェクト名	プロジェクト名の長さは 100 文字を超えることはできず、空白で開始または終了することはできません。
プロジェクトの説明	使用できる文字の組み合わせの長さは 0 ~ 1,024 文字です。プロジェクトの説明はオプションです。

AWS CodeStar プロジェクトのチームメンバー	100
ユーザープロファイルの表示名	使用できる文字の組み合わせの長さは 1 ~ 100 文字です。表示名は 1 文字以上にする必要があります。その文字にスペースは使用できません。表示名をスペースで開始または終了することはできません。
ユーザープロファイル内の E メールアドレス	E メールアドレスには @ が含まれ、有効なドメイン拡張で終わる必要があります。
フェデレーションアクセス、root アカウントへのアクセス、AWS CodeStar への一時アクセス	AWS CodeStar では、フェデレーティッドユーザー、および一時アクセスの認証情報の使用をサポートしていません。ルートアカウントで AWS CodeStar を使用することは推奨されません。
IAM ロール	IAM ロールに添付されている管理ポリシーでは、最大 5,120 文字です。

トラブルシューティング AWS CodeStar

以下の情報は、 AWS CodeStarでの一般的な問題のトラブルシューティングに役立ちます。

トピック

- プロジェクトの作成の失敗: プロジェクトが作成されませんでした
- プロジェクトの作成: プロジェクトの作成時に Amazon EC2 の設定を編集しようとするとエラーが発生します
- プロジェクトの削除: AWS CodeStar プロジェクトは削除されましたが、リソースはまだ存在します
- チーム管理の失敗: IAM ユーザーを AWS CodeStar プロジェクトのチームに追加できませんでした
- アクセス障害: フェデレーティッドユーザーが AWS CodeStar プロジェクトにアクセスできない
- アクセス障害: フェデレーティッドユーザーが AWS Cloud9 環境にアクセスしたり、環境を作成したりできない
- アクセスの失敗: フェデレーティッドユーザーは AWS CodeStar プロジェクトを作成できますが、プロジェクトリソースを表示できません
- サービスロールの問題: サービスロールを作成できませんでした
- サービスロールの問題: サービスロールが有効でないか、または存在しません
- プロジェクトロールの問題: AWS CodeStar プロジェクト内のインスタンスの AWS Elastic Beanstalk ヘルスステータスチェックが失敗する
- プロジェクトロールの問題: プロジェクトロールが有効でないか、または存在しません
- プロジェクトの拡張機能: JIRA に接続できません
- GitHub: リポジトリのコミット履歴、問題、またはコードにアクセスできない
- AWS CloudFormation: アクセス許可の不足により、スタックの作成がロールバックされた
- AWS CloudFormation は iam:PassRole on Lambda 実行ロールを実行する権限がありません
- GitHub リポジトリの接続を作成できません

プロジェクトの作成の失敗: プロジェクトが作成されませんでした

問題: プロジェクトを作成しようとすると、作成に失敗した旨のメッセージが表示されます。

解決方法: 失敗の最も一般的な原因は次のとおりです。

- その ID を持つプロジェクトは、 AWS アカウント、場合によっては別の AWS リージョンに既に存在します。
- へのサインインに使用した IAM ユーザーには、プロジェクトの作成に必要なアクセス許可 AWS Management Console がありません。
- AWS CodeStar サービスロールに必要なアクセス許可が 1 つ以上ありません。
- プロジェクトの 1 つ以上のリソースの上限 (IAM、Amazon S3 バケット、または のパイプラインのカスタマー管理ポリシーの制限など CodePipeline) に達しました。

プロジェクトを作成する前に、`AWSCodeStarFullAccess` ポリシーが IAM ユーザーに適用されていることを確認します。詳細については、「AWSCodeStarFullAccess ポリシー」を参照してください。

プロジェクトを作成するときは、ID が一意で、 AWS CodeStar 要件を満たしていることを確認します。リソースAWS CodeStar を管理するアクセス許可をユーザー AWS に代わって選択したことを確認してください。

その他の問題をトラブルシューティングするには、 AWS CloudFormation コンソールを開き、作成しようとしたプロジェクトのスタックを選択し、イベントタブを選択します。1 つのプロジェクトに複数のスタックが存在する可能性があります。スタック名は `awscodestar-` で始まり、プロジェクト ID が続きます。スタックは [Deleted] (削除済み) フィルタービューにある場合があります。スタックイベント内のすべての失敗メッセージを確認し、失敗の原因としてリストされている問題を修正します。

プロジェクトの作成: プロジェクトの作成時に Amazon EC2 の設定を編集しようとするとエラーが発生します

問題: プロジェクトの作成中に Amazon EC2 の設定オプションを編集すると、エラーメッセージまたはグレイアウトされたオプションが表示され、プロジェクトの作成を続行できません。

解決方法: エラーメッセージの最も一般的な原因は次のとおりです：

- AWS CodeStar プロジェクトテンプレートの VPC (デフォルト VPC または Amazon EC2 設定の編集時に使用された VPC) には、専用インスタンステナンシーがあり、専用インスタンスではインスタンスタイプはサポートされていません。別のインスタンスタイプ、または、別の Amazon VPC を選択します。
- AWS アカウントに Amazon VPCsがありません。デフォルトの VPC を削除し、他に作成していない。https://console.aws.amazon.com/vpc/ で、Amazon VPC コンソールを開き、[VPC] を選択

し、少なくとも 1 つの VPC が設定されていることを確認します。設定されていない場合は、作成します。詳細については、「Amazon VPC 入門ガイド」の「Amazon Virtual Private Cloud の概要」を参照してください。

- Amazon VPC にサブネットがない。別の VPC を選択するか、VPC のサブネットを作成します。詳細については、「VPC とサブネットの基本」を参照してください。

プロジェクトの削除: AWS CodeStar プロジェクトは削除されましたが、リソースはまだ存在します

問題： AWS CodeStar プロジェクトは削除されましたが、そのプロジェクト用に作成されたリソースはまだ存在します。デフォルトでは、プロジェクト AWS CodeStar が削除されると、 はプロジェクトリソースを削除します。バケットにデータが含まれている可能性があるため、ユーザーが [Delete resources] (リソースの削除) チェックボックスをオンにしても、一部のリソース (Amazon S3 バケットなど)は保持されます。

解決方法： AWS CloudFormation コンソールを開き、プロジェクトの作成に使用された 1 つ以上の AWS CloudFormation スタックを検索します。スタック名は `awscodestar-` で始まり、プロジェクト ID が続きます。スタックは [Deleted] (削除済み) フィルタービューにある場合があります。スタックに関連付けられたイベントを確認し、プロジェクトに作成されたリソースを検出します。AWS CodeStar プロジェクトを作成した AWS リージョン内の各リソースのコンソールを開き、リソースを手動で削除します。

残っているプロジェクトリソースには次のようなものが含まれている場合があります：

- Amazon S3 の 1 つ以上のプロジェクトバケット。他のプロジェクトリソースとは異なり、Amazon S3 のプロジェクトバケットは、 AWS CodeStar プロジェクトとともに関連付けられた AWS リソースを削除チェックボックスがオンになっている場合、削除されません。

  https://console.aws.amazon.com/s3/でAmazon S3 コンソールを開きます。
- のプロジェクトのソースリポジトリ CodeCommit。

  https://console.aws.amazon.com/codecommit/ で CodeCommit コンソールを開きます。
- でのプロジェクトのパイプライン CodePipeline。

  https://console.aws.amazon.com/codepipeline/ で CodePipeline コンソールを開きます。
- 内のアプリケーションおよび関連するデプロイグループ CodeDeploy。

https://console.aws.amazon.com/codedeploy/ で CodeDeploy コンソールを開きます。

- AWS Elastic Beanstalkのアプリケーションおよび関連する環境。

  https://console.aws.amazon.com/elasticbeanstalk/ で、Elastic Beanstalk コンソールを開きます。
- AWS Lambdaの関数。

  https://console.aws.amazon.com/lambda/ で AWS Lambda コンソールを開きます。
- API Gateway の 1 つ以上の API。

  https://console.aws.amazon.com/apigateway で、API Gateway コンソールを開きます。
- 1 つ以上の IAM ポリシーまたは IAM のロール。

  にサインイン AWS Management Console し、https://console.aws.amazon.com/iam/ で IAM コンソールを開きます。
- Amazon EC2 インスタンス。

  Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
- の 1 つ以上の開発環境 AWS Cloud9。

  開発環境を表示、アクセス、管理するには、https://console.aws.amazon.com/cloud9/ で AWS Cloud9 コンソールを開きます。

プロジェクトが 以外のリソース AWS （ 例えば、 GitHub リポジトリや Atlassian JIRA の問題) を使用している場合、 CodeStar プロジェクトとともに関連付けられたリソースを削除ボックスが選択されていても、それらの AWS リソースは削除されません。

チーム管理の失敗: IAM ユーザーを AWS CodeStar プロジェクトのチームに追加できませんでした

問題: プロジェクトにユーザーを追加しようとすると、追加に失敗したことを示すエラーメッセージが表示されます。

解決方法: このエラーの最も一般的な原因は、ユーザーが IAM でユーザーに適用できる管理ポリシーの制限に達していることです。また、ユーザーを追加しようとした AWS CodeStar プロジェクトに所有者ロールがない場合、または IAM ユーザーが存在しないか削除された場合にも、このエラーが表示されることがあります。

その AWS CodeStar プロジェクトの所有者であるユーザーとしてサインインしていることを確認します。詳細については、「AWS CodeStar プロジェクトにチームメンバーを追加する 」を参照してください。

他の問題のトラブルシューティングを行うには、IAM コンソールを開き、追加しようとしたユーザーを選択し、その IAM ユーザーに適用されている管理ポリシーの数を確認します。

詳細については、「IAM エンティティおよびオブジェクトの制限」を参照してください。変更できる制限の詳細については、「AWS サービスの制限」を参照してください。

アクセス障害: フェデレーティッドユーザーが AWS CodeStar プロジェクトにアクセスできない

問題： フェデレーティッドユーザーが AWS CodeStar コンソールでプロジェクトを表示できない。

解決方法: フェデレーティッドユーザーとしてサインインしている場合は、サインインを引き受けるロールに適切な管理ポリシーが添付されていることを確認します。詳細については、「プロジェクトの AWS CodeStar ビューワー/コントリビューター/所有者管理ポリシーをフェデレーティッドユーザーロールにアタッチする」を参照してください。

ポリシーを手動でアタッチして、フェデレーティッドユーザーを AWS Cloud9 環境に追加します。フェデレーティッドユーザーロールに AWS Cloud9 管理ポリシーをアタッチする を参照してください。

アクセス障害: フェデレーティッドユーザーが AWS Cloud9 環境にアクセスしたり、環境を作成したりできない

問題： フェデレーティッドユーザーが AWS Cloud9 コンソールで AWS Cloud9 環境を表示または作成できない。

解決方法: フェデレーティッドユーザーとしてサインインしている場合は、適切な管理ポリシーがフェデレーティッドユーザーのロールにアタッチされていることを確認します。

フェデレーティッドユーザーのロールにポリシーを手動でアタッチして、フェデレーティッドユーザーを AWS Cloud9 環境に追加します。フェデレーティッドユーザーロールに AWS Cloud9 管理ポリシーをアタッチする を参照してください。

アクセスの失敗: フェデレーティッドユーザーは AWS CodeStar プロジェクトを作成できますが、プロジェクトリソースを表示できません

問題: フェデレーティッドユーザーは、プロジェクトを作成できたが、プロジェクトパイプラインなどのプロジェクトリソースを表示できない。

解決方法： **AWSCodeStarFullAccess**管理ポリシーをアタッチしている場合は、 でプロジェクトを作成するアクセス許可があります AWS CodeStar。ただし、すべてのプロジェクトリソースにアクセスするには、所有者の管理ポリシーをアタッチする必要があります。

がプロジェクトリソース AWS CodeStar を作成すると、すべてのプロジェクトリソースへのプロジェクトアクセス許可が、所有者、寄稿者、および閲覧者管理ポリシーで利用可能になります。すべてのリソースにアクセスするには、所有者のポリシーをロールに手動でアタッチする必要があります。ステップ 3: ユーザーの IAM アクセス許可を設定する を参照してください。

サービスロールの問題: サービスロールを作成できませんでした

問題： でプロジェクトを作成しようとすると AWS CodeStar、サービスロールの作成を求めるメッセージが表示されます。作成するオプションを選択するとエラーが表示されます。

解決方法： このエラーの最も一般的な理由は、サービスロールを作成するのに十分なアクセス許可がないアカウント AWS で にサインインしていることです。 AWS CodeStar サービスロール (`aws-codestar-service-role`) を作成するには、管理ユーザーとして、またはルートアカウントでサインインする必要があります。コンソールからサインアウトし、`AdministratorAccess` 管理ポリシーが適用されている IAM ユーザーでサインインします。

サービスロールの問題: サービスロールが有効でないか、または存在しません

問題： AWS CodeStar コンソールを開くと、 AWS CodeStar サービスロールが欠落しているか、有効でないというメッセージが表示されます。

解決方法: このエラーの最も一般的な原因は、管理ユーザーがサービスロール (`aws-codestar-service-role`) を編集または削除したことです。サービスロールが削除された場合は、作成する

よう求められます。管理ユーザーとして、またはルートアカウントでサインインし、ロールを作成する必要があります。ロールが編集された場合は、もはや有効ではありません。管理ユーザーとして IAM コンソールにサインインし、ロールのリストでサービスロールを見つけて削除します。 AWS CodeStar コンソールに切り替え、指示に従ってサービスロールを作成します。

プロジェクトロールの問題: AWS CodeStar プロジェクト内のインスタンスの AWS Elastic Beanstalk ヘルスステータスチェックが失敗する

問題： 2017 年 9 月 22 日より前に Elastic Beanstalk を含む AWS CodeStar プロジェクトを作成した場合、Elastic Beanstalk のヘルスステータスチェックが失敗する可能性があります。プロジェクト作成後 Elastic Beanstalk 設定を変更していない場合、ヘルスステータスチェックは失敗し、灰色の状態がレポートされます。ヘルスチェックの失敗にもかかわらず、アプリケーションは正常に実行します。プロジェクト作成後、Elastic Beanstalk 設定を変更した場合は、ヘルスステータスチェックが失敗するだけでなく、アプリケーションは正常に実行されない可能性があります。

修正: 1 つ以上の IAM ロールで、必要な IAM ポリシーステートメントが不足しています。不足しているポリシーを AWS アカウントの影響のあるロールに追加します。

1. にサインイン AWS Management Console し、https://console.aws.amazon.com/iam/ で IAM コンソールを開きます。

   （これができない場合は、 AWS アカウント管理者にお問い合わせください）。
2. ナビゲーションペインで、[ロール] を選択します。
3. ロールのリストで、-CodeStarWorker***Project-ID*** -EB を選択します。ここで、*Project-ID* は影響を受けるプロジェクトの 1 つの ID です。(リストでロールを見つけるのが困難な場合は、ロールの名前の一部またはすべてを [検索] ボックスに入力します。)
4. [Permissions] (アクセス許可) タブで [Attach Policy] (ポリシーの添付) を選択します。
5. ポリシーのリストで、 AWSElasticBeanstalkEnhancedHealthと を選択しますAWSElasticBeanstalkService。(リストでポリシーを見つけるのが困難な場合は、ポリシーの名前の一部またはすべてを [検索] ボックスに入力します。)
6. [Attach Policy] (ポリシーの添付) を選択します。
7. -CodeStarWorker***Project-ID -EB ################################ 3 # 6*** を繰り返します。

プロジェクトロールの問題: プロジェクトロールが有効でないか、または存在しません

問題: プロジェクトにユーザーを追加しようとすると、プロジェクトロールのポリシーが存在しないか、または無効であるため、追加に失敗したことを示すエラーメッセージが表示されます。

解決方法: このエラーの最も一般的な原因は、1 つ以上のプロジェクトポリシーが編集または IAM から削除されたことです。プロジェクトポリシーは AWS CodeStar プロジェクトに固有であり、再作成することはできません。プロジェクトは使用できません。でプロジェクトを作成し AWS CodeStar、データを新しいプロジェクトに移行します。使用できないプロジェクトのリポジトリからプロジェクトコードをクローンし、そのコードを新しいプロジェクトのリポジトリにプッシュします。チームの wiki 情報を古いプロジェクトから新しいプロジェクトにコピーします。新しいプロジェクトにユーザーを追加します。すべてのデータと設定を移行したら、使用できないプロジェクトを削除します。

プロジェクトの拡張機能: JIRA に接続できません

問題： Atlassian JIRA 拡張機能を使用してプロジェクトを JIRA インスタンスに接続しようとする AWS CodeStar と、「URL は有効な JIRA URL ではありません。URL が正しいことを確認してください。」

解決方法:

- JIRA URL が正しいことを確認してから、再接続を試みます。
- お客様のセルフホスト型の JIRA インスタンスは、公開インターネットからアクセスできない可能性があります。ネットワーク管理者に連絡して、公共のインターネットから JIRA インスタンスにアクセスできることを確認してから、再度接続してみてください。

GitHub: リポジトリのコミット履歴、問題、またはコードにアクセスできない

問題： にコードを保存するプロジェクトのダッシュボードで GitHub、コミット履歴とGitHub問題タイルに接続エラーが表示されるか、これらのタイルで Open GitHub in または Create issue を選択するとエラーが表示されます。

考えられる原因:

- AWS CodeStar プロジェクトは GitHub リポジトリにアクセスできなくなる可能性があります。
- リポジトリが で削除または名前が変更された可能性があります GitHub。

AWS CloudFormation: アクセス許可の不足により、スタックの作成がロールバックされた

リソースを `template.yml` ファイルに追加したら、 AWS CloudFormation スタック更新を表示して、エラーメッセージがないか確認します。特定の条件が満たされない場合、スタック更新は失敗します (例：必要なリソースに対するアクセス許可が不足している)。

> **Note**
>
> 2019 年 5 月 2 日をもって、既存のすべてのプロジェクトの AWS CloudFormation ワーカーロールポリシーが更新されました。この更新により、プロジェクトパイプラインに付与されるアクセス権限の範囲が減り、プロジェクトのセキュリティが向上します。

トラブルシューティングを行うには、プロジェクトのパイプラインの AWS CodeStar ダッシュボードビューで障害ステータスを表示します。

次に、パイプラインのデプロイステージのCloudFormationリンクを選択して、 AWS CloudFormation コンソールで障害をトラブルシューティングします。スタック作成の詳細を表示するには、プロジェクトの [Events] (イベント) リストを展開し、障害メッセージがあれば表示します。このメッセージによって、不足しているアクセス許可が分かります。 AWS CloudFormation ワーカーロールポリシーを修正してから、パイプラインを再実行します。

AWS CloudFormation は iam:PassRole on Lambda 実行ロールを実行する権限がありません

2018 年 12 月 6 日より前に作成されたプロジェクトで Lambda 関数を作成する場合、次のような AWS CloudFormation エラーが表示されることがあります。

```
User: arn:aws:sts::id:assumed-role/CodeStarWorker-project-id-CloudFormation/
AWSCloudFormation is not authorized to perform: iam:PassRole on resource:
 arn:aws:iam::id:role/CodeStarWorker-project-id-Lambda (Service: AWSLambdaInternal;
 Status Code: 403; Error Code: AccessDeniedException; Request ID: id)
```

このエラーは、 AWS CloudFormation ワーカーロールに新しい Lambda 関数をプロビジョニングするためのロールを渡すアクセス許可がないために発生します。

このエラーを修正するには、 AWS CloudFormation ワーカーロールポリシーを次のスニペットで更新する必要があります。

```
{
        "Action":[ "iam:PassRole" ],
        "Resource": [
           "arn:aws:iam::account-id:role/CodeStarWorker-project-id-Lambda",
           ],

        "Effect": "Allow"
}
```

ポリシーを更新したら、パイプラインを再実行します。

または、「既存のプロジェクトにアクセスIAM許可の境界を追加する」で説明しているように、プロジェクトにアクセス許可の境界を追加して、Lambda 関数のカスタムロールを使用できます。

GitHub リポジトリの接続を作成できません

問題:

GitHub リポジトリへの接続は AWS Connector for を使用するため GitHub、接続を作成するには、リポジトリに対する組織所有者のアクセス許可または管理者のアクセス許可が必要です。

解決方法： GitHub リポジトリのアクセス許可レベルの詳細については、https://docs.github.com/en/free-pro-team@latest/github/setting-up-and-managing-organizations-and-teams/permission-levels-for-anorganization を参照してください。

AWS CodeStar ユーザーガイドリリースノート

次の表に、AWS CodeStar ユーザーガイドのリリース別の重要な変更点を示します。このドキュメントの更新に関する通知については、RSS フィードでサブスクライブできます。

変更	説明	日付
アクセスポリシーの更新	AWS CodeStar アクセスロールポリシーが更新されました。ポリシーの結果は同じですが、CloudFormation には DescribeStacks に加えて ListStacks が必要です。これはすでに必須になっています。更新されたポリシーを確認するには、「AWSCodeStarServiceRole ポリシー」を参照してください。	2023 年 3 月 24 日
[Service role policy updates] (サービスロールポリシーの更新)	AWS CodeStar サービスロールポリシーが更新されました。更新されたポリシーを参照するには、[AWSCodeStarServiceRolePolicy] (AWSCodeStar サービスロールポリシー) を参照してください。	2021 年 9 月 23 日
GitHub ソースリポジトリを持つプロジェクト用に接続リソースを使用する	コンソールを使用して GitHub リポジトリを使用するプロジェクトを AWS CodeStar で作成する場合、接続リソースを使用して GitHub アクションを管理します。接続は GitHub Apps を使用します。以前の GitHub 認可では OAuth を使用していました	2021 年 4 月 27 日

	。GitHub への接続を使用するプロジェクトを作成する方法を示すチュートリアルについては、[Tutorial: Create a Project with a GitHub Source Repository] (チュートリアル: GitHub ソースリポジトリを使用してプロジェクトを作成する) を参照してください。このチュートリアルでは、プロジェクトソースリポジトリのプルリクエストを作成、レビュー、マージする方法についても説明します。	
AWS CodeStar は米国西部 (北カリフォルニア) リージョンで AWS Cloud9 をサポートします	AWS CodeStar は、米国西部 (北カリフォルニア) リージョンで AWS Cloud9 の使用をサポートするようになりました。詳細については、[Setting up Cloud9] (Cloud9 の設定) を参照してください。	2021 年 2 月 16 日
新しいコンソールエクスペリエンスを反映するようにドキュメントを更新する	2020 年 8 月 12 日、AWS CodeStar サービスは、AWS コンソールにおける新しいユーザーエクスペリエンスに移行しました。新しいコンソールエクスペリエンスに合わせてユーザーガイドが更新されました。	2020 年 8 月 12 日

AWS CodeStar プロジェクトは AWS CodeStar CLI を使用して作成できます	AWS CodeStar プロジェクトは、CLI コマンドを使用して作成できます。AWS CodeStar では、指定したソースコードおよびツールチェーンテンプレートを使用してプロジェクトおよびインフラストラクチャを作成します。「AWS CodeStar でプロジェクトを作成する (AWS CLI))」を参照してください。	2018 年 10 月 24 日
すべての AWS CodeStar プロジェクトテンプレートにインフラストラクチャの更新用の AWS CloudFormation ファイルが含まれるようになりました	AWS CodeStar では、AWS CloudFormation と連携して、コードを使用してクラウド内にサポートサービスとサーバ、またはサーバレスプラットフォームを作成できるようにします。AWS CloudFormation ファイルは、すべての AWS CodeStar プロジェクトのテンプレートタイプ (Lambda、EC2、または Elastic Beanstalk コンピューティングプラットフォームを使用するテンプレート) で使用できるようになりました。ファイルは、ソースリポジトリの `template.yml` に保存されています。ファイルを表示および変更して、プロジェクトにリソースを追加できます。[Project Templates] (プロジェクトテンプレート) を参照してください。	2018 年 8 月 3 日

AWS CodeStarユーザーガイドの更新の通知が RSS を介して利用可能に	HTML 版の AWS CodeStar ユーザーガイドで、「ドキュメントの更新のリリースノート」ページに説明されている更新の RSS フィードがサポートされるようになりました。RSS フィードには、2018 年 6 月 30 日以降に行われた更新が含まれています。以前に発表された更新は、「ドキュメントの更新のリリースノート」ページで引き続き利用できます。このフィードをサブスクライブするには、トップメニューパネルの RSS ボタンを使用します。	2018 年 6 月 30 日

次の表に、2018 年 6 月 30 日以前の AWS CodeStar ユーザーガイドの各リリースにおける重要な変更点を示します。

変更	説明	変更日
GitHub で AWS CodeStar ユーザーガイドを入手できるようになりました。	このガイドが GitHub で利用可能になりました。GitHub を使用して、フィードバックの送信およびこのガイドの内容に対する変更リクエストの送信もできます。詳細については、ガイドのナビゲーションバーの [Edit on GitHub] (GitHub で編集) アイコンを選択するか、GitHub ウェブサイトで awsdocs/aws-codestar-user-guide リポジトリを参照してください。	2018 年 2 月 22 日
AWS CodeStar がアジアパシフィック (ソウル) で利用可能に	AWS CodeStar が、アジアパシフィック (ソウル) リージョンで使用できるようになりました。詳細については、「Amazon Web Services 全般のリファレンス」の「AWS CodeStar」を参照してください。	2018 年 2 月 14 日

変更	説明	変更日
AWS CodeStar がアジアパシフィック (東京) およびカナダ (中部) で利用可能に	AWS CodeStar が、アジアパシフィック (東京) リージョンと カナダ (中部) リージョンで利用できるようになりました。詳細については、「Amazon Web Services 全般のリファレンス」の「AWS CodeStar」を参照してください。	2017 年 12 月 20 日
AWS CodeStar で AWS Cloud9 がサポートされるようになりました	AWS CodeStar では、ウェブブラウザベースのオンライン IDE AWS Cloud9 を使用してプロジェクトコードを操作できるようになりました。詳細については、「AWS CodeStar で AWS Cloud9 を使用する」を参照してください。 サポートされている AWS リージョンのリストについては、「Amazon Web Services 全般のリファレンス」の「AWS Cloud9」を参照してください。	2017 年 11 月 30 日
AWS CodeStar で GitHub がサポートされるようになりました	AWS CodeStar がプロジェクトコードを GitHub に保存する機能をサポートしました。詳細については、[Create a Project] (プロジェクトの作成) を参照してください。	2017 年 10 月 12 日
AWS CodeStar が米国西部 (北カリフォルニア)、欧州 (ロンドン)で利用可能に	AWS CodeStar が、米国西部 (北カリフォルニア)リージョン、欧州 (ロンドン) リージョンで利用できるようになりました。詳細については、「Amazon Web Services 全般のリファレンス」の「AWS CodeStar」を参照してください。	2017 年 8 月 17 日
AWS CodeStar がアジアパシフィック (シドニー)、アジアパシフィック (シンガポール)、欧州 (フランクフルト)で利用可能に	アジアパシフィック (シドニー)、アジアパシフィック (シンガポール)、欧州 (フランクフルト) の各リージョンで AWS CodeStar が利用できるようになりました。詳細については、「Amazon Web Services 全般のリファレンス」の「AWS CodeStar」を参照してください。	2017 年 7 月 25 日

変更	説明	変更日
AWS CloudTrail で AWS CodeStar がサポートされるようになりました	AWS CodeStar は、CloudTrail に統合されました。これは、AWS アカウントで AWS CodeStar によりまたは代理で行われた API コールをキャプチャし、指定した Amazon S3 バケットにログファイルを配信するサービスです。詳細については、「AWS CloudTrail を使用した AWS CodeStar API コールの口グ記録」を参照してください。	2017 年 6 月 14 日
初回リリース	これは AWS CodeStar ユーザーガイドの最初のリリースです。	2017 年 4 月 19 日

AWS 用語集

AWS の最新の用語については、「AWS の用語集リファレンス」の「AWS 用語集」を参照してください。