aws

開発者ガイド

Amazon Pinpoint

Amazon Pinpoint: 開発者ガイド

Table of Contents

Amazon Pinpoint とは

Amazon Pinpoint は、複数のメッセージングチャンネルをまたがってお客様とやり取りするための AWS サービスです。Amazon Pinpoint を使用して、プッシュ通知、E メール、SMS テキストメッセージ、または音声メッセージを送信できます。

この開発者ガイドの情報は、アプリケーション開発者を対象にしています。このガイドには、Amazon Pinpoint の機能をプログラムとして使用することに関する情報が含まれています。また、「アプリケーションに分析と測定機能を統合する」ための手順など、モバイルアプリ開発者に特に有益な情報も含まれています。

このドキュメントに付随する他の文書がいくつかあります。次のドキュメントは、Amazon Pinpoint API に関連するリファレンス情報を提供します。

- Amazon Pinpoint API リファレンス
- Amazon Pinpoint SMS および音声 API

Amazon Pinpoint を初めて使用する方は、『Amazon Pinpoint ユーザーガイド』を参照してからこのドキュメントを読み進めるようにしてください。

Amazon Pinpoint の機能

このセクションでは、Amazon Pinpoint の主な機能とそれらを使用して実行できるタスクについて説明します。

対象者セグメントの定義

対象者セグメントを定義することで、正しい対象者にメッセージを届けます。セグメントは、キャンペーンから送信されるメッセージを受信するユーザーを指定します。オペレーティングシステムやモバイルデバイスタイプなど、アプリケーションによって報告されたデータに基づいて動的なセグメントを定義できます。他のサービスまたはアプリケーションを使用して定義した静的なセグメントをインポートすることもできます。

メッセージングキャンペーンで対象者を取り込む

メッセージングキャンペーンを作成して、対象者を取り込みます。キャンペーンは、カスタマイズされたメッセージを定義したスケジュールで送信します。モバイルプッシュ、E メール、または SMS メッセージを送信するキャンペーンを作成できます。

別のキャンペーン戦略を試すには、キャンペーンを A/B テストとして設定し、結果を Amazon Pinpoint 分析で分析します。

トランザクションメッセージの送信

新しいアカウントのアクティベーションメッセージ、注文確認、パスワードリセット通知といったトランザクションモバイルプッシュや SMS メッセージを特定のユーザーに直接送信することで、お客様に情報を提供しましょう。Amazon Pinpoint REST API を使用してトランザクションメッセージを送信できます。

ユーザー行動の分析

Amazon Pinpoint で提供された分析を使用して、対象者およびキャンペーンの効果に関する洞察を得られます。ユーザーの取り組みレベル、購入アクティビティ、デモグラフィックに関する傾向を表示できます。キャンペーンやアプリケーションに対して送信されたまたは開かれたメッセージの総数などのメトリクスを表示して、メッセージトラフィックを監視することもできます。Amazon Pinpoint APIを通して、アプリケーションは Amazon Pinpoint が分析に利用できるようになる、カスタムデータを報告することができます。また特定の標準的なメトリクスについて分析データにクエリを実行できます。

Amazon Pinpoint の外でイベントデータを分析および保存するには、Amazon Kinesis にデータをストリーミングするよう Amazon Pinpoint を設定します。

リージョナルな可用性

Amazon Pinpoint は、北米、ヨーロッパ、アジア、およびオセアニアにある複数の AWS リージョンで利用できます。各地域で、AWS は複数のアベイラビリティーゾーンを維持しています。これらのアベイラビリティーゾーンは物理的に相互に分離されていますが、低レイテンシーで高スループットの冗長性に優れたプライベートネットワーク接続で統合されています。これらのアベイラビリティーゾーンにより、非常に高いレベルの可用性と冗長性を提供しながら、レイテンシーを最小限に抑えています。

AWS リージョン の詳細については、「Amazon Web Services 全般のリファレンス」の「AWS リージョン の管理」を参照してください。Amazon Pinpoint が現在利用可能なすべてのリージョンの一覧については、「Amazon Web Services 全般のリファレンス」の「Amazon Pinpoint エンドポイントとクォータ」と「AWS サービスエンドポイント」を参照してください。各リージョンで利用できるアベイラビリティーゾーンの数の詳細については、「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。

チュートリアル

このセクションのチュートリアルは、いくつかの重要なタスクを完了する方法を、新規の Amazon Pinpoint ユーザーに説明することを目的としています。Amazon Pinpoint を初めて使用する場合、または特定の機能についてまだよくわからない場合、これらのチュートリアルから始めることをお勧めします。

このガイドのチュートリアルには、開発者やシステム管理者を対象として作られたタスクが含まれています。これらのチュートリアルでは、Amazon Pinpoint API、AWS SDK、AWS CLI を使用してタスクを実行する方法が示されます。主にウェブベースのコンソールを使用して Amazon Pinpoint を操作する場合、『Amazon Pinpoint ユーザーガイド』の「チュートリアル」セクションを参照してください。

チュートリアル

- チュートリアル: Amazon Pinpoint API で Postman を使用する
- チュートリアル: SMS 登録システムの設定

チュートリアル: Amazon Pinpoint API で Postman を使用する

Postman は使いやすいグラフィカル環境で API をテストする、人気の高いツールです。Postman を使用して、API リクエストを任意の REST API へ送信し、リクエストに対する応答を受け取ることができます。Postman を使用するのは、Amazon Pinpoint API への呼び出しをテストおよびトラブルシューティングするのに便利な方法です。このチュートリアルには、Amazon Pinpoint で Postman を設定し、使用する手順が含まれています。

> **Note**
>
> Postman は、サードパーティー企業によって開発されています。Amazon Web Services (AWS) が開発およびサポートするものではありません。Postman の使用方法または Postman に関連する問題のサポートの詳細については、Postman ウェブサイトでサポートセンターを参照してください。

このチュートリアルの内容

このセクションでは、このチュートリアルの概要について説明します。

対象者

このチュートリアルは、開発者とシステム実装者を対象としています。このチュートリアルの手順を完了するために Amazon Pinpoint または Postman に精通している必要はありません。IAM ポリシーの管理および JSON のコード例の変更に慣れている必要があります。

このチュートリアルの手順は、新しいユーザーが、Amazon Pinpoint リソースを完全に削除する可能性のある API オペレーションを使用できないように設計されました。上級ユーザーは、 ユーザーに関連付けられているポリシーを変更して、この制限を解除できます。

使用される機能

このチュートリアルには、次の Amazon Pinpoint 機能の使用例が含まれています。

- Postman を使用して Amazon Pinpoint APIと対話する

所要時間

このチュートリアルは完了までに約 15 分かかります。

リージョン別制限

このソリューションの使用に関連するリージョン別制限はありません。

リソース使用量のコスト

AWS アカウントを作成するための料金はかかりません。ただし、Postman を使用して以下のいずれかの操作を実行する場合は、このソリューションを実装することによって、AWS 使用料金が発生する可能性があります。

- E メール、SMS、モバイルプッシュ、または音声メッセージを送信する
- キャンペーンを作成して送信する
- 電話番号検証機能を使用する

Amazon Pinpoint の使用に関連する料金の詳細については、「Amazon Pinpoint pricing」を参照してください。

前提条件

このチュートリアルを開始する前に、次の前提条件を完了してください。

- AWS アカウントが必要です。AWS アカウントを作成するには、https://console.aws.amazon.com/から、[Create a new AWS account] を選択します。
- AWS Management Console へのサインインに使用するアカウントで新しい IAM ポリシーおよびロールを作成できることを確認します。
- 作成した少なくとも 1 つのサンプルプロジェクトで E メールが有効になっており、検証済みの E メール ID があることを確認します。「Amazon Pinpoint ユーザーガイド」の「E メールをサポートする Amazon Pinpoint プロジェクトの作成」を参照してください。
- AWS アカウント ID を持っていることを確認します。AWS アカウント ID はコンソールの右上に表示されます。また、コマンドラインインターフェイス (CLI) を使用することもできます。「AWS アカウント ID の検索」を参照してください。
- Postman をコンピュータにダウンロードしてインストールする必要があります。Postman は、Postman ウェブサイトからダウンロードできます。
- コンピュータに Postman をインストールしたら、Postman アカウントを作成します。Postman アプリケーションを最初に起動するときは、ログインするか、新しいアカウントを作成するように求められます。Postman の指示に従ってアカウントにログインします。まだアカウントを持っていない場合は作成します。

ステップ 1: IAM ポリシーおよびロールを作成する

Postman を使用して Amazon Pinpoint API をテストするとき、最初のステップはユーザーを作成することです。このセクションでは、すべての Amazon Pinpoint リソースの操作をユーザーに許可するポリシーを作成します。その後、ユーザーを作成し、そのユーザーにポリシーを直接アタッチします。

IAM ポリシーを作成する

IAM ポリシーの作成方法について説明します。このポリシーを使用するユーザーとロールは、Amazon Pinpoint API ですべてのリソースを操作できます。また、Amazon Pinpoint Email API、Amazon Pinpoint SMS and Voice API に関連するリソースへのアクセスも提供します。

ポリシーを作成するには

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。
2. ナビゲーションペインで ポリシーを選択してから ポリシーの作成を選択します。

3. ポリシーエディタで JSON を選択します。ポリシーエディタで最新の JSON を削除して、空白にします。次の JSON をコピーしてポリシーエディタに貼り付け、ポリシーエディタで *123456789012* のすべてのインスタンスを AWS アカウント ID に置き換えます。

   AWS アカウント ID はコンソールの右上隅にあるか、CLI を使用できます。AWS 「アカウント ID の検索」を参照してください。

   > **Note**
   >
   > Amazon Pinpoint アカウント内のデータを保護するため、このポリシーには、リソースの読み取り、作成、変更ができるアクセス許可のみが含まれています。リソースの削除を許可するアクセス許可はこのポリシーに含まれていません。IAM コンソールのビジュアルエディタを使用してポリシーを変更できます。IAM ポリシーの詳細については、『IAM ユーザーガイド』の「Managing IAM policies」を参照してください。IAM API の CreatePolicyVersionオペレーションを使用して、このポリシーを更新することもできます。
   >
   > また、このポリシーには、mobiletargeting サービスに加えて、ses サービスや sms-voice サービスを操作するためのアクセス許可も含まれています。ses および sms-voice のアクセス許可により、それぞれ Amazon Pinpoint Email API および Amazon Pinpoint SMS and Voice API を利用することができます。mobiletargeting アクセス許可では、Amazon Pinpoint API を操作することができます。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "mobiletargeting:Update*",
                "mobiletargeting:Get*",
                "mobiletargeting:Send*",
                "mobiletargeting:Put*",
                "mobiletargeting:Create*"
            ],
            "Resource": [
                "arn:aws:mobiletargeting:*:123456789012:apps/*",
                "arn:aws:mobiletargeting:*:123456789012:apps/*/campaigns/*",
```

```
                "arn:aws:mobiletargeting:*:123456789012:apps/*/segments/*"
            ]
        },
        {
            "Sid": "VisualEditor1",
            "Effect": "Allow",
            "Action": [
                "mobiletargeting:TagResource",
                "mobiletargeting:PhoneNumberValidate",
                "mobiletargeting:ListTagsForResource",
                "mobiletargeting:CreateApp"
            ],
            "Resource": "arn:aws:mobiletargeting:*:123456789012:*"
        },
        {
            "Sid": "VisualEditor2",
            "Effect": "Allow",
            "Action": [
                "ses:TagResource",
                "ses:Send*",
                "ses:Create*",
                "ses:Get*",
                "ses:List*",
                "ses:Put*",
                "ses:Update*",
                "sms-voice:SendVoiceMessage",
                "sms-voice:List*",
                "sms-voice:Create*",
                "sms-voice:Get*",
                "sms-voice:Update*"
            ],
            "Resource": "*"
        }
    ]
}
```

[次へ] をクリックします。

4. ポリシー名 には、 など、ポリシーの名前を入力します**PostmanAccessPolicy**。[ポリシーの作成] を選択します。
5. (オプション) ポリシーにタグを追加するには、[タグの追加] を選択します。
6. [次へ: レビュー] を選択します。

IAM ユーザーの作成

> **Warning**
>
> IAM ユーザーには長期的な認証情報があり、セキュリティ上のリスクがあります。このリスクを軽減するために、これらのユーザーにはタスクの実行に必要な権限のみを付与し、不要になったユーザーを削除することをお勧めします。

ポリシーを作成したら、ユーザーを作成してポリシーをアタッチできます。ユーザーを作成するとき、IAM では、Postman に Amazon Pinpoint API オペレーションの実行を許可する一連の認証情報が提供されます。

ユーザーを作成するには

1. IAM コンソール (https://console.aws.amazon.com/iam/) を開きます。
2. IAM コンソールのナビゲーションペインで、ユーザー を選択し、ユーザーの作成 を選択します。
3. [ユーザーの詳細] の [ユーザー名] に、**PostmanUser** など、ユーザーを識別する名前を入力します。次いで、[次へ] を選択します。
4. [許可を設定] ページの [許可のオプション] で、[ポリシーを直接アタッチする] を選択します。
5. アクセス許可ポリシー で、「IAM ポリシーの作成」で作成したポリシー (**PostmanAccessPolicy**) を選択します。次いで、[次へ] を選択します。
6. [確認して作成] ページで、必要に応じて、ユーザーを識別するのに役立つタグを追加します。タグの使用に関する詳細については、「IAM ユーザーガイド」の「IAM リソースのタグ付け」を参照してください。
7. ユーザーを作成する準備ができたら、[ユーザーの作成] を選択します。

アクセスキーを作成する

> **Warning**
>
> このシナリオでは、プログラムによるアクセスと長期的な認証情報を持つ IAM ユーザーが必要です。これはセキュリティ上のリスクをもたらします。このリスクを軽減するために、これらのユーザーにはタスクの実行に必要な権限のみを付与し、不要になったユーザーを削除

することをお勧めします。アクセスキーは、必要に応じて更新できます。詳細については、「IAM ユーザーガイド」の「アクセスキーの更新」を参照してください。

IAM では、Postman に Amazon Pinpoint API オペレーションの実行を許可するために使用できる一連の認証情報が提供されます。

ユーザーを作成するには

1. IAM コンソール (https://console.aws.amazon.com/iam/) を開きます。
2. IAM コンソールのナビゲーションペインで [ユーザー] を選択します。「IAM ユーザーの作成」で作成したユーザー (**PostmanUser**) を選択し、「セキュリティ認証情報」タブを選択します。
3. [Access keys (アクセスキー)] セクションで、[Create access key (アクセスキーを作成)] を選択します。
4. アクセスキーのベストプラクティスと代替ページで、 の外部で実行されているアプリケーション AWSを選択します。

   次いで、[次へ] を選択します。
5. (オプション) ポリシーには説明タグを追加できます。
6. [Create access key] (アクセスキーの作成) を選択します。
7. [アクセスキーを取得] ページで、[アクセスキー] 列と [シークレットアクセスキー] 列に表示されている認証情報をコピーします。

   > **Note**
   >
   > このチュートリアルの後半で、アクセスキー ID とシークレットアクセスキーの両方を入力する必要があります。シークレットアクセスキーを表示できるのは、このときのみとなります。コピーして安全な場所に保存することをお勧めします。
8. 両方のキーを保存したら、[完了] を選択します。

ステップ 2: Postman をセットアップする

Amazon Pinpoint API にアクセスできるユーザーアカウントを作成したので、Postman をセットアップできます。このセクションでは、Postman で 1 つ以上の環境を作成します。次に、Amazon Pinpoint API の各オペレーションのリクエストテンプレートが含まれているコレクションをインポートします。

Postman ワークスペースを作成する

Postman において、ワークスペースはプロジェクトと環境用の組織のコンテナです。このセクションでは、Amazon Pinpoint で使用するワークスペースを少なくとも 1 つ作成します。

ワークスペースの作成

Postman で、その他のアクションを選択し、[ファイル]、[新規] の順に選択します。

1. [新規作成] ウィンドウで [ワークスペース] を選択します。
2. 名前と概要を入力し、[可視性] を [個人] に設定します。次に、[ワークスペースの作成] を選択します。

Postman 環境を作成する

Postman では、環境とは、キーと値のペアとして保存される一連の変数です。環境を使用すると、Postman を介してリクエストの設定を変更でき、API リクエスト自体を変更する必要はありません。

このセクションでは、Amazon Pinpoint で使用するには、少なくとも 1 つの環境を作成します。作成する各環境には、1 つの AWS リージョンのアカウントに固有の変数のセットが含まれています。このセクションの手順を使用して複数の環境を作成する場合、Postman で [環境] メニューから別の環境を選択して、リージョン間で変更できます。

環境を作成するには

1. Postman で、その他のアクションメニューを選択し、[ファイル]、[新規] の順に選択します。
2. [新規作成] ウィンドウで、[環境] を選択します。
3. [MANAGE ENVIRONMENTS] ウィンドウで、[環境名] に「**Amazon Pinpoint - *Region Name***」と入力します。*Region Name* を次のいずれかのリージョンに置き換えます。

   - 米国東部 (バージニア北部)
   - 米国西部 (オレゴン)
   - アジアパシフィック (ムンバイ)
   - アジアパシフィック (シドニー)
   - 欧州 (フランクフルト)
   - 欧州 (アイルランド)

> **Note**
>
> 少なくとも、1 つの に対して 1 つの環境を作成するだけで AWS リージョン、1 つのプロジェクトを含める AWS リージョン 必要があります。前述の のいずれかでプロジェクトを作成していない場合は AWS リージョン、Amazon Pinpoint ユーザーガイド」の「Eメールをサポートする Amazon Pinpoint プロジェクトの作成 Amazon Pinpoint」を参照してください。

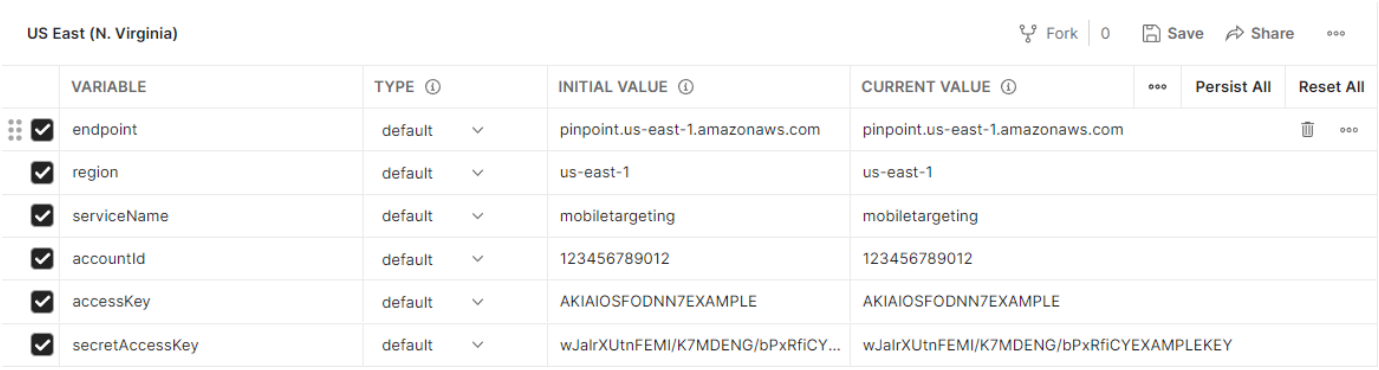
4. 6 つの新しい変数、`endpoint`、`region`、`serviceName`、`accountId`、`accessKey`、および `secretAccessKey` を作成します。次の表を使用して、各変数の [初期値] 列と [現在の値] に入力する値を決定します。

リージョン	変数	初期値と現在の値
米国東部 (バージニア北部)	`endpoint`	**`pinpoint.us-east-1.amazonaws.com`**
	`region`	**`us-east-1`**
	`serviceName`	**`mobiletargeting`**
	`accountId`	（ AWS アカウント ID)
	`accessKey`	(お客様の IAM アクセスキー ID)
	`secretAccessKey`	(お客様の IAM シークレットアクセスキー)
米国西部 (オレゴン)	`endpoint`	**`pinpoint.us-west-2.amazonaws.com`**
	`region`	**`us-west-2`**
	`serviceName`	**`mobiletargeting`**

リージョン	変数	初期値と現在の値
	accountId	（ AWS アカウント ID)
	accessKey	(お客様の IAM アクセスキー ID)
	secretAccessKey	(お客様の IAM シークレットアクセスキー)
アジアパシフィック (ムンバイ)	endpoint	**pinpoint.ap-south-1.amazonaws.com**
	region	**ap-south-1**
	serviceName	**mobiletargeting**
	accountId	（ AWS アカウント ID)
	accessKey	(お客様の IAM アクセスキー ID)
	secretAccessKey	(お客様の IAM シークレットアクセスキー)
アジアパシフィック (シドニー)	endpoint	**pinpoint.ap-southeast-2.amazonaws.com**
	region	**ap-southeast-2**
	serviceName	**mobiletargeting**
	accountId	（ AWS アカウント ID)
	accessKey	(お客様の IAM アクセスキー ID)

リージョン	変数	初期値と現在の値
	secretAccessKey	(お客様の IAM シークレットアクセスキー)
欧州 (フランクフルト)	endpoint	**pinpoint.eu-central-1.amazonaws.com**
	region	**eu-central-1**
	serviceName	**mobiletargeting**
	accountId	（ AWS アカウント ID)
	accessKey	(お客様の IAM アクセスキー ID)
	secretAccessKey	(お客様の IAM シークレットアクセスキー)
欧州 (アイルランド)	endpoint	**pinpoint.eu-west-1.amazonaws.com**
	region	**eu-west-1**
	serviceName	**mobiletargeting**
	accountId	（ AWS アカウント ID)
	accessKey	(お客様の IAM アクセスキー ID)
	secretAccessKey	(お客様の IAM シークレットアクセスキー)

これらの変数を作成すると、[MANAGE ENVIRONMENTS] ウィンドウは、次の図に示す例のようになります。



完了したら、保存 を選択します。

> **Important**
>
> 先ほどの画像に示されているアクセスキーは架空のものです。IAM アクセスキーは他のユーザーと共有してはいけません。
> Postman には、環境の共有とエクスポートのための機能があります。これらの機能を使用する場合、これらの認証情報にアクセスすべきではない人と、アクセスキー ID やシークレットアクセスキーを共有しないでください。
> 詳細については、『IAM ユーザーガイド』の「IAM ベストプラクティス」を参照してください。

5. (オプション) 作成する環境を追加するごとに、手順 1～4 を繰り返します。

> **Tip**
>
> Postman では、必要な数だけ環境を作成することができます。環境は次の方法で使用できます。
>
> - Amazon Pinpoint API をテストする必要があるリージョンごとに個別の環境を作成します。
> - 異なる AWS アカウントアカウントに関連付けられた環境を作成します。
> - 他のユーザーに関連付けられている認証情報を使用する環境を作成します。

6. 環境の作成が完了したら、次のセクションに進みます。

Postman で Amazon Pinpoint コレクションを作成する

Postman では、コレクション は API リクエストのグループです。コレクション内のリクエストは通常、共通の目的で結合されます。このセクションでは、Amazon Pinpoint API での各オペレーションに対するリクエストテンプレートを含む新しいコレクションを作成します。

Amazon Pinpoint コレクションを作成する

1. Postman で、その他のアクションメニューを選択し、[ファイル] を選択し、[インポート] を選択します。
2. インポートウィンドウで、リンクからインポート を選択し、次の URL を入力します: https://raw.githubusercontent.com/awsdocs/amazon-pinpoint-developer-guide/master/Amazon%20Pinpoint.postman_collection.json。

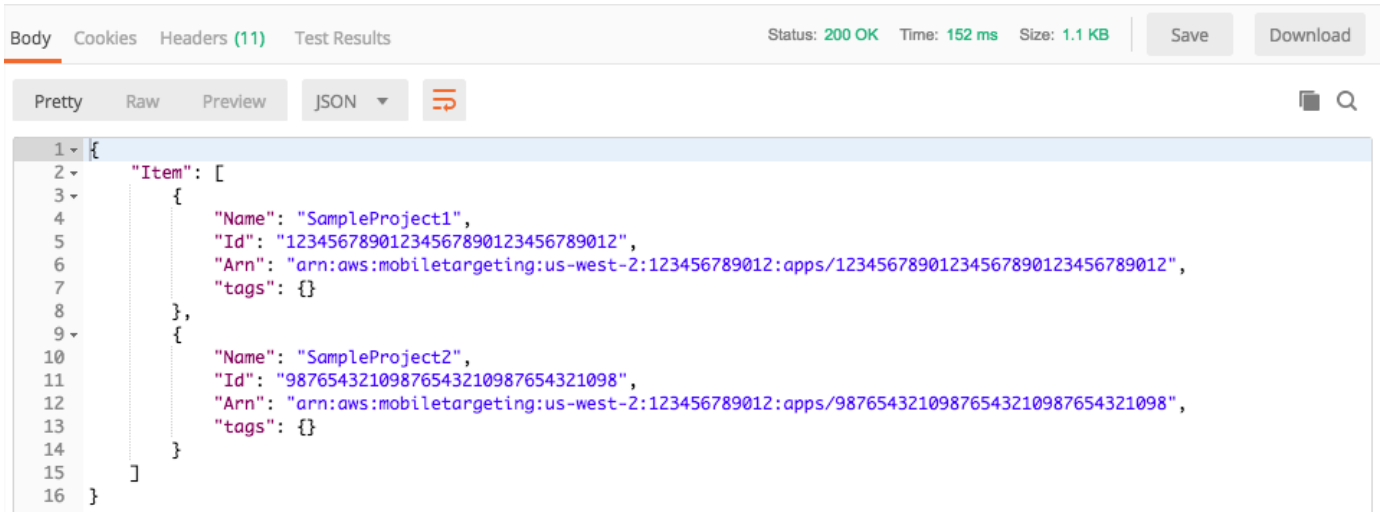
   Import (インポート) を選択します。Postman は、120 のサンプルリクエストが含まれている Amazon Pinpoint コレクションをインポートします。

Postman 設定のテスト

Amazon Pinpoint コレクションをインポートした後、簡単なテストを実行して、すべてのコンポーネントが適切に設定されていることを検証することをお勧めします。`GetApps` リクエストを送信することで、設定をテストできます。このリクエストでは、現在の AWS リージョンで自分の Amazon Pinpoint アカウントに存在するすべてのプロジェクトのリストが返されます。このリクエストは、他に追加の設定を必要としないため、設定をテストする方法に適しています。

Amazon Pinpoint コレクションの設定をテストするには

1. ナビゲーションペインで、[Collections] を選択し、Amazon Pinpoint コレクションを展開してから、Apps フォルダを展開します。
2. リクエストのリストで、 を選択しますGetApps。
3. 環境セレクタを使用して、Postman 環境の作成 で作成した環境を選択します。
4. [Send] を選択します。リクエストが正常に送信されると、[Response] ペインに `200 OK` のステータスが表示されます。次のイメージの例のようなレスポンスが表示されます。



> **Note**
>
> にプロジェクトが作成されていない場合 AWS リージョン 、Amazon Pinpoint は を返します`{ "Item": [] }`。

このレスポンスには、「ステップ 3」で選択したリージョンにある自分のアカウントに存在するすべての Amazon Pinpoint プロジェクトの一覧が表示されます。

トラブルシューティング

リクエストを送信すると、エラーが表示される可能性があります。発生する可能性のあるいくつかの一般的なエラーと問題を解決するための手順については、以下のリストをご覧ください。

エラーメッセージ	問題	解決方法
何もレスポンスが得られませんでした https://%7B%7Bendpoint%7D%7D/v1/apps に接続中にエラーが発生しました。	環境を選択したときに設定される、`{{endpoint}}` 変数の現在の値がありません。	環境セレクタを使用して環境を選択します。
リクエストに含まれているセキュリティトークンが無効です。	Postman では、アクセスキー ID またはシークレットアクセ	環境セレクタの近くにある歯車アイコンを選択し、現在の環境を選択し

エラーメッセージ	問題	解決方法
	スキーの現在の値を見つけることができませんでした。	ます。`accessKey` および `secretAccessKey` の値が [INITIAL VALUE] と [CURRENT VALUE] の両方の列に表示されること、および認証情報が正しく入力されていることを確認します。
「メッセージ」：「User: arn:aws:iam::123456789012:user/PinpointPostmanUser is not authorized to perform: mobiletargeting:GetApps on resource: arn:aws:mobiletargeting:us-west-2:123456789012:*」	ユーザーに関連付けられている IAM ポリシーに、適切なアクセス許可が含まれていません。	IAM ポリシーの作成 で説明されているアクセス許可がユーザーに付与されていること、および Postman ワークスペースの作成 で環境を作成したときに正しい認証情報を指定したことを確認します。

ステップ 3: 追加のリクエストを送信する

Postman の設定とテストが完了したら、Amazon Pinpoint API に追加のリクエストの送信を開始することができます。このセクションには、リクエストの送信を開始する前に知っておく必要がある情報が記載されています。また、Amazon Pinpoint コレクションの使用方法の説明に役立つリクエスト例も 2 つ含まれています。

> **Important**
>
> このセクションの手順を完了したら、Amazon Pinpoint API へのリクエストを送信します。これらのリクエストでは、Amazon Pinpoint アカウントでの新しいリソースの作成、既存のリソースの変更、メッセージの送信、Amazon Pinpoint プロジェクトの設定の変更、その他の Amazon Pinpoint 機能の使用を行います。これらのリクエストを実行するときには注意が必要です。

Amazon Pinpoint Postman コレクションの例について

Amazon Pinpoint Postman コレクションのほとんどのオペレーションは、使用する前に設定する必要があります。GET と DELETE オペレーションの場合、通常は、[Pre-request Script] タブで設定されている変数を変更するだけです。

> **Note**
>
> 「IAM ポリシーの作成」に示されている IAM ポリシーを使用する場合、このコレクションに含まれるDELETEリクエストを実行することはできません。

例えば、GetCampaign オペレーションでは、projectId と campaignId を指定する必要があります。[Pre-request Script] タブには、これら両方の変が存在し、値の例が設定されています。値の例を削除し、Amazon Pinpoint プロジェクトとキャンペーンに該当する値に置き換えます。

これらの変数のうち、最も一般的に使用されるのは projectId 変数です。この変数の値は、リクエストが適用されるプロジェクトの一意の識別子であることが必要です。プロジェクトの識別子のリストを取得するには、このチュートリアルの前のステップで送信した GetApps リクエストに対するレスポンスを参照してください。そのレスポンスで、Id フィールドにプロジェクトの一意の識別子が表示されています。GetApps オペレーションおよびレスポンスの各フィールドの意味の詳細については、「Amazon Pinpoint APIリファレンス」の「Apps」を参照してください。

> **Note**
>
> Amazon Pinpoint では、「プロジェクト」は「アプリ」や「アプリアプリケーション」と同じです。

POST オペレーションと PUT オペレーションの場合、リクエスト本文を変更して、API に送信する値が含まれるようにする必要もあります。例えば、POST リクエストである CreateApp リクエストを送信するときには、作成するプロジェクトの名前を指定する必要があります。[本文] タブで、リクエストを変更することができます。この例では、"Name" の横にある値をプロジェクトの名前に置き換えます。プロジェクトにタグを追加するには、tags オブジェクトで指定します。または、タグを追加しない場合は、tags オブジェクト全体を削除することができます。

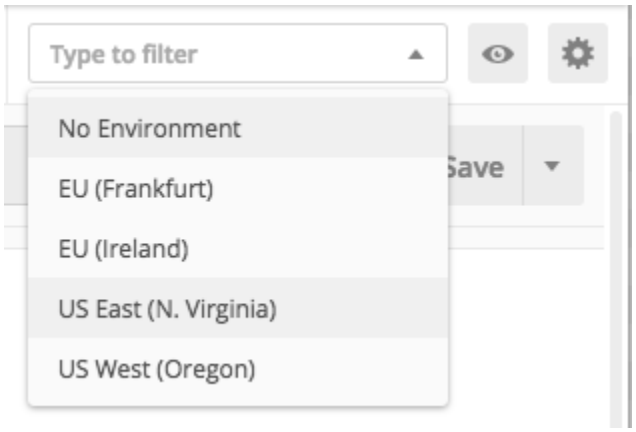
> **Note**
>
> `UntagResource` オペレーションでは、URL パラメータを指定する必要もあります。これらのパラメータは、[Params] タブで指定することができます。[VALUE] 列の値を、指定されたリソースについて削除するタグに置き換えます。

リクエストの例: **`CreateApp`** オペレーションを使用したプロジェクトの作成

Amazon Pinpoint でセグメントおよびキャンペーンを作成する前に、まずプロジェクトを作成する必要があります。Amazon Pinpoint では、プロジェクトは、セグメント、キャンペーン、設定、および共通の目的で結合されたデータで構成されます。例えば、特定のアプリケーション、または特定のブランドまたはマーケティングイニシアティブに関連するすべてのコンテンツを含むプロジェクトを使用できます。Amazon Pinpoint にお客様の情報を追加すると、その情報はプロジェクトに関連付けられます。

CreateApp API リクエストを送信してプロジェクトを作成するには

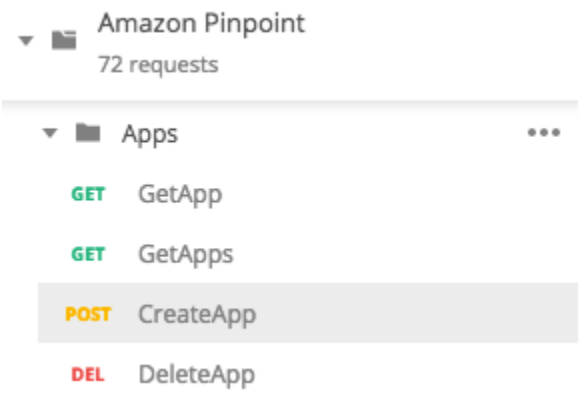
1. 環境メニューで、プロジェクト AWS リージョン を作成する を選択します。



   この例では、[環境] メニューには次の 4 つのオプションが表示されるように Postman が設定されています。

   - 米国東部 (バージニア北部)
   - 米国西部 (オレゴン)
   - 欧州 (フランクフルト)
   - 欧州 (アイルランド)

2. Apps フォルダで、 CreateAppオペレーション > を選択します。



Amazon Pinpoint Postman コレクション内の [Apps] フォルダが展開され、次のリクエストが表示されます。

- `GetApp`
- `GetApps`
- `CreateApp`
- `DeleteApp`

3. [本文] タブの、`"Name"` の横にあるプレースホルダーの値 (`"string"`) を、**`"MySampleProject"`** など、キャンペーンの名前に置き換えます。
4. キャンペーン名の後ろにあるカンマを削除し、3～5 行目にある `tags` オブジェクト全体を削除します。終了すると、リクエストは次のコードスニペットの例のようになります。

```
{
    "Name": "MySampleProject"
}
```

Postman は未加工の JSON ペイロードとしてリクエストを送信するように設定されています。

5. [送信] を選択します。キャンペーンが正常に作成された場合、[レスポンス] ペインに `201 Created` のステータスが表示されます。

```
{
    "Name": "MySampleProject"
    "Id": "12345678901234567890123456789012",
    "Arn": "arn:aws:mobiletargeting:us-
east-1:123456789012:apps/12345678901234567890123456789012",
    "tags": {}
}
```

例: **SendMessages** オペレーションを使用する E メールの送信

トランザクションメッセージを送信するために Amazon Pinpoint SendMessages API を使用するのは非常に一般的です。キャンペーンを作成するのではなく、SendMessages API を使用してメッセージを送信する利点の 1 つは、E メールアドレス、電話番号、デバイストークンなど任意のアドレスにメッセージを送信できることです。メッセージの送信先のアドレスは、Amazon Pinpoint アカウント内に存在している必要はありません。この方法を、キャンペーンを作成してメッセージを送信する方法と比較してみましょう。Amazon Pinpoint でキャンペーンを送信する前に、Amazon Pinpoint アカウントにエンドポイントを追加し、セグメントを作成し、キャンペーンを作成し実行する必要があります。

このセクションの例は、特定の E メールアドレスに直接トランザクションメールメッセージを送信する方法を説明しています。このリクエストを変更して、SMS、モバイルプッシュ、または音声など、他のチャネル経由でメッセージを送信できます。

SendMessages リクエストを送信して E メールメッセージを送信するには

1. E メールチャネルがプロジェクトで有効になっており、メッセージの送受信に使用する E メールアドレスやドメインが設定されていることを確認します。詳細については、『Amazon Pinpoint ユーザーガイド』の「E メールチャネルの有効化と無効化」と「E メール ID の検証」を参照してください。

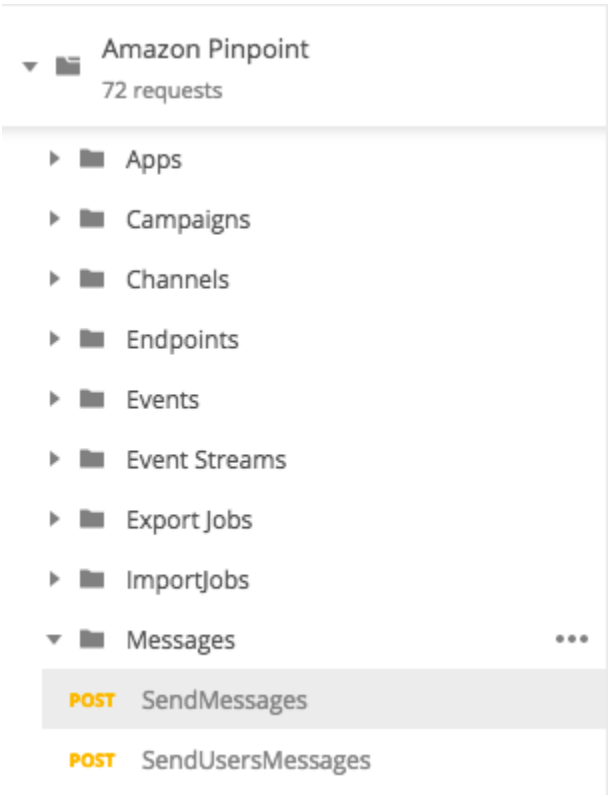
   > **Note**
   >
   > このセクションの手順を実行するには、E メールアドレスを検証する必要があります。

2. 環境メニューで、メッセージを送信 AWS リージョン する を選択します。

   この例では、[環境] メニューには次の 4 つのオプションが表示されるように Postman が設定されています。

   - 米国東部 (バージニア北部)
   - 米国西部 (オレゴン)
   - 欧州 (フランクフルト)
   - 欧州 (アイルランド)

3. メッセージフォルダで、 SendMessagesオペレーションを選択します。



4. [Pre-request Script] タブで、`projectId` 変数の値をこのセクションのステップ 2 で選択したリージョンに既に存在するプロジェクトの ID と置き換えます。
5. [本文] タブで、リクエストエディタに示されているリクエストの例を削除します。次のコードを貼り付けます。

```
{
    "MessageConfiguration":{
        "EmailMessage":{
            "FromAddress":"sender@example.com",
            "SimpleEmail":{
                "Subject":{
                    "Data":"Sample Amazon Pinpoint message"
                },
                "HtmlPart":{
                    "Data":"<h1>Test message</h1><p>This is a sample message sent
 from <a href=\"https://aws.amazon.com/pinpoint\">Amazon Pinpoint</a> using the
 SendMessages API.</p>"
                },
                "TextPart":{
                    "Data":"This is a sample message sent from Amazon Pinpoint
 using the SendMessages API."
                }
```

```
            }
        }
    },
    "Addresses":{
        "recipient@example.com": {
            "ChannelType": "EMAIL"
        }
    }
}
```

6. 前述のコードで、*sender@example.com* を検証済みの E メールアドレスに置き換えます。*recipient@example.com* を、メッセージの送信先にする検証済み E メールアドレスに置き換えます。

> **Note**
>
> アカウントがまだ Amazon Pinpoint E メールサンドボックスにある場合は、Amazon Pinpoint アカウントで検証済みのアドレスまたはドメインにのみ E メールを送信できます。アカウントをサンドボックスから削除する方法については、『Amazon Pinpoint ユーザーガイド』の「Requesting production access for email」を参照してください。

7. [送信] を選択します。メッセージが正常に送信された場合、[レスポンス] ペインに `200 OK` のステータスが表示されます。

```
{
    "ApplicationId": "12345678901234567890123456789012",
    "RequestId": "<sampleValue>",
    "Result": {
        "recipient@example.com": {
            "DeliveryStatus": "SUCCESSFUL",
            "StatusCode": 200,
            "StatusMessage": "<sampleValue>",
            "MessageId": "<sampleValue>"
        }
    }
}
```

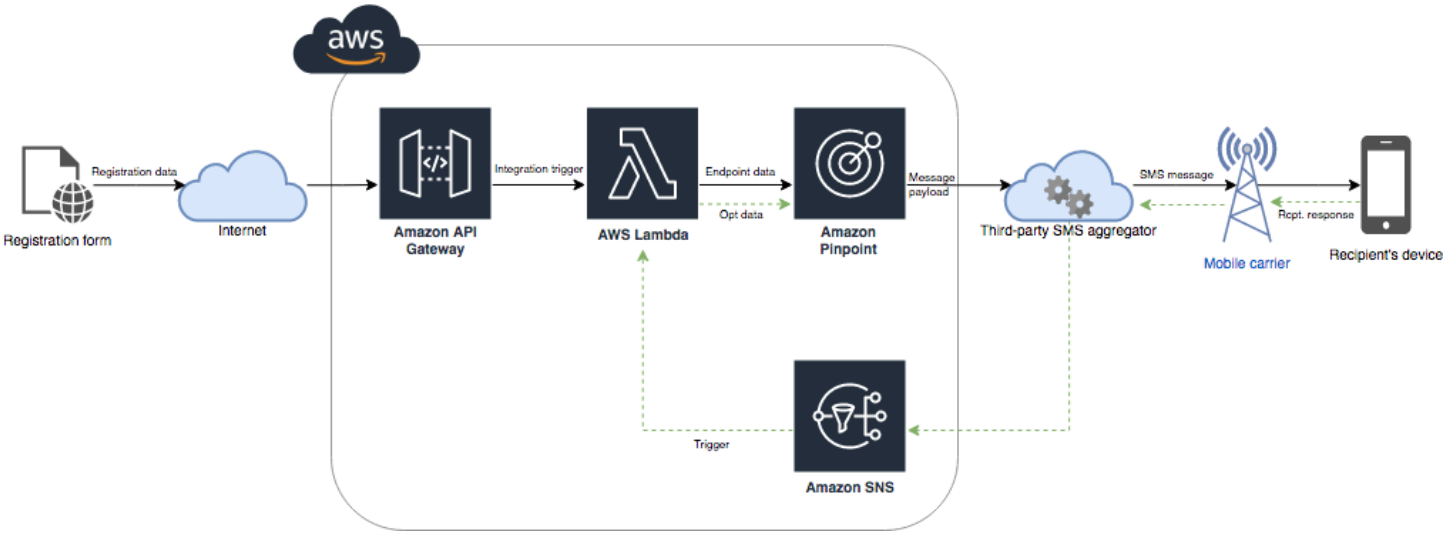
チュートリアル: SMS 登録システムの設定

SMS メッセージ (テキストメッセージ) は、時間的制約のあるメッセージをお客様に送信するのに最適な方法です。最近では、多くの人が常に自分の電話を近くに置いています。また、SMS メッセージはプッシュ通知、E メール、または電話よりも人々の注意を引く傾向があります。

お客様の携帯電話番号を取り込むには、普通、ウェブベースのフォームを使用します。お客様の電話番号を確認し、サブスクリプションを確認したら、そのお客様にプロモーション、トランザクション、販売促進、取引、および情報提供の SMS メッセージを送信できます。

このチュートリアルでは、お客様の連絡先情報を取り込むためのウェブフォームの設定について説明します。ウェブフォームはこの情報を Amazon Pinpoint に送信します。次に、Amazon Pinpoint は電話番号が有効であることを確認し、その電話番号に関連するその他のメタデータをキャプチャします。その後で、Amazon Pinpoint は、お客様にサブスクリプションについて確認するメッセージを送信します。お客様がサブスクリプションを確認したら、Amazon Pinpoint はお客様がメッセージを受信するように選択します。

次のアーキテクチャダイアグラムは、このソリューションでのデータフローを示しています。



ダブルオプトインについて

このチュートリアルでは、双方向 SMS メッセージを使用する Amazon Pinpoint でのダブルオプトインのセットアップ方法について説明します。

SMS ダブルオプトインシステムでは、お客様は、ウェブフォームまたはアプリ内で送信することによって、電話番号を提供します。お客様からのリクエストを受信したら、Amazon Pinpoint に新しいエンドポイントを作成します。新しいエンドポイントはコミュニケーションからオプトアウトされて

いるはずです。次に、その電話番号にメッセージを送信します。メッセージの中で、受信者に特定の単語やフレーズ (「はい」や「確認」など) で返事をしてサブスクリプションを確認します。お客様が指定した単語またはフレーズを使用してメッセージに応答した場合は、エンドポイントのステータスをオプトインに変更します。それ以外の場合、お客様が応答しなかったり、別の単語やフレーズで応答したりした場合は、エンドポイントをオプトアウトのステータスのままにしておくことができます。

このソリューションについて

このセクションには、このチュートリアルで構築しようとしているソリューションに関する情報が含まれています。

対象者

このチュートリアルは、開発者とシステム実装者を対象としています。このチュートリアルの手順を行うのに Amazon Pinpoint に精通している必要はありません。ただし、IAM ポリシーの管理、Node.js での Lambda 関数の作成、およびウェブコンテンツのデプロイについて慣れている必要があります。

使用される機能

このチュートリアルには、次の Amazon Pinpoint 機能の使用例が含まれています。

- トランザクション SMS メッセージの送信
- 電話番号検証を使用して電話番号に関する情報を取得する
- 双方向 SMS メッセージングを使用して着信 SMS メッセージを受信する
- 動的なセグメントを作成する
- キャンペーンの作成
- を使用して Amazon Pinpoint API を操作する AWS Lambda

所要時間

このチュートリアルを完了するには、約 1 時間かかります。このソリューションを実装した後は、独自のユースケースに合わせてソリューションを改良するために実行できる追加のステップがあります。

リージョン別制限

このチュートリアルでは、Amazon Pinpoint コンソールを使用してロングコードをリースする必要があります。Amazon Pinpoint コンソールを使用して、複数の国を拠点とする専用のロングコードをリースできます。ただし、SMS メッセージの送信に使用できるのは、カナダを拠点とするロングコードだけです。(他の国や地域を拠点とするロングコードを使用してボイスメッセージを送信できます。)

この制限を念頭に置いて、このチュートリアルのコード例を作成しました。例えば、コード例では、受信者の電話番号は常に 10 桁で国コードは 1 であると想定しています。米国またはカナダ以外の国またはリージョンでこのソリューションを実装する場合は、コード例を適切に修正する必要があります。

リソース使用量のコスト

AWS アカウントの作成には料金はかかりません。ただし、このソリューションを実装することによって、以下のコストが発生する可能性があります。

- [Long code lease costs] – このチュートリアルを完了するにはロングコードをリースする必要があります。カナダを拠点とするロングコードは、1 か月あたり 1.00 USD のコストがかかります。
- [Phone number validation usage] – このチュートリアルのソリューションでは、Amazon Pinpoint の電話番号を検証する機能を使用して、受信した各番号が有効で正しい形式であることを確認し、電話番号に関する追加情報を取得します。電話番号検証のリクエストには、1 件あたり 0.006 米ドル (USD) の料金が発生します。
- [Message sending costs] – このチュートリアルのソリューションは、アウトバウンド SMS メッセージを送信します。Amazon Pinpoint を通じて送信するメッセージごとに料金が発生します。各メッセージに対して支払う料金は、受信者の国または地域によって異なります。米国内の受信者 (米国領を除く) にメッセージを送信すると、メッセージあたり 0.00645 米ドル (USD) をお支払いいただきます。カナダの受信者にメッセージを送信すると、受信者の通信事業者と場所に応じて、0.00109 ~ 0.02 米ドル (USD) をお支払いいただきます。
- [Message receiving costs] – このソリューションは、着信 SMS メッセージも受信して処理します。Amazon Pinpoint アカウントに関連付けられている電話番号に送信される各着信メッセージに対してお支払いいただきます。支払い額は、受信側の電話番号によって異なります。受信番号が米国 (米国領を除く) に拠点を置いている場合、受信メッセージあたり 0.0075米ドル (USD) をお支払いいただきます。番号がカナダを拠点としている場合、受信メッセージあたり 0.00155 米ドル (USD) をお支払いいただきます。
- [Lambda usage] – このソリューションでは Amazon Pinpoint API とやり取りする 2 つの Lambda 関数を使用します。Lambda 関数を呼び出すと、関数のリクエストの数、コードの実行に要する時間、および関数が使用するメモリの量に基づいて課金されます。このチュートリアルの関数は

ごくわずかなメモリしか使用せず、通常 1～3 秒間実行されます。このソリューションの使用の一部またはすべての使用量が Lambda 無料利用枠内に該当します。詳細については、「Lambda pricing」を参照してください。

- [API Gateway usage] – このソリューションのウェブフォームは、 によって管理されている API を呼び出します。API Gateway への 100 万回の呼び出しごとに、Amazon Pinpoint を使用する AWS リージョンに応じて 3.50～3.70 USD を支払います。 Amazon Pinpoint 詳細については、[API Gateway pricing] を参照してください。
- [Web hosting costs] – このソリューションには、ウェブサイトでホストする必要があるウェブベースのフォームが含まれています。このコンテンツをホストするための支払い額は、ウェブホスティングプロバイダーによって異なります。

> **Note**
>
> このリストに示されているすべての価格は米ドル (USD)。

次の手順: 前提条件

前提条件

このチュートリアルを開始する前に、次の前提条件を完了する必要があります。

- AWS アカウントを持っている必要があります。AWS アカウントを作成するには、「https://console.aws.amazon.com/」、[Create a new AWS account] を選択します。
- AWS Management Console へのサインインに使用するアカウントは、次のタスクを実行できる必要があります。
  - 新しい IAM ポリシーおよびロールを作成する
  - 新しい Amazon Pinpoint プロジェクトを作成する
  - 新しい Lambda 関数を作成する
  - API Gateway で新しい API を作成する
- ウェブページをホストする方法がある必要があり、ウェブページを発行する方法を知っている必要があります。ウェブページをホストするために AWS サービスを使用できますが、必須ではありません。

> **Tip**
>
> AWS サービスを使用してウェブページをホストする方法の詳細については、「静的ウェブサイトをホスティングする」を参照してください。

Next: Amazon Pinpoint を設定する

ステップ 1: Amazon Pinpoint を設定する

このソリューションを実装するための最初のステップは、Amazon Pinpoint を設定することです。このセクションでは、以下の作業を行います。

- Amazon Pinpoint でプロジェクトを作成する
- SMS チャネルを有効にして電話番号をリースする
- 双方向 SMS メッセージングを設定する

このチュートリアルを開始する前に、前提条件を確認してください。

Amazon Pinpoint でプロジェクトを作成する

開始するには、Amazon Pinpoint プロジェクトを作成する必要があります。Amazon Pinpoint では、プロジェクトは、セグメント、キャンペーン、設定、および共通の目的で結合されたデータで構成されます。例えば、特定のアプリケーション、または特定のブランドまたはマーケティングイニシアティブに関連するすべてのコンテンツを含むプロジェクトを使用できます。Amazon Pinpoint にお客様の情報を追加すると、その情報はプロジェクトに関連付けられます。

新しいプロジェクトを作成するためのステップは、以前 Amazon Pinpoint にプロジェクトを作成したことがあるかどうかによって異なります。

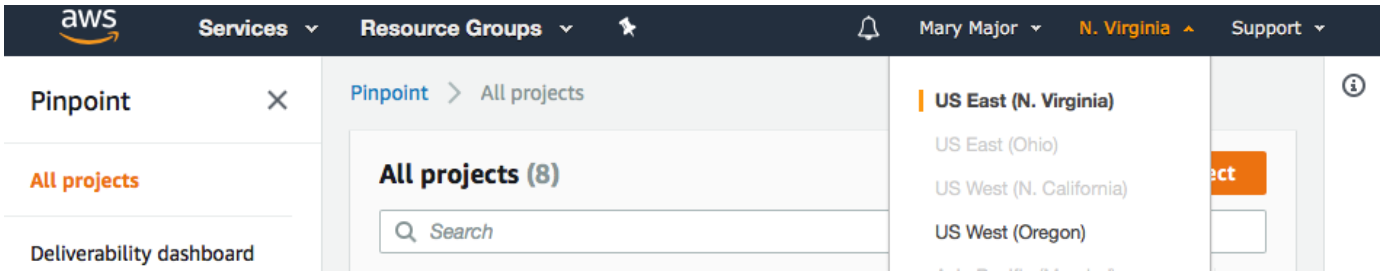
プロジェクトを作成する (新しい Amazon Pinpoint ユーザー)

このステップでは、現在の AWS リージョンでプロジェクトを作成したことがない場合に、新しい Amazon Pinpoint プロジェクトを作成するプロセスについて説明します。

プロジェクトを作成する

1. にサインイン AWS Management Console し、https://console.aws.amazon.com/pinpoint/ で Amazon Pinpoint コンソールを開きます。

2. 次の図に示すように、リージョンセレクタを使用して、使用する AWS リージョンを選択します。不明な場合は、最寄りに配置されたリージョンを選択します。



3. [開始する] の [名前] にキャンペーン (**SMSRegistration** など) の名前を入力して、[プロジェクトを作成する] を選択します。
4. [Configure features] ページで、[Skip this step] を選択します。
5. ナビゲーションペインで、[All projects] を選択します。
6. [すべてのプロジェクト] ページで、先ほど作成したプロジェクトの横に、[プロジェクト ID] 列に表示されている値をコピーします。

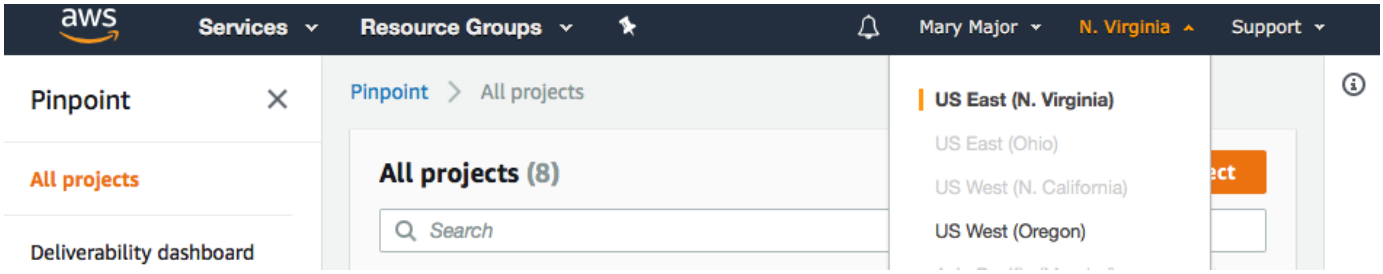
> **Tip**
>
> このチュートリアルでは、いくつかの異なる場所でこの ID を使用する必要があります。プロジェクト ID は、後でコピーできるように都合の良い場所に保管してください。

プロジェクトを作成する (既存の Amazon Pinpoint ユーザー)

このステップでは、現在の AWS リージョンでプロジェクトを既に作成している場合に、新しい Amazon Pinpoint プロジェクトを作成するプロセスについて説明します。

プロジェクトを作成する

1. にサインイン AWS Management Console し、https://console.aws.amazon.com/pinpoint/ で Amazon Pinpoint コンソールを開きます。
2. 次の図に示すように、リージョンセレクタを使用して、使用する AWS リージョンを選択します。不明な場合は、最寄りに配置されたリージョンを選択します。



3. [All projects] ページで、[Create a project] を選択します。
4. [プロジェクトを作成する] ウィンドウの [プロジェクト名] に、プロジェクトの名前 (**SMSRegistration** など) を入力します。[作成] を選択します。
5. [Configure features] ページで、[Skip this step] を選択します。
6. ナビゲーションペインで、[All projects] を選択します。
7. [すべてのプロジェクト] ページで、先ほど作成したプロジェクトの横に、[プロジェクト ID] 列に表示されている値をコピーします。

> **Tip**
>
> このチュートリアルでは、いくつかの異なる場所でこの ID を使用する必要があります。プロジェクト ID は、後でコピーできるように都合の良い場所に保管してください。

専用の電話番号を取得する

> **Note**
>
> Amazon Pinpoint のユーザーガイドドキュメントを更新しました。Amazon Pinpoint SMS と音声リソースを作成、設定、管理する方法に関する最新情報を入手するには、新しい「Amazon Pinpoint SMS ユーザーガイド」を参照してください。

プロジェクトを作成すると、そのプロジェクトの機能の設定を開始できます。このセクションでは、SMS チャネルを有効にし、SMS メッセージを送信するときに使用する専用の電話番号を取得します。

> **Note**
>
> このセクションでは、ブランドおよびキャンペーン登録後に米国の 10DLC 電話番号をリースする、または米国の通話料無料番号やカナダのロングコードをリースすることを前提としています。このセクションの手順に従い、米国またはカナダ以外の国を選択すると、その番号を使用して SMS メッセージを送信することはできません。米国やカナダ以外の国で SMS 対応のロングコードをリースする方法については、「Amazon Pinpoint SMS ユーザーガイド」の「サポートされている国とリージョン (SMS チャネル)」を参照してください。

Amazon Pinpoint コンソールを使用して SMS チャネルを有効にするには、次の手順に従います。

SMS チャンネルを有効にする

1. にサインイン AWS Management Console し、https://console.aws.amazon.com/pinpoint/ で Amazon Pinpoint コンソールを開きます。
2. ナビゲーションペインで、[設定] の [SMS および音声] を選択します。
3. [SMS 設定] の横の [編集] を選択します。
4. [全般設定] で、[Enable the SMS channel for this project] を選択し、[変更の保存] を選択します。

Amazon Pinpoint SMS コンソールを使用して電話番号をリクエストするには、次の手順に従います。

電話番号をリクエストする (コンソール)

1. Amazon Pinpoint SMS コンソール (https://console.aws.amazon.com/sms-voice/) を開きます。

   > **Note**
   >
   > Amazon Pinpoint プロジェクトを AWS リージョン 作成したのと同じ で電話番号をリクエストしてください。

2. ナビゲーションペインの [設定] で、[電話番号]、[リクエスト発信者] の順を選択します。
3. [メッセージ送信先の国] の [国を選択]] ページで、[米国] または [カナダ] を選択します。[次へ] をクリックします。
4. [メッセージングのユースケース] セクションで、以下を入力します。

- [番号機能] で [SMS] を選択します。

> **Important**
> 電話番号を購入すると、SMS と音声の機能は変更できなくなります。

- [双方向メッセージング] で、[はい] を選択します。

5. [次へ] をクリックします。
6. [発信者タイプを選択] で [ロングコード] または [10DLC] を選択します。

   10DLC を選択し、既にキャンペーンを登録している場合は、[登録済みキャンペーンに関連付ける] からキャンペーンを選択できます。
7. [次へ] をクリックします。
8. [確認とリクエスト] で、リクエストを送信する前に検証して編集できます。[リクエスト] を選択します。
9. リクエストした電話番号の種類によっては、[登録が必須です] ウィンドウが表示される場合があります。電話番号はこの登録に関連付けられるため、登録が承認されるまではメッセージを送信できません。登録要件の詳細については、「登録」を参照してください。

   a. [登録フォーム名] に、わかりやすい名前を入力します。
   b. [登録を開始] を選択して電話番号の登録を完了するか、[後で登録] を選択します。

> **Important**
> 登録が承認されるまでは、電話番号からメッセージを送信できません。
> 登録状況にかかわらず、電話番号の月額リース料は引き続き請求されます。登録要件の詳細については、「登録」を参照してください。

双方向 SMS を有効にする

これで、専用の電話番号があるので、双方向 SMS をセットアップできます。双方向 SMS を有効にすると、お客様は送信された SMS メッセージに応答することができるようになります。このソリューションでは、双方向 SMS を使用して、お客様が SMS プログラムにサブスクライブすることを希望することを確認する手段を提供します。

Amazon Pinpoint SMS コンソールを使用して双方向 SMS を有効にするには、次の手順に従います。

双方向 SMS を有効にする

1. Amazon Pinpoint SMS コンソール (https://console.aws.amazon.com/sms-voice/) を開きます。
2. ナビゲーションペインの [設定] で、[電話番号] を選択します。
3. [電話番号] ページで、電話番号を選択します。
4. [双方向 SMS] タブで、[設定を編集] ボタンを選択します。
5. [設定を編集] ページで [双方向メッセージを有効にする] を選択します。
6. [送信先タイプ] で、[Amazon SNS] を選択します。
   - [新しい Amazon SNS トピック] - Amazon Pinpoint SMS はアカウントにトピックを作成します。トピックは、必要なすべてのパーミッションがあれば、自動的に作成されます。Amazon SNS トピックの詳細については、「Amazon SNS デベロッパーガイド」の「Amazon SNS を設定する」を参照してください。
   - [受信メッセージの送信先] に、トピック名 (**SMSRegistrationFormTopic** など) を入力します。
7. [双方向チャネルの役割] で、[SNS トピックポリシーを使用] を選択します。
8. [変更を保存] を選択します。

Amazon Pinpoint SMS コンソールを使用して、お客様がサブスクリプションを確認するために送信する電話番号にキーワード (**Confirm** や **Yes** など) を追加します。

キーワードを追加する

1. Amazon Pinpoint SMS コンソール (https://console.aws.amazon.com/sms-voice/) を開きます。
2. ナビゲーションペインで、[設定] の [電話番号] を選択します。
3. [電話番号] ページで、キーワードを追加する電話番号を選択します。
4. [キーワード] タブで [キーワードを追加] ボタンを選択します。
5. [カスタムキーワード] ペインで、以下を追加します。
   - キーワード — 追加する新しいキーワード (**Yes** や **Confirm** など)。
   - 応答メッセージ — 受信者に送り返すメッセージ。
   - キーワードアクション — キーワードの受信時に実行するアクション。[自動応答] を選択します。
6. [キーワードを追加] を選択します。

Next: IAM ポリシーおよびロールを作成する

ステップ 2: IAM ポリシーおよびロールを作成する

SMS 登録ソリューションを実装する次のステップは、 AWS Identity and Access Management (IAM) でポリシーとロールを設定することです。このソリューションでは、Amazon Pinpoint に関連付けられている特定のリソースへのアクセスを提供するポリシーを作成する必要があります。次に、ロールを作成して、ポリシーをアタッチします。このチュートリアルの後半では、このロールを使用して Amazon Pinpoint API で特定のオペレーションを呼び出す AWS Lambda 関数を作成します。

IAM ポリシーを作成する

このセクションでは、IAM ポリシーを作成する方法を説明します。このポリシーを使用するユーザーとロールは、以下を実行できます。

- 電話番号検証機能を使用する
- Amazon Pinpoint エンドポイントを表示、作成、および更新する
- Amazon Pinpoint エンドポイントにメッセージを送信する

このチュートリアルでは、Lambda にこれらのタスクを実行する機能を付与できます。ただし、セキュリティを強化するために、このポリシーでは、最小特権を付与する原則が使用されます。つまり、このソリューションを完了するために必要なアクセス許可のみが付与され、それ以上は付与されません。このポリシーには、次のような制限事項があります。

- 特定のリージョンで 電話番号検証 API を呼び出すためにのみ使用できます。
- 特定の Amazon Pinpoint プロジェクトに関連付けられているエンドポイントを表示、作成、または更新するためにのみ使用できます。
- 特定の Amazon Pinpoint プロジェクトに関連付けられているエンドポイントにメッセージを送信するためにのみ使用できます。

ポリシーを作成するには

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。
2. ナビゲーションペインで ポリシーを選択してから ポリシーの作成を選択します。
3. [JSON] タブに、以下のコードを貼り付けます。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogStream",
                "logs:PutLogEvents",
                "logs:CreateLogGroup"
            ],
            "Resource": "arn:aws:logs:*:*:*"
        },
        {
            "Effect": "Allow",
            "Action": "mobiletargeting:SendMessages",
            "Resource": "arn:aws:mobiletargeting:region:accountId:apps/projectId/*"
        },
        {
            "Effect": "Allow",
            "Action": [
              "mobiletargeting:GetEndpoint",
              "mobiletargeting:UpdateEndpoint",
              "mobiletargeting:PutEvents"
            ],
            "Resource": "arn:aws:mobiletargeting:region:accountId:apps/projectId/
endpoints/*"
        },
        {
          "Effect": "Allow",
          "Action": "mobiletargeting:PhoneNumberValidate",
          "Resource": "arn:aws:mobiletargeting:region:accountId:phone/number/
validate"
        }
    ]
}
```

前述の例で、以下を実行します。

- *region* は、 us-east-1や など、Amazon Pinpoint を使用する AWS リージョンに置き換えますeu-central-1。

> **Tip**
>
> Amazon Pinpoint が利用可能な AWS リージョンの完全なリストについては、「」のAWS 「リージョンとエンドポイント」を参照してくださいAWS 全般のリファレンス。

- *accountId* を AWS アカウントの一意の ID に置き換えます。
- *projectId* を、このチュートリアルのAmazon Pinpoint プロジェクトの作成」で作成したプロジェクトの一意の ID に置き換えます。

> **Note**
>
> logs アクションにより、Lambda は出力を Logs CloudWatch に記録できます。

4. [次へ] をクリックします。
5. ポリシー名 には、 など、ポリシーの名前を入力します**RegistrationFormPolicy**。[ポリシーの作成] を選択します。

IAM ロールを作成する

ロールを作成するには

1. IAM コンソール (https://console.aws.amazon.com/iam/) を開きます。
2. IAM コンソールのナビゲーションペインで、[ロール]、[ロールを作成する] の順に選択します。
3. 信頼されたエンティティタイプ でサービス AWS を選択し、サービスまたはユーザーケースでドロップダウンリストから Lambda を選択します。
4. [次へ] をクリックします。
5. アクセス許可ポリシー で、前のセクションで作成したポリシーを選択または検索し、次へ を選択します。
6. ロールの詳細 のロール名 に、 などのロールの名前を入力します**SMSRegistrationForm**。[ロールを作成] を選択します。

Next:Lambda 関数を作成する

ステップ 3: Lambda 関数を作成する

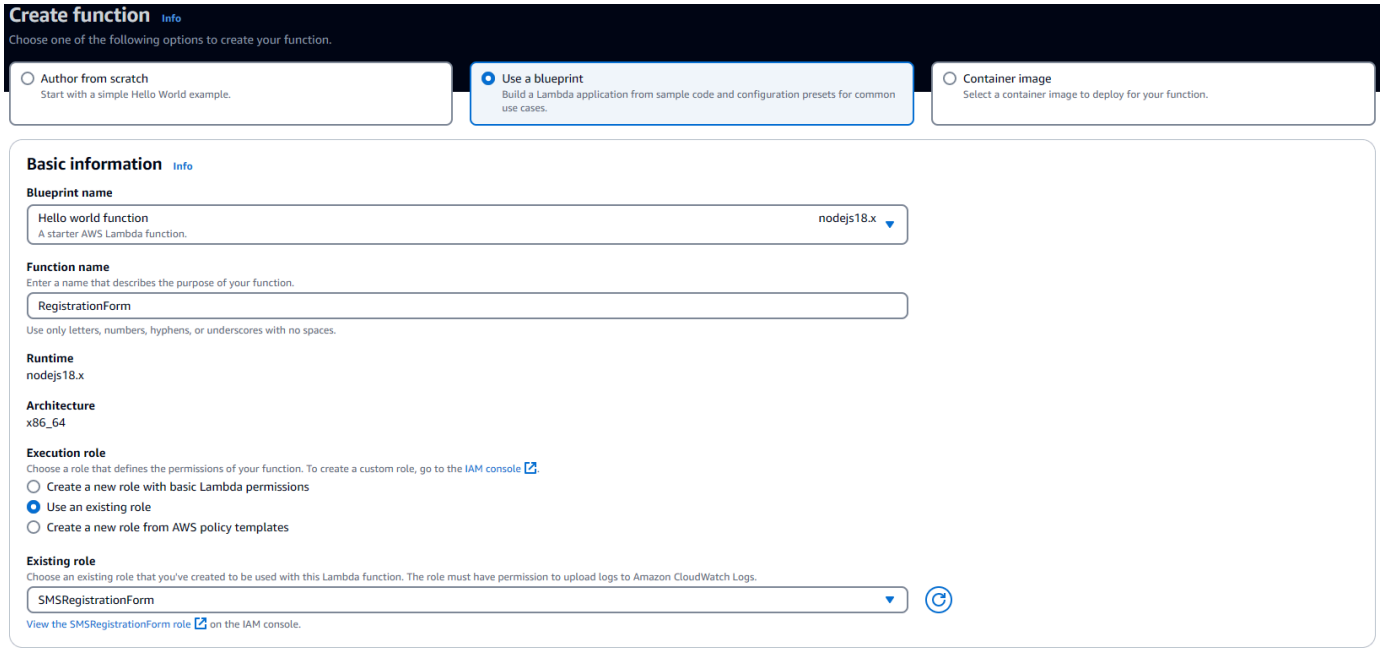
このソリューションでは、2 つの Lambda 関数を使用します。このセクションでは、これらの関数を作成して設定する方法について説明します。後で、特定のイベントが発生したときにこれらの関数が実行するように API Gateway および Amazon Pinpoint をセットアップします。これらの関数のどちらも、指定した Amazon Pinpoint プロジェクトにエンドポイントを作成し、更新します。最初の関数では、電話番号検証機能も使用します。

お客様の情報を検証する関数を作成し、エンドポイントを作成する

最初の関数は登録フォームから入力を受け取り、Amazon API Gateway から受け取ります。この情報を使用して、Amazon Pinpoint の電話番号検証機能を使用して顧客の電話番号に関する情報を取得します。次に、検証されたデータを使用して、指定された Amazon Pinpoint プロジェクトに新しいエンドポイントを作成します。デフォルトでは、関数に作成されるエンドポイントはユーザーからの今後の通信からオプトアウトされますが、このステータスは 2 番目の関数で変更できます。最後に、この関数では、ユーザーから SMS 通信を受信することをお客様が希望することを確認するように求めるメッセージをお客様に送信します。

Lambda 関数を作成するには

1. https://console.aws.amazon.com/lambda/ で AWS Lambda コンソールを開きます。
2. [関数を作成] を選択します。
3. 「関数の作成」で、「設計図の使用」を選択します。
4. 検索フィールドに **`hello`** と入力して Enter キーを押します。結果のリストで、次のイメージに示すように、`hello-world` Node.js 関数を選択します。



5. [基本的な情報] で、以下を実行します。

   - [名前] には、**RegistrationForm** など、関数の名前を入力します。
   - [Role] で、[既存のロールを選択] を選択します。
   - 既存のロール で、「IAM ロールの作成」で作成した SMSRegistrationForm ロールを選択します。 ???

   完了したら、[関数の作成] を選択します。

6. コードソースの場合、コードエディタでサンプル関数を削除し、次のコードを貼り付けます。

```
import { PinpointClient, PhoneNumberValidateCommand, UpdateEndpointCommand,
 SendMessagesCommand } from "@aws-sdk/client-pinpoint"; // ES Modules import
const pinClient = new PinpointClient({region: process.env.region});

// Make sure the SMS channel is enabled for the projectId that you specify.
// See: https://docs.aws.amazon.com/pinpoint/latest/userguide/channels-sms-
setup.html
var projectId = process.env.projectId;

// You need a dedicated long code in order to use two-way SMS.
// See: https://docs.aws.amazon.com/pinpoint/latest/userguide/channels-voice-
manage.html#channels-voice-manage-request-phone-numbers
var originationNumber = process.env.originationNumber;
```

```
// This message is spread across multiple lines for improved readability.
var message = "ExampleCorp: Reply YES to confirm your subscription. 2 msgs per "
            + "month. No purchase req'd. Msg&data rates may apply. Terms: "
            + "example.com/terms-sms";

var messageType = "TRANSACTIONAL";

export const handler = async (event, context) => {
  console.log('Received event:', event);
  await validateNumber(event);
};

async function validateNumber (event) {
  var destinationNumber = event.destinationNumber;
  if (destinationNumber.length == 10) {
    destinationNumber = "+1" + destinationNumber;
  }
  var params = {
    NumberValidateRequest: {
      IsoCountryCode: 'US',
      PhoneNumber: destinationNumber
    }
  };
  try{
    const PhoneNumberValidateresponse = await pinClient.send( new
 PhoneNumberValidateCommand(params));
    console.log(PhoneNumberValidateresponse);
     if (PhoneNumberValidateresponse['NumberValidateResponse']['PhoneTypeCode'] ==
 0) {
        await createEndpoint(PhoneNumberValidateresponse, event.firstName,
 event.lastName, event.source);

     } else {
       console.log("Received a phone number that isn't capable of receiving "
                  +"SMS messages. No endpoint created.");
     }
  }catch(err){
    console.log(err);
  }
}

async function createEndpoint(data, firstName, lastName, source) {
  var destinationNumber = data['NumberValidateResponse']
['CleansedPhoneNumberE164'];
```

```
  var endpointId = data['NumberValidateResponse']
['CleansedPhoneNumberE164'].substring(1);

  var params = {
    ApplicationId: projectId,
    // The Endpoint ID is equal to the cleansed phone number minus the leading
    // plus sign. This makes it easier to easily update the endpoint later.
    EndpointId: endpointId,
    EndpointRequest: {
      ChannelType: 'SMS',
      Address: destinationNumber,
      // OptOut is set to ALL (that is, endpoint is opted out of all messages)
      // because the recipient hasn't confirmed their subscription at this
      // point. When they confirm, a different Lambda function changes this
      // value to NONE (not opted out).
      OptOut: 'ALL',
      Location: {
        PostalCode:data['NumberValidateResponse']['ZipCode'],
        City:data['NumberValidateResponse']['City'],
        Country:data['NumberValidateResponse']['CountryCodeIso2'],
      },
      Demographic: {
        Timezone:data['NumberValidateResponse']['Timezone']
      },
      Attributes: {
        Source: [
          source
        ]
      },
      User: {
        UserAttributes: {
          FirstName: [
            firstName
          ],
          LastName: [
            lastName
          ]
        }
      }
    }
  };
  try{
    const UpdateEndpointresponse = await pinClient.send(new
 UpdateEndpointCommand(params));
```

```
    console.log(UpdateEndpointresponse);
    await sendConfirmation(destinationNumber);
  }catch(err){
    console.log(err);
  }
}

async function sendConfirmation(destinationNumber) {
  var params = {
    ApplicationId: projectId,
    MessageRequest: {
      Addresses: {
        [destinationNumber]: {
          ChannelType: 'SMS'
        }
      },
      MessageConfiguration: {
        SMSMessage: {
          Body: message,
          MessageType: messageType,
          OriginationNumber: originationNumber
        }
      }
    }
  };
  try{
    const SendMessagesCommandresponse = await pinClient.send(new
 SendMessagesCommand(params));
    console.log("Message sent! "
          + SendMessagesCommandresponse['MessageResponse']['Result']
[destinationNumber]['StatusMessage']);
  }catch(err){
    console.log(err);
  }
}
```

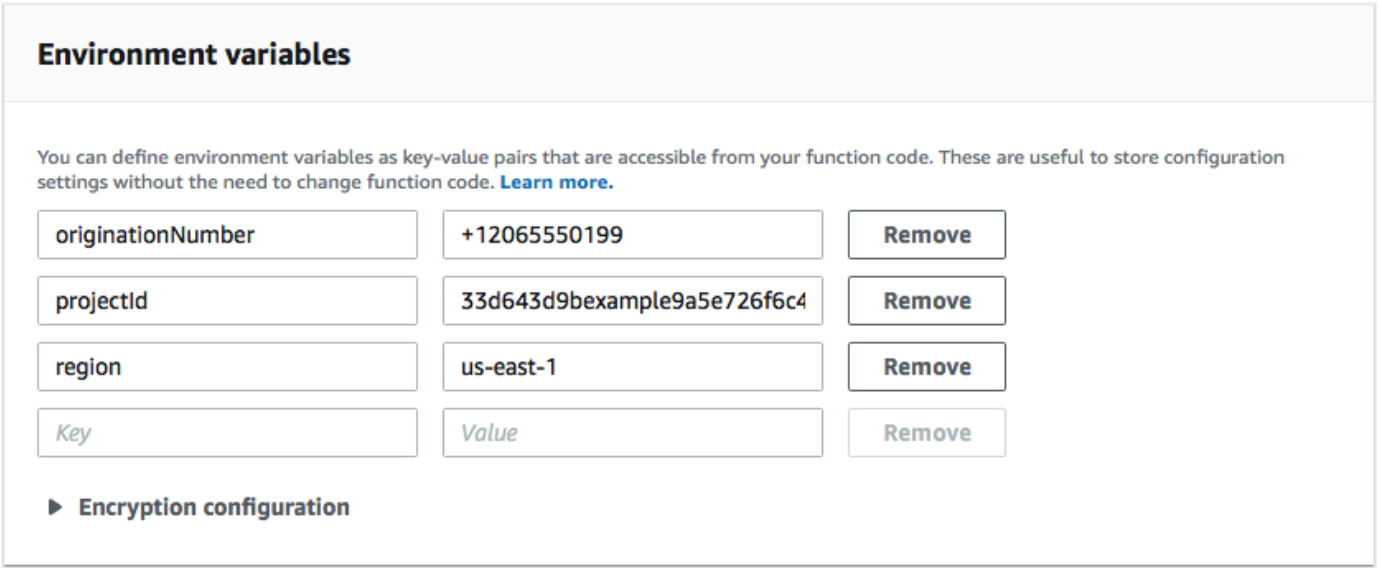
7. 環境変数 の設定 タブで、編集 を選択し、環境変数 を追加 を選択し、次の操作を行います。

   - 最初の行で、**originationNumber** のキーで変数を作成します。次に、「ステップ 1.2」で受け取った専用ロングコードの電話番号に値を設定します。

> **Note**
>
> 電話番号には、必ずプラス記号 (+) および国コードを含めてください。ハイフン (-)、ピリオド (.)、または括弧など、その他の特殊文字を含めないでください。

- 2 行目で、**`projectId`** のキーで変数を作成します。次に、「ステップ 1.1」で作成したプロジェクトの一意の ID に値を設定します。
- 3 行目で、**`region`** のキーで変数を作成します。次に、**`us-east-1`** または **`us-west-2`** など、Amazon Pinpoint を使用するリージョンに値を設定します。

終了したら、[環境変数] セクションは、次のイメージに示す例のようになります。



8. ページの上部で、[保存] を選択します。

関数をテストする

関数を作成した後、テストして適切に設定されているかどうかを確認します。また、作成した IAM ロールに適切なアクセス許可があることを確認します。

関数をテストするには

1. [テスト] タブを選択します。
2. 新しいイベントの作成 を選択し、次の操作を行います。
   - [Event name] で、**`MyPhoneNumber`** など、テストイベントの名前を入力します。

- コードエディタのサンプルコードを消去します。次のコードを貼り付けます。

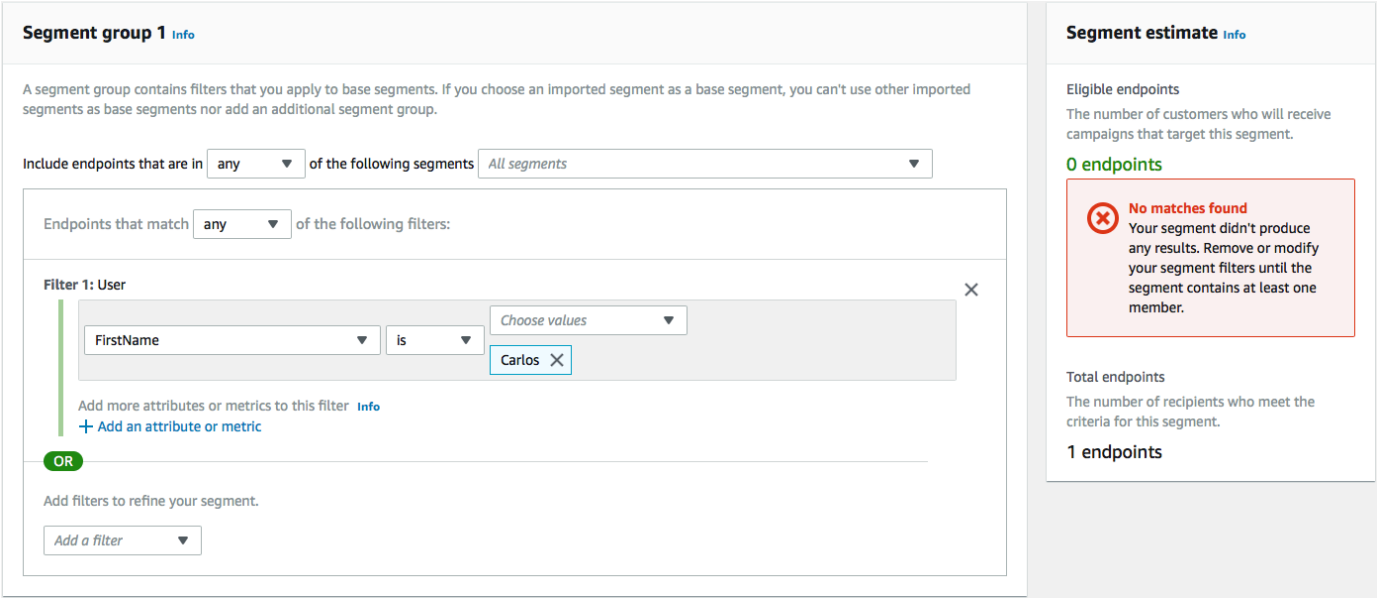
```
{
  "destinationNumber": "+12065550142",
  "firstName": "Carlos",
  "lastName": "Salazar",
  "source": "Registration form test"
}
```

- 上記のコード例で、destinationNumber、firstName、および lastName 属性の値を、個人の連絡先情報など、テストに使用する値に置き換えます。この関数をテストすると、destinationNumber 属性で指定した電話番号に SMS メッセージが送信されます。指定した電話番号が SMS メッセージを受信できることを確認します。
- [作成] を選択します。

3. [Test] を選択します。
4. [実行結果: 成功] で、[詳細] を選択します。[ログ出力] セクションで、関数の出力を確認します。関数がエラーなしで実行されたことを確認します。

   指定した destinationNumber に関連付けられたデバイスを確認し、テストメッセージを受信したことを確認します。
5. Amazon Pinpoint コンソール (https://console.aws.amazon.com/pinpoint/) を開きます。
6. 「すべてのプロジェクト」ページで、Amazon Pinpoint プロジェクトの作成」で作成したプロジェクトを選択します。
7. ナビゲーションペインの [Segments] を選択します。[セグメント] ページで、[セグメントの作成] を選択します。
8. [セグメントグループ 1] の [Add filters to refine your segment] で、[Filter by user] を選択します。
9. ユーザー属性の選択 で、 を選択しますFirstName。次に、[Choose values] に、テストイベントで指定した名を選択します。

   [Segment estimate] セクションに、次のイメージに示すように、適格なエンドポイントがゼロであり、合計エンドポイントが 1 つであることが表示されます。この結果は正常です。関数が新しいエンドポイントを作成すると、エンドポイントがオプトアウトされます。Amazon Pinpoint のセグメントは、オプトアウトされたエンドポイントを自動的に除外します。



通信にお客様をオプトインする関数を作成する

2 番目の関数は、お客様が最初の関数により送信されるメッセージに返信した場合のみ実行されます。顧客の返信に、「双方向 SMS を有効にする」で指定したキーワードが含まれている場合、関数はエンドポイントレコードを更新して将来の通信にオプトインします。また、Amazon Pinpoint は、「双方向 SMS を有効にする」で指定したメッセージで自動的に応答します。

お客様が応答しない場合、または指定されたキーワード以外で応答する場合、何も起こりません。お客様のエンドポイントは Amazon Pinpoint に残っていますが、セグメントの対象となることはできません。

Lambda 関数を作成するには

1. https://console.aws.amazon.com/lambda/ で AWS Lambda コンソールを開きます。
2. [関数を作成] を選択します。
3. [関数の作成] で、[設計図] を選択します。
4. 検索フィールドに **`hello`** と入力して Enter キーを押します。結果のリストで、次のイメージに示すように、`hello-world` Node.js 関数を選択します。[Configure] (設定) を選択します。
5. [Basic information] (基本的な情報) で、以下を実行します。
   - [名前] には、**`RegistrationForm_OptIn`** など、関数の名前を入力します。
   - [Role] で、[既存のロールを選択] を選択します。

- 既存のロール で、「IAM ロールの作成」で作成した SMSRegistrationForm ロールを選択します。 ???

完了したら、[関数の作成] を選択します。

6. コードエディタでサンプル関数を削除し、次のコードを貼り付けます。

```
import { PinpointClient, UpdateEndpointCommand } from "@aws-sdk/client-
pinpoint"; // ES Modules import

// Create a new Pinpoint client instance with the region specified in the
 environment variables
const pinClient = new PinpointClient({ region: process.env.region });

// Get the Pinpoint project ID and the confirm keyword from environment variables
const projectId = process.env.projectId;
const confirmKeyword = process.env.confirmKeyword.toLowerCase();

// This is the main handler function that is invoked when the Lambda function is
 triggered
export const handler = async (event, context) => {
    console.log('Received event:', event);

    try {
        // Extract the timestamp, message, and origination number from the SNS
 event
        const timestamp = event.Records[0].Sns.Timestamp;
        const message = JSON.parse(event.Records[0].Sns.Message);
        const originationNumber = message.originationNumber;
        const response = message.messageBody.toLowerCase();

        // Check if the response message contains the confirm keyword
        if (response.includes(confirmKeyword)) {
            // If the confirm keyword is found, update the endpoint's opt-in status
            await updateEndpointOptIn(originationNumber, timestamp);
        }
    }catch (error) {
        console.error('An error occurred:', error);
        throw error; // Rethrow the error to handle it upstream
    }
};

// This function updates the opt-in status of a Pinpoint endpoint
```

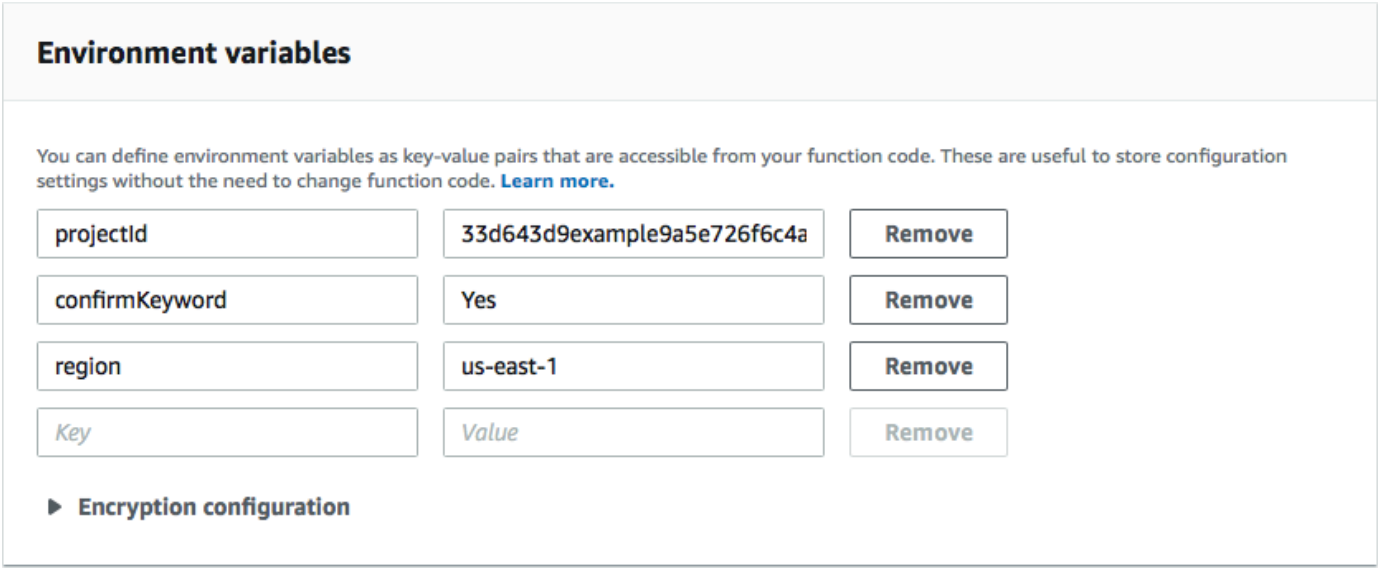
```
async function updateEndpointOptIn(originationNumber, timestamp) {
    // Extract the endpoint ID from the origination number
    const endpointId = originationNumber.substring(1);

     // Prepare the parameters for the UpdateEndpointCommand
    const params = {
        ApplicationId: projectId,
        EndpointId: endpointId,
        EndpointRequest: {
            Address: originationNumber,
            ChannelType: 'SMS',
            OptOut: 'NONE',
            Attributes: {
                OptInTimestamp: [timestamp]
            },
        }
    };

    try {
        // Send the UpdateEndpointCommand to update the endpoint's opt-in status
        const updateEndpointResponse = await pinClient.send(new
 UpdateEndpointCommand(params));
        console.log(updateEndpointResponse);
        console.log(`Successfully changed the opt status of endpoint ID
 ${endpointId}`);
    } catch (error) {
        console.error('An error occurred while updating endpoint:', error);
        throw error; // Rethrow the error to handle it upstream
    }
}
```

7. [Environment variables (環境変数)] で、以下の操作を実行します。

   - 最初の行で、**projectId** のキーで変数を作成します。次に、Amazon Amazon Pinpoint プロジェクトの作成 で作成したプロジェクトの一意の ID に値を設定します。
   - 2 行目で、**region** のキーで変数を作成します。次に、**us-east-1** または **us-west-2** など、Amazon Pinpoint を使用するリージョンに値を設定します。
   - 3 行目で、**confirmKeyword** のキーで変数を作成します。次に、 値を「双方向 SMS を有効にする」で作成した確認キーワードに設定します。

> **Note**
>
> キーワードでは大文字と小文字は区別されません。この関数は、受信メッセージを小文字に変換します。

終了したら、[環境変数] セクションは、次のイメージに示す例のようになります。



8. ページの上部で、[保存] を選択します。

関数をテストする

関数を作成した後、テストして適切に設定されているかどうかを確認します。また、作成した IAM ロールに適切なアクセス許可があることを確認します。

関数をテストするには

1. [テスト] を選択します。
2. [テストイベントを設定する] ページで、以下の操作を行います。
   a. Create new test eventを選択します。
   b. [Event name] で、**MyResponse** など、テストイベントの名前を入力します。
   c. コードエディタのサンプルコードを消去します。次のコードを貼り付けます。

```
{
```

```
  "Records":[
    {
      "Sns":{
        "Message":"{\"originationNumber\":\"+12065550142\",\"messageBody\":
\"Yes\"}",
        "Timestamp":"2019-02-20T17:47:44.147Z"
      }
    }
  ]
}
```

上記のコード例では、originationNumber 属性の値を、前の Lambda 関数をテストしたときに使用した電話番号と置き換えます。の値を、「双方向 SMS を有効にする」で指定した双方向 SMS キーワードmessageBodyに置き換えます。オプションで、Timestamp の値を現在の日付と時刻と置き換えることができます。

d. [作成] を選択します。

3. [テスト] を再度選択します。
4. [実行結果: 成功] で、[詳細] を選択します。[ログ出力] セクションで、関数の出力を確認します。関数がエラーなしで実行されたことを確認します。
5. Amazon Pinpoint コンソール (https://console.aws.amazon.com/pinpoint/) を開きます。
6. 「すべてのプロジェクト」ページで、Amazon Pinpoint プロジェクトの作成」で作成したプロジェクトを選択します。
7. ナビゲーションペインの [Segments] を選択します。[セグメント] ページで、[セグメントの作成] を選択します。
8. [セグメントグループ 1] の [Add filters to refine your segment] で、[Filter by user] を選択します。
9. ユーザー属性の選択 で、 を選択しますFirstName。次に、[Choose values] に、テストイベントで指定した名を選択します。

   [Segment estimate (セグメントの見積もり)] セクションに、1 つの適格なエンドポイントがあり、合計エンドポイントが 1 つであることが表示されます。

Next:Amazon API Gateway を設定する

ステップ 4: Amazon API Gateway を設定する

このセクションでは、Amazon API Gateway を使用して新しい API を作成します。本ソリューションでデプロイする登録フォームは、この API を呼び出します。API Gateway は、登録フォームでキャプチャされた情報を、Lambda 関数の作成 で作成した Lambda 関数に渡します。

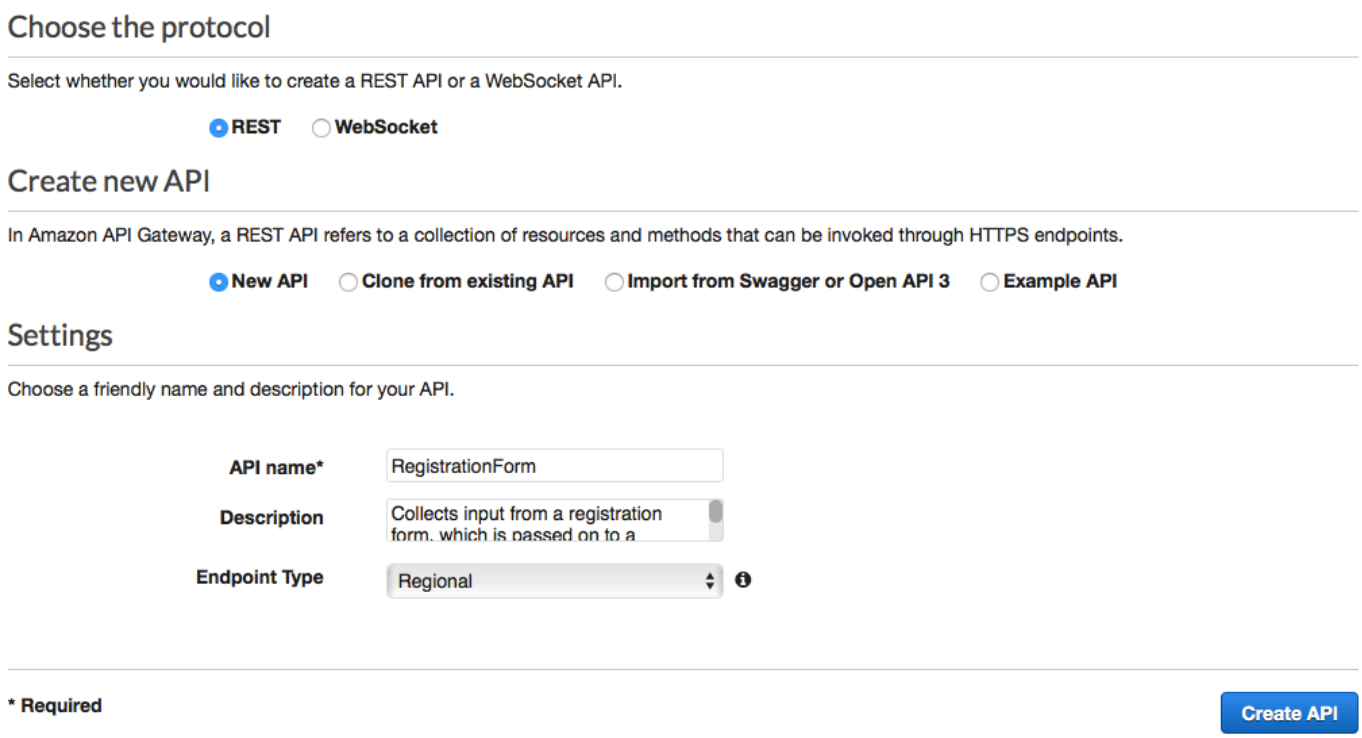
API を作成する

まず、API Gateway に新しい API を作成する必要があります。次の手順では、新しい REST API を作成する方法を示します。

新しい API を作成するには

1. API Gateway コンソール (「https://console.aws.amazon.com/apigateway」) を開きます。
2. API の作成 を選択します。以下の選択を行います。
   - [Choose the protocol (プロトコルの選択)] で、[REST] を選択します。
   - [Create new API (新しい API の作成)] で、[New API (新しい API)] を選択します。
   - [設定] の [名前] に、**`RegistrationForm`** などの名前を入力します。[説明] に、必要に応じて、API の目的を説明するテキストを入力します。[エンドポイントタイプ] で、[リージョン] を選択します。続いて、[API の作成] を選択します。

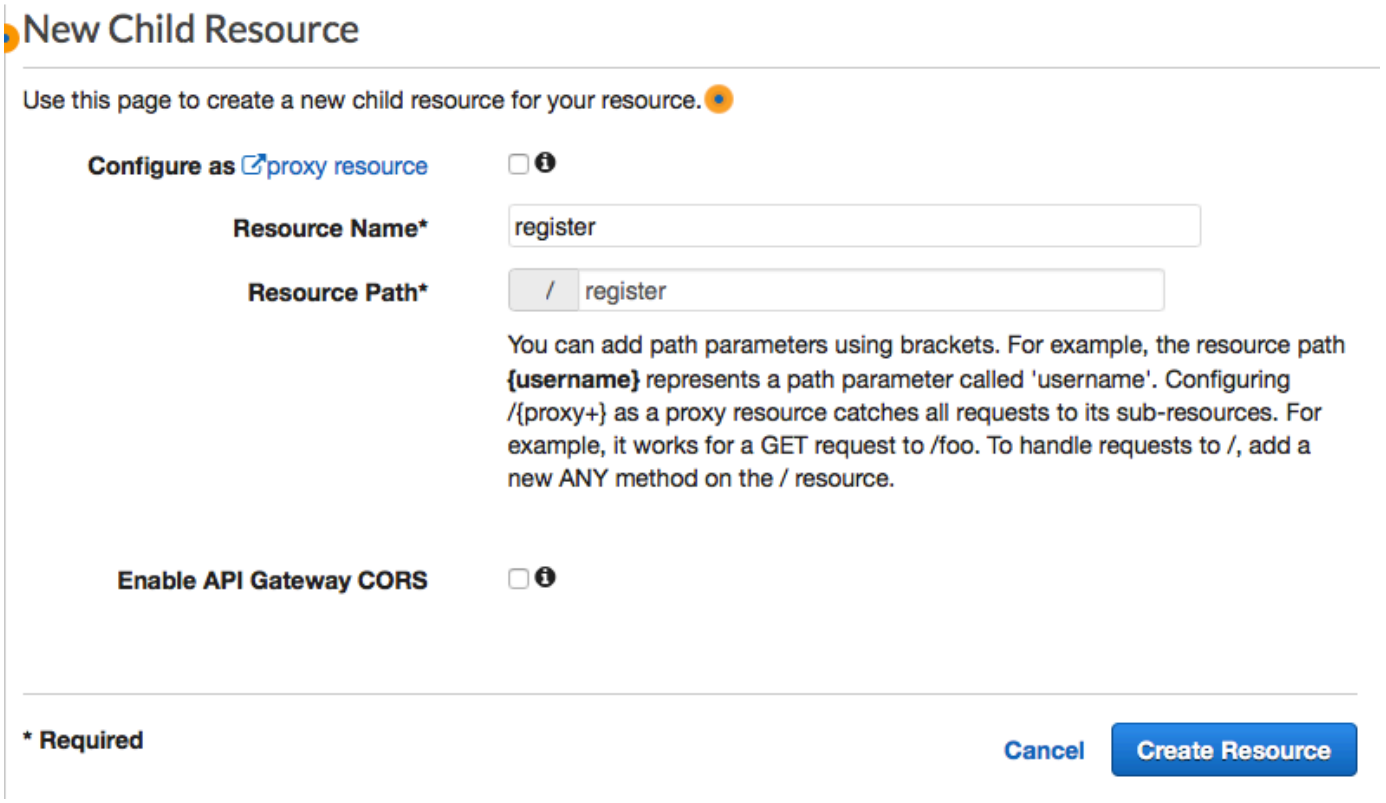
   これらの設定の一例を次のイメージに示します。



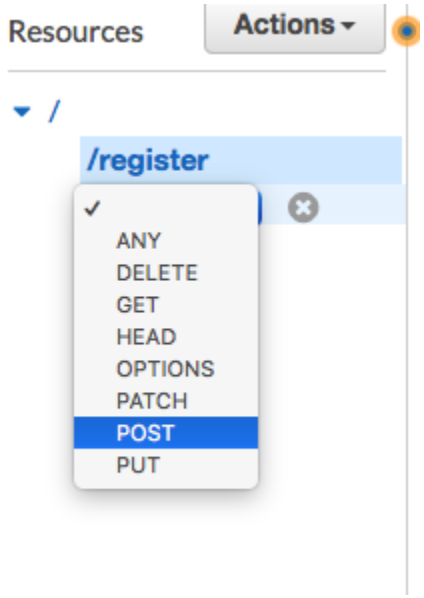
リソースを作成する

API を作成したので、リソースの追加を開始することができます。その後、リソースに POST メソッドを追加して、API Gateway にこのメソッドから受け取ったデータを Lambda 関数に渡すように指示します。

1. [アクション] メニューで、[リソースの作成] を選択します。次のイメージに示すように、[New Child Resource] ペインの [リソース名] に **`register`** を入力します。[リソースの作成] を選択します。



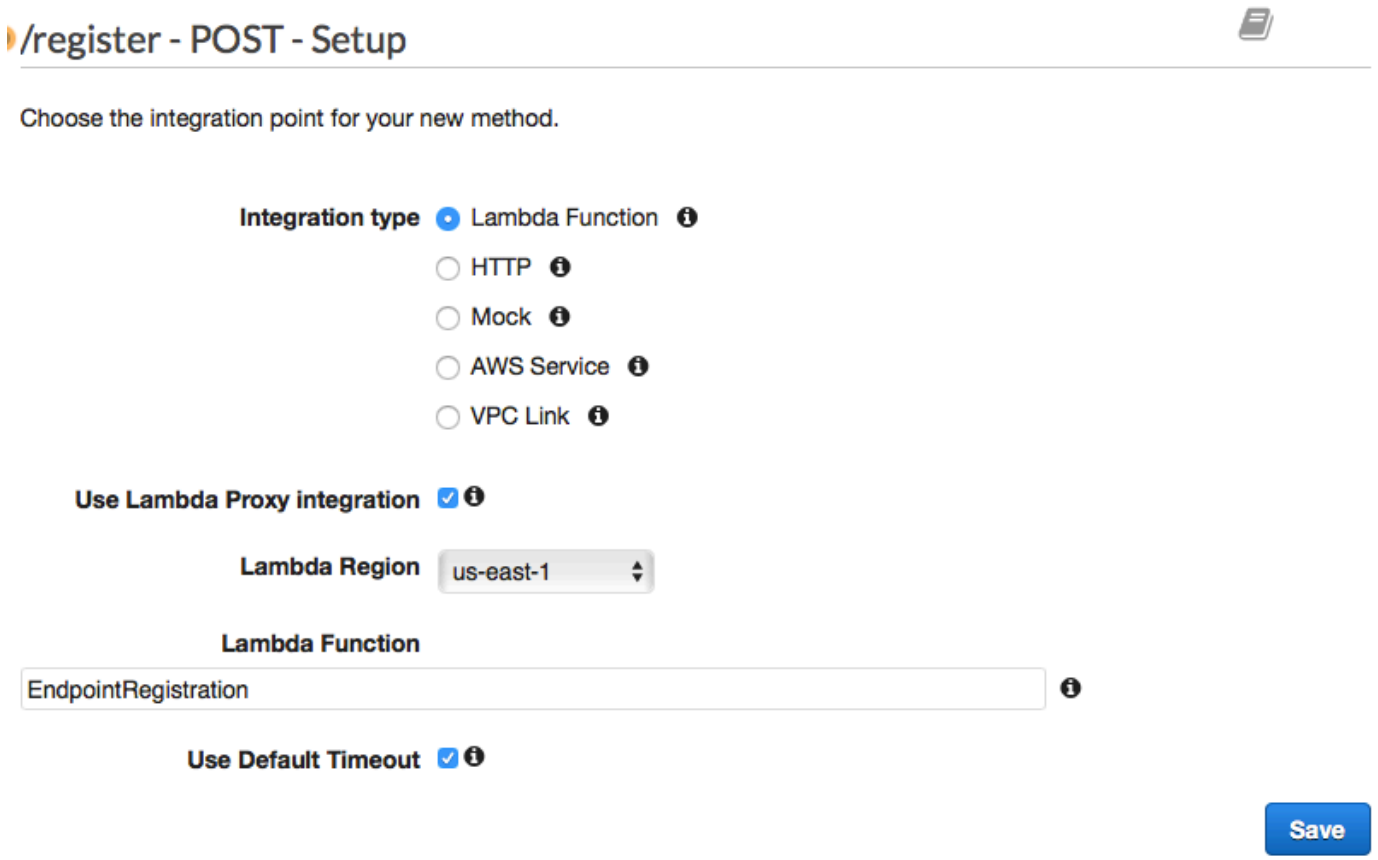
2. [アクション] メニューから、[メソッドの作成] を選択します。次のイメージに示すように、表示されたメニューから、[POST] を選択します。次に、[チェックマーク] ( ) ボタンを選択します。



3. [/register - POST - Setup (/register - POST - セットアップ)] ペインで、以下を指定します。

- [Integration type (統合タイプ)] で、[Lambda Function (Lambda 関数)] を選択します。
- [Use Lambda Proxy integration] を選択します。
- [Lambda Region]で、Lambda 関数を作成したリージョンを選択します。
- Lambda 関数 で RegisterEndpoint、Lambda 関数の作成 で作成した関数を選択します。

これらの設定の一例を次のイメージに示します。



[保存] を選択します。表示されたウィンドウで、[OK] を選択して、API Gateway に Lambda 関数を実行するためのアクセス許可を付与します。

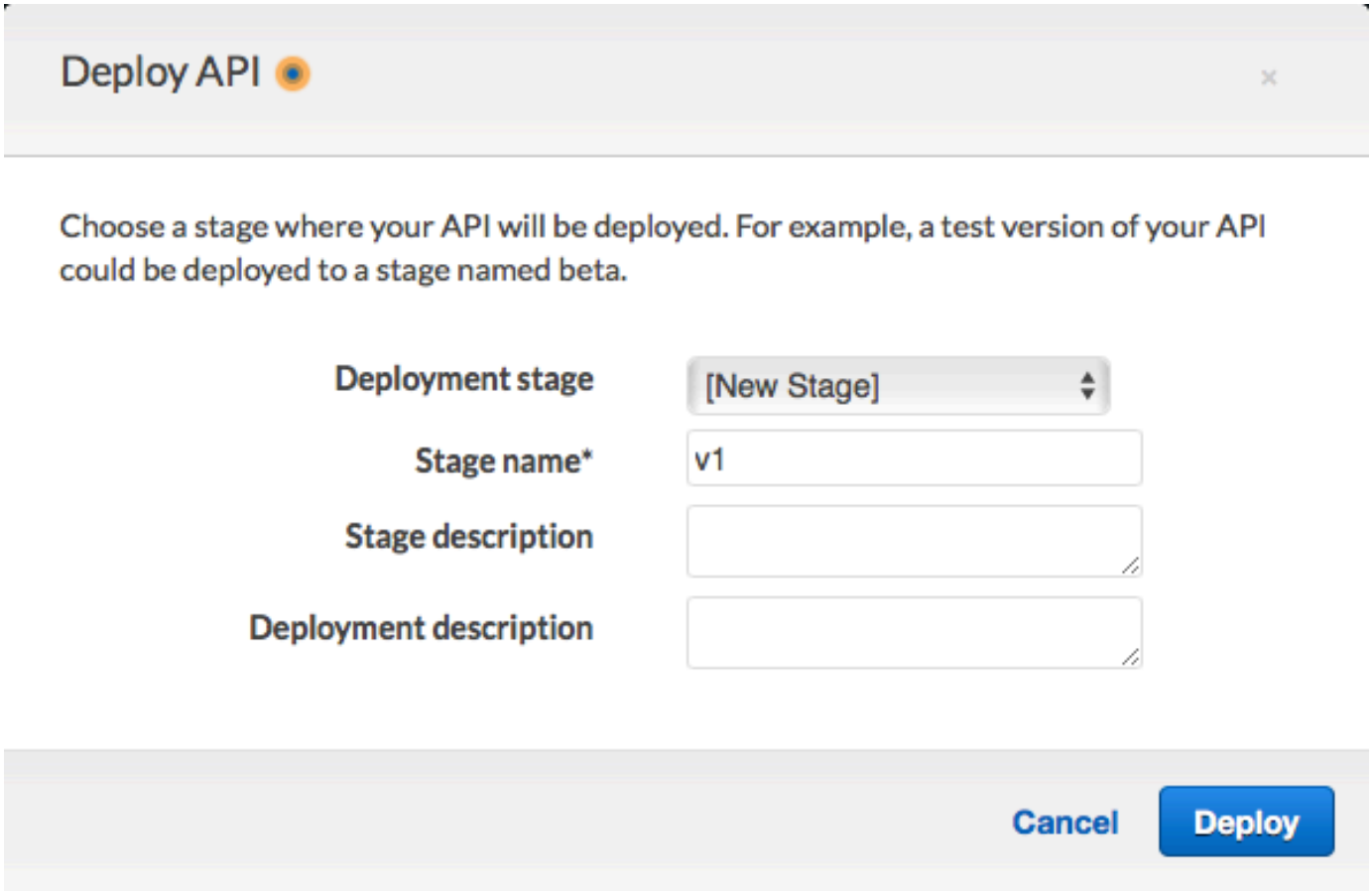
API をデプロイする

これで、API を使用できるようになりました。この時点で、パブリックアクセス可能なエンドポイントを作成するために、これをデプロイする必要があります。

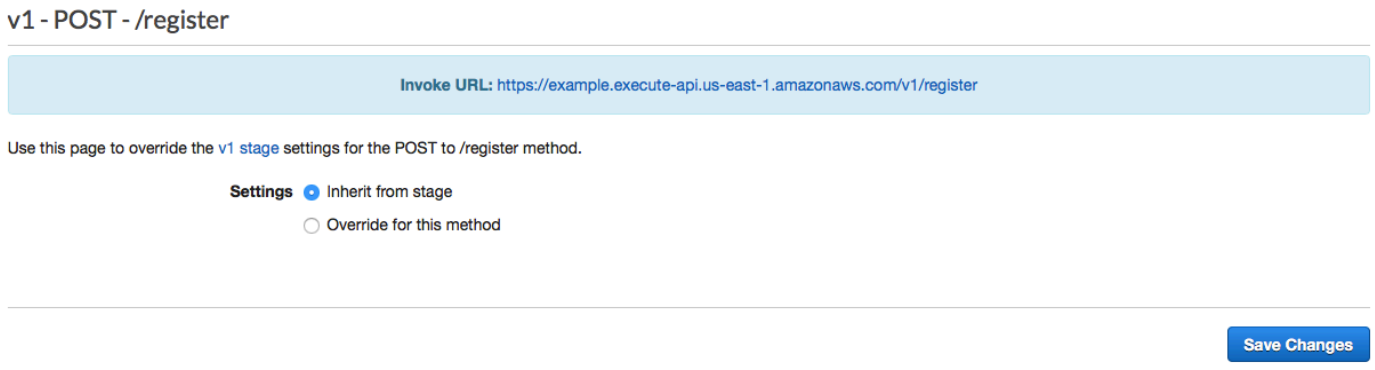
1. [アクション] メニューから、[API のデプロイ] を選択します。[API のデプロイ] ウィンドウで、以下の選択を行います。

- [デプロイされるステージ] で、[新しいステージ] を選択します。
- [Stage name (ステージ名)] に **v1** と入力します。
- デプロイを選択します。

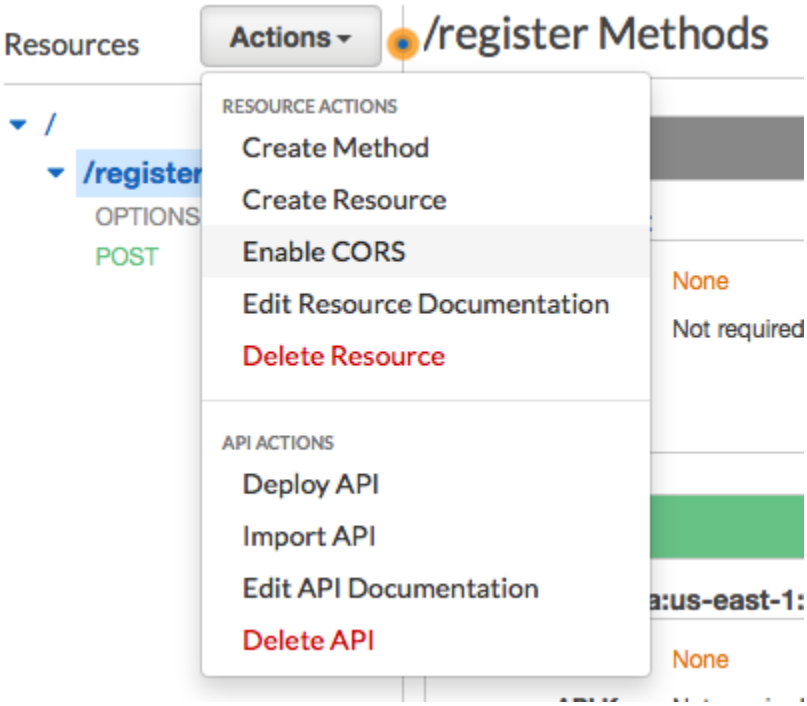
これらの選択の一例を次のイメージに示します。



2. [v1 Stage Editor] ペインで、[/register] リソースを選択してから [POST] メソッドを選択します。次のイメージに示すように、[呼び出し URL] の横に表示されているアドレスをコピーします。



3. ナビゲーションペインで、[リソース] を選択します。リソースのリストで、[/register] リソースを選択します。最後に、次のイメージに示すように、[アクション] メニューで、[CORS を有効にする] を選択します。



4. [CORS を有効にする] ペインで[CORS を有効にして既存の CORS ヘッダーを置換] を選択します。

次の手順: ウェブフォームを作成してデプロイする

ステップ 5: ウェブフォームを作成してデプロイする

これで、 AWS サービスを使用するこのソリューションのすべてのコンポーネントが導入されました。最後のステップは、お客様のデータをキャプチャするウェブフォームを作成してデプロイすることです。

JavaScript フォームハンドラーを作成する

このセクションでは、次のセクションで作成したウェブフォームのコンテンツを解析する JavaScript 関数を作成します。コンテンツを解析した後、この関数は Amazon API Gateway のセットアップ で作成した API にデータを送信します。

フォームハンドラを作成するには

1. テキストエディタで新規ファイルを作成します。

2. エディタで、次のコードを貼り付けます。

```
$(document).ready(function() {

  // Handle form submission.
  $("#submit").click(function(e) {

    var firstName = $("#firstName").val(),
        lastName = $("#lastName").val(),
        source = window.location.pathname,
        optTimestamp = undefined,
        utcSeconds = Date.now() / 1000,
        timestamp = new Date(0),
        phone = $("#areaCode").val()
              + $("#phone1").val()
              + $("#phone2").val();

    e.preventDefault();

    if (firstName == "") {
      $('#form-response').html('<div class="mt-3 alert alert-info"
 role="alert">Please enter your first name.</div>');
    } else if (lastName == "") {
      $('#form-response').html('<div class="mt-3 alert alert-info"
 role="alert">Please enter your last name.</div>');
    } else if (phone.match(/[^0-9]/gi)) {
      $('#form-response').html('<div class="mt-3 alert alert-info"
 role="alert">Your phone number contains invalid characters. Please check the phone
 number that you supplied.</div>');
    } else if (phone.length < 10) {
      $('#form-response').html('<div class="mt-3 alert alert-info"
 role="alert">Please enter your phone number.</div>');
    } else if (phone.length > 10) {
      $('#form-response').html('<div class="mt-3 alert alert-info"
 role="alert">Your phone number contains too many digits. Please check the phone
 number that you supplied.</div>');
    } else {
      $('#submit').prop('disabled', true);
      $('#submit').html('<span class="spinner-border spinner-border-sm"
 role="status" aria-hidden="true"></span>  Saving your preferences</button>');

      timestamp.setUTCSeconds(utcSeconds);
```

```
      var data = JSON.stringify({
        'destinationNumber': phone,
        'firstName': firstName,
        'lastName': lastName,
        'source': source,
        'optTimestamp': timestamp.toString()
      });

      $.ajax({
        type: 'POST',
        url: 'https://example.execute-api.us-east-1.amazonaws.com/v1/register',
        contentType: 'application/json',
        data: data,
        success: function(res) {
          $('#form-response').html('<div class="mt-3 alert alert-success"
 role="alert"><p>Congratulations! You&apos;ve successfully registered for SMS
 Alerts from ExampleCorp.</p><p>We just sent you a message. Follow the instructions
 in the message to confirm your subscription. We won&apos;t send any additional
 messages until we receive your confirmation.</p><p>If you decide you don&apos;t
 want to receive any additional messages from us, just reply to one of our messages
 with the keyword STOP.</p></div>');
          $('#submit').prop('hidden', true);
          $('#unsubAll').prop('hidden', true);
          $('#submit').text('Preferences saved!');
        },
        error: function(jqxhr, status, exception) {
          $('#form-response').html('<div class="mt-3 alert alert-danger"
 role="alert">An error occurred. Please try again later.</div>');
          $('#submit').text('Save preferences');
          $('#submit').prop('disabled', false);
        }
      });
    }
  });
});
```

3. 前の例では、*https://example.execute-api.us-east-1.amazonaws.com/v1/register* を API をデプロイ で取得した呼び出し URL に置き換えます。
4. ファイルを保存します。

フォームファイルを作成する

このセクションでは、お客様が SMS プログラムを登録するために使用するフォームを格納する HTML ファイルを作成します。このファイルは、前のセクションで作成した JavaScript フォームハンドラーを使用して、フォームデータを Lambda 関数に送信します。

> **Important**
>
> ユーザーがこのフォームを送信すると、複数の Amazon Pinpoint API オペレーションを呼び出す Lambda 関数をトリガーします。悪意のあるユーザーが、多数のリクエストが発生する原因になる攻撃をフォームで起動する可能性があります。このソリューションを本番環境でのユースケースで使用する予定がある場合は、Google reCAPTCHA などのシステムを使用してセキュリティで保護する必要があります。

フォームを作成するには

1. テキストエディタで新規ファイルを作成します。
2. エディタで、次のコードを貼り付けます。

```
<!doctype html>
<html lang="en">

<head>
  <!-- Meta tags required by Bootstrap -->
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-
fit=no">

  <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/
css/bootstrap.min.css" integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/
iJTQUOhcWr7x9JvoRxT2MZw1T" crossorigin="anonymous">
  <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-
q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo"
 crossorigin="anonymous"></script>
  <script src="https://cdnjs.cloudflare.com/ajax/libs/
popper.js/1.14.7/umd/popper.min.js" integrity="sha384-
UO2eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dIHNDz0W1"
 crossorigin="anonymous"></script>
```

```
  <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/
bootstrap.min.js" integrity="sha384-JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/
nJGzIxFDsf4x0xIM+B07jRM" crossorigin="anonymous"></script>
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></
script>

  <script type="text/javascript" src="SMSFormHandler.js"></script>
  <title>SMS Registration Form</title>
</head>

<body>
  <div class="container">
    <div class="row justify-content-center mt-3">
      <div class="col-md-6">
        <h1>Register for SMS Alerts</h1>
        <p>Enter your phone number below to sign up for PromotionName messages from
 ExampleCorp.</p>
        <p>We don't share your contact information with anyone else. For more
 information, see our <a href="http://example.com/privacy">Privacy Policy</a>.</p>
        <p>ExampleCorp alerts are only available to recipients in the United
 States.</p>
      </div>
    </div>
    <div class="row justify-content-center">
      <div class="col-md-6">
        <form>
          <div class="form-group">
            <label for="firstName" class="font-weight-bold">First name</label>
            <input type="text" class="form-control" id="firstName"
 placeholder="Your first name" required>
          </div>
          <div class="form-group">
            <label for="lastName" class="font-weight-bold">Last name</label>
            <input type="text" class="form-control" id="lastName" placeholder="Your
 last name" required>
          </div>
          <label for="areaCode" class="font-weight-bold">Phone number</label>
          <div class="input-group">
            <span class="h3">( </span>
            <input type="tel" class="form-control" id="areaCode" placeholder="Area
 code" required>
            <span class="h3"> ) </span>
            <input type="tel" class="form-control" id="phone1" placeholder="555"
 required>
```

```
            <span class="h3"> - </span>
            <input type="tel" class="form-control" id="phone2" placeholder="0199"
 required>
          </div>
          <div id="form-response"></div>
          <button id="submit" type="submit" class="btn btn-primary btn-block
 mt-3">Submit</button>
        </form>
      </div>
    </div>
    <div class="row mt-3">
      <div class="col-md-12 text-center">
        <small class="text-muted">Copyright © 2019, ExampleCorp or its
 affiliates.</small>
      </div>
    </div>
  </div>
</body>

</html>
```

3. 前の例では、*SMS FormHandler.js* を前のセクションで作成したフォームハンドラー JavaScript ファイルへのフルパスに置き換えます。
4. ファイルを保存します。

フォームファイルをアップロードする

HTML フォームと JavaScript フォームハンドラーを作成したので、最後のステップは、これらのファイルをインターネットに公開することです。このセクションでは、既存のウェブホスティングプロバイダーがあることを前提としています。既存のホスティングプロバイダーがない場合は、Amazon Route 53、Amazon Simple Storage Service (Amazon S3)、および Amazon を使用してウェブサイトを起動できます CloudFront。詳細については、「静的ウェブサイトをホスティングする」を参照してください。

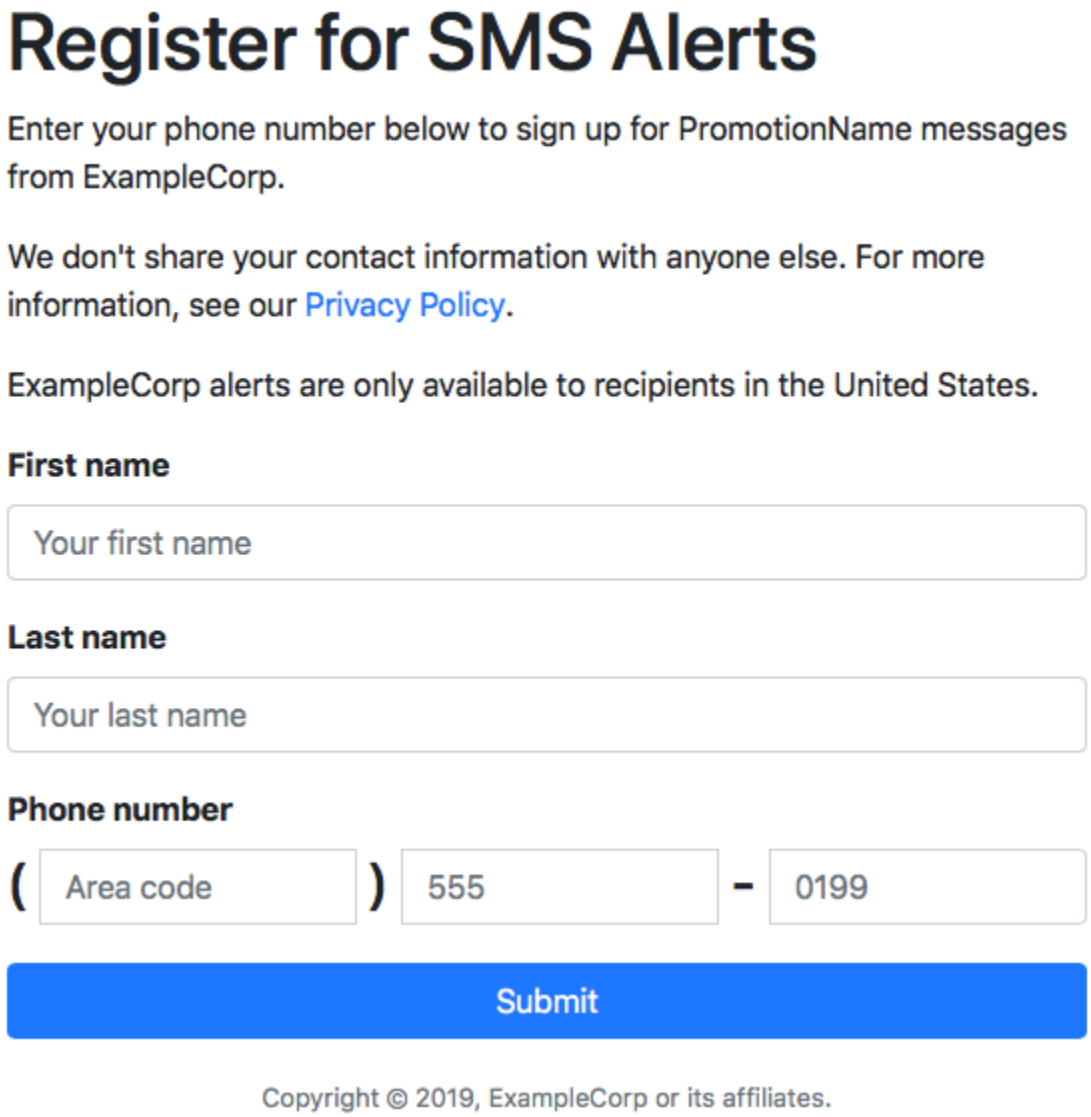
別のウェブホスティングプロバイダーを使用する場合は、ウェブページの発行の詳細について、プロバイダーのドキュメントを参照してください。

フォームをテストする

フォームを発行した後は、想定どおりに動作するかどうかを確認するためにいくつかのテストイベントを送信する必要があります。

登録フォームをテストするには

1. ウェブブラウザで、登録フォームのアップロード先に移動します。JavaScript「フォームハンドラーの作成」のコード例を使用した場合、次の図の例に似たフォームが表示されます。



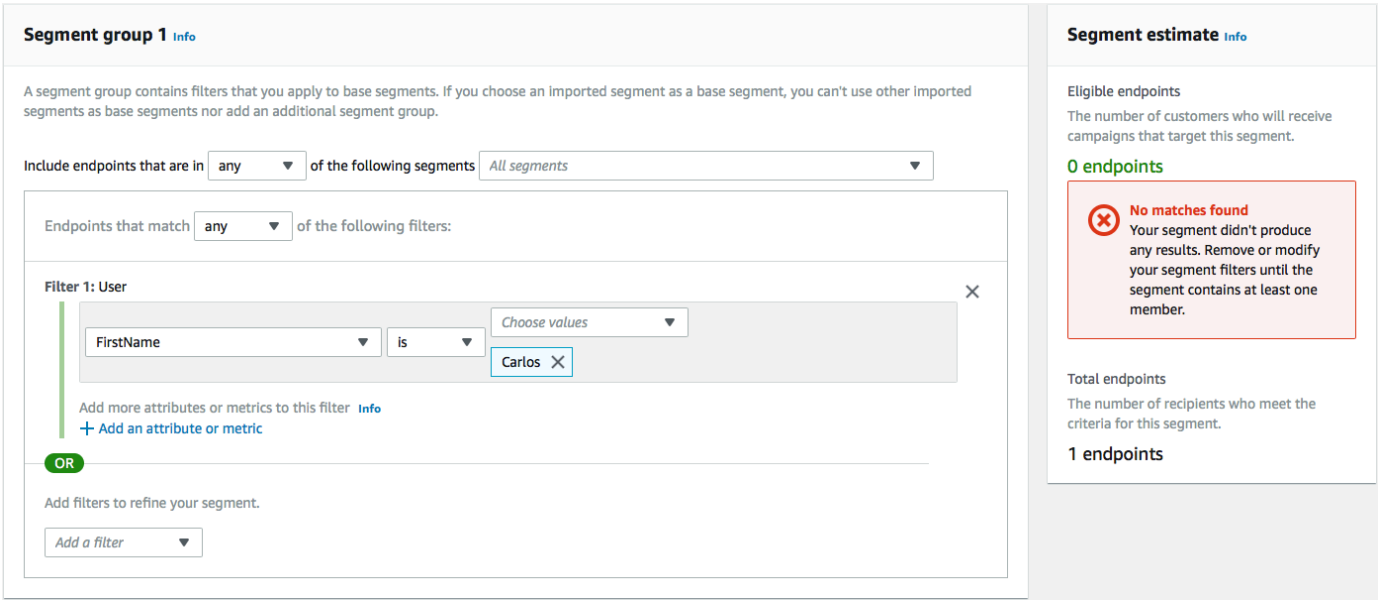
2. 連絡先情報を、[名]、[姓]、[電話番号] の各フィールドに入力します。

> **Note**
>
> フォームを送信すると、Amazon Pinpoint により指定した電話番号へのメッセージの送信が試行されます。この機能のため、ソリューションをテストするには、最初から最後まで実際の電話番号を使用する必要があります。
> Lambda 関数の作成 で Lambda 関数をテストした場合、Amazon Pinpoint プロジェクトにはすでに少なくとも 1 つのエンドポイントが含まれています。このフォームをテスト

するときは、フォームに別の電話番号を送信するか、DeleteEndpoint API オペレーションを使用して既存のエンドポイントを削除する必要があります。

3. 指定した電話番号に関連付けられたデバイスを確認し、そのデバイスがテストメッセージを受信したことを確認します。
4. Amazon Pinpoint コンソール (https://console.aws.amazon.com/pinpoint/) を開きます。
5. 「すべてのプロジェクト」ページで、Amazon Pinpoint プロジェクトの作成」で作成したプロジェクトを選択します。
6. ナビゲーションペインの [Segments] を選択します。[セグメント] ページで、[セグメントの作成] を選択します。
7. [セグメントグループ 1] の [Add filters to refine your segment] で、[Filter by user] を選択します。
8. ユーザー属性の選択 で、 を選択しますFirstName。次に、[Choose values] に、フォームを送信したときに指定した名前を選択します。

   [Segment estimate] セクションに、次の例に示すように、適格なエンドポイントがゼロであり、エンドポイントが 1 つあることが表示されます。この結果は正常です。Lambda 関数が新しいエンドポイントを作成すると、デフォルトでエンドポイントがオプトアウトされます。



9. メッセージを受信したデバイスで、「双方向 SMS を有効にする」で指定した双方向 SMS キーワードを使用してメッセージに返信します。Amazon Pinpoint はすぐに応答メッセージを送信します。

10. Amazon Pinpoint コンソールで、ステップ 4 ~ 8 を繰り返します。今回は、セグメントを作成するときに、1 つの適格なエンドポイントと、合計 1 つのエンドポイントが表示されます。エンドポイントがオプトインされたため、これは想定される動作です。

次のステップ

このチュートリアルを完了すると、以下のことが完了していることを示します。

- Amazon Pinpoint プロジェクトを作成し、SMS チャネルを設定して、専用のロングコードを取得する。
- 最小特権を使用する IAM ポリシーを作成して、アクセス権を付与し、そのポリシーとロールを関連付けます。
- PhoneNumberValidate、UpdateEndpoint、および SendMessages オペレーションを使用する 2 つの Lambda 関数を Amazon Pinpoint API に作成します。
- API Gateway を使用して REST API を作成します。
- お客様の連絡先情報を収集するウェブベースのフォームを作成およびデプロイする。
- ソリューションでテストを実行し、動作することを確認する。

このセクションでは、このソリューションを使用して収集するお客様の情報を使用できる方法をいくつか説明します。また、このソリューションをカスタマイズして、独自のユースケースに適合させる方法の推奨事項をいくつかご紹介します。

カスタマーセグメントの作成

このフォームを介して収集するすべてのカスタマーの詳細は、エンドポイントとして保存されます。このソリューションでは、セグメンテーションのために使用できる複数の属性を含むエンドポイントを作成します。

たとえば、このソリューションでは、`Source` と呼ばれるエンドポイント属性をキャプチャします。この属性には、フォームがホストされた場所への完全パスが含まれています。セグメントを作成するとき、エンドポイントでセグメントをフィルタリングすることができ、`Source` 属性を選択して、フィルターをさらに詳細なものにすることができます。

`Source` 属性に基づいてセグメントを作成すると、複数の方法で役立ちます。まず、SMS メッセージ受信にサインアップしたお客様のセグメントを迅速に作成できます。また、Amazon Pinpoint のセグメンテーションツールでは、メッセージの受信を選択していないエンドポイントが自動的に除外されます。

Source 属性は、複数の異なる場所から登録フォームをホスティングする場合にも役立ちます。たとえば、ユーザーのマーケティングマテリアルが、1 つの場所でホストされているフォームを参照し、ユーザーのウェブサイトを閲覧中にこのフォームを見つけたお客様が、別の場所でホストされているバージョンを見る可能性があります。この場合、マーケティングマテリアルを見た後に、フォームに記入したお客様のソース属性が、ウェブサイトでフォームを見つけた後で記入したお客様のソース属性と異なることになります。この違いを利用して、異なるセグメントを作成し、それぞれの受信者にカスタマイズしたコミュニケーションを送信できます。

パーソナライズされたキャンペーンメッセージの送信

セグメントを作成した後、これらのセグメントにキャンペーンの送信を開始することができます。キャンペーンメッセージを作成するとき、メッセージに含めるエンドポイント属性を指定することで、パーソナライズできます。たとえば、このソリューションで使用されるウェブフォームでは、カスタマーが氏名を入力する必要があります。これらの値は、エンドポイントに関連付けられているユーザーレコードに保存されます。

たとえば、GetEndpoint API オペレーションを使用して、このソリューションを使用して作成されたエンドポイントに関する情報を取得する場合、次の例に似たセクションが表示されます。

```
...
"User": {
  "UserAttributes": {
    "FirstName": [
      "Carlos"
    ],
    "LastName": [
      "Salazar"
    ]
  }
}
...
```

これらの属性の値をキャンペーンメッセージに含める場合は、ドット表記を使用して属性を参照することができます。次に、参照全体を二重の中括弧で囲みます。例えば、キャンペーンメッセージにそれぞれの受信者の名を含めるには、メッセージに文字列 {{User.UserAttributes.FirstName}} を含めます。Amazon Pinpoint によりメッセージが送信されるとき、文字列が FirstName 属性の値に置き換えられます。

フォームを使用して追加情報を収集する

このソリューションを変更して、登録フォームの追加情報を収集できます。たとえば、Endpoint リソースに Location.City、Location.Country、Location.Region、Location.PostalCode フィールドを生成して、お客様に住所を入力するよう依頼して、住所データを使用することができます。登録フォームでアドレスを収集すると、エンドポイントに含まれる情報がより正確になる可能性があります。この変更を行うには、ウェブフォームに適切なフィールドを追加する必要があります。また、新しい値を渡すためにフォームの JavaScript コードを変更する必要があります。最後に、新しい着信情報を処理するためのエンドポイントを作成する Lambda 関数を変更する必要があります。

他のチャネルで連絡先情報を収集するように、フォームを変更することもできます。たとえば、フォームを使用して、お客様の電話番号に加えて E メールアドレスを収集することができます。この変更を行うには、ウェブフォームの HTML と JavaScript を変更する必要があります。また、2 つの個別のエンドポイント (1 つは E メールのエンドポイント、もう 1 つは SMS のエンドポイント用) を作成するように、エンドポイントを作成する Lambda 関数も変更する必要があります。また、Lambda 関数を変更して、User.UserId 属性に一意の値を生成し、その値を両方のエンドポイントのに関連付ける必要もあります。

監査のために追加の属性を記録する

このソリューションは、エンドポイントを作成および更新するときに 2 つの重要な属性を記録します。まず、1 つ目の Lambda 関数が最初にエンドポイントを作成し、Attributes.Source 属性にフォーム自体の URL を記録します。お客様がメッセージに応答した場合、2 番目の Lambda 関数により Attributes.OptInTimestamp 属性が作成されます。この属性には、お客様がメッセージの受信に同意した正確な日時が含まれています。

これら両方のフィールドは、モバイルキャリアまたは規制機関からお客様の同意を得ている証拠の提出を依頼された場合に、役立ちます。この情報は、GetEndpoint API オペレーションを使用して、いつでも取得することができます。

また、Lambda 関数を変更して、登録リクエストの送信元の IP アドレスなど、監査に役立つ可能性がある追加のデータを記録することもできます。

Amazon Pinpoint とアプリケーションの統合

クライアントコードに Amazon Pinpoint を統合して、ユーザーを理解して交流します。

統合後、ユーザーがアプリケーションを起動すると、エンドポイントを追加したり、更新するために Amazon Pinpoint サービスに接続します。エンドポイントは、ユーザーのデバイス、E メールアドレスや電話番号など、メッセージを送信できる宛先を表します。

また、アプリケーションには、使用状況のデータやイベントが表示されます。Amazon Pinpoint コンソールのイベントデータを表示して、ユーザー数、アプリケーションの使用頻度、いつ使用しているかなど、知ることができます。

アプリケーションによってエンドポイントとイベントが提供されたら、この情報を使用して特定の閲覧者やセグメントを対象としたカスタムメッセージングキャンペーンができます。(キャンペーンを作成しないで、受信者のシンプルなリストに直接メッセージを送信することもできます)。

このセッションのトピックを使用して、Amazon Pinpoint をモバイルアプリケーションやウェブアプリケーションと統合します。これらのトピックには、、Android JavaScript、Swift、または Flutter アプリケーションと統合するためのコード例と手順が含まれています。アプリの統合を開始するには、「the section called “AWS Amplify を使用してフロントエンドアプリケーションを接続する”」を参照してください。

クライアントのほかには、サポートされている AWS SDK あるいは Amazon Pinpoint API を使用して、エンドポイントのインポート、イベントデータのエクスポート、カスタマーセグメントの定義、キャンペーンの作成と実行などができます。

トピック

- AWS SDK での Amazon Pinpoint の使用
- AWS Amplify を使用してフロントエンドアプリケーションを Amazon Pinpoint に接続する
- アプリケーションでエンドポイントを登録する
- アプリケーションでのイベントのレポート
- プッシュ通知の処理

AWS SDK での Amazon Pinpoint の使用

AWS Software Development Kit (SDKsは、多くの一般的なプログラミング言語で使用できます。各 SDK には、デベロッパーが好みの言語でアプリケーションを簡単に構築できるようにする API、コード例、およびドキュメントが提供されています。

SDK ドキュメント	コード例
AWS SDK for C++	AWS SDK for C++ コード例
AWS CLI	AWS CLI コード例
AWS SDK for Go	AWS SDK for Go コード例
AWS SDK for Java	AWS SDK for Java コード例
AWS SDK for JavaScript	AWS SDK for JavaScript コード例
AWS SDK for Kotlin	AWS SDK for Kotlin コード例
AWS SDK for .NET	AWS SDK for .NET コード例
AWS SDK for PHP	AWS SDK for PHP コード例
AWS Tools for PowerShell	PowerShell コード例のツール
AWS SDK for Python (Boto3)	AWS SDK for Python (Boto3) コード例
AWS SDK for Ruby	AWS SDK for Ruby コード例
AWS SDK for Rust	AWS SDK for Rust コード例
AWS SDK for SAP ABAP	AWS SDK for SAP ABAP コード例
AWS SDK for Swift	AWS SDK for Swift コード例

Amazon Pinpoint 固有の例については、「AWS SDK を使用した Amazon Pinpoint のコード例」を参照してください。

> **可用性の例**
>
> 必要なものが見つからなかった場合。このページの下側にある [Provide feedback (フィードバックを送信)] リンクから、コードの例をリクエストしてください。

AWS Amplify を使用してフロントエンドアプリケーションを Amazon Pinpoint に接続する

AWS Amplify を使用してアプリを AWS を統合します。Swift アプリケーションについては、Amplify for Swift ドキュメントの「Getting started」を参照してください。Android アプリケーションについては、Amplify for Android SDK ドキュメントの「Getting started」を参照してください。React Native アプリケーションについては、Amplify JavaScript ドキュメントの「Getting started」を参照してください。Flutter アプリケーションについては、Flutter SDK ドキュメントの「Getting started」を参照してください。このトピックは次の目的に対して有用です。

- バックエンドリソースをセットアップする
- Amplify ライブラリを使用してアプリをバックエンドリソースに接続する

フロントエンドアプリを Amazon Pinpoint に接続して分析、アプリ内メッセージング、プッシュ通知を行う方法の詳細については、「AmplifyAWS」を参照してください。

次のステップ

AWS Amplify をアプリケーションと統合しました。次に、コードを更新してユーザーのデバイスをエンドポイントとして登録します。「アプリケーションでエンドポイントを登録する」を参照してください。

アプリケーションでエンドポイントを登録する

ユーザーがセッションを開始すると (モバイルアプリケーションを起動するなど)、モバイルまたはウェブアプリケーションは、エンドポイントを自動的に Amazon Pinpoint に登録できます。このエンドポイントは、ユーザーがセッションを開始したデバイスを表します。これにはデバイスを説明する属性が含まれており、また定義するカスタム属性を含めることもできます。エンドポイントはまた、E メールアドレスや携帯電話番号といった、顧客とやり取りするための他の方法を表します。

アプリケーションにエンドポイントを登録したら、エンドポイントの属性に基づいた対象者のセグメントができます。続いて、このセグメントをカスタマイズしたメッセージキャンペーンと関連付けることができます。Amazon Pinpoint コンソールで [Analytics] ページを使用して、[New endpoints] や [Daily active endpoints] など、エンドポイント登録およびアクティビティに関するグラフを表示することもできます。

単一のユーザー ID を複数のエンドポイントに割り当てることができます。ユーザー ID は単一のユーザーを表しますが、このユーザー ID に割り当てられる各エンドポイントはユーザーのデバイスのいずれかを表します。エンドポイントにユーザー ID を割り当てると、[Daily active users] や [Monthly active users] など、ユーザーアクティビティに関するグラフをコンソールに表示することができます。

開始する前に

まだ統合していない場合は、SDK for Android または iOS 用の AWS Mobile SDK、あるいは AWS Amplify JavaScript ライブラリをアプリケーションに統合します。「AWS Amplify を使用してフロントエンドアプリケーションを Amazon Pinpoint に接続する」を参照してください。

Android または iOS 用 AWS Mobile SDK でエンドポイントを登録する

Android または iOS 用 AWS Mobile SDK を使用して、エンドポイントの登録とカスタマイズを行うことができます。詳細情報、およびコード例の表示については、次のドキュメントを参照してください。

- Android SDK ドキュメントの「アプリケーションでエンドポイントを登録する」。
- iOS SDK ドキュメントの「アプリケーションでエンドポイントを登録する」。

AWS Amplify JavaScript ライブラリでエンドポイントを登録する

AWS Amplify JavaScript ライブラリを使用して、アプリ内のエンドポイントを登録および更新することができます。詳細とコード例については、『AWS Amplify JavaScriptドキュメント』の「エンドポイントの更新」を参照してください。

次のステップ

アプリケーションを更新して、エンドポイントを登録します。これにより、ユーザーがアプリケーションを起動すると、デバイス情報とカスタム属性が Amazon Pinpoint に提供されます。この情報

を使用して、閲覧者のセグメントを定義できます。コンソールで、エンドポイントに関するメトリックス、そして、該当する場合には、ユーザー ID を割り当てられたユーザーを表示できます。

次に、アプリケーションでのイベントのレポート のステップを実行して、使用状況データを報告するようにアプリケーションを更新します。

アプリケーションでのイベントのレポート

モバイルアプリケーションまたはウェブアプリケーションで、AWS Mobile SDK または Amazon Pinpoint イベント API を使用して、使用状況データやイベント にレポートすることができます。イベントを報告して、セッション時間、ユーザーの購入行動、サインインの試行、必要なカスタムイベントタイプなどの情報を取得できます。

アプリケーションがイベントを報告したら、Amazon Pinpoint コンソールで分析を表示できます。[Analytics] ページのグラフは、ユーザーの行動の多くの側面に関するメトリクスを提供します。詳細については、『Amazon Pinpoint ユーザーガイド』の「Chart reference for Amazon Pinpoint analytics」を参照してください。

Amazon Pinpoint の外でイベントデータを分析または保存するには、Amazon Kinesis にデータをストリーミングするよう Amazon Pinpoint を設定します。詳細については、「Amazon Pinpoint のイベントを Kinesis にストリーミングする」を参照してください。

AWS Mobile SDK および AWS Amplify JavaScript ライブラリを使用すると、Amazon Pinpoint API を呼び出して以下の種類のイベントが報告できます。

セッションイベント

ユーザーがアプリを開き、終了する日時と回数を示します。

アプリケーションがセッションイベントを報告したら、Amazon Pinpoint コンソールの [Analytics] ページを使用して、[セッション]、[Daily active endpoints]、[7-day retention rate] などのグラフを表示します。

カスタムイベント

カスタムイベントタイプを割り当てることで、標準外のイベントを定義します。カスタムイベントにカスタム属性およびメトリクスを追加できます。

コンソールの [Analytics] ページの [Events] タブに、アプリケーションで報告されるすべてのカスタムイベントのメトリクスが表示されます。

収益化イベント

アプリケーションによって生成される収益およびユーザーが購入した商品数を報告します。

[Analytics] ページの [Revenue] タブには、[Revenue]、[Paying users]、[Units sold] などのグラフが表示されます。

認証イベント

ユーザーがアプリケーションで認証する頻度を示しています。

[Analytics] ページの [Users] タブには、[Sign-ins]、[Sign-ups] および [Authentication failures] のグラフが表示されます。

開始する前に

をまだ実行していない場合は、次を実行します。

- AWS Amplify とアプリを統合します。「AWS Amplify を使用してフロントエンドアプリケーションを Amazon Pinpoint に接続する」を参照してください。
- アプリケーションを更新して、エンドポイントを登録します。「アプリケーションでエンドポイントを登録する」を参照してください。

Android または iOS 用 AWS Mobile SDK でイベントを報告する

iOS および Android 用の AWS Mobile SDK を使用すると、モバイルアプリケーションで Amazon Pinpoint にイベントを報告することができます。

イベントを記録して Amazon Pinpoint に送信するようにアプリケーションを更新する方法の詳細については、AWS Amplify のドキュメントの以下のページを参照してください。

- iOS SDK ドキュメントの分析
- Android SDK ドキュメントの分析

AWS Amplify JavaScript ライブラリでイベントを報告する

AWS Amplify JavaScript ライブラリを使用して、JavaScript アプリケーションおよび React Native アプリケーションから Amazon Pinpoint にアプリケーション使用状況イベントをレポートできま

す。イベントを記録して Amazon Pinpoint に送信するようにアプリケーションを更新する方法の詳細については、AWS 『Amplify JavaScript documentation』の「Analytics」のページを参照してください。

Amazon Pinpoint API を使用してイベントをレポートする

Amazon Pinpoint にイベントを一括送信するには、Amazon Pinpoint API または AWS SDK を使用します。詳細については、「Amazon Pinpoint API リファレンス」の「イベント」を参照してください。

次のステップ

アプリケーションを更新して、イベントを報告します。これで、ユーザーがアプリケーションを操作すると、使用状況データが Amazon Pinpoint に送信されます。このデータをコンソールで表示でき、また、Amazon Kinesis にストリーミングすることができます。

次に、アプリケーションをアップデートして Amazon Pinpoint で送信するプッシュ通知を処理します。「プッシュ通知の処理」を参照してください。

プッシュ通知の処理

以下のトピックでは、Amazon Pinpoint を使用して送信したプッシュ通知を受信するように Swift、Android、React Native、または Flutter アプリケーションを変更する方法を説明します。

トピック

- Amazon Pinpoint のプッシュ通知のセットアップ
- プッシュ通知の処理

Amazon Pinpoint のプッシュ通知のセットアップ

アプリケーションにプッシュ通知を送信できるように Amazon Pinpoint を設定するには、まず、アプリケーションにメッセージを送信することを Amazon Pinpoint に許可する認証情報を提供する必要があります。提供する認証情報は、使用するプッシュ通知システムによって異なります。

- iOS のアプリケーションでは、Apple の Developer ポータルから取得した SSL 証明書を提供します。証明書により、Amazon Pinpoint は Apple Push Notification Service を通してアプリケーションにメッセージを送信できます。

- Android のアプリケーションでは、Firebase コンソールから取得したウェブ API キーを提供します。これらの認証情報により、Amazon Pinpoint は Firebase クラウドメッセージングを通してアプリケーションにメッセージを送信できます。

プッシュ通知チャンネルの認証情報を取得したら、Amazon Pinpoint でプロジェクトを作成し、それにプッシュ通知サービスの認証情報を提供する必要があります。

トピック

Swift プッシュ通知の設定

iOS アプリのプッシュ通知は Apple Push Notification Service (APNs) を使用して送信されます。iOS デバイスにプッシュ通知を送信するには、Apple 開発者ポータルでアプリ ID を作成する必要があり、必要な証明書を作成する必要があります。これらのステップを完了する方法の詳細については、AWS Amplify ドキュメントの「プッシュ通知をセットアップする」を参照してください。

APNs トークンの使用

ベストプラクティスとして、アプリケーションの再インストール時に顧客のデバイストークンが再生成されるようにアプリケーションを開発する必要があります。

受信者がデバイスを新しいメジャーバージョンの iOS (iOS 12 から iOS 13 など) にアップグレードし、後でアプリを再インストールした場合、アプリケーションにより新しいトークンが生成されます。アプリケーションによりトークンが更新されない場合、古いトークンを使用して通知が送信されます。その結果、トークンが無効になったため、Apple Push Notification Service (APNs) は通知を拒否します。通知を送信しようとすると、APNs からメッセージ失敗通知を受け取ります。

Android プッシュ通知のセットアップ

Android アプリケーションのプッシュ通知は、Google Cloud Messaging (GCM) の代わりに Firebase Cloud Messaging (FCM) を使用して送信されます。Android デバイスにプッシュ通知を送信する前に、FCM 認証情報を取得する必要があります。その後それらの認証情報により、Android プロジェ

クトを作成し、プッシュ通知を受け取るサンプルアプリを起動することができます。これらのステップを完了する方法の詳細については、AWS Amplify ドキュメントの「Push notifications」のセクションを参照してください。

Flutter プッシュ通知のセットアップ

Flutter アプリケーションのプッシュ通知は、Android の場合は Firebase Cloud Messaging (FCM)、iOS の場合は APN を使用して送信されます。これらのステップを完了する方法の詳細については、AWS Amplify Flutter ドキュメントの「Push notifications」のセクションを参照してください。

React Native プッシュ通知のセットアップ

React Native アプリケーションのプッシュ通知は、Android の場合は Firebase Cloud Messaging (FCM)、iOS の場合は APN を使用して送信されます。これらのステップを完了する方法の詳細については、AWS Amplify JavaScript ドキュメントの「Push notifications」のセクションを参照してください。

Amazon Pinpoint でプロジェクトを作成する

Amazon Pinpoint では、プロジェクトは設定、データ、キャンペーン、および共通の目的を共有するセグメントの集まりです。Amazon Pinpoint API では、プロジェクトはアプリケーションとも呼ばれます。このセクションでは、この概念を参照するときだけ「プロジェクト」という単語を使用します。

Amazon Pinpoint でプッシュ通知の送信を開始するには、プロジェクトを作成する必要があります。次に、適切な認証情報を入力して、使用するプッシュ通知チャンネルを有効にする必要があります。

Amazon Pinpoint コンソールを使用して、新しいプロジェクトを作成してプッシュ通知チャンネルを設定できます。詳細については、『Amazon Pinpoint ユーザーガイド』の「Amazon Pinpoint プッシュ通知チャンネルの設定」を参照してください。

Amazon Pinpoint API、AWS SDK、または AWS Command Line Interface (AWS CLI) を使用してプロジェクトを作成および設定することもできます。プロジェクトを作成するには、`Apps` リソースを使用します。プッシュ通知チャンネルを設定するには、次のリソースを使用してください。

- Apple Push Notification Service を利用して、iOSデバイスのユーザーにメッセージを送信するための APNs チャンネルです。
- Amazon Kindle Fire デバイスのユーザーにメッセージを送信する ADM チャンネル。

- Baidu ユーザーにメッセージを送信する Baidu チャンネル。
- Google Cloud Messaging (GCM) に代わるFirebase Cloud Messaging (FCM) を利用して、Android 端末にメッセージを送信する GCM チャンネルです。

プッシュ通知の処理

プッシュ通知を送信するために必要な認証情報を取得したら、プッシュ通知を受信できるようにアプリを更新できます。詳細については、『AWS Amplify ドキュメント』内の「Push notifications—Getting started」を参照してください。

Amazon Pinpoint へ対象の定義

Amazon Pinpoint では、対象者の各メンバーは、1 つ以上のエンドポイントで表されます。Amazon Pinpoint を使用してメッセージを送信すると、そのメッセージは対象者のメンバーを表すエンドポイントに直接送信されます。エンドポイント定義にはそれぞれ、メッセージの送信先 (例: デバイストークン、E メールアドレス、電話番号) が含まれます。また、ユーザーやデバイスに関するデータも含まれます。対象者を分析、分類し、取り込むには、まず Amazon Pinpoint プロジェクトにエンドポイントを追加します。

エンドポイントを追加するには、次の操作を行います。

- ユーザーがアプリケーションにアクセスするとエンドポイントが自動的に追加されるように、Amazon Pinpoint を Android、iOS、または JavaScript クライアントに統合します。
- Amazon Pinpoint API を使用して、エンドポイントを個別またはバッチで追加します。
- Amazon Pinpoint の外部に保存されているエンドポイント定義をインポートします。

エンドポイントを追加すると、次のことを行うことができます。

- Amazon Pinpoint コンソールで対象者に関する分析を表示します。
- エンドポイントデータを検索またはエクスポートして対象者の詳細を確認します。
- エンドポイント属性に基づき、対象者のセグメント (例: 人口動態データ、ユーザーの関心) を定義します。
- カスタマイズしたメッセージキャンペーンで対象者を取り込む。
- エンドポイントのリストに直接メッセージを送信する。

Amazon Pinpoint API を使用してエンドポイントを追加、更新、削除するには、このセクションのトピックを使用します。Android、iOS、または JavaScript クライアントから自動的にエンドポイントを追加するには、「アプリケーションでエンドポイントを登録する」を参照してください。

トピック

- Amazon Pinpoint へエンドポイントを追加する
- ユーザーおよび Amazon Pinpoint エンドポイントの関連付け
- エンドポイントのバッチを Amazon Pinpoint に追加する
- Amazon Pinpoint へエンドポイントをインポートする

- Amazon Pinpoint からエンドポイントを削除する
- オーディエンスメンバーの最大エンドポイント数の管理

Amazon Pinpoint へエンドポイントを追加する

エンドポイントは、ユーザーのモバイルデバイスや電話番号、E メールアドレスなど、メッセージの送信先を表します。対象者のメンバーにメッセージを送信するには、メンバーごとにエンドポイントを 1 つ以上定義する必要があります。

エンドポイントを定義する際、チャンネルとアドレスを指定します。チャンネルは、エンドポイントへのメッセージの送信に使用するプラットフォームのタイプです。チャンネルの例として、プッシュ通知サービス、SMS、E メールがあります。アドレスは、デバイストークン、電話番号、E メールアドレスなど、エンドポイントにメッセージを送信する場所を指定します。

対象者の詳細を追加するには、カスタム属性と標準属性を使用して、エンドポイントを強化します。これらの属性には、ユーザーやユーザーの設定、デバイスに加え、ユーザーが使用するクライアントのバージョンや、ユーザーの場所に関するデータが含まれます。このタイプのデータをエンドポイントに追加すると、次のことを実行できるようになります。

- Amazon Pinpoint コンソールで対象者に関するグラフを表示する。
- 適切な対象者にメッセージを送信できるように、エンドポイント属性に基づき、対象者を分類する。
- エンドポイント属性値に置き換えられているメッセージ変数を組み込み、メッセージをパーソナライズする。

AWS Mobile SDK または AWS Amplify JavaScript ライブラリを使用して Amazon Pinpoint を統合すると、モバイルまたは JavaScript クライアントアプリケーションによってエンドポイントが自動的に登録されます。クライアントは、各新規ユーザーのエンドポイントを登録し、リピートユーザーのエンドポイントを更新します。モバイルまたは JavaScript クライアントからエンドポイントを登録するには、「アプリケーションでエンドポイントを登録する」を参照してください。

例

Amazon Pinpoint プロジェクトにエンドポイントを追加する方法を以下の例に示します。エンドポイントは、シアトル在住で iPhone を使用している対象メンバーを表します。このユーザーには、Apple Push Notification Service (APNs) を使用して、メッセージを送信できます。エンドポイントのアドレスは、APN より送信されるデバイストークンです。

AWS CLI

Amazon Pinpoint を使用するには、AWS CLI でコマンドを実行します。

Example Update Endpoint コマンド

エンドポイントを追加または更新するには、update-endpoint コマンドを使用します。

```
$ aws pinpoint update-endpoint \
> --application-id application-id \
> --endpoint-id endpoint-id \
> --endpoint-request file://endpoint-request-file.json
```

実行する条件は以下のとおりです。

- application-id は、エンドポイントを追加または更新する Amazon Pinpoint プロジェクトの ID です。
- example-endpoint は、新しいエンドポイントに割り当てる ID、または更新するエンドポイントの ID です。
- endpoint-request-file.json は、ローカル JSON ファイルへのファイルパスを表しており、`--endpoint-request` パラメータの入力を含みます。

Example エンドポイントリクエストファイル

例の `update-endpoint` コマンドでは、`--endpoint-request` パラメータの引数として、JSON ファイルを使用します。このファイルには、次のようなエンドポイント定義が含まれます。

```
{
  "ChannelType": "APNS",
  "Address": "1a2b3c4d5e6f7g8h9i0j1k2l3m4n5o6p7q8r9s0t1u2v3w4x5y6z7a8b9c0d1e2f",
  "Attributes": {
    "Interests": [
      "Technology",
      "Music",
      "Travel"
    ]
  },
  "Metrics": {
    "technology_interest_level": 9.0,
    "music_interest_level": 6.0,
```

```
    "travel_interest_level": 4.0
  },
  "Demographic": {
    "AppVersion": "1.0",
    "Make": "apple",
    "Model": "iPhone",
    "ModelVersion": "8",
    "Platform": "ios",
    "PlatformVersion": "11.3.1",
    "Timezone": "America/Los_Angeles"
  },
  "Location": {
    "Country": "US",
    "City": "Seattle",
    "PostalCode": "98121",
    "Latitude": 47.61,
    "Longitude": -122.33
  }
}
```

エンドポイントのバッチの定義に使用する属性については、「Amazon Pinpoint API リファレンス」の「EndpointRequest スキーマ」を参照してください。

AWS SDK for Java

AWS SDK for Java が提供するクライアントにより、Java アプリケーションで Amazon Pinpoint API を使用できます。

Example Code

エンドポイントを追加するには、`EndpointRequest` オブジェクトを初期化し、`AmazonPinpoint` クライアントの `updateEndpoint` メソッドに渡します。

```
import com.amazonaws.regions.Regions;
import com.amazonaws.services.pinpoint.AmazonPinpoint;
import com.amazonaws.services.pinpoint.AmazonPinpointClientBuilder;
import com.amazonaws.services.pinpoint.model.*;
import java.util.Arrays;

public class AddExampleEndpoint {

 public static void main(String[] args) {
```

```
 final String USAGE = "\n" +
   "AddExampleEndpoint - Adds an example endpoint to an Amazon Pinpoint
application." +
   "Usage: AddExampleEndpoint <applicationId>" +
   "Where:\n" +
   "  applicationId - The ID of the Amazon Pinpoint application to add the example
" +
   "endpoint to.";

 if (args.length < 1) {
  System.out.println(USAGE);
  System.exit(1);
 }

 String applicationId = args[0];

 // The device token assigned to the user's device by Apple Push Notification
 // service (APNs).
 String deviceToken =
"1a2b3c4d5e6f7g8h9i0j1k2l3m4n5o6p7q8r9s0t1u2v3w4x5y6z7a8b9c0d1e2f";

 // Initializes an endpoint definition with channel type and address.
 EndpointRequest wangXiulansIphoneEndpoint = new EndpointRequest()
   .withChannelType(ChannelType.APNS)
   .withAddress(deviceToken);

 // Adds custom attributes to the endpoint.
 wangXiulansIphoneEndpoint.addAttributesEntry("interests", Arrays.asList(
   "technology",
   "music",
   "travel"));

 // Adds custom metrics to the endpoint.
 wangXiulansIphoneEndpoint.addMetricsEntry("technology_interest_level", 9.0);
 wangXiulansIphoneEndpoint.addMetricsEntry("music_interest_level", 6.0);
 wangXiulansIphoneEndpoint.addMetricsEntry("travel_interest_level", 4.0);

 // Adds standard demographic attributes.
 wangXiulansIphoneEndpoint.setDemographic(new EndpointDemographic()
   .withAppVersion("1.0")
   .withMake("apple")
   .withModel("iPhone")
   .withModelVersion("8")
   .withPlatform("ios")
```

```
    .withPlatformVersion("11.3.1")
    .withTimezone("America/Los_Angeles"));

  // Adds standard location attributes.
  wangXiulansIphoneEndpoint.setLocation(new EndpointLocation()
    .withCountry("US")
    .withCity("Seattle")
    .withPostalCode("98121")
    .withLatitude(47.61)
    .withLongitude(-122.33));

  // Initializes the Amazon Pinpoint client.
  AmazonPinpoint pinpointClient = AmazonPinpointClientBuilder.standard()
    .withRegion(Regions.US_EAST_1).build();

  // Updates or creates the endpoint with Amazon Pinpoint.
  UpdateEndpointResult result = pinpointClient.updateEndpoint(new
 UpdateEndpointRequest()
    .withApplicationId(applicationId)
    .withEndpointId("example_endpoint")
    .withEndpointRequest(wangXiulansIphoneEndpoint));

  System.out.format("Update endpoint result: %s\n",
 result.getMessageBody().getMessage());

 }
}
```

HTTP

HTTP リクエストを直接 REST API に送信して Amazon Pinpoint を使用することができます。

Example PUT Endpoint リクエスト

エンドポイントを追加するには、次の URI の Endpoint リソースに対して PUT リクエストを発行します。

/v1/apps/*application-id*/endpoints/*endpoint-id*

実行する条件は以下のとおりです。

- application-id は、エンドポイントを追加または更新する Amazon Pinpoint プロジェクトの ID です。

- endpoint-id は、新しいエンドポイントに割り当てる ID、または更新するエンドポイントの ID です。

リクエストに、必要なヘッダーを含め、EndpointRequest JSON を本文として指定します。

```
PUT /v1/apps/application_id/endpoints/example_endpoint HTTP/1.1
Host: pinpoint.us-east-1.amazonaws.com
X-Amz-Date: 20180415T182538Z
Content-Type: application/json
Accept: application/json
X-Amz-Date: 20180428T004705Z
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20180428/us-
east-1/mobiletargeting/aws4_request, SignedHeaders=accept;content-length;content-
type;host;x-amz-date,
 Signature=c25cbd6bf61bd3b3667c571ae764b9bf2d8af61b875cacced95d1e68d91b4170
Cache-Control: no-cache

{
  "ChannelType": "APNS",
  "Address": "1a2b3c4d5e6f7g8h9i0j1k2l3m4n5o6p7q8r9s0t1u2v3w4x5y6z7a8b9c0d1e2f",
  "Attributes": {
    "Interests": [
      "Technology",
      "Music",
      "Travel"
    ]
  },
  "Metrics": {
    "technology_interest_level": 9.0,
    "music_interest_level": 6.0,
    "travel_interest_level": 4.0
  },
  "Demographic": {
    "AppVersion": "1.0",
    "Make": "apple",
    "Model": "iPhone",
    "ModelVersion": "8",
    "Platform": "ios",
    "PlatformVersion": "11.3.1",
    "Timezone": "America/Los_Angeles"
  },
  "Location": {
    "Country": "US",
```

```
    "City": "Seattle",
    "PostalCode": "98121",
    "Latitude": 47.61,
    "Longitude": -122.33
  }
}
```

リクエストが成功すると、次のようなレスポンスが表示されます。

```
{
    "RequestID": "67e572ed-41d5-11e8-9dc5-db288f3cbb72",
    "Message": "Accepted"
}
```

関連情報

Amazon Pinpoint API のエンドポイントリソースに関する詳細 (例: サポートされている HTTP メソッドやリクエストパラメータ) については、「Amazon Pinpoint API リファレンス」の「エンドポイント」を参照してください。

変数を使用したメッセージのパーソナライズの詳細については、「Amazon Pinpoint ユーザーガイド」の「メッセージ変数」を参照してください。

エンドポイントに適用するクォータに関する情報 (割り当て可能な属性の数など) については、「the section called “エンドポイントクォータ”」を参照してください。

ユーザーおよび Amazon Pinpoint エンドポイントの関連付け

エンドポイントには、ユーザーを定義する属性を含めることができます。このユーザーは、対象者のユーザーを表します。例えば、ユーザーはモバイルアプリケーションをインストールしたユーザー、またはウェブサイトのアカウントを持つユーザーを表します。

ユーザーを定義するには、一意の ID を指定し、オプションでカスタムのユーザー属性を指定します。複数のデバイスでアプリを使用する場合、または複数のアドレスでメッセージを送信できる場合は、同一のユーザー ID を複数のエンドポイントに割り当てることができます。この場合、ユーザー属性は、Amazon Pinpoint によってこれらのエンドポイント間で同期されます。そのため、1 つのエンドポイントにユーザー属性を追加すると、その属性は Amazon Pinpoint によって、同一のユーザー ID が含まれる各エンドポイントに追加されます。

個々の属性に適用するデータを追跡するには、ユーザー属性を追加します。この属性は、ユーザーが使用しているデバイスによって異なります。例えば、ユーザーの名前、年齢、アカウントステータスの属性を追加することができます。

> **Tip**
>
> アプリケーションで Amazon Cognito ユーザープールを使用してユーザー認証を行っている場合、Amazon Cognito はユーザー ID と属性を自動的にエンドポイントに追加します。Amazon Cognito は、ユーザープールのユーザーに割り当てられている sub をエンドポイントユーザー ID 値に割り当てます。Amazon Cognito でユーザーを追加する方法については、『Amazon Cognito デベロッパーガイド』の「Using amazon pinpoint analytics with amazon cognito user pools」を参照してください。

エンドポイントにユーザー定義を追加すると、さらに多くの方法で対象者を分類することができます。セグメントの定義は、ユーザー属性に基づき、またはユーザー ID のリストをインポートして行うことができます。ユーザーに基づき、セグメントにメッセージを送信する場合の送信先の例として、セグメント内の各ユーザー ID に関連付けられている各エンドポイントがあります。

また、メッセージは他の方法で対象者に送信することもできます。キャンペーンを使用してユーザーのセグメントにメッセージを送信したり、ユーザー ID のリストに直接メッセージを送信したりできます。メッセージをパーソナライズするには、ユーザーの属性値に置き換えられているメッセージ変数を含めることができます。

例

エンドポイントにユーザー定義を追加する方法を以下の例に示します。

AWS CLI

Amazon Pinpoint を使用するには、AWS CLI でコマンドを実行します。

Example Update Endpoint コマンド

ユーザーをエンドポイントに追加するには、update-endpoint コマンドを使用します。`--endpoint-request` パラメータでは、新しいエンドポイントを定義することができます。この定義でユーザーを含めることができます。または、変更する属性のみ指定して、既存のエンドポイントを更新します。ユーザー属性のみ指定して、既存のエンドポイントにユーザーを追加する方法を以下の例に示します。

```
$ aws pinpoint update-endpoint \
> --application-id application-id \
> --endpoint-id endpoint-id \
> --endpoint-request file://endpoint-request-file.json
```

実行する条件は以下のとおりです。

- *application-id* は、エンドポイントを追加または更新する Amazon Pinpoint プロジェクトの ID です。
- *endpoint-id* は、新しいエンドポイントに割り当てる ID、または更新するエンドポイントの ID です。
- *endpoint-request-file.json* は、ローカル JSON ファイルへのファイルパスを表しており、`--endpoint-request` パラメータの入力を含みます。

Example エンドポイントリクエストファイル

例の `update-endpoint` コマンドでは、`--endpoint-request` パラメータの引数として、JSON ファイルを使用します。このファイルには、次のようなユーザー定義が含まれます。

```
{
    "User":{
        "UserId":"example_user",
        "UserAttributes":{
            "FirstName":["Wang"],
            "LastName":["Xiulan"],
            "Gender":["Female"],
            "Age":["39"]
        }
    }
}
```

ユーザーの定義に使用する属性については、Amazon Pinpoint API リファレンスの「EndpointRequest スキーマ」の「`User` オブジェクト」を参照してください。

AWS SDK for Java

AWS SDK for Java が提供するクライアントにより、Java アプリケーションで Amazon Pinpoint API を使用できます。

Example Code

エンドポイントにユーザーを追加するには、EndpointRequest オブジェクトを初期化し、AmazonPinpoint クライアントの updateEndpoint メソッドに渡します。このオブジェクトを使用して新しいエンドポイントを定義することができます。この定義でユーザーを含めることができます。または、変更するプロパティのみ更新して、既存のエンドポイントを更新します。次の例では、EndpointUser オブジェクトを EndpointRequest オブジェクトに追加して、ユーザーを既存のエンドポイントに追加します。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.EndpointRequest;
import software.amazon.awssdk.services.pinpoint.model.EndpointUser;
import software.amazon.awssdk.services.pinpoint.model.ChannelType;
import software.amazon.awssdk.services.pinpoint.model.UpdateEndpointRequest;
import software.amazon.awssdk.services.pinpoint.model.UpdateEndpointResponse;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
```

```
    public static void updatePinpointEndpoint(PinpointClient pinpoint, String
 applicationId, String endPointId) {
        try {
            List<String> wangXiList = new ArrayList<>();
            wangXiList.add("cooking");
            wangXiList.add("running");
            wangXiList.add("swimming");

            Map myMapWang = new HashMap<>();
            myMapWang.put("interests", wangXiList);

            List<String> myNameWang = new ArrayList<>();
            myNameWang.add("Wang ");
            myNameWang.add("Xiulan");

            Map wangName = new HashMap<>();
            wangName.put("name", myNameWang);

            EndpointUser wangMajor = EndpointUser.builder()
                    .userId("example_user_10")
```

```
                .userAttributes(wangName)
                .build();

        // Create an EndpointBatchItem object for Mary Major.
        EndpointRequest wangXiulanEndpoint = EndpointRequest.builder()
                .channelType(ChannelType.EMAIL)
                .address("wang_xiulan@example.com")
                .attributes(myMapWang)
                .user(wangMajor)
                .build();

        // Adds multiple endpoint definitions to a single request object.
        UpdateEndpointRequest endpointList = UpdateEndpointRequest.builder()
                .applicationId(applicationId)
                .endpointRequest(wangXiulanEndpoint)
                .endpointId(endPointId)
                .build();

        UpdateEndpointResponse result = pinpoint.updateEndpoint(endpointList);
        System.out.format("Update endpoint result: %s\n",
result.messageBody().message());

    } catch (PinpointException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
```

SDK の完全な例については、「GitHub」の「AddExampleUser.java」を参照してください。

HTTP

HTTP リクエストを直接 REST API に送信して Amazon Pinpoint を使用することができます。

Example ユーザー定義を含む Put Endpoint リクエスト

エンドポイントにユーザーを送信するには、次の URI の Endpoint リソースに対して PUT リクエストを発行します。

/v1/apps/*application-id*/endpoints/*endpoint-id*

実行する条件は以下のとおりです。

- *application-id* は、エンドポイントを追加または更新する Amazon Pinpoint プロジェクトの ID です。
- *endpoint-id* は、新しいエンドポイントに割り当てる ID、または更新するエンドポイントの ID です。

リクエストに、必要なヘッダーを含め、EndpointRequest JSON を本文として指定します。リクエストボディで、新しいエンドポイントを定義することができます。この定義でユーザーを含めることができます。または、変更する属性のみ指定して、既存のエンドポイントを更新します。ユーザー属性のみ指定して、既存のエンドポイントにユーザーを追加する方法を以下の例に示します。

```
PUT /v1/apps/application_id/endpoints/example_endpoint HTTP/1.1
Host: pinpoint.us-east-1.amazonaws.com
X-Amz-Date: 20180415T182538Z
Content-Type: application/json
Accept: application/json
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20180501/us-
east-1/mobiletargeting/aws4_request, SignedHeaders=accept;content-length;content-
type;host;x-amz-date,
 Signature=c25cbd6bf61bd3b3667c571ae764b9bf2d8af61b875cacced95d1e68d91b4170
Cache-Control: no-cache

{
    "User":{
        "UserId":"example_user",
        "UserAttributes":{
            "FirstName":"Wang",
            "LastName":"Xiulan",
            "Gender":"Female",
            "Age":"39"
        }
    }
}
```

リクエストが成功すると、次のようなレスポンスが表示されます。

```
{
    "RequestID": "67e572ed-41d5-11e8-9dc5-db288f3cbb72",
    "Message": "Accepted"
}
```

関連情報

Amazon Pinpoint API のエンドポイントリソースに関する詳細 (例: サポートされている HTTP メソッドやリクエストパラメータ) については、「Amazon Pinpoint API リファレンス」の「エンドポイント」を参照してください。

変数を使用したメッセージのパーソナライズの詳細については、『Amazon Pinpoint ユーザーガイド』の「Message variables」を参照してください。

ユーザー ID のリストをインポートしてセグメントを定義するには、『Amazon Pinpoint ユーザーガイド』の「Importing segments」を参照してください。

ダイレクトメッセージを最大 100 のユーザー ID に送信するには、「Amazon Pinpoint API リファレンス」の「ユーザーメッセージ」を参照してください。

エンドポイントに適用するクォータに関する情報 (割り当て可能なユーザー属性の数など) については、「the section called “エンドポイントクォータ”」を参照してください。

エンドポイントのバッチを Amazon Pinpoint に追加する

バッチでエンドポイントを送信して、複数のエンドポイントを 1 回のオペレーションで追加または更新することができます。バッチリクエストごとに最大 100 個のエンドポイント定義を含めることができます。

100 を超えるエンドポイントを 1 回のオペレーションで追加または更新するには、「Amazon Pinpoint へエンドポイントをインポートする」を参照してください。

例

次の例は、2 つのエンドポイントをバッチリクエストに含めて一度に追加する方法を示します。

AWS CLI

Amazon Pinpoint を使用するには、AWS CLI でコマンドを実行します。

Example Update Endpoints Batch コマンド

エンドポイントのバッチリクエストを送信するには、`update-endpoints-batch` コマンドを使用します。

```
$ aws pinpoint update-endpoints-batch \
> --application-id application-id \
> --endpoint-batch-request file://endpoint_batch_request_file.json
```

実行する条件は以下のとおりです。

- *application-id* は、エンドポイントを追加または更新する Amazon Pinpoint プロジェクトの ID です。
- *endpoint_batch_request_file.json* は、--endpoint-batch-request パラメータの入力が含まれているローカル JSON ファイルへのファイルパスです。

Example エンドポイントバッチリクエストファイル

例の update-endpoints-batch コマンドでは、--endpoint-request パラメータの引数として、JSON ファイルを使用します。このファイルには、次のようなエンドポイント定義のバッチが含まれます。

```
{
    "Item": [
        {
            "ChannelType": "EMAIL",
            "Address": "richard_roe@example.com",
            "Attributes": {
                "Interests": [
                    "Music",
                    "Books"
                ]
            },
            "Metrics": {
                "music_interest_level": 3.0,
                "books_interest_level": 7.0
            },
            "Id": "example_endpoint_1",
            "User":{
                "UserId": "example_user_1",
                "UserAttributes": {
                    "FirstName": "Richard",
                    "LastName": "Roe"
                }
            }
        },
```

```
        {
            "ChannelType": "SMS",
            "Address": "+16145550100",
            "Attributes": {
                "Interests": [
                    "Cooking",
                    "Politics",
                    "Finance"
                ]
            },
            "Metrics": {
                "cooking_interest_level": 5.0,
                "politics_interest_level": 8.0,
                "finance_interest_level": 4.0
            },
            "Id": "example_endpoint_2",
            "User": {
                "UserId": "example_user_2",
                "UserAttributes": {
                    "FirstName": "Mary",
                    "LastName": "Major"
                }
            }
        }
    ]
}
```

エンドポイントのバッチの定義に使用する属性については、Amazon Pinpoint API リファレンスの「EndpointBatchRequest スキーマ」を参照してください。

AWS SDK for Java

AWS SDK for Java が提供するクライアントにより、Java アプリケーションで Amazon Pinpoint API を使用できます。

Example Code

エンドポイントバッチリクエストを送信するには、`EndpointBatchRequest` オブジェクトを初期化し、`AmazonPinpoint` クライアントの `updateEndpointsBatch` メソッドに渡します。次の例では、`EndpointBatchRequest` オブジェクトを 2 つの `EndpointBatchItem` オブジェクトに追加します。

```
import software.amazon.awssdk.regions.Region;
```

```
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.UpdateEndpointsBatchResponse;
import software.amazon.awssdk.services.pinpoint.model.EndpointUser;
import software.amazon.awssdk.services.pinpoint.model.EndpointBatchItem;
import software.amazon.awssdk.services.pinpoint.model.ChannelType;
import software.amazon.awssdk.services.pinpoint.model.EndpointBatchRequest;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import software.amazon.awssdk.services.pinpoint.model.UpdateEndpointsBatchRequest;
import java.util.Map;
import java.util.List;
import java.util.ArrayList;
import java.util.HashMap;
```

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.UpdateEndpointsBatchResponse;
import software.amazon.awssdk.services.pinpoint.model.EndpointUser;
import software.amazon.awssdk.services.pinpoint.model.EndpointBatchItem;
import software.amazon.awssdk.services.pinpoint.model.ChannelType;
import software.amazon.awssdk.services.pinpoint.model.EndpointBatchRequest;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import software.amazon.awssdk.services.pinpoint.model.UpdateEndpointsBatchRequest;
import java.util.Map;
import java.util.List;
import java.util.ArrayList;
import java.util.HashMap;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class AddExampleEndpoints {

        public static void main(String[] args) {
                final String usage = """

                                Usage:    <appId>

                                Where:
```

```
                                appId - The ID of the application.

                            """;

                if (args.length != 1) {
                        System.out.println(usage);
                        System.exit(1);
                }

                String applicationId = args[0];
                PinpointClient pinpoint = PinpointClient.builder()
                                .region(Region.US_EAST_1)
                                .build();

                updateEndpointsViaBatch(pinpoint, applicationId);
                pinpoint.close();
        }

        public static void updateEndpointsViaBatch(PinpointClient pinpoint, String
applicationId) {
                try {
                        List<String> myList = new ArrayList<>();
                        myList.add("music");
                        myList.add("books");

                        Map myMap = new HashMap<String, List>();
                        myMap.put("attributes", myList);

                        List<String> myNames = new ArrayList<String>();
                        myList.add("Richard");
                        myList.add("Roe");

                        Map myMap2 = new HashMap<String, List>();
                        myMap2.put("name", myNames);

                        EndpointUser richardRoe = EndpointUser.builder()
                                        .userId("example_user_1")
                                        .userAttributes(myMap2)
                                        .build();

                        // Create an EndpointBatchItem object for Richard Roe.
                        EndpointBatchItem richardRoesEmailEndpoint =
EndpointBatchItem.builder()
                                        .channelType(ChannelType.EMAIL)
```

```
                                        .address("richard_roe@example.com")
                                        .id("example_endpoint_1")
                                        .attributes(myMap)
                                        .user(richardRoe)
                                        .build();

                        List<String> myListMary = new ArrayList<String>();
                        myListMary.add("cooking");
                        myListMary.add("politics");
                        myListMary.add("finance");

                        Map myMapMary = new HashMap<String, List>();
                        myMapMary.put("interests", myListMary);

                        List<String> myNameMary = new ArrayList<String>();
                        myNameMary.add("Mary ");
                        myNameMary.add("Major");

                        Map maryName = new HashMap<String, List>();
                        myMapMary.put("name", myNameMary);

                        EndpointUser maryMajor = EndpointUser.builder()
                                        .userId("example_user_2")
                                        .userAttributes(maryName)
                                        .build();

                        // Create an EndpointBatchItem object for Mary Major.
                        EndpointBatchItem maryMajorsSmsEndpoint =
EndpointBatchItem.builder()
                                        .channelType(ChannelType.SMS)
                                        .address("+16145550100")
                                        .id("example_endpoint_2")
                                        .attributes(myMapMary)
                                        .user(maryMajor)
                                        .build();

                        // Adds multiple endpoint definitions to a single request
object.
                        EndpointBatchRequest endpointList =
EndpointBatchRequest.builder()
                                        .item(richardRoesEmailEndpoint)
                                        .item(maryMajorsSmsEndpoint)
                                        .build();
```

```
                        // Create the UpdateEndpointsBatchRequest.
                        UpdateEndpointsBatchRequest batchRequest =
 UpdateEndpointsBatchRequest.builder()
                                            .applicationId(applicationId)
                                            .endpointBatchRequest(endpointList)
                                            .build();

                        // Updates the endpoints with Amazon Pinpoint.
                        UpdateEndpointsBatchResponse result =
 pinpoint.updateEndpointsBatch(batchRequest);
                        System.out.format("Update endpoints batch result: %s\n",
 result.messageBody().message());

                } catch (PinpointException e) {
                        System.err.println(e.awsErrorDetails().errorMessage());
                        System.exit(1);
                }
        }
}
```

SDK の完全な例については、「GitHub」 の「AddExampleEndpoints.java」を参照してください。

HTTP

HTTP リクエストを直接 REST API に送信して Amazon Pinpoint を使用することができます。

Example Put Endpoints リクエスト

エンドポイントバッチリクエストを送信するには、次の URI のPUTEndpoints リソースに対してリクエストを発行します。

/v1/apps/*application-id*/endpoints

application-id は、エンドポイントを追加または更新する Amazon Pinpoint プロジェクトの ID です。

リクエストに、必要なヘッダーを含め、EndpointBatchRequest JSON を本文として指定します。

```
PUT /v1/apps/application_id/endpoints HTTP/1.1
Host: pinpoint.us-east-1.amazonaws.com
Content-Type: application/json
Accept: application/json
X-Amz-Date: 20180501T184948Z
```

```
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20180501/us-
east-1/mobiletargeting/aws4_request, SignedHeaders=accept;content-length;content-
type;host;x-amz-date,
 Signature=c25cbd6bf61bd3b3667c571ae764b9bf2d8af61b875cacced95d1e68d91b4170
Cache-Control: no-cache

{
    "Item": [
        {
            "ChannelType": "EMAIL",
            "Address": "richard_roe@example.com",
            "Attributes": {
                "Interests": [
                    "Music",
                    "Books"
                ]
            },
            "Metrics": {
                "music_interest_level": 3.0,
                "books_interest_level": 7.0
            },
            "Id": "example_endpoint_1",
            "User":{
                "UserId": "example_user_1",
                "UserAttributes": {
                    "FirstName": "Richard",
                    "LastName": "Roe"
                }
            }
        },
        {
            "ChannelType": "SMS",
            "Address": "+16145550100",
            "Attributes": {
                "Interests": [
                    "Cooking",
                    "Politics",
                    "Finance"
                ]
            },
            "Metrics": {
                "cooking_interest_level": 5.0,
                "politics_interest_level": 8.0,
                "finance_interest_level": 4.0
```

```
            },
            "Id": "example_endpoint_2",
            "User": {
                "UserId": "example_user_2",
                "UserAttributes": {
                    "FirstName": "Mary",
                    "LastName": "Major"
                }
            }
        }
    ]
}
```

リクエストが成功すると、次のようなレスポンスが表示されます。

```
{
    "RequestID": "67e572ed-41d5-11e8-9dc5-db288f3cbb72",
    "Message": "Accepted"
}
```

関連情報

Amazon Pinpoint API のエンドポイントリソースに関する詳細 (例: サポートされている HTTP メソッドやリクエストパラメータ) については、Amazon Pinpoint API リファレンスの「エンドポイント」を参照してください。

Amazon Pinpoint へエンドポイントをインポートする

多数のエンドポイントを追加または更新するには、Amazon S3 バケットからインポートします。対象者に関するレコードが Amazon Pinpoint の外部にあり、この情報を Amazon Pinpoint プロジェクトに追加する場合は、エンドポイントをインポートすることをお勧めします。この場合は、以下のように行います。

1. 独自の対象者データに基づくエンドポイント定義を作成します。
2. これらのエンドポイント定義を 1 つ以上のファイルに保存し、そのファイルを Amazon S3 バケットにアップロードします。
3. バケットからそれらをインポートして、エンドポイントを Amazon Pinpoint プロジェクトに追加します。

インポートジョブごとに最大 1 GB のデータを転送できます。各エンドポイントが 4 KB 以下の一般的なジョブでは、約 250,000 のエンドポイントをインポートできます。AWS アカウントごとに最大 2 つのインポートジョブを同時に実行できます。インポートジョブで帯域幅を追加する必要がある場合は、サービスクォータの引き上げリクエストを AWS Support に送信できます。詳細については、「クォータ引き上げのリクエスト」を参照してください。

開始する前に

エンドポイントをインポートする前に、AWS アカウントの次のリソースが必要です。

- Amazon S3 バケット バケットの作成方法については、『Amazon Simple Storage Service ユーザーガイド』の「バケットの作成」を参照してください。
- Amazon S3 バケットに対する Amazon Pinpoint の読み取りのアクセス許可を付与する AWS Identity and Access Management (IAM) のロール。ロールを作成するには、「エンドポイントまたはセグメントをインポートするための IAM ロール」を参照してください。

例

次の例では、Amazon S3 バケットにエンドポイント定義を追加し、Amazon Pinpoint プロジェクトにそれらのエンドポイントをインポートする方法を示します。

エンドポイント定義を含むファイル

エンドポイント定義は、CSV または改行で区切られた JSON ファイルで、Amazon S3 バケットに追加するファイルに含めることができます。エンドポイントのバッチの定義に使用する属性については、「Amazon Pinpoint API リファレンスの」の「EndpointRequest JSON スキーマ」を参照してください。

CSV

以下の例のように、CSV ファイルで定義されたエンドポイントをインポートできます。

```
ChannelType,Address,Location.Country,Demographic.Platform,Demographic.Make,User.UserId
SMS,12065550182,CN,Android,LG,example-user-id-1
APNS,1a2b3c4d5e6f7g8h9i0j1a2b3c4d5e6f,US,iOS,Apple,example-user-id-2
EMAIL,john.stiles@example.com,US,iOS,Apple,example-user-id-2
```

最初の行はヘッダーで、エンドポイント属性が含まれています。入れ子の属性を指定するには、ドット表記を指定します (例: `Location.Country`)。

以下の行では、ヘッダーの属性ごとの値を指定することでエンドポイントを定義しています。

カンマ、または二重引用符を値に含めるには、"aaa,bbb" のように値を二重引用符で囲みます。

CSV 内の値は、改行はサポートされていません。

JSON

以下の例のように、改行で区切られた JSON ファイルで定義されたエンドポイントをインポートできます。

```
{"ChannelType":"SMS","Address":"12065550182","Location":
{"Country":"CN"},"Demographic":{"Platform":"Android","Make":"LG"},"User":
{"UserId":"example-user-id-1"}}
{"ChannelType":"APNS","Address":"1a2b3c4d5e6f7g8h9i0j1a2b3c4d5e6f","Location":
{"Country":"US"},"Demographic":{"Platform":"iOS","Make":"Apple"},"User":
{"UserId":"example-user-id-2"}}
{"ChannelType":"EMAIL","Address":"john.stiles@example.com","Location":
{"Country":"US"},"Demographic":{"Platform":"iOS","Make":"Apple"},"User":
{"UserId":"example-user-id-2"}}
```

この形式で、各行は個別のエンドポイント定義を含む完全な JSON オブジェクトです。

Import Job リクエスト

以下の例では、ローカルファイルをバケットにアップロードし、エンドポイント定義を Amazon S3 に追加する方法を示します。また、この例では、エンドポイント定義を Amazon Pinpoint プロジェクトにインポートします。

AWS CLI

Amazon Pinpoint を使用するには、AWS CLI でコマンドを実行します。

Example S3 CP コマンド

ローカルファイルを Amazon S3 バケットにアップロードするには、Amazon S3 cp コマンドを使用します。

```
$ aws s3 cp ./endpoints-file s3://bucket-name/prefix/
```

実行する条件は以下のとおりです。

- /endpoints-file は、エンドポイント定義を含むローカルファイルへのファイルパスを表します。
- bucket-name/prefix/ は、Amazon S3 バケットの名前です。また、バケットのオブジェクトを階層別に整理するのに役立つプレフィックスでもあります。例えば、このようなプレフィックスには `pinpoint/imports/endpoints/` があります。

Example Create Import Job コマンド

Amazon S3 バケットからエンドポイント定義をインポートするには、`create-import-job` コマンドを使用します。

```
$ aws pinpoint create-import-job \
> --application-id application-id \
> --import-job-request \
> S3Url=s3://bucket-name/prefix/key,\
> RoleArn=iam-import-role-arn,\
> Format=format,\
> RegisterEndpoints=true
```

実行する条件は以下のとおりです。

- application-id は、エンドポイントをインポートする Amazon Pinpoint プロジェクトの ID です。
- bucket-name/prefix/key は、インポートする 1 つ以上のオブジェクトを含む、Amazon S3 内の場所です。この場所を表す末尾は、各オブジェクトのキー、または複数のオブジェクトを指定するプレフィックスになります。
- iam-import-role-arn は、Amazon Pinpoint の読み込みのアクセス許可をバケットに付与する IAM ロールの Amazon リソースネーム (ARN) です。
- 形式は、エンドポイントの定義に使用した形式に基づき、`JSON` または `CSV` になります。Amazon S3 の場所にさまざまな形式のオブジェクトが複数含まれている場合、Amazon Pinpoint では、指定した形式に一致するオブジェクトのみインポートします。
- RegisterEndpoints は、`true` または `false` のどちらかに設定できます。true に設定すると、エンドポイント定義がインポートされるときに、インポートジョブにより、エンドポイントが Amazon Pinpoint に登録されます。

RegisterEndpoints と DefineSegments の組み合わせ

RegisterEndpoints	DefineSegments	説明
true	true	Amazon Pinpoint では、エンドポイントがインポートされ、エンドポイントが含まれたセグメントが作成されます。
true	false	Amazon Pinpoint では、エンドポイントはインポートされますが、セグメントは作成されません。
false	true	Amazon Pinpoint では、エンドポイントがインポートされ、エンドポイントが含まれたセグメントが作成されます。エンドポイントは保存されず、既存のエンドポイントが上書きされることもありません。
false	false	Amazon Pinpoint では、このリクエストは拒否されます。

このレスポンスには、インポートジョブに関する詳細が含まれます。

```
{
    "ImportJobResponse": {
        "CreationDate": "2018-05-24T21:26:33.995Z",
        "Definition": {
            "DefineSegment": false,
            "ExternalId": "463709046829",
            "Format": "JSON",
            "RegisterEndpoints": true,
```

```
            "RoleArn": "iam-import-role-arn",
            "S3Url": "s3://bucket-name/prefix/key"
        },
        "Id": "d5ecad8e417d498389e1d5b9454d4e0c",
        "JobStatus": "CREATED",
        "Type": "IMPORT"
    }
}
```

このレスポンスでは、Id 属性のジョブ ID が返ります。この ID を使用して、インポートジョブの現在のステータスを確認できます。

Example Get Import Job コマンド

インポートジョブの現在のステータスを確認するには、get-import-job コマンドを使用します。

```
$ aws pinpoint get-import-job \
> --application-id application-id \
> --job-id job-id
```

実行する条件は以下のとおりです。

- application-id は、インポートジョブを開始した Amazon Pinpoint プロジェクトの ID です。
- job-id は、確認中のインポートジョブの ID です。

このコマンドのレスポンスには、インポートジョブの現在の状態が含まれます。

```
{
    "ImportJobResponse": {
        "ApplicationId": "application-id",
        "CompletedPieces": 1,
        "CompletionDate": "2018-05-24T21:26:45.308Z",
        "CreationDate": "2018-05-24T21:26:33.995Z",
        "Definition": {
            "DefineSegment": false,
            "ExternalId": "463709046829",
            "Format": "JSON",
            "RegisterEndpoints": true,
            "RoleArn": "iam-import-role-arn",
            "S3Url": "s3://s3-bucket-name/prefix/endpoint-definitions.json"
        },
```

```
        "FailedPieces": 0,
        "Id": "job-id",
        "JobStatus": "COMPLETED",
        "TotalFailures": 0,
        "TotalPieces": 1,
        "TotalProcessed": 3,
        "Type": "IMPORT"
    }
}
```

このレスポンスでは、JobStatus 属性のジョブステータスを返します。

AWS SDK for Java

AWS SDK for Java が提供するクライアントにより、Java アプリケーションで Amazon Pinpoint API を使用できます。

Example Code

エンドポイント定義を含むファイルを Amazon S3 にアップロードするには、AmazonS3 クライアントの putObject メソッドを使用します。

エンドポイントを Amazon Pinpoint プロジェクトにインポートするには、CreateImportJobRequest オブジェクトを初期化します。次に、このオブジェクトを AmazonPinpoint クライアントの createImportJob メソッドに渡します。

```
package com.amazonaws.examples.pinpoint;

import com.amazonaws.AmazonServiceException;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.pinpoint.AmazonPinpoint;
import com.amazonaws.services.pinpoint.AmazonPinpointClientBuilder;
import com.amazonaws.services.pinpoint.model.CreateImportJobRequest;
import com.amazonaws.services.pinpoint.model.CreateImportJobResult;
import com.amazonaws.services.pinpoint.model.Format;
import com.amazonaws.services.pinpoint.model.GetImportJobRequest;
import com.amazonaws.services.pinpoint.model.GetImportJobResult;
import com.amazonaws.services.pinpoint.model.ImportJobRequest;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.AmazonS3Exception;
import java.io.File;
import java.nio.file.Path;
```

```
import java.nio.file.Paths;
import java.util.List;
import java.util.concurrent.TimeUnit;

public class ImportEndpoints {

    public static void main(String[] args) {

        final String USAGE = "\n" +
                "ImportEndpoints - Adds endpoints to an Amazon Pinpoint application
 by: \n" +
                "1.) Uploading the endpoint definitions to an Amazon S3 bucket. \n"
 +
                "2.) Importing the endpoint definitions from the bucket to an Amazon
 Pinpoint " +
                "application.\n\n" +
                "Usage: ImportEndpoints <endpointsFileLocation> <s3BucketName>
 <iamImportRoleArn> " +
                "<applicationId>\n\n" +
                "Where:\n" +
                "  endpointsFileLocation - The relative location of the JSON file
 that contains the " +
                "endpoint definitions.\n" +
                "  s3BucketName - The name of the Amazon S3 bucket to upload the
 JSON file to. If the " +
                "bucket doesn't exist, a new bucket is created.\n" +
                "  iamImportRoleArn - The ARN of an IAM role that grants Amazon
 Pinpoint read " +
                "permissions to the S3 bucket.\n" +
                "  applicationId - The ID of the Amazon Pinpoint application to add
 the endpoints to.";

        if (args.length < 1) {
            System.out.println(USAGE);
            System.exit(1);
        }

        String endpointsFileLocation = args[0];
        String s3BucketName = args[1];
        String iamImportRoleArn = args[2];
        String applicationId = args[3];

        Path endpointsFilePath = Paths.get(endpointsFileLocation);
```

```
        File endpointsFile = new
File(endpointsFilePath.toAbsolutePath().toString());
        uploadToS3(endpointsFile, s3BucketName);

        importToPinpoint(endpointsFile.getName(), s3BucketName, iamImportRoleArn,
applicationId);

    }

    private static void uploadToS3(File endpointsFile, String s3BucketName) {

        // Initializes Amazon S3 client.
        final AmazonS3 s3 = AmazonS3ClientBuilder.defaultClient();

        // Checks whether the specified bucket exists. If not, attempts to create
one.
        if (!s3.doesBucketExistV2(s3BucketName)) {
            try {
                s3.createBucket(s3BucketName);
                System.out.format("Created S3 bucket %s.\n", s3BucketName);
            } catch (AmazonS3Exception e) {
                System.err.println(e.getErrorMessage());
                System.exit(1);
            }
        }

        // Uploads the endpoints file to the bucket.
        String endpointsFileName = endpointsFile.getName();
        System.out.format("Uploading %s to S3 bucket %s . . .\n", endpointsFileName,
s3BucketName);
        try {
            s3.putObject(s3BucketName, "imports/" + endpointsFileName,
endpointsFile);
            System.out.println("Finished uploading to S3.");
        } catch (AmazonServiceException e) {
            System.err.println(e.getErrorMessage());
            System.exit(1);
        }
    }

    private static void importToPinpoint(String endpointsFileName, String
s3BucketName,
            String iamImportRoleArn, String applicationId) {
```

```
        // The S3 URL that Amazon Pinpoint requires to find the endpoints file.
        String s3Url = "s3://" + s3BucketName + "/imports/" + endpointsFileName;

        // Defines the import job that Amazon Pinpoint runs.
        ImportJobRequest importJobRequest = new ImportJobRequest()
                .withS3Url(s3Url)
                .withRegisterEndpoints(true)
                .withRoleArn(iamImportRoleArn)
                .withFormat(Format.JSON);
        CreateImportJobRequest createImportJobRequest = new CreateImportJobRequest()
                .withApplicationId(applicationId)
                .withImportJobRequest(importJobRequest);

        // Initializes the Amazon Pinpoint client.
        AmazonPinpoint pinpointClient = AmazonPinpointClientBuilder.standard()
                .withRegion(Regions.US_EAST_1).build();

        System.out.format("Importing endpoints in %s to Amazon Pinpoint application
%s . . .\n",
                endpointsFileName, applicationId);

        try {

            // Runs the import job with Amazon Pinpoint.
            CreateImportJobResult importResult =
pinpointClient.createImportJob(createImportJobRequest);

            String jobId = importResult.getImportJobResponse().getId();
            GetImportJobResult getImportJobResult = null;
            String jobStatus = null;

            // Checks the job status until the job completes or fails.
            do {
                getImportJobResult = pinpointClient.getImportJob(new
GetImportJobRequest()
                        .withJobId(jobId)
                        .withApplicationId(applicationId));
                jobStatus =
getImportJobResult.getImportJobResponse().getJobStatus();
                System.out.format("Import job %s . . .\n", jobStatus.toLowerCase());
                TimeUnit.SECONDS.sleep(3);
            } while (!jobStatus.equals("COMPLETED") && !jobStatus.equals("FAILED"));

            if (jobStatus.equals("COMPLETED")) {
```

```
                System.out.println("Finished importing endpoints.");
            } else {
                System.err.println("Failed to import endpoints.");
                System.exit(1);
            }

            // Checks for entries that failed to import.
            // getFailures provides up to 100 of the first failed entries for the
 job, if
            // any exist.
            List<String> failedEndpoints =
 getImportJobResult.getImportJobResponse().getFailures();
            if (failedEndpoints != null) {
                System.out.println("Failed to import the following entries:");
                for (String failedEndpoint : failedEndpoints) {
                    System.out.println(failedEndpoint);
                }
            }

        } catch (AmazonServiceException | InterruptedException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }

    }

}
```

HTTP

HTTP リクエストを直接 REST API に送信して Amazon Pinpoint を使用することができます。

Example S3 PUT Object リクエスト

エンドポイント定義をバケットに追加するには、Amazon S3 の PUT object オペレーションを使用して、エンドポイント定義を本文として入力します。

```
PUT /prefix/key HTTP/1.1
Content-Type: text/plain
Accept: application/json
Host: bucket-name.s3.amazonaws.com
X-Amz-Content-Sha256:
 c430dc094b0cec2905bc88d96314914d058534b14e2bc6107faa9daa12fdff2d
X-Amz-Date: 20180605T184132Z
```

```
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20180605/
us-east-1/s3/aws4_request, SignedHeaders=accept;cache-control;content-
length;content-type;host;postman-token;x-amz-content-sha256;x-amz-date,
 Signature=c25cbd6bf61bd3b3667c571ae764b9bf2d8af61b875cacced95d1e68d91b4170
Cache-Control: no-cache

{"ChannelType":"SMS","Address":"2065550182","Location":
{"Country":"CAN"},"Demographic":{"Platform":"Android","Make":"LG"},"User":
{"UserId":"example-user-id-1"}}
{"ChannelType":"APNS","Address":"1a2b3c4d5e6f7g8h9i0j1a2b3c4d5e6f","Location":
{"Country":"USA"},"Demographic":{"Platform":"iOS","Make":"Apple"},"User":
{"UserId":"example-user-id-2"}}
{"ChannelType":"EMAIL","Address":"john.stiles@example.com","Location":
{"Country":"USA"},"Demographic":{"Platform":"iOS","Make":"Apple"},"User":
{"UserId":"example-user-id-2"}}
```

実行する条件は以下のとおりです。

- /prefix/key は、オブジェクトのプレフィックスおよびキーの名前であり、アップロード後にエンドポイント定義が含まれます。オブジェクトを階層的に整理するには、このプレフィックスを使用します。例えば、このようなプレフィックスには pinpoint/imports/endpoints/ があります。
- bucket-name は、エンドポイント定義に追加する Amazon S3 バケットの名前です。

Example POST Import Job リクエスト

Amazon S3 バケットからエンドポイント定義をインポートするには、インポートジョブリソースに対して POST リクエストを発行します。リクエストに、必要なヘッダーを含め、ImportJobRequest JSON を本文として指定します。

```
POST /v1/apps/application_id/jobs/import HTTP/1.1
Content-Type: application/json
Accept: application/json
Host: pinpoint.us-east-1.amazonaws.com
X-Amz-Date: 20180605T214912Z
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20180605/
us-east-1/mobiletargeting/aws4_request, SignedHeaders=accept;cache-
control;content-length;content-type;host;postman-token;x-amz-date,
 Signature=c25cbd6bf61bd3b3667c571ae764b9bf2d8af61b875cacced95d1e68d91b4170
Cache-Control: no-cache
```

```
{
  "S3Url": "s3://bucket-name/prefix/key",
  "RoleArn": "iam-import-role-arn",
  "Format": "format",
  "RegisterEndpoints": true
}
```

実行する条件は以下のとおりです。

- application-id は、エンドポイントをインポートする Amazon Pinpoint プロジェクトの ID です。
- bucket-name/prefix/key は、インポートする 1 つ以上のオブジェクトを含む、Amazon S3 内の場所です。この場所を表す末尾は、各オブジェクトのキー、または複数のオブジェクトを指定するプレフィックスになります。
- iam-import-role-arn は、Amazon Pinpoint の読み込みのアクセス許可をバケットに付与する IAM ロールの Amazon リソースネーム (ARN) です。
- 形式は、エンドポイントの定義に使用した形式に基づき、JSON または CSV になります。Amazon S3 の場所にさまざまな形式のファイルが複数含まれている場合、Amazon Pinpoint では、指定した形式に一致するファイルのみインポートします。

リクエストが成功すると、次のようなレスポンスが表示されます。

```
{
    "Id": "a995ce5d70fa44adb563b7d0e3f6c6f5",
    "JobStatus": "CREATED",
    "CreationDate": "2018-06-05T21:49:15.288Z",
    "Type": "IMPORT",
    "Definition": {
        "S3Url": "s3://bucket-name/prefix/key",
        "RoleArn": "iam-import-role-arn",
        "ExternalId": "external-id",
        "Format": "JSON",
        "RegisterEndpoints": true,
        "DefineSegment": false
    }
}
```

このレスポンスでは、Id 属性のジョブ ID が返ります。この ID を使用して、インポートジョブの現在のステータスを確認できます。

Example GET Import Job リクエスト

インポートジョブの現在のステータスを確認するには、GETImport Job リソースに対して リクエストを発行します。

```
GET /v1/apps/application_id/jobs/import/job_id HTTP/1.1
Content-Type: application/json
Accept: application/json
Host: pinpoint.us-east-1.amazonaws.com
X-Amz-Date: 20180605T220744Z
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20180605/us-
east-1/mobiletargeting/aws4_request, SignedHeaders=accept;cache-control;content-
type;host;postman-token;x-amz-date,
 Signature=c25cbd6bf61bd3b3667c571ae764b9bf2d8af61b875cacced95d1e68d91b4170
Cache-Control: no-cache
```

実行する条件は以下のとおりです。

- application_id は、インポートジョブを開始した Amazon Pinpoint プロジェクトの ID です。
- job_id は、確認中のインポートジョブの ID です。

リクエストが成功すると、次のようなレスポンスが表示されます。

```
{
    "ApplicationId": "application_id",
    "Id": "70a51b2cf442447492d2c8e50336a9e8",
    "JobStatus": "COMPLETED",
    "CompletedPieces": 1,
    "FailedPieces": 0,
    "TotalPieces": 1,
    "CreationDate": "2018-06-05T22:04:49.213Z",
    "CompletionDate": "2018-06-05T22:04:58.034Z",
    "Type": "IMPORT",
    "TotalFailures": 0,
    "TotalProcessed": 3,
    "Definition": {
        "S3Url": "s3://bucket-name/prefix/key.json",
        "RoleArn": "iam-import-role-arn",
        "ExternalId": "external-id",
        "Format": "JSON",
        "RegisterEndpoints": true,
        "DefineSegment": false
```

```
    }
}
```

このレスポンスでは、JobStatus 属性のジョブステータスを返します。

関連情報

Amazon Pinpoint API のインポートジョブリソースに関する詳細 (例: サポートされている HTTP メソッドやリクエストパラメータ) については、「Amazon Pinpoint API リファレンス」の「インポートジョブ」を参照してください。

Amazon Pinpoint からエンドポイントを削除する

宛先不明でエラーになる場合や、顧客がアカウントを削除した場合など、特定の送信先へのメッセージの送信が不要になった場合は、そのエンドポイントを削除することができます。

例

エンドポイントの削除方法を次の例に示します。

AWS CLI

Amazon Pinpoint を使用するには、AWS CLI でコマンドを実行します。

Example Delete Endpoint コマンド

エンドポイントを削除するには、`delete-endpoint` コマンドを使用します。

```
$ aws pinpoint delete-endpoint \
> --application-id application-id \
> --endpoint-id endpoint-id
```

実行する条件は以下のとおりです。

- application-id は、エンドポイントを含む Amazon Pinpoint プロジェクトの ID です。
- endpoint-id は、削除するエンドポイントの ID です。

このコマンドのレスポンスは、削除したエンドポイントの JSON 定義です。

AWS SDK for Java

AWS SDK for Java が提供するクライアントにより、Java アプリケーションで Amazon Pinpoint API を使用できます。

Example Code

エンドポイントを削除するには、AmazonPinpoint クライアントの deleteEndpoint メソッドを使用します。メソッドの引数として DeleteEndpointRequest オブジェクトを指定します。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.DeleteEndpointRequest;
import software.amazon.awssdk.services.pinpoint.model.DeleteEndpointResponse;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
```

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.DeleteEndpointRequest;
import software.amazon.awssdk.services.pinpoint.model.DeleteEndpointResponse;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class DeleteEndpoint {
    public static void main(String[] args) {
        final String usage = """

                Usage:   <appName> <endpointId >

                Where:
                  appId - The id of the application to delete.
                  endpointId - The id of the endpoint to delete.
                """;

        if (args.length != 2) {
```

```
            System.out.println(usage);
            System.exit(1);
        }

        String appId = args[0];
        String endpointId = args[1];
        System.out.println("Deleting an endpoint with id: " + endpointId);
        PinpointClient pinpoint = PinpointClient.builder()
                .region(Region.US_EAST_1)
                .build();

        deletePinEncpoint(pinpoint, appId, endpointId);
        pinpoint.close();
    }

    public static void deletePinEncpoint(PinpointClient pinpoint, String appId,
 String endpointId) {
        try {
            DeleteEndpointRequest appRequest = DeleteEndpointRequest.builder()
                    .applicationId(appId)
                    .endpointId(endpointId)
                    .build();

            DeleteEndpointResponse result = pinpoint.deleteEndpoint(appRequest);
            String id = result.endpointResponse().id();
            System.out.println("The deleted endpoint id  " + id);

        } catch (PinpointException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        System.out.println("Done");
    }
}
```

SDK の完全な例については、「GitHub」の「DeleteEndpoint.java」を参照してください。

HTTP

HTTP リクエストを直接 REST API に送信して Amazon Pinpoint を使用することができます。

Example DELETE Endpoint リクエスト

エンドポイントを削除するには、Endpoint リソースに対して DELETE リクエストを発行します。

```
DELETE /v1/apps/application-id/endpoints/endpoint-id HTTP/1.1
Host: pinpoint.us-east-1.amazonaws.com
Content-Type: application/json
Accept: application/json
Cache-Control: no-cache
```

実行する条件は以下のとおりです。

- application-id は、エンドポイントを含む Amazon Pinpoint プロジェクトの ID です。
- endpoint-id は、削除するエンドポイントの ID です。

このリクエストのレスポンスは、削除したエンドポイントの JSON 定義です。

オーディエンスメンバーの最大エンドポイント数の管理

オーディエンスの各メンバーは、UserId に最大 15 のエンドポイントを関連付けることができます。「エンドポイントクォータ」を参照してください。16 番目のエンドポイントを追加しようとすると、ChannelType に応じて、BadRequestException が発生するか、または EffectiveDate が最も古いエンドポイントを削除すれば成功となります。

16 番目のエンドポイントの追加

- エンドポイントの新しいチャネルタイプが SMS、PUSH、VOICE、EMAIL、CUSTOM、または IN_APP の場合、オーディエンスメンバーはエンドポイントの数が最大に達しているため、BadRequestException が返されます。オーディエンスメンバーに関連付けられているエンドポイントを削除してやり直す必要があります。「Amazon Pinpoint からエンドポイントを削除する」を参照してください。
- エンドポイントの新しいチャネルタイプが ADM、GCM、APNS、APNS_VOIP、APNS_VOIP_SANDBOX、または BAIDU の場合:
  - オーディエンスメンバーに現在関連付けられている少なくとも 1 つのエンドポイントの ChannelType が ADM、GCM、APNS、APNS_VOICE、APNS_VOIP_SANDBOX、または BAIDU であることを確認します。該当するエンドポイントが存在しない場合は BadRequestException が返されます。再試行する前にエンドポイントを削除する必要があります。「Amazon Pinpoint からエンドポイントを削除する」を参照してください。

- それ以外の場合は、EffectiveDate が最も古いエンドポイントが `INACTIVE` に設定されます。ここで、ChannelType は ADM、GCM、APNS、APNS_VOIP、APNS_VOIP_SANDBOX、または BAIDU です。
  - 古いエンドポイントから UserId が削除されます。
  - 新しいエンドポイントはオーディエンスメンバーに関連付けられますが、まだエンドポイントの数が最大になっています。

Status を `ACTIVE` に設定し、UserId をエンドポイントに追加し直すことで、エンドポイントを再度有効にできます。

Amazon Pinpoint の対象者データへアクセス

Amazon Pinpoint に追加するにつれて、エンドポイントは対象者データのリポジトリとして大きくなります。このデータの内容は以下のとおりです。

- Amazon Pinpoint API を使用して、追加または更新するエンドポイント。
- ユーザーがアプリケーションにアクセスする度にクライアントコードによって追加または更新されるエンドポイント。

エンドポイントデータは、対象者の増加や変更に伴い、増減します。Amazon Pinpoint が対象者について所有している最新情報を表示するには、エンドポイントを個別に検索するか、Amazon Pinpoint プロジェクトに属するエンドポイントをすべてエクスポートします。エンドポイントデータを表示することで、エンドポイントに記録した対象者の特性を確認することができます。例えば、次のような内容があります。

- ユーザーのデバイスやプラットフォーム。
- ユーザーのタイムゾーン。
- ユーザーのデバイスにインストールされているアプリのバージョン。
- ユーザーの場所 (例: 都市、国)。
- 記録したカスタムの属性やメトリクス。

また、Amazon Pinpoint コンソールには、デモグラフィック分析や、エンドポイントにキャプチャしたカスタム属性などがあります。

エンドポイントを検索するには、Amazon Pinpoint プロジェクトに追加する必要があります。エンドポイントを追加するには、「Amazon Pinpoint へ対象の定義」を参照してください。

このセクションのトピックを使用して、Amazon Pinpoint API でエンドポイントを検索またはエクスポートします。

トピック

- Amazon Pinpoint を使用したエンドポイントの検索
- Amazon Pinpoint からエンドポイントをエクスポートする
- Amazon Pinpoint で使用しているエンドポイント ID の一覧表示

Amazon Pinpoint を使用したエンドポイントの検索

Amazon Pinpoint プロジェクトに追加されたエンドポイントの詳細を個々に検索することができます。これらの詳細には、メッセージの送信先アドレス、メッセージングチャネル、ユーザーのデバイスに関するデータ、ユーザーの場所に関するデータ、エンドポイントに記録するカスタム属性が含まれます。

エンドポイントを検索するには、エンドポイント ID が必要です。ID が不明な場合は、エクスポートしてエンドポイントデータを取得できます。エンドポイントをエクスポートするには、「the section called “エンドポイントのエクスポート”」を参照してください。

例

以下の例は、ID を指定して、エンドポイントを個々に検索する方法を示します。

AWS CLI

Amazon Pinpoint を使用するには、AWS CLI でコマンドを実行します。

Example Get Endpoint コマンド

エンドポイントを検索するには、`get-endpoint` コマンドを使用します。

```
$ aws pinpoint get-endpoint \
> --application-id application-id \
> --endpoint-id endpoint-id
```

実行する条件は以下のとおりです。

- *application-id* は、エンドポイントを含む Amazon Pinpoint プロジェクトの ID です。
- *endpoint-id* は、検索するエンドポイントの ID です。

このコマンドのレスポンスは、次の例のように、エンドポイントの JSON 定義です。

```
{
    "EndpointResponse": {
        "Address":
 "1a2b3c4d5e6f7g8h9i0j1k2l3m4n5o6p7q8r9s0t1u2v3w4x5y6z7a8b9c0d1e2f",
        "ApplicationId": "application-id",
        "Attributes": {
            "Interests": [
```

```
                "Technology",
                "Music",
                "Travel"
            ]
        },
        "ChannelType": "APNS",
        "CohortId": "63",
        "CreationDate": "2018-05-01T17:31:01.046Z",
        "Demographic": {
            "AppVersion": "1.0",
            "Make": "apple",
            "Model": "iPhone",
            "ModelVersion": "8",
            "Platform": "ios",
            "PlatformVersion": "11.3.1",
            "Timezone": "America/Los_Angeles"
        },
        "EffectiveDate": "2018-05-07T19:03:29.963Z",
        "EndpointStatus": "ACTIVE",
        "Id": "example_endpoint",
        "Location": {
            "City": "Seattle",
            "Country": "US",
            "Latitude": 47.6,
            "Longitude": -122.3,
            "PostalCode": "98121"
        },
        "Metrics": {
            "music_interest_level": 6.0,
            "travel_interest_level": 4.0,
            "technology_interest_level": 9.0
        },
        "OptOut": "ALL",
        "RequestId": "7f546cac-6858-11e8-adcd-2b5a07aab338",
        "User": {
            "UserAttributes": {
                "Gender": "Female",
                "FirstName": "Wang",
                "LastName": "Xiulan",
                "Age": "39"
            },
            "UserId": "example_user"
        }
    }
```

```
}
```

AWS SDK for Java

AWS SDK for Java が提供するクライアントにより、Java アプリケーションで Amazon Pinpoint API を使用できます。

Example Code

エンドポイントを検索するには、GetEndpointRequest オブジェクトを初期化します。次に、このオブジェクトを クライアントの getEndpointAmazonPinpoint クライアントに渡します。

```
import com.google.gson.FieldNamingPolicy;
import com.google.gson.Gson;
import com.google.gson.GsonBuilder;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.EndpointResponse;
import software.amazon.awssdk.services.pinpoint.model.GetEndpointResponse;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import software.amazon.awssdk.services.pinpoint.model.GetEndpointRequest;
```

```
import com.google.gson.FieldNamingPolicy;
import com.google.gson.Gson;
import com.google.gson.GsonBuilder;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.EndpointResponse;
import software.amazon.awssdk.services.pinpoint.model.GetEndpointResponse;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import software.amazon.awssdk.services.pinpoint.model.GetEndpointRequest;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class LookUpEndpoint {
    public static void main(String[] args) {
```

```
        final String usage = """

                Usage:   <appId> <endpoint>

                Where:
                  appId - The ID of the application to delete.
                  endpoint - The ID of the endpoint.\s
                  """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String appId = args[0];
        String endpoint = args[1];
        System.out.println("Looking up an endpoint point with ID: " + endpoint);
        PinpointClient pinpoint = PinpointClient.builder()
                .region(Region.US_EAST_1)
                .build();

        lookupPinpointEndpoint(pinpoint, appId, endpoint);
        pinpoint.close();
    }

    public static void lookupPinpointEndpoint(PinpointClient pinpoint, String appId,
String endpoint) {
        try {
            GetEndpointRequest appRequest = GetEndpointRequest.builder()
                    .applicationId(appId)
                    .endpointId(endpoint)
                    .build();

            GetEndpointResponse result = pinpoint.getEndpoint(appRequest);
            EndpointResponse endResponse = result.endpointResponse();

            // Uses the Google Gson library to pretty print the endpoint JSON.
            Gson gson = new GsonBuilder()
                    .setFieldNamingPolicy(FieldNamingPolicy.UPPER_CAMEL_CASE)
                    .setPrettyPrinting()
                    .create();

            String endpointJson = gson.toJson(endResponse);
            System.out.println(endpointJson);
```

```
        } catch (PinpointException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        System.out.println("Done");
    }
}
```

読み取り可能な形式でエンドポイントデータを出力するために、この例では、Google GSON ライブラリを使用して、EndpointResponse オブジェクトを JSON 文字列に変換します。

HTTP

HTTP リクエストを直接 REST API に送信して Amazon Pinpoint を使用することができます。

Example GET Endpoint リクエスト

エンドポイントを検索するには、エンドポイントリソースに対して GET リクエストを発行します。

```
GET /v1/apps/application_id/endpoints/endpoint_id HTTP/1.1
Host: pinpoint.us-east-1.amazonaws.com
Content-Type: application/json
Accept: application/json
Cache-Control: no-cache
```

実行する条件は以下のとおりです。

- *application-id* は、エンドポイントを含む Amazon Pinpoint プロジェクトの ID です。
- *endpoint-id* は、検索するエンドポイントの ID です。

このリクエストのレスポンスは、次の例のように、エンドポイントの JSON 定義です。

```
{
    "ChannelType": "APNS",
    "Address": "1a2b3c4d5e6f7g8h9i0j1k2l3m4n5o6p7q8r9s0t1u2v3w4x5y6z7a8b9c0d1e2f",
    "EndpointStatus": "ACTIVE",
    "OptOut": "NONE",
    "RequestId": "b720cfa8-6924-11e8-aeda-0b22e0b0fa59",
```

```
    "Location": {
        "Latitude": 47.6,
        "Longitude": -122.3,
        "PostalCode": "98121",
        "City": "Seattle",
        "Country": "US"
    },
    "Demographic": {
        "Make": "apple",
        "Model": "iPhone",
        "ModelVersion": "8",
        "Timezone": "America/Los_Angeles",
        "AppVersion": "1.0",
        "Platform": "ios",
        "PlatformVersion": "11.3.1"
    },
    "EffectiveDate": "2018-06-06T00:58:19.865Z",
    "Attributes": {
        "Interests": [
            "Technology",
            "Music",
            "Travel"
        ]
    },
    "Metrics": {
        "music_interest_level": 6,
        "travel_interest_level": 4,
        "technology_interest_level": 9
    },
    "User": {},
    "ApplicationId": "application_id",
    "Id": "example_endpoint",
    "CohortId": "39",
    "CreationDate": "2018-06-06T00:58:19.865Z"
}
```

関連情報

Amazon Pinpoint API のエンドポイントリソースの詳細については、「Amazon Pinpoint API リファレンス」の「エンドポイント」を参照してください。

Amazon Pinpoint からエンドポイントをエクスポートする

Amazon Pinpoint が対象者について所有している情報をすべて取得するには、プロジェクトに属するエンドポイント定義をエクスポートします。エクスポートすると、Amazon Pinpoint は指定した Amazon S3 バケットにエンドポイント定義を配置します。エンドポイントのエクスポートは、次のような場合に役立ちます。

- クライアントアプリケーションで Amazon Pinpoint を使用して登録された新規および既存のエンドポイントに関する最新のデータの表示。
- Amazon Pinpoint のエンドポイントデータと独自の顧客関係管理 (CRM) システムの同期。
- 顧客データに関するレポートの作成、または顧客データの分析。

開始する前に

エンドポイントをエクスポートする前に、AWS アカウントの次のリソースが必要です。

- Amazon S3 バケット バケットの作成方法については、『Amazon Simple Storage Service ユーザーガイド』の「バケットの作成」を参照してください。
- Amazon S3 バケットに対する Amazon Pinpoint の書き込みのアクセス許可を付与する AWS Identity and Access Management (IAM)のロール。ロールを作成するには、「エンドポイントまたはセグメントをエクスポートするための IAM ロール」を参照してください。

例

次の例では、Amazon Pinpoint プロジェクトからエンドポイントをエクスポートし、Amazon S3 バケットからそれらのエンドポイントをダウンロードする方法を示します。

AWS CLI

Amazon Pinpoint を使用するには、AWS CLI でコマンドを実行します。

Example Create Export Job コマンド

Amazon Pinpoint プロジェクトのエンドポイントをエクスポートするには、`create-export-job` コマンドを使用します。

```
$ aws pinpoint create-export-job \
> --application-id application-id \
```

```
> --export-job-request \
> S3UrlPrefix=s3://bucket-name/prefix/,\
> RoleArn=iam-export-role-arn
```

実行する条件は以下のとおりです。

- *application-id* は、エンドポイントを含む Amazon Pinpoint プロジェクトの ID です。
- *bucket-name/prefix/* は、Amazon S3 バケットの名前です。また、バケットのオブジェクトを階層別に整理するのに役立つプレフィックスでもあります。例えば、このようなプレフィックスには pinpoint/exports/endpoints/ があります。
- *iam-export-role-arn* は、バケットの書き込み権限を Amazon Pinpoint に付与する IAM ロールの Amazon リソースネーム (ARN) です。

このコマンドのレスポンスには、エクスポートジョブに関する詳細が含まれます。

```
{
    "ExportJobResponse": {
        "CreationDate": "2018-06-04T22:04:20.585Z",
        "Definition": {
            "RoleArn": "iam-export-role-arn",
            "S3UrlPrefix": "s3://s3-bucket-name/prefix/"
        },
        "Id": "7390e0de8e0b462380603c5a4df90bc4",
        "JobStatus": "CREATED",
        "Type": "EXPORT"
    }
}
```

このレスポンスでは、Id 属性のジョブ ID が返ります。この ID を使用して、エクスポートジョブの現在のステータスを確認できます。

Example Get Export Job コマンド

エクスポートジョブの現在のステータスを確認するには、get-export-job コマンドを使用します。

```
$ aws pinpoint get-export-job \
> --application-id application-id \
> --job-id job-id
```

実行する条件は以下のとおりです。

- *application-id* は、エンドポイントをエクスポートした Amazon Pinpoint プロジェクトの ID です。
- *job-id* は、確認中のジョブの ID です。

このコマンドのレスポンスには、エクスポートジョブの現在の状態が含まれます。

```
{
    "ExportJobResponse": {
        "ApplicationId": "application-id",
        "CompletedPieces": 1,
        "CompletionDate": "2018-05-08T22:16:48.228Z",
        "CreationDate": "2018-05-08T22:16:44.812Z",
        "Definition": {},
        "FailedPieces": 0,
        "Id": "6c99c463f14f49caa87fa27a5798bef9",
        "JobStatus": "COMPLETED",
        "TotalFailures": 0,
        "TotalPieces": 1,
        "TotalProcessed": 215,
        "Type": "EXPORT"
    }
}
```

このレスポンスでは、JobStatus 属性のジョブステータスを返します。ジョブステータス値が COMPLETED の場合は、Amazon S3 バケットからエクスポートされたエンドポイントを取得できます。

Example S3 CP コマンド

エクスポートされたエンドポイントをダウンロードするには、Amazon S3 の cp コマンドを使用します。

```
$ aws s3 cp s3://bucket-name/prefix/key.gz /local/directory/
```

実行する条件は以下のとおりです。

- *bucket-name/prefix/key* は、エンドポイントエクスポート時に Amazon Pinpoint によってバケットに追加された .gz ファイルの場所を表します。このファイルには、エクスポートされたエンドポイント定義が含まれます。例えば、https://PINPOINT-EXAMPLE-

BUCKET.s3.us-west-2.amazonaws.com/Exports/example.csv という URL で、「PINPOINT-EXAMPLE-BUCKET」がバケットの名前で、「Exports/example.csv」がキーです。キーの詳細については、「Amazon S3 ユーザーガイド」の「キー」を参照してください。

- */local/directory/* は、エンドポイントをダウンロードするローカルディレクトリのファイルパスを表します。

AWS SDK for Java

AWS SDK for Java が提供するクライアントにより、Java アプリケーションで Amazon Pinpoint API を使用できます。

Example Code

Amazon Pinpoint プロジェクトからエンドポイントをエクスポートするには、CreateExportJobRequest オブジェクトを初期化します。次に、このオブジェクトを AmazonPinpoint クライアントの createExportJob メソッドに渡します。

Amazon Pinpoint からエクスポートされたエンドポイントをダウンロードするには、クライアントの getObjectAmazonS3 メソッドを使用します。

```
import software.amazon.awssdk.core.ResponseBytes;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.ExportJobRequest;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import software.amazon.awssdk.services.pinpoint.model.CreateExportJobRequest;
import software.amazon.awssdk.services.pinpoint.model.CreateExportJobResponse;
import software.amazon.awssdk.services.pinpoint.model.GetExportJobResponse;
import software.amazon.awssdk.services.pinpoint.model.GetExportJobRequest;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.GetObjectRequest;
import software.amazon.awssdk.services.s3.model.ListObjectsV2Request;
import software.amazon.awssdk.services.s3.model.ListObjectsV2Response;
import software.amazon.awssdk.services.s3.model.S3Object;
import software.amazon.awssdk.services.s3.model.GetObjectResponse;
import software.amazon.awssdk.services.s3.model.S3Exception;
import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.OutputStream;
```

```
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Date;
import java.util.List;
import java.util.concurrent.TimeUnit;
import java.util.stream.Collectors;
```

```
import software.amazon.awssdk.core.ResponseBytes;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.ExportJobRequest;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import software.amazon.awssdk.services.pinpoint.model.CreateExportJobRequest;
import software.amazon.awssdk.services.pinpoint.model.CreateExportJobResponse;
import software.amazon.awssdk.services.pinpoint.model.GetExportJobResponse;
import software.amazon.awssdk.services.pinpoint.model.GetExportJobRequest;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.GetObjectRequest;
import software.amazon.awssdk.services.s3.model.ListObjectsV2Request;
import software.amazon.awssdk.services.s3.model.ListObjectsV2Response;
import software.amazon.awssdk.services.s3.model.S3Object;
import software.amazon.awssdk.services.s3.model.GetObjectResponse;
import software.amazon.awssdk.services.s3.model.S3Exception;
import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.OutputStream;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Date;
import java.util.List;
import java.util.concurrent.TimeUnit;
import java.util.stream.Collectors;

/**
 * To run this code example, you need to create an AWS Identity and Access
 * Management (IAM) role with the correct policy as described in this
 * documentation:
 * https://docs.aws.amazon.com/pinpoint/latest/developerguide/audience-data-
export.html
 *
 * Also, set up your development environment, including your credentials.
 *
```

```
 * For information, see this documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */

public class ExportEndpoints {
    public static void main(String[] args) {
        final String usage = """

                This program performs the following steps:

                1. Exports the endpoints to an Amazon S3 bucket.
                2. Downloads the exported endpoints files from Amazon S3.
                3. Parses the endpoints files to obtain the endpoint IDs and prints
 them.
                Usage: ExportEndpoints <applicationId> <s3BucketName>
 <iamExportRoleArn> <path>

                Where:
                  applicationId - The ID of the Amazon Pinpoint application that has
 the endpoint.
                  s3BucketName - The name of the Amazon S3 bucket to export the JSON
 file to.\s
                  iamExportRoleArn - The ARN of an IAM role that grants Amazon
 Pinpoint write permissions to the S3 bucket.  path - The path where the files
 downloaded from the Amazon S3 bucket are written (for example, C:/AWS/).
                """;

        if (args.length != 4) {
            System.out.println(usage);
            System.exit(1);
        }

        String applicationId = args[0];
        String s3BucketName = args[1];
        String iamExportRoleArn = args[2];
        String path = args[3];
        System.out.println("Deleting an application with ID: " + applicationId);

        Region region = Region.US_EAST_1;
        PinpointClient pinpoint = PinpointClient.builder()
                .region(region)
                .build();
```

```
        S3Client s3Client = S3Client.builder()
                .region(region)
                .build();

        exportAllEndpoints(pinpoint, s3Client, applicationId, s3BucketName, path,
 iamExportRoleArn);
        pinpoint.close();
        s3Client.close();
    }

    public static void exportAllEndpoints(PinpointClient pinpoint,
            S3Client s3Client,
            String applicationId,
            String s3BucketName,
            String path,
            String iamExportRoleArn) {

        try {
            List<String> objectKeys = exportEndpointsToS3(pinpoint, s3Client,
 s3BucketName, iamExportRoleArn,
                    applicationId);
            List<String> endpointFileKeys = objectKeys.stream().filter(o ->
 o.endsWith(".gz"))
                    .collect(Collectors.toList());
            downloadFromS3(s3Client, path, s3BucketName, endpointFileKeys);

        } catch (PinpointException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static List<String> exportEndpointsToS3(PinpointClient pinpoint, S3Client
 s3Client, String s3BucketName,
            String iamExportRoleArn, String applicationId) {

        SimpleDateFormat dateFormat = new SimpleDateFormat("yyyy-MM-dd-
HH_mm:ss.SSS_z");
        String endpointsKeyPrefix = "exports/" + applicationId + "_" +
 dateFormat.format(new Date());
        String s3UrlPrefix = "s3://" + s3BucketName + "/" + endpointsKeyPrefix +
 "/";
        List<String> objectKeys = new ArrayList<>();
        String key;
```

```
        try {
            // Defines the export job that Amazon Pinpoint runs.
            ExportJobRequest jobRequest = ExportJobRequest.builder()
                    .roleArn(iamExportRoleArn)
                    .s3UrlPrefix(s3UrlPrefix)
                    .build();

            CreateExportJobRequest exportJobRequest =
CreateExportJobRequest.builder()
                    .applicationId(applicationId)
                    .exportJobRequest(jobRequest)
                    .build();

            System.out.format("Exporting endpoints from Amazon Pinpoint application
%s to Amazon S3 " +
                    "bucket %s . . .\n", applicationId, s3BucketName);

            CreateExportJobResponse exportResult =
pinpoint.createExportJob(exportJobRequest);
            String jobId = exportResult.exportJobResponse().id();
            System.out.println(jobId);
            printExportJobStatus(pinpoint, applicationId, jobId);

            ListObjectsV2Request v2Request = ListObjectsV2Request.builder()
                    .bucket(s3BucketName)
                    .prefix(endpointsKeyPrefix)
                    .build();

            // Create a list of object keys.
            ListObjectsV2Response v2Response = s3Client.listObjectsV2(v2Request);
            List<S3Object> objects = v2Response.contents();
            for (S3Object object : objects) {
                key = object.key();
                objectKeys.add(key);
            }

            return objectKeys;

        } catch (PinpointException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return null;
```

```
    }

    private static void printExportJobStatus(PinpointClient pinpointClient,
            String applicationId,
            String jobId) {

        GetExportJobResponse getExportJobResult;
        String status;

        try {
            // Checks the job status until the job completes or fails.
            GetExportJobRequest exportJobRequest = GetExportJobRequest.builder()
                    .jobId(jobId)
                    .applicationId(applicationId)
                    .build();

            do {
                getExportJobResult = pinpointClient.getExportJob(exportJobRequest);
                status =
getExportJobResult.exportJobResponse().jobStatus().toString().toUpperCase();
                System.out.format("Export job %s . . .\n", status);
                TimeUnit.SECONDS.sleep(3);

            } while (!status.equals("COMPLETED") && !status.equals("FAILED"));

            if (status.equals("COMPLETED")) {
                System.out.println("Finished exporting endpoints.");
            } else {
                System.err.println("Failed to export endpoints.");
                System.exit(1);
            }

        } catch (PinpointException | InterruptedException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }

    // Download files from an Amazon S3 bucket and write them to the path location.
    public static void downloadFromS3(S3Client s3Client, String path, String
s3BucketName, List<String> objectKeys) {

        String newPath;
        try {
```

```
            for (String key : objectKeys) {
                GetObjectRequest objectRequest = GetObjectRequest.builder()
                        .bucket(s3BucketName)
                        .key(key)
                        .build();

                ResponseBytes<GetObjectResponse> objectBytes =
 s3Client.getObjectAsBytes(objectRequest);
                byte[] data = objectBytes.asByteArray();

                // Write the data to a local file.
                String fileSuffix = new
 SimpleDateFormat("yyyyMMddHHmmss").format(new Date());
                newPath = path + fileSuffix + ".gz";
                File myFile = new File(newPath);
                OutputStream os = new FileOutputStream(myFile);
                os.write(data);
            }
            System.out.println("Download finished.");

        } catch (S3Exception | NullPointerException | IOException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
}
```

SDK の完全な例については、「GitHub」 の「ExportEndpoints.java」を参照してください。

HTTP

HTTP リクエストを直接 REST API に送信して Amazon Pinpoint を使用することができます。

Example POST Export Job リクエスト

Amazon Pinpoint プロジェクトのエンドポイントをエクスポートするには、Export Jobs リソースに対して POST リクエストを発行します。

```
POST /v1/apps/application_id/jobs/export HTTP/1.1
Content-Type: application/json
Accept: application/json
Host: pinpoint.us-east-1.amazonaws.com
X-Amz-Date: 20180606T001238Z
```

```
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20180606/
us-east-1/mobiletargeting/aws4_request, SignedHeaders=accept;cache-
control;content-length;content-type;host;postman-token;x-amz-date,
 Signature=c25cbd6bf61bd3b3667c571ae764b9bf2d8af61b875cacced95d1e68d91b4170
Cache-Control: no-cache

{
  "S3UrlPrefix": "s3://bucket-name/prefix",
  "RoleArn": "iam-export-role-arn"
}
```

実行する条件は以下のとおりです。

- *application-id* は、エンドポイントを含む Amazon Pinpoint プロジェクトの ID です。
- *bucket-name/prefix* は、Amazon S3 バケットの名前です。また、バケットのオブジェクトを階層別に整理するのに役立つプレフィックスでもあります。例えば、このようなプレフィックスには pinpoint/exports/endpoints/ があります。
- *iam-export-role-arn* は、バケットの書き込み権限を Amazon Pinpoint に付与する IAM ロールの Amazon リソースネーム (ARN) です。

このリクエストのレスポンスには、エクスポートジョブに関する詳細が含まれます。

```
{
    "Id": "611bdc54c75244bfa51fe7001ddb2e36",
    "JobStatus": "CREATED",
    "CreationDate": "2018-06-06T00:12:43.271Z",
    "Type": "EXPORT",
    "Definition": {
        "S3UrlPrefix": "s3://bucket-name/prefix",
        "RoleArn": "iam-export-role-arn"
    }
}
```

このレスポンスでは、Id 属性のジョブ ID が返ります。この ID を使用して、エクスポートジョブの現在のステータスを確認できます。

Example GET Export Job リクエスト

エクスポートジョブの現在のステータスを確認するには、Export Job リソースに対して GET リクエストを発行します。

```
GET /v1/apps/application_id/jobs/export/job_id HTTP/1.1
Content-Type: application/json
Accept: application/json
Host: pinpoint.us-east-1.amazonaws.com
X-Amz-Date: 20180606T002443Z
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20180606/us-
east-1/mobiletargeting/aws4_request, SignedHeaders=accept;cache-control;content-
type;host;postman-token;x-amz-date,
 Signature=c25cbd6bf61bd3b3667c571ae764b9bf2d8af61b875cacced95d1e68d91b4170
Cache-Control: no-cache
```

実行する条件は以下のとおりです。

- *application-id* は、エンドポイントをエクスポートした Amazon Pinpoint プロジェクトの ID です。
- *job-id* は、確認中のジョブの ID です。

このリクエストのレスポンスには、エクスポートジョブの現在の状態が含まれます。

```
{
    "ApplicationId": "application_id",
    "Id": "job_id",
    "JobStatus": "COMPLETED",
    "CompletedPieces": 1,
    "FailedPieces": 0,
    "TotalPieces": 1,
    "CreationDate": "2018-06-06T00:12:43.271Z",
    "CompletionDate": "2018-06-06T00:13:01.141Z",
    "Type": "EXPORT",
    "TotalFailures": 0,
    "TotalProcessed": 217,
    "Definition": {}
}
```

このレスポンスでは、JobStatus 属性のジョブステータスを返します。ジョブステータス値が COMPLETED の場合は、Amazon S3 バケットからエクスポートされたエンドポイントを取得できます。

関連情報

API のエクスポートジョブリソースに関する詳細 (例: サポートされている HTTP メソッドやリクエストパラメータ) については、Amazon Pinpoint API リファレンスの「Export Jobs」を参照してください。

Amazon Pinpoint で使用しているエンドポイント ID の一覧表示

エンドポイントを更新または削除するには、エンドポイント ID が必要です。そのため、Amazon Pinpoint プロジェクトのすべてのエンドポイントでこれらのオペレーションを実行するには、まずそのプロジェクトに属するすべてのエンドポイント ID を一覧表示します。次に、これらの ID を反復処理します。例えば、属性をグローバルに追加するか、プロジェクトのエンドポイントを削除します。

次の例では、AWS SDK for Java を使用して、以下のことを行います。

1. `exportEndpointsToS3` のサンプルコードから Amazon Pinpoint からエンドポイントをエクスポートする メソッドの例を呼び出します。このメソッドでは、Amazon Pinpoint プロジェクトからエンドポイント定義をエクスポートします。エンドポイント定義は、gzip ファイルとして Amazon S3 バケットに追加されます。
2. エクスポートされた gzip ファイルをダウンロードします。
3. gzip ファイルを読み込み、各エンドポイントの JSON 定義からエンドポイント ID を取得します。
4. エンドポイント ID がコンソールに出力されます。
5. Amazon Pinpoint によって Amazon S3 に追加されたファイルを削除してクリーンアップします。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.EndpointResponse;
import software.amazon.awssdk.services.pinpoint.model.GetUserEndpointsRequest;
import software.amazon.awssdk.services.pinpoint.model.GetUserEndpointsResponse;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import java.util.List;
```

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.EndpointResponse;
```

```
import software.amazon.awssdk.services.pinpoint.model.GetUserEndpointsRequest;
import software.amazon.awssdk.services.pinpoint.model.GetUserEndpointsResponse;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class ListEndpointIds {
    public static void main(String[] args) {
        final String usage = """

                Usage:    <applicationId> <userId>

                Where:
                   applicationId - The ID of the Amazon Pinpoint application that has
 the endpoint.
                   userId - The user id applicable to the endpoints""";

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String applicationId = args[0];
        String userId = args[1];
        PinpointClient pinpoint = PinpointClient.builder()
                .region(Region.US_EAST_1)
                .build();

        listAllEndpoints(pinpoint, applicationId, userId);
        pinpoint.close();
    }

    public static void listAllEndpoints(PinpointClient pinpoint,
            String applicationId,
            String userId) {

        try {
```

```
            GetUserEndpointsRequest endpointsRequest =
 GetUserEndpointsRequest.builder()
                    .userId(userId)
                    .applicationId(applicationId)
                    .build();

            GetUserEndpointsResponse response =
 pinpoint.getUserEndpoints(endpointsRequest);
            List<EndpointResponse> endpoints = response.endpointsResponse().item();

            // Display the results.
            for (EndpointResponse endpoint : endpoints) {
                System.out.println("The channel type is: " + endpoint.channelType());
                System.out.println("The address is  " + endpoint.address());
            }

        } catch (PinpointException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

SDK の完全な例については、「GitHub」 の「ListEndpointIs.java」を参照してください。

セグメントの作成

ユーザーセグメントは、ユーザーが最近いつ頃アプリケーションを使用したか、またはどのデバイスプラットフォームを使用しているか、などの共有特性に基づくユーザーのサブセットを表します。セグメントは、キャンペーンにより配信されたメッセージを受信するユーザーを指定します。セグメントを指定することで、ユーザーがアプリケーションに戻ってくるように、招待が適切な対象者に届くようにしたり、特典を提供したり、ユーザーエンゲージメントや購入を促進するようにします。

作成したセグメントは、1 つ以上のキャンペーンで使用できます。キャンペーンはセグメントのユーザーに合わせたメッセージを送信します。

詳細については、「セグメント」を参照してください。

トピック

- セグメントの構築
- セグメントのインポート
- AWS Lambda を使用したセグメントのカスタマイズ

セグメントの構築

キャンペーンの対象ユーザーに到達するためには、アプリケーションによって報告されたデータに基づいて、セグメントを構築します。たとえば、最近アプリを使用していないユーザーにアプローチするために、過去 30 日間アプリを使用していないユーザーのセグメントを指定できます。

AWS SDK for Java を使用したセグメントの構築

次の例では、AWS SDK for Java でセグメントを構築する方法を示します。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.AttributeDimension;
import software.amazon.awssdk.services.pinpoint.model.SegmentResponse;
import software.amazon.awssdk.services.pinpoint.model.AttributeType;
import software.amazon.awssdk.services.pinpoint.model.RecencyDimension;
import software.amazon.awssdk.services.pinpoint.model.SegmentBehaviors;
import software.amazon.awssdk.services.pinpoint.model.SegmentDemographics;
import software.amazon.awssdk.services.pinpoint.model.SegmentLocation;
import software.amazon.awssdk.services.pinpoint.model.SegmentDimensions;
```

```
import software.amazon.awssdk.services.pinpoint.model.WriteSegmentRequest;
import software.amazon.awssdk.services.pinpoint.model.CreateSegmentRequest;
import software.amazon.awssdk.services.pinpoint.model.CreateSegmentResponse;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import java.util.HashMap;
import java.util.Map;
```

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.AttributeDimension;
import software.amazon.awssdk.services.pinpoint.model.SegmentResponse;
import software.amazon.awssdk.services.pinpoint.model.AttributeType;
import software.amazon.awssdk.services.pinpoint.model.RecencyDimension;
import software.amazon.awssdk.services.pinpoint.model.SegmentBehaviors;
import software.amazon.awssdk.services.pinpoint.model.SegmentDemographics;
import software.amazon.awssdk.services.pinpoint.model.SegmentLocation;
import software.amazon.awssdk.services.pinpoint.model.SegmentDimensions;
import software.amazon.awssdk.services.pinpoint.model.WriteSegmentRequest;
import software.amazon.awssdk.services.pinpoint.model.CreateSegmentRequest;
import software.amazon.awssdk.services.pinpoint.model.CreateSegmentResponse;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import java.util.HashMap;
import java.util.Map;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class CreateSegment {
        public static void main(String[] args) {
                final String usage = """

                                Usage:   <appId>

                                Where:
                                  appId - The application ID to create a segment for.

                                """;
```

```
                if (args.length != 1) {
                        System.out.println(usage);
                        System.exit(1);
                }

                String appId = args[0];
                PinpointClient pinpoint = PinpointClient.builder()
                                .region(Region.US_EAST_1)
                                .build();

                SegmentResponse result = createSegment(pinpoint, appId);
                System.out.println("Segment " + result.name() + " created.");
                System.out.println(result.segmentType());
                pinpoint.close();
        }

        public static SegmentResponse createSegment(PinpointClient client, String
appId) {
                try {
                        Map<String, AttributeDimension> segmentAttributes = new
HashMap<>();
                        segmentAttributes.put("Team", AttributeDimension.builder()
                                        .attributeType(AttributeType.INCLUSIVE)
                                        .values("Lakers")
                                        .build());

                        RecencyDimension recencyDimension = RecencyDimension.builder()
                                        .duration("DAY_30")
                                        .recencyType("ACTIVE")
                                        .build();

                        SegmentBehaviors segmentBehaviors = SegmentBehaviors.builder()
                                        .recency(recencyDimension)
                                        .build();

                        SegmentDemographics segmentDemographics = SegmentDemographics
                                        .builder()
                                        .build();

                        SegmentLocation segmentLocation = SegmentLocation
                                        .builder()
                                        .build();

                        SegmentDimensions dimensions = SegmentDimensions
```

```
                                        .builder()
                                        .attributes(segmentAttributes)
                                        .behavior(segmentBehaviors)
                                        .demographic(segmentDemographics)
                                        .location(segmentLocation)
                                        .build();

                        WriteSegmentRequest writeSegmentRequest =
 WriteSegmentRequest.builder()
                                        .name("MySegment")
                                        .dimensions(dimensions)
                                        .build();

                        CreateSegmentRequest createSegmentRequest =
 CreateSegmentRequest.builder()
                                        .applicationId(appId)
                                        .writeSegmentRequest(writeSegmentRequest)
                                        .build();

                        CreateSegmentResponse createSegmentResult =
 client.createSegment(createSegmentRequest);
                        System.out.println("Segment ID: " +
 createSegmentResult.segmentResponse().id());
                        System.out.println("Done");
                        return createSegmentResult.segmentResponse();

                } catch (PinpointException e) {
                        System.err.println(e.awsErrorDetails().errorMessage());
                        System.exit(1);
                }
                return null;
        }
}
```

この例を実行すると、IDE のコンソールウィンドウに次のように出力されます。

```
Segment ID: 09cb2967a82b4a2fbab38fead8d1f4c4
```

SDK の完全な例については、「GitHub」の「CreateSegment.java」を参照してください。

セグメントのインポート

Amazon Pinpoint では、セグメントに属するエンドポイントに関する情報をインポートしてユーザーセグメントを定義できます。エンドポイントは、モバイルプッシュデバイストークン、携帯電話番号、または E メールアドレスなどの、単一のメッセージ送信先です。

Amazon Pinpoint の外にすでにユーザーのセグメントを作成しているが、Amazon Pinpoint キャンペーンにユーザーを関与させたい場合は、セグメントのインポートが役立ちます。

セグメントをインポートする際、Amazon Pinpoint は Amazon Simple Storage Service (Amazon S3) からセグメントのエンドポイントを取得します。インポートする前に、Amazon S3 にエンドポイントを追加し、Amazon S3 に対して Amazon Pinpoint へのアクセス権限を付与する IAM ロールを作成します。次に、エンドポイントが保存される Amazon S3 の場所を Amazon Pinpoint に与えると、Amazon Pinpoint が各エンドポイントをセグメントに追加します。

IAM ロールを作成するには、「エンドポイントまたはセグメントをインポートするための IAM ロール」を参照してください。Amazon Pinpoint コンソールを使用したセグメントのインポートに関する詳細については、『Amazon Pinpoint ユーザーガイド』の「セグメントのインポート」を参照してください。

セグメントのインポート

次の例では、AWS SDK for Java を使用してセグメントをインポートする方法を示します。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.CreateImportJobRequest;
import software.amazon.awssdk.services.pinpoint.model.ImportJobResponse;
import software.amazon.awssdk.services.pinpoint.model.ImportJobRequest;
import software.amazon.awssdk.services.pinpoint.model.Format;
import software.amazon.awssdk.services.pinpoint.model.CreateImportJobResponse;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
```

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.CreateImportJobRequest;
import software.amazon.awssdk.services.pinpoint.model.ImportJobResponse;
import software.amazon.awssdk.services.pinpoint.model.ImportJobRequest;
import software.amazon.awssdk.services.pinpoint.model.Format;
import software.amazon.awssdk.services.pinpoint.model.CreateImportJobResponse;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
```

```
/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class ImportSegment {
    public static void main(String[] args) {
        final String usage = """

                Usage:   <appId> <bucket> <key> <roleArn>\s

                Where:
                  appId - The application ID to create a segment for.
                  bucket - The name of the Amazon S3 bucket that contains the segment
 definitons.
                  key - The key of the S3 object.
                  roleArn - ARN of the role that allows Amazon Pinpoint to access S3.
 You need to set trust management for this to work. See https://docs.aws.amazon.com/
IAM/latest/UserGuide/reference_policies_elements_principal.html
                  """;

        if (args.length != 4) {
            System.out.println(usage);
            System.exit(1);
        }

        String appId = args[0];
        String bucket = args[1];
        String key = args[2];
        String roleArn = args[3];

        PinpointClient pinpoint = PinpointClient.builder()
                .region(Region.US_EAST_1)
                .build();

        ImportJobResponse response = createImportSegment(pinpoint, appId, bucket, key,
 roleArn);
        System.out.println("Import job for " + bucket + " submitted.");
        System.out.println("See application " + response.applicationId() + " for import
 job status.");
```

```
        System.out.println("See application " + response.jobStatus() + " for import job
 status.");
        pinpoint.close();
    }

    public static ImportJobResponse createImportSegment(PinpointClient client,
            String appId,
            String bucket,
            String key,
            String roleArn) {

        try {
            ImportJobRequest importRequest = ImportJobRequest.builder()
                    .defineSegment(true)
                    .registerEndpoints(true)
                    .roleArn(roleArn)
                    .format(Format.JSON)
                    .s3Url("s3://" + bucket + "/" + key)
                    .build();

            CreateImportJobRequest jobRequest = CreateImportJobRequest.builder()
                    .importJobRequest(importRequest)
                    .applicationId(appId)
                    .build();

            CreateImportJobResponse jobResponse = client.createImportJob(jobRequest);
            return jobResponse.importJobResponse();

        } catch (PinpointException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return null;
    }
}
```

SDK の例については、「GitHub」の「ImportingSegments.java」を参照してください。

AWS Lambda を使用したセグメントのカスタマイズ

これはパブリックベータリリースの機能に関するプレリリースドキュメントです。このドキュメントは変更される可能性があります。

AWS Lambda 関数を使用して、Amazon Pinpoint キャンペーンに対象者を取り込む方法を調整できます。AWS Lambda では、Amazon Pinpoint がキャンペーンのメッセージを送信するときに、キャンペーンのセグメントを変更することができます。

AWS Lambda はサーバーをプロビジョニングしたり管理しなくてもコードを実行するために使用できるコンピューティングサービスです。コードをパッケージ化し、Lambda 関数として Lambda にアップロードします。関数が呼び出されたときに、Lambda によって関数が実行されます。この操作は手動で行うことも、イベントに応じて自動的に行うこともできます。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。

キャンペーンに Lambda 関数を割り当てるには、Amazon Pinpoint API のキャンペーンリソースを使用して、キャンペーンの `CampaignHook` 設定を定義します。これらの設定には、Lambda 関数名が含まれます。`CampaignHook` モードも含まれます。これは、Amazon Pinpoint が関数から戻り値を受け取るかどうかを指定します。

キャンペーンに割り当てる Lambda 関数を、Amazon Pinpoint の拡張機能と呼びます。

`CampaignHook` 設定が定義されると、Amazon Pinpoint はキャンペーンを実行する際、メッセージの送信前に自動的に Lambda 関数を呼び出します。Amazon Pinpoint が関数を呼び出すと、メッセージ配信に関するイベントデータを提供します。このデータには、キャンペーンのセグメントが含まれます。これは Amazon Pinpoint がメッセージを送信するエンドポイントのリストです。

`CampaignHook` モードが `FILTER` に設定されている場合、Amazon Pinpoint により、関数はメッセージを送信する前に、セグメントを変更して返すことができます。例えば、関数は、Amazon Pinpoint の外部にあるソースからのデータを含む属性でエンドポイント定義を更新する場合があります。または、関数コードの条件に基づいて、特定のエンドポイントを削除してセグメントをフィルタリングする場合があります。Amazon Pinpoint が変更されたセグメントを関数から受け取ると、キャンペーンの配信チャネルを使用して、各セグメントのエンドポイントにメッセージを送信します。

AWS Lambda でセグメントを処理することにより、メッセージの送信先のユーザーと、メッセージの内容をより詳細に管理できます。キャンペーンメッセージが送信されるときに、キャンペーンをリアルタイムで調整できます。セグメントをフィルタリングすると、サブセットの定義されたサブセットを絞り込んで適用できます。エンドポイント属性を追加または更新しても、メッセージ変数で新しいデータを使用できるようになります。

> **Note**
>
> また、`CampaignHook` 設定を使用して、メッセージ配信を処理する Lambda 関数を割り当てることもできます。このタイプの関数は、ソーシャルメディアプラットフォームな

ど、Amazon Pinpoint がサポートしないカスタムチャネルを通じてメッセージを配信するときに役立ちます。詳細については、「Amazon Pinpoint でカスタムチャンネルの作成」を参照してください。
Amazon Pinpoint を使用して Lambda フックを呼び出す場合、Lambda 関数も Amazon Pinpoint プロジェクトと同じリージョンにある必要があります。

AWS Lambda でキャンペーンセグメントを変更するには、まず、Amazon Pinpoint によって送信されたイベントデータを処理し、変更されたセグメントを返す関数を作成します。次に、Lambda 関数ポリシーを割り当てて関数を呼び出すことを Amazon Pinpoint に許可します。最後に、CampaignHook 設定を定義して、1 つ以上のキャンペーンに関数を割り当てます。

イベントデータ

Amazon Pinpoint が Lambda 関数を呼び出すときに、イベントデータとして次のペイロードを提供します。

```
{
  "MessageConfiguration": {Message configuration}
  "ApplicationId": ApplicationId,
  "CampaignId": CampaignId,
  "TreatmentId": TreatmentId,
  "ActivityId": ActivityId,
  "ScheduledTime": Scheduled Time,
  "Endpoints": {
    EndpointId: {Endpoint definition}
    . . .
  }
}
```

AWS Lambda はイベントデータをユーザーの関数コードに渡します。イベントデータは次の属性を提供します。

- MessageConfiguration – Amazon Pinpoint API のメッセージリソースの DirectMessageConfiguration オブジェクトと同じ構造を持つ。
- ApplicationId – キャンペーンが属する Amazon Pinpoint プロジェクトの ID。
- CampaignId – 関数が呼び出される Amazon Pinpoint キャンペーンの ID。
- TreatmentId – A/B テストに使用されるキャンペーンバリエーションの ID。

- `ActivityId` – キャンペーンが実行しているアクティビティの ID。
- `ScheduledTime` – キャンペーンのメッセージが配信される日時 (ISO 8601 形式)。
- `Endpoints` – エンドポイント ID をエンドポイント定義と関連付けるマップ。各イベントデータのペイロードには最大 50 のエンドポイントが含まれます。キャンペーンセグメントに 50 以上のエンドポイントが含まれる場合、Amazon Pinpoint はすべてのエンドポイントが処理されるまで、繰り返し関数を呼び出します (最大で一度に 50 のエンドポイント)。

Lambda 関数の作成

Lambda 関数の作成方法については、『AWS Lambda デベロッパーガイド』の「開始方法」を参照してください。関数を作成するときは、以下の条件ではメッセージ配信が失敗することに注意してください。

- Lambda 関数では、変更されたセグメントを返すために 15 秒以上かかります。
- Amazon Pinpoint は関数の戻り値をデコードすることはできません。
- この関数では、正常な呼び出しのためには、Amazon Pinpoint から 3 回を超える試行が必要です。

Amazon Pinpoint は、関数の戻り値でのみエンドポイントの定義を受け入れます。関数は、イベントデータで他の要素を変更することはできません。

Lambda 関数の例

Lambda 関数は、Node.js で書かれた次の例のように、Amazon Pinpoint によって送信されたイベントデータを処理し、変更されたエンドポイントを返します。

```
'use strict';

exports.handler = (event, context, callback) => {
    for (var key in event.Endpoints) {
        if (event.Endpoints.hasOwnProperty(key)) {
            var endpoint = event.Endpoints[key];
            var attr = endpoint.Attributes;
            if (!attr) {
                attr = {};
                endpoint.Attributes = attr;
            }
            attr["CreditScore"] = [ Math.floor(Math.random() * 200) + 650];
        }
```

```
    }
    console.log("Received event:", JSON.stringify(event, null, 2));
    callback(null, event.Endpoints);
};
```

Lambda は event パラメータとしてイベントデータをハンドラに渡します。

この例では、ハンドラは event.Endpoints オブジェクトの各エンドポイントで反復処理し、エンドポイントに新しい属性 CreditScore を追加します。CreditScore 属性の値は、単純にランダムな数値です。

console.log() ステートメントは CloudWatch Logs にイベントのログを記録します。

callback() ステートメントは変更されたエンドポイントを Amazon Pinpoint に返します。通常、Node.js Lambda 関数で callback パラメータはオプションですが、この関数は更新されたエンドポイントを Amazon Pinpoint に返す必要があるため、このコンテキストで必要です。

関数は、イベントデータで提供されるのと同じ形式でエンドポイントを返す必要があります。これは、次の例に示すように、エンドポイント ID とエンドポイント定義を関連付けるマップです。

```
{
    "eqmj8wpxszeqy/b3vch04sn41yw": {
        "ChannelType": "GCM",
        "Address": "4d5e6f1a2b3c4d5e6f7g8h9i0j1a2b3c",
        "EndpointStatus": "ACTIVE",
        "OptOut": "NONE",
        "Demographic": {
            "Make": "android"
        },
        "EffectiveDate": "2017-11-02T21:26:48.598Z",
        "User": {}
    },
    "idrexqqtn8sbwfex0ouscod0yto": {
        "ChannelType": "APNS",
        "Address": "1a2b3c4d5e6f7g8h9i0j1a2b3c4d5e6f",
        "EndpointStatus": "ACTIVE",
        "OptOut": "NONE",
        "Demographic": {
            "Make": "apple"
        },
        "EffectiveDate": "2017-11-02T21:26:48.598Z",
        "User": {}
```

```
    }
}
```

この例の関数では、イベントデータで受信した event.Endpoints オブジェクトを変更して返します。

必要に応じて、返されるエンドポイント定義に TitleOverride および BodyOverride 属性を含めることができます。

> **Note**
>
> このソリューションを使用してメッセージを送信すると、Amazon Pinpoint は、ChannelType 属性の値が次のいずれかの場合、エンドポイントの TitleOverride および BodyOverride 属性のみを尊重します: ADM、APNS、APNS_SANDBOX、APNS_VOIP、APNS_VOIP_SANDBOX、BAIDU、GCM、または SMS。
> Amazon Pinpoint は、ChannelType 属性の値が EMAIL であるエンドポイントについては、これらの属性を尊重しません。

Lambda 関数ポリシーの割り当て

Lambda 関数を使用してエンドポイントを処理する前に、Amazon Pinpoint に Lambda 関数の呼び出しを許可する必要があります。呼び出しのアクセス許可を付与するには、関数に Lambda 関数ポリシーを割り当てます。Lambda 関数ポリシーはリソースベースのアクセス許可ポリシーで、関数を使用できるエンティティと、それらのエンティティが実行できるアクションを指定します。

詳細については、『AWS Lambda デベロッパーガイド』の「AWS Lambda のリソースベースのポリシーを使用する」を参照してください。

関数ポリシーの例

以下のポリシーは、Amazon Pinpoint サービスプリンシパルに、特定のキャンペーン (*campaign-id*) に対して lambda:InvokeFunction アクションを使用する許可を与えるものです。

```
{
  "Sid": "sid",
  "Effect": "Allow",
  "Principal": {
```

```
    "Service": "pinpoint.us-east-1.amazonaws.com"
  },
  "Action": "lambda:InvokeFunction",
  "Resource": "{arn:aws:lambda:us-east-1:account-id:function:function-name}",
  "Condition": {
    "StringEquals": {
      "AWS:SourceAccount": "111122223333"
    },
    "ArnLike": {
      "AWS:SourceArn": "arn:aws:mobiletargeting:us-east-1:account-id:apps/application-
id/campaigns/campaign-id"
    }
  }
}
```

関数ポリシーは、Condition キーを含む AWS:SourceArn ブロックを必要とします。このコードは、関数を呼び出すために許可される Amazon Pinpoint キャンペーンを指定します。この例では、ポリシーは 1 つのキャンペーンのみにアクセス許可を付与します。Condition ブロックは AWS:SourceAccount キーも含まなければなりません。このキーは、どの AWS アカウントがアクションを呼び出すことができるかを制御します。

より一般的なポリシーを記述するには、複数文字に一致するワイルドカード (*) を使用します。例えば、次の Condition ブロックを使用して、特定の Amazon Pinpoint プロジェクト (*application-id*) の任意のキャンペーンに関数の呼び出しを許可できます。

```
...
"Condition": {
  "StringEquals": {
    "AWS:SourceAccount": "111122223333"
  },
  "ArnLike": {
    "AWS:SourceArn": "arn:aws:mobiletargeting:us-east-1:account-id:apps/application-id/
campaigns/*"
  }
}
...
```

Lambda 関数をプロジェクトのすべてのキャンペーンで使用されるデフォルト関数にする場合は、前述の方法でポリシーの Condition ブロックを構成することをお勧めします。プロジェクト内のすべてのキャンペーンのデフォルトとして Lambda 関数を設定する方法については、キャンペーンへの Lambda 関数の割り当て を参照してください。

Amazon Pinpoint の呼び出し許可の付与

AWS Command Line Interface (AWS CLI) を使用して、Lambda 関数に割り当てられた Lambda 関数ポリシーにアクセス許可を追加できます。Amazon Pinpoint に特定のキャンペーンの関数の呼び出しを許可するには、次の例に示すように Lambda add-permission コマンドを使用します。

```
$ aws lambda add-permission \
> --function-name function-name \
> --statement-id sid \
> --action lambda:InvokeFunction \
> --principal pinpoint.us-east-1.amazonaws.com \
> --source-account 111122223333
> --source-arn arn:aws:mobiletargeting:us-east-1:account-id:apps/application-id/
campaigns/campaign-id
```

キャンペーン ID は、AWS CLI のget-campaigns コマンドで調べることができます。また、アプリケーション ID は get-apps コマンドで調べることができます。

Lambda add-permission のコマンドを実行すると、Lambda により以下のような出力が表示されます。

```
{
  "Statement": "{\"Sid\":\"sid\",
    \"Effect\":\"Allow\",
    \"Principal\":{\"Service\":\"pinpoint.us-east-1.amazonaws.com\"},
    \"Action\":\"lambda:InvokeFunction\",
    \"Resource\":\"arn:aws:lambda:us-east-1:111122223333:function:function-name\",
    \"Condition\":
      {\"ArnLike\":
        {\"AWS:SourceArn\":
         \"arn:aws:mobiletargeting:us-east-1:111122223333:apps/application-id/
campaigns/campaign-id\"}}
      {\"StringEquals\":
        {\"AWS:SourceAccount\":
          \"111122223333\"}}}
}
```

Statement 値は、Lambda 関数ポリシーに追加されたステートメントの JSON 文字列バージョンです。

キャンペーンへの Lambda 関数の割り当て

Lambda 関数は個別の Amazon Pinpoint キャンペーンに割り当てることができます。または、個別に関数を割り当てるキャンペーンを除いて、プロジェクトのすべてのキャンペーンで使用されるデフォルトとして、Lambda 関数を設定できます。

Lambda 関数を個別のキャンペーンに割り当てるには、Amazon Pinpoint API を使用して `Campaign` オブジェクトを作成または更新し、その `CampaignHook` 属性を定義します。プロジェクトのすべてのキャンペーン用にデフォルトとして Lambda 関数を設定するには、そのプロジェクト用の `Settings` リソースを作成または更新し、その `CampaignHook` オブジェクトを定義します。

いずれの場合も、次の `CampaignHook` 属性を設定します。

- `LambdaFunctionName` – キャンペーンのメッセージを送信する前に Amazon Pinpoint が呼び出す Lambda 関数の名前または ARN。
- `Mode` – `FILTER` に設定します。このモードでは、Amazon Pinpoint はこの関数を呼び出し、変更されたエンドポイントが返されるのを待ちます。エンドポイントを受け取った後、Amazon Pinpointは、メッセージを送信します。Amazon Pinpoint は最大 15 秒待ってから、メッセージ配信を失敗させます。

キャンペーンに対して `CampaignHook` 設定を定義した状態で、Amazon Pinpoint はキャンペーンのメッセージを送信する前に、指定された Lambda 関数を呼び出します。Amazon Pinpoint は、変更されたエンドポイントの関数からの受け取りを待機します。Amazon Pinpoint が更新されたエンドポイントを受信する場合、更新されたエンドポイントデータを使用して、メッセージの配信を続行します。

キャンペーンの作成

アプリケーションとユーザーの間のエンゲージメントを向上するために、Amazon Pinpoint を使用してユーザーの特定のセグメントを対象とするプッシュ通知キャンペーンを作成し管理します。

例えば、キャンペーンにより、最近アプリを実行していないユーザーをアプリケーションに戻るよう招待し、最近購入していないユーザーに特別なプロモーションを提供します。

キャンペーンでは、指定したセグメントのユーザーに向けたメッセージを送信します。キャンペーンではセグメントのすべてのユーザーにメッセージを送信できます。または、メッセージを受け取らないユーザーの割合である保留 (Holdout) を割り当てることもできます。

メッセージを 1 回、または 1 週間に一度など、定期的な間隔で送信するように、キャンペーンのスケジュールを設定できます。ユーザーが都合の悪い時間にメッセージを受信しないように、スケジュールにメッセージを送信しないクワイエットタイムを含めることができます。

別のキャンペーン戦略を試すには、キャンペーンを A/B テストとして設定します。A / B テストには、メッセージまたはスケジュールの 2 つ以上の処理が含まれます。処理によって、メッセージまたはスケジュールに違いを持たせます。ユーザーがキャンペーンに応答すると、キャンペーン分析を表示してそれぞれの処理の効果性を比較できます。

詳細については、「キャンペーン」を参照してください。

標準キャンペーンの作成

標準キャンペーンは、定義したスケジュールに基づいて、指定したセグメントにカスタムのプッシュ通知を送信します。

を使用したキャンペーンの作成 AWS SDK for Java

次の例では、 AWS SDK for Javaでキャンペーンを作成する方法を示します。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.CampaignResponse;
import software.amazon.awssdk.services.pinpoint.model.Message;
import software.amazon.awssdk.services.pinpoint.model.Schedule;
import software.amazon.awssdk.services.pinpoint.model.Action;
import software.amazon.awssdk.services.pinpoint.model.MessageConfiguration;
```

```
import software.amazon.awssdk.services.pinpoint.model.WriteCampaignRequest;
import software.amazon.awssdk.services.pinpoint.model.CreateCampaignResponse;
import software.amazon.awssdk.services.pinpoint.model.CreateCampaignRequest;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
```

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.CampaignResponse;
import software.amazon.awssdk.services.pinpoint.model.Message;
import software.amazon.awssdk.services.pinpoint.model.Schedule;
import software.amazon.awssdk.services.pinpoint.model.Action;
import software.amazon.awssdk.services.pinpoint.model.MessageConfiguration;
import software.amazon.awssdk.services.pinpoint.model.WriteCampaignRequest;
import software.amazon.awssdk.services.pinpoint.model.CreateCampaignResponse;
import software.amazon.awssdk.services.pinpoint.model.CreateCampaignRequest;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class CreateCampaign {
    public static void main(String[] args) {

        final String usage = """

                Usage:   <appId> <segmentId>

                Where:
                  appId - The ID of the application to create the campaign in.
                  segmentId - The ID of the segment to create the campaign from.
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String appId = args[0];
```

```
        String segmentId = args[1];
        PinpointClient pinpoint = PinpointClient.builder()
                .region(Region.US_EAST_1)
                .build();

        createPinCampaign(pinpoint, appId, segmentId);
        pinpoint.close();
    }

    public static void createPinCampaign(PinpointClient pinpoint, String appId, String
segmentId) {
        CampaignResponse result = createCampaign(pinpoint, appId, segmentId);
        System.out.println("Campaign " + result.name() + " created.");
        System.out.println(result.description());
    }

    public static CampaignResponse createCampaign(PinpointClient client, String appID,
String segmentID) {

        try {
            Schedule schedule = Schedule.builder()
                    .startTime("IMMEDIATE")
                    .build();

            Message defaultMessage = Message.builder()
                    .action(Action.OPEN_APP)
                    .body("My message body.")
                    .title("My message title.")
                    .build();

            MessageConfiguration messageConfiguration = MessageConfiguration.builder()
                    .defaultMessage(defaultMessage)
                    .build();

            WriteCampaignRequest request = WriteCampaignRequest.builder()
                    .description("My description")
                    .schedule(schedule)
                    .name("MyCampaign")
                    .segmentId(segmentID)
                    .messageConfiguration(messageConfiguration)
                    .build();

            CreateCampaignResponse result =
client.createCampaign(CreateCampaignRequest.builder()
```

```
                    .applicationId(appID)
                    .writeCampaignRequest(request).build());

            System.out.println("Campaign ID: " + result.campaignResponse().id());
            return result.campaignResponse();

        } catch (PinpointException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }

        return null;
    }
}
```

この例を実行すると、IDE のコンソールウィンドウに次のように出力されます。

```
Campaign ID: b1c3de717aea4408a75bb3287a906b46
```

SDK の完全な例については、 のCreateCampaign「.java」を参照してくださいGitHub。

A/B テストキャンペーンの作成

A/B テストは標準キャンペーンのように動作しますが、キャンペーンメッセージまたはスケジュールに異なる処理を定義できます。

を使用した A/B テストキャンペーンの作成 AWS SDK for Java

次の例では、 AWS SDK for Javaで A/B テストキャンペーンを作成する方法を示します。

```
import com.amazonaws.services.pinpoint.AmazonPinpointClient;
import com.amazonaws.services.pinpoint.model.Action;
import com.amazonaws.services.pinpoint.model.CampaignResponse;
import com.amazonaws.services.pinpoint.model.CreateCampaignRequest;
import com.amazonaws.services.pinpoint.model.CreateCampaignResult;
import com.amazonaws.services.pinpoint.model.Message;
import com.amazonaws.services.pinpoint.model.MessageConfiguration;
import com.amazonaws.services.pinpoint.model.Schedule;
import com.amazonaws.services.pinpoint.model.WriteCampaignRequest;
import com.amazonaws.services.pinpoint.model.WriteTreatmentResource;

import java.util.ArrayList;
```

```
import java.util.List;

public class PinpointCampaignSample {

    public CampaignResponse createAbCampaign(AmazonPinpointClient client, String appId,
 String segmentId) {
        Schedule schedule = new Schedule()
                .withStartTime("IMMEDIATE");

        // Default treatment.
        Message defaultMessage = new Message()
                .withAction(Action.OPEN_APP)
                .withBody("My message body.")
                .withTitle("My message title.");

        MessageConfiguration messageConfiguration = new MessageConfiguration()
                .withDefaultMessage(defaultMessage);

        // Additional treatments
        WriteTreatmentResource treatmentResource = new WriteTreatmentResource()
                .withMessageConfiguration(messageConfiguration)
                .withSchedule(schedule)
                .withSizePercent(40)
                .withTreatmentDescription("My treatment description.")
                .withTreatmentName("MyTreatment");

        List<WriteTreatmentResource> additionalTreatments = new
 ArrayList<WriteTreatmentResource>();
        additionalTreatments.add(treatmentResource);

        WriteCampaignRequest request = new WriteCampaignRequest()
                .withDescription("My description.")
                .withSchedule(schedule)
                .withSegmentId(segmentId)
                .withName("MyCampaign")
                .withMessageConfiguration(messageConfiguration)
                .withAdditionalTreatments(additionalTreatments)
                .withHoldoutPercent(10); // Hold out of A/B test

        CreateCampaignRequest createCampaignRequest = new CreateCampaignRequest()
                .withApplicationId(appId).withWriteCampaignRequest(request);

        CreateCampaignResult result = client.createCampaign(createCampaignRequest);
```

```
        System.out.println("Campaign ID: " + result.getCampaignResponse().getId());

        return result.getCampaignResponse();
    }

}
```

この例を実行すると、IDE のコンソールウィンドウに次のように出力されます。

```
Campaign ID: b1c3de717aea4408a75bb3287a906b46
```

Amazon Pinpoint SMS および音声 API バージョン 2 の使用

> **Note**
>
> Amazon Pinpoint のユーザーガイドドキュメントを更新しました。Amazon Pinpoint SMS と音声リソースを作成、設定、管理する方法に関する最新情報を入手するには、新しい「Amazon Pinpoint SMS ユーザーガイド」を参照してください。
> 次のトピックは、新しい Amazon Pinpoint SMS ユーザーガイドに移動されました。
>
> - 電話番号の管理
> - 送信者 IDs の管理
> - プールの管理
> - オプトアウトリストの管理
> - 設定セットの管理
> - キーワードの管理
> - イベント送信先の管理
> - メッセージの送信

Amazon Pinpoint には、SMS と音声メッセージの送信用に設計された API (SMS および音声 API バージョン 2 と呼ばれる) が含まれています。Amazon Pinpoint API は、スケジュールされたイベント主導型のキャンペーンやジャーニーを通じてメッセージを送信することに重点を置いていますが、SMS および音声 API は、SMS と音声メッセージを個々の受信者に直接送信するための新機能を提供します。SMS および音声 API は、Amazon Pinpoint のキャンペーンとジャーニーの機能から独立して使用できます。また、さまざまなユースケースに対応するために両方を同時に使用できます。既に Amazon Pinpoint を使用して SMS または音声メッセージを送信している場合、アカウントはこの API を使用するように既に設定されています。

この API は、独立系ソフトウェアベンダー (ISV) など、マルチテナントアーキテクチャのユーザーに適したソリューションです。この API を使用すると、イベントデータ、送信元電話番号、およびオプトアウトリストをテナントごとに分けることが容易になります。

SMS および音声 API を使用するときは、設定セットとイベント送信先を設定することをお勧めします。SMS および音声 API は、送信したメッセージのイベントデータを自動的には出力しません。イベント送信先を設定すると、メッセージ配信や失敗イベントなどの重要なイベントデータを確実にキャプチャできます。

この API のバージョン 2 の前に、バージョン 1 がありました。現在、この API のバージョン 1 を使用している場合は、引き続き使用可能で、使用を継続することは可能です。ただし、バージョン 2 に移行すると、電話番号プールの作成、プログラムによる新しい電話番号のリクエスト、電話番号の特定の機能の有効化/無効化などの追加機能を利用できます。

> **Note**
>
> 現時点では、一部のタスクは、Amazon Pinpoint コンソールを使用する方法でのみ完了できます。例えば、アカウントが SMS サンドボックスにあるときに使用する電話番号を検証する場合や、10DLC を使用するために登録する場合は、Amazon Pinpoint コンソールを使用する必要があります。

このセクションには、この API に関する情報と使用方法の例が記載されています。リファレンスドキュメントは、「SMS and Voice version 2 API Reference」にもあります。

Amazon Pinpoint でのワンタイムパスワード (OTPsの送信と検証

Amazon Pinpoint には、ワンタイムパスワード (OTP) を管理する機能があります。この機能を使用して、新しいワンタイムパスワードを生成し、SMS メッセージとして受信者に送信できます。その後、アプリケーションは Amazon Pinpoint API を呼び出して、これらのパスワードを検証します。

> **Important**
>
> この機能を使用するには、アカウントに本番稼働用アクセスとアクティブな発信元 ID が必要です。詳細については、Amazon Pinpoint SMS ユーザーガイド」の「Amazon Pinpoint SMS サンドボックスについて」および「電話番号をリクエストする」を参照してください。
> Amazon Pinpoint
> 一部の国や地域では、SMS メッセージを送信する前に、専用の電話番号または発信元 ID を取得する必要があります。例えば、米国の受信者にメッセージを送信する場合、専用の通話料無料の番号、10DLC 番号、またはショートコードが必要です。インドにいる受信者にメッセージを送信する場合、PEID (プリンシパル エンティティ ID) および テンプレート ID を含む送信者 ID を登録する必要があります。これらの要件は、OTP 機能を使用する際にも適用されます。
> この機能を使用するには、OTP メッセージの送信と検証のためのアクセス許可が必要です。「ワンタイムパスワード」を参照してください。アクセス許可の決定についてサポートが必要な場合は、「Amazon Pinpoint Identity and Access Management のトラブルシューティング」を参照してください。

OTP メッセージの送信

Amazon Pinpoint API の `SendOtpMessages` オペレーションを利用して、アプリケーションのユーザーに OTP コードを送信できます。この API を使用すると、Amazon Pinpoint はランダムなコードを生成し、SMS メッセージとしてユーザーに送信します。リクエストには、次のパラメータを含めることができます。

- `Channel` — OTP コードが送信される通信チャネル。現在、SMS メッセージのみがサポートされているため、許容される値は SMS のみです。
- `BrandName` — OTP コードに関連するブランド名、会社名、商品名。この名前の長さは最大 20 文字です。

> **Note**
>
> Amazon Pinpointが OTP メッセージを送信する際、以下のメッセージテンプレートにブランド名が自動的に挿入されます。
>
> ```
> This is your One Time Password: {{otp}} from {{brand}}
> ```
>
> したがって、ブランド名 ExampleCorp として を指定し、Amazon Pinpoint が 123456 のワンタイムパスワードを生成すると、ユーザーに次のメッセージを送信します。
>
> ```
> This is your One Time Password: 123456 from ExampleCorp
> ```

- `CodeLength` — 受信者に送信されるワンタイムコードの桁数 OTP コードは 5 桁から 8 桁の数字で構成されます。
- `ValidityPeriod` — OTPコードが有効になる時間 (分) です。有効期間は 5 分から 60 分までです。
- `AllowedAttempts` – 受信者が OTP の検証で失敗できる回数。試行回数がこの値を超えると、OTP は自動的に無効になります。試行回数の上限は 5 回です。
- `Language` — メッセージを送信するときに使用する言語 (IETF BCP-47 形式)。許容値は次のとおりです。
  - `de-DE` – ドイツ語
  - `en-GB` – 英語 (英国)
  - `en-US` – 英語 (米国)
  - `es-419` – スペイン語 (ラテンアメリカ)
  - `es-ES` – スペイン語
  - `fr-CA` – フランス語 (カナダ)
  - `fr-FR` – フランス語
  - `it-IT` – イタリア語
  - `ja-JP` – 日本語
  - `ko-KR` – 韓国語
  - `pt-BR` – ポルトガル語 (ブラジル)
  - `zh-CN` – 簡体字中国語

  - zh-TW – 繁体字中国語
- OriginationIdentity – OTP コードの送信に使用される発信元 ID (ロングコード、ショートコード、送信者 ID など)。OTP 送信にロングコードまたは通話料無料を使用する場合、電話番号は E.164 形式である必要があります。
- DestinationIdentity — OTP コードの送信先の電話番号 (E.164形式)。
- ReferenceId — リクエストの一意の参照 ID。参照 ID は、OTP を検証するときに入力した参照 ID と完全に一致します。参照 ID は、1 文字以上 48 文字以下になります。
- EntityId — 規制機関に登録されているエンティティ ID。このパラメータは現在、インドの受信者にメッセージを送信する場合にのみ使用されます。インドの受信者に送信しない場合は、このパラメータを省略できます。
- TemplateId — 規制機関にテンプレートされているエンティティ ID。このパラメータは現在、インドの受信者にメッセージを送信する場合にのみ使用されます。インドの受信者に送信しない場合は、このパラメータを省略できます。

> **Note**
>
> インドの受信者にメッセージを送信するための要件の詳細については、『Amazon Pinpoint ユーザーガイド』の「Special requirements for sending SMS messages to recipients in India」を参照してください。

Amazon Pinpoint アカウントが OTP メッセージを送信するように適切に設定されていることを確認するには、 AWS CLI を使用してテストメッセージを送信できます。の詳細については AWS CLI、「 AWS Command Line Interface ユーザーガイド」を参照してください。

を使用してテスト OTP メッセージを送信するには AWS CLI、ターミナルで send-otp-message コマンドを実行します。

```
aws pinpoint send-otp-message --application-id 7353f53e6885409fa32d07cedexample --send-
otp-message-request-parameters
 Channel=SMS,BrandName=ExampleCorp,CodeLength=5,ValidityPeriod=20,AllowedAttempts=5,Origination
```

上記のコマンドで、次の操作を行います。

- *7353f53e6885409fa32d07cedexample* をアプリケーション ID に置き換えます。
- を会社名*ExampleCorp*に置き換えます。

- の *5* を、受信者に送信される OTP コードに含まれる桁数CodeLegthに置き換えます。
- の *20* を、OTP コードが有効である分単位の時間ValidityPeriodに置き換えます。
- の *5* を、受信者が OTP の検証に失敗した回数AllowedAttemptsに置き換えます。
- の *+18555550142* を、OTP コードの送信に使用される発信元 ID OriginationIdentityに置き換えます。
- の *+12065550007* を、OTP コードの送信先の電話番号DestinationIdentityに置き換えます。
- *SampleReferenceId* の をリクエストの一意の参照 ID ReferenceIdに置き換えます。

SendOtpMessage 応答

OTP メッセージを正常に送信すると、次の例のようなレスポンスが表示されます。

```
{
    "MessageResponse": {
        "ApplicationId": "7353f53e6885409fa32d07cedexample",
        "RequestId": "255d15ea-75fe-4040-b919-096f2example",
        "Result": {
            "+12065550007": {
                "DeliveryStatus": "SUCCESSFUL",
                "MessageId": "nvrmgq9kq4en96qgp0tlqli3og1at6aexample",
                "StatusCode": 200,
                "StatusMessage": "MessageId: nvrmgq9kq4en96qgp0tlqli3og1at6aexample"
            }
        }
    }
}
```

OTP メッセージの認証

OTP コードを検証するには、VerifyOtpMessages API を呼び出します。リクエストには次のパラメータが含まれます。

- DestinationIdentity — OTP コードの送信先の電話番号 (E.164形式)。
- ReferenceId — OTP コードを受信者に送信したときに使用した参照 ID。参照 ID は完全に一致する必要があります。
- Otp — 認証している OTP コード。

を使用して検証プロセスを AWS CLI テストできます。のインストールと設定の詳細については AWS CLI、「 AWS Command Line Interface ユーザーガイド」を参照してください。

を使用して OTP を検証するには AWS CLI、ターミナルで verify-otp-message コマンドを実行します。

```
aws pinpoint verify-otp-message --application-id 7353f53e6885409fa32d07cedexample --
verify-otp-message-request-parameters
 DestinationIdentity=+12065550007,ReferenceId=SampleReferenceId,Otp=01234
```

上記のコマンドで、次の操作を行います。

- *7353f53e6885409fa32d07cedexample* をアプリケーション ID に置き換えます。
- の *+12065550007* を、OTP コードの送信先の電話番号DestinationIdentityに置き換えます。
- *SampleReferenceId* の をリクエストの一意の参照 ID ReferenceIdに置き換えます。この値は、リクエストの送信にReferenceID使用された と一致する必要があります。
- の *01234* を、 に送信された Otp Otpに置き換えますDestinationIdentity。

VerifyOtpMessage 応答

VerifyOTPMessage API にリクエストを送信すると、VerificationResponse オブジェクトが返されます。これには 1 つのプロパティ、Valid が含まれています。参照 ID、電話番号、OTP のすべてが Amazon Pinpoint が期待する値と一致し、OTP が有効期限内であれば、Valid は true となり、それ以外は false となります。以下は、OTP 検証に成功した場合のレスポンスの例です。

```
{
    "VerificationResponse": {
        "Valid": true
    }
}
```

コードの例

SDK for Python (Boto3) を使って、OTP コードを送信および認証するためのコードの例を紹介します。

参照 ID を生成する

次の機能は、受信者の電話番号、受信者が OTP を受信する製品やブランド、リクエスト元 (例えば、サイトやアプリケーションのページ名など) から、受信者ごとに一意の参照 ID を生成するものです。OTP コードを認証するときは、検証を成功させるために同一の参照 ID を渡す必要があります。送信コードおよび検証コードの例では、いずれもこのユーティリティ関数を使用しています。

この関数は必須ではありませんが、OTP 送信および検証プロセスの範囲を特定のトランザクションに限定し、検証ステップ中に簡単に再送信できるようにするために使用すると便利です。任意の参照 ID を使用できます。これは基本的な例に過ぎません。ただし、このセクションの他のコード例は、この関数に依存しています。

```
# Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
# SPDX-License-Identifier:  Apache-2.0

import hashlib

def generate_ref_id(destinationNumber,brandName,source):
    refId = brandName + source + destinationNumber
    return hashlib.md5(refId.encode()).hexdigest()
```

OTP コードの送信

次のコード例では、SDK for Python (Boto3) を使用して OTP コードを送信する方法を示しています。

```
# Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
# SPDX-License-Identifier:  Apache-2.0

import boto3
from botocore.exceptions import ClientError
from generate_ref_id import generate_ref_id

### Some variables that are unlikely to change from request to request. ###

# The AWS Region that you want to use to send the message.
region = "us-east-1"

# The phone number or short code to send the message from.
originationNumber = "+18555550142"
```

```
# The project/application ID to use when you send the message.
appId = "7353f53e6885409fa32d07cedexample"

# The number of times the user can unsuccessfully enter the OTP code before it becomes
 invalid.
allowedAttempts = 3

# Function that sends the OTP as an SMS message.
def send_otp(destinationNumber,codeLength,validityPeriod,brandName,source,language):
    client = boto3.client('pinpoint',region_name=region)
    try:
        response = client.send_otp_message(
            ApplicationId=appId,
            SendOTPMessageRequestParameters={
                'Channel': 'SMS',
                'BrandName': brandName,
                'CodeLength': codeLength,
                'ValidityPeriod': validityPeriod,
                'AllowedAttempts': allowedAttempts,
                'Language': language,
                'OriginationIdentity': originationNumber,
                'DestinationIdentity': destinationNumber,
                'ReferenceId': generate_ref_id(destinationNumber,brandName,source)
            }
        )

    except ClientError as e:
        print(e.response)
    else:
        print(response)

# Send a message to +14255550142 that contains a 6-digit OTP that is valid for 15
 minutes. The
# message will include the brand name "ExampleCorp", and the request originated from a
 part of your
# site or application called "CreateAccount". The US English message template should be
 used to
# send the message.
send_otp("+14255550142",6,15,"ExampleCorp","CreateAccount","en-US")
```

OTP コードの認証

次のコードの例は、SDK for Python (Boto3) を使用して既に送信した OTP コードを検証する方法を示しています。認証のステップを成功させるためには、お客様のリクエストに、メッセージの送信に使用された参照 ID と完全に一致する参照 ID が含まれている必要があります。

```
# Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
# SPDX-License-Identifier:  Apache-2.0

import boto3
from botocore.exceptions import ClientError
from generate_ref_id import generate_ref_id

# The AWS Region that you want to use to send the message.
region = "us-east-1"

# The project/application ID to use when you send the message.
appId = "7353f53e6885409fa32d07cedexample"

# Function that verifies the OTP code.
def verify_otp(destinationNumber,otp,brandName,source):
    client = boto3.client('pinpoint',region_name=region)
    try:
        response = client.verify_otp_message(
            ApplicationId=appId,
            VerifyOTPMessageRequestParameters={
                'DestinationIdentity': destinationNumber,
                'ReferenceId': generate_ref_id(destinationNumber,brandName,source),
                'Otp': otp
            }
        )

    except ClientError as e:
        print(e.response)
    else:
        print(response)

# Verify the OTP 012345, which was sent to +14255550142. The brand name ("ExampleCorp")
 and the
# source name ("CreateAccount") are used to generate the correct reference ID.
verify_otp("+14255550142","012345","ExampleCorp","CreateAccount")
```

Amazon Pinpoint でアプリケーション内のメッセージの送信および取得

Amazon Pinpoint でアプリケーション内メッセージを作成および送信し、アプリがそれを取得および表示するように設定できます。アプリケーション内メッセージは、高度なカスタマイズができます。ウェブサイトを開いたり、アプリケーション内の特定の場所に移動するためのボタンを配置することができます。背景色や文字色の設定、テキストの配置、ボタンやイメージの追加など、通知に必要な設定を行うことができます。1 つのメッセージだけを送信することも、最大 5 つまでの異なるメッセージを格納するカルーセルを作成することもできです。アプリケーション内のメッセージの概要、アプリケーション内のメッセージテンプレートの作成方法については、『Amazon Pinpoint ユーザーガイド』の「Creating in-app templates」を参照してください。

AWS Amplify を利用することで、Amazon Pinpoint のアプリケーション内のメッセージング機能をお客様のアプリケーションにシームレスに統合することができます。Amplify は、メッセージの取得、メッセージのレンダリング、Amazon Pinpoint への分析データの送信プロセスを自動的に処理することができます。この統合は現在、React Native アプリケーションでサポートされています。詳細については、『Amplify Framework Documentation』の「In-App Messaging」を参照してください。

このセクションでは、アプリケーションのエンドポイントのためのアプリケーション内のメッセージをリクエストする方法と、そのリクエストの結果を解釈する方法について説明します。

エンドポイントのアプリケーション内のメッセージの取得

アプリケーションは、GetInAppMessages API を呼び出すことで、指定されたエンドポイントに権利があるすべてのアプリケーション内のメッセージを取得することができます。`GetInAppMessages` API を呼び出す際には、以下のパラメータを指定します。

- `ApplicationId` – アプリケーション内のメッセージキャンペーンが関連付けられている Amazon Pinpoint プロジェクトの一意の ID です。
- `EndpointId` — メッセージを取得するエンドポイントの一意の ID です。

これらの値を指定して API を呼び出すと、メッセージの一覧が返されます。このオペレーションで生成されるレスポンスの詳細については、「`GetInAppMessages` API レスポンスの理解」を参照してください。

GetInAppMessages SDK を使用して AWS オペレーションを呼び出すことができます。次のコード例には、アプリケーション内のメッセージを取得する関数が含まれています。

JavaScript

別のモジュールでクライアントを作成し、エクスポートします。

```
import { PinpointClient } from "@aws-sdk/client-pinpoint";
const REGION = "us-east-1";
const pinClient = new PinpointClient({ region: REGION });
export { pinClient };
```

エンドポイントのアプリケーション内のメッセージを取得します。

```

// Import required AWS SDK clients and commands for Node.js
import { PinpointClient, GetInAppMessagesCommand } from "@aws-sdk/client-pinpoint";
import { pinClient } from "./lib/pinClient.js";

("use strict");

//The Amazon Pinpoint application ID.
const projectId = "4c545b28d21a490cb51b0b364example";

//The ID of the endpoint to retrieve messages for.
const endpointId = "c5ac671ef67ee3ad164cf7706example";

const params = {
  ApplicationId: projectId,
  EndpointId: endpointId
};

const run = async () => {
  try {
    const data = await pinClient.send(new GetInAppMessagesCommand(params));
    console.log(JSON.stringify(data, null, 4));
    return data;
  } catch (err) {
    console.log("Error", err);
  }
};
run();
```

Python

```
import logging
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)

def retrieve_inapp_messages(
            pinpoint_client, project_id, endpoint_id):
    """
    Retrieves the in-app messages that a given endpoint is entitled to.

    :param pinpoint_client: A Boto3 Pinpoint client.
    :param project_id: An Amazon Pinpoint project ID.
    :param endpoint_id: The ID of the endpoint to retrieve messages for.
    :return: A JSON object that contains information about the in-app message.
    """

    try:
        response = pinpoint_client.get_in_app_messages(
            ApplicationId=project_id,
            EndpointId=endpoint_id)
    except ClientError:
        logger.exception("Couldn't retrieve messages.")
        raise
    else:
        return response

def main():
    project_id = "4c545b28d21a490cb51b0b364example"
    endpoint_id = "c5ac671ef67ee3ad164cf7706example"
    inapp_response = retrieve_inapp_messages(
        boto3.client('pinpoint'), project_id, endpoint_id)
    print(inapp_response)

if __name__ == '__main__':
    main()
```

GetInAppMessages API レスポンスの理解

GetInAppMessages API オペレーションを呼び出すと、指定されたエンドポイントに権利があるメッセージのリストが返されます。そして、アプリケーションはレスポンスの値に基づいてメッセージをレンダリングすることができます。

以下は、GetInAppMessages APIを呼び出した際に返される JSON オブジェクトの例です。

```
{
  "InAppMessagesResponse":{
    "InAppMessageCampaigns":[
      {
        "CampaignId":"inAppTestCampaign-4c545b28d21a490cb51b0b364example",
        "DailyCap":0,
        "InAppMessage":{
          "Content":[
            {
              "BackgroundColor":"#f8e71c",
              "BodyConfig":{
                "Alignment":"CENTER",
                "Body":"This is a sample in-app message sent using Amazon Pinpoint.",
                "TextColor":"#d0021b"
              },
              "HeaderConfig":{
                "Alignment":"CENTER",
                "Header":"Sample In-App Message",
                "TextColor":"#d0021b"
              },
              "ImageUrl":"https://example.com/images/thumbnail.png",
              "PrimaryBtn":{
                "DefaultConfig":{
                  "BackgroundColor":"#d0021b",
                  "BorderRadius":50,
                  "ButtonAction":"CLOSE",
                  "Text":"Dismiss",
                  "TextColor":"#f8e71c"
                }
              }
            }
          ],
          "Layout":"MIDDLE_BANNER"
        },
        "Priority":3,
```

```
        "Schedule":{
          "EndDate":"2021-11-06T00:08:05Z",
          "EventFilter":{
            "Dimensions":{
              "Attributes":{

              },
              "EventType":{
                "DimensionType":"INCLUSIVE",
                "Values":[
                  "_session.start"
                ]
              },
              "Metrics":{

              }
            }
          }
        },
        "SessionCap":0,
        "TotalCap":0,
        "TreatmentId":"0"
      }
    ]
  }
}
```

次のセクションでは、このレスポンスの構成要素に関する詳細について説明します。

InAppMessageCampaigns オブジェクト

InAppMessageCampaigns オブジェクトには、次の属性が含まれます。

属性	説明	設定場所は
CampaignId	メッセージの送信元である Amazon Pinpoint キャンペーンの名前と一意のキャンペーン ID を格納する文字列です。キャンペーン ID の前に名前が	キャンペーンの作成時に Amazon Pinpoint によって自動的に作成されます。

属性	説明	設定場所は
	付加されます。2 つの値はハイフン (-) で区切られます。	
TreatmentId	このメッセージのキャンペーン処理の ID を表す整数値です。キャンペーンに 1 つの処理しかない場合、値は 0 です。	
Priority	アプリケーション内メッセージの優先度。1 から 5 までの整数で表され、1 が最も高い優先度、5 が最も低い優先度を表します。	キャンペーン作成手順のステップ 1。
InAppMessage	メッセージのレンダリング方法に関する情報を格納する InAppMessage オブジェクト。	キャンペーンに指定されたアプリケーション内メッセージテンプレートの内容に基づきます。
Schedule	メッセージの送信日時に関する情報を格納する Schedule オブジェクトです。	キャンペーン作成プロセスのステップ4 (コンソールでキャンペーンを作成した場合) または Schedule オブジェクト (API または SDK を使用してキャンペーンを作成した場合)。
DailyCap	24 時間以内にアプリケーション内のメッセージを表示できる回数を整数で表したものです。	プロジェクトレベルの設定から継承されます。キャンペーンにプロジェクトの設定が上書きされるように設定されている場合、それらの設定が代わりに使用されます。

属性	説明	設定場所は
SessionCap	アプリケーションセッションにおいて、アプリケーション内のメッセージをユーザーに表示できる回数を整数で表したものです。	
TotalCap	キャンペーンごとにエンドポイントへ表示できるアプリケーション内のメッセージの合計回数を整数で表したものです。	

InAppMessage オブジェクト

InAppMessage オブジェクトには、次の属性が含まれます。

属性	説明	設定場所は
Content	メッセージの内容を記述した InAppMessageContent オブジェクトを格納する配列です。	キャンペーンに指定されたアプリケーション内メッセージテンプレートの内容に基づきます。
Layout	アプリケーション内のメッセージが受信者の端末にどのように表示されるかを記述する文字列です。可能な値は以下のとおりです。 • BOTTOM_BANNER — ページの下部にバナーとして表示されるメッセージ。 • TOP_BANNER — ページの上部にバナーとして表示されるメッセージ。	

属性	説明	設定場所は
	• OVERLAYS — 画面全体をカバーするメッセージ。 • MOBILE_FEED — ページの手前にあるウィンドウに表示されるメッセージ。 • MIDDLE_BANNER — ページの中央にバナーとして表示されるメッセージ。 • CAROUSEL — 最大 5 つの一意のメッセージをスクロールできるレイアウト。	

HeaderConfig オブジェクト

HeaderConfig オブジェクトには、次の属性が含まれます。

属性	説明	設定場所は
Alignment	ヘッダーのテキストのアライメントを指定する文字列です。指定できる値は LEFT、CENTER、および RIGHT です。	キャンペーンに指定されたアプリケーション内メッセージテンプレートの内容に基づきます。
Header	メッセージのヘッダーのテキスト。	
TextColor	ヘッダーのテキストの色。16 進数のカラーコードを格納する文字列で表示されます (例: 黒の場合は「#000000」)。	

BodyConfig オブジェクト

BodyConfig オブジェクトには、次の属性が含まれます。

属性	説明	設定場所は
Alignment	メッセージの本文のアライメントを指定する文字列です。指定できる値は LEFT、CENTER、および RIGHT です。	キャンペーンに指定されたアプリケーション内メッセージテンプレートの内容に基づきます。
Body	メッセージの本文	
TextColor	本文のテキストの色。16 進数のカラーコードを格納する文字列で表示されます (例: 黒の場合は「#000000」)。	

InAppMessageContent オブジェクト

InAppMessageContent オブジェクトには、次の属性が含まれます。

属性	説明	設定場所は
BackgroundColor	アプリケーション内のメッセージの背景色。16 進数のカラーコードを格納する文字列で表されます (例: 黒の場合は「#000000」)。	キャンペーンに指定されたアプリケーション内メッセージテンプレートの内容に基づきます。
BodyConfig	メッセージの本文に関連する情報を格納する BodyConfig オブジェクトです。	
HeaderConfig	メッセージのヘッダやタイトルに関連する情報を格納する	

属性	説明	設定場所は
	HeaderConfig オブジェクトです。	
ImageUrl	メッセージに表示されるイメージの URL です。	
PrimaryBtn	メッセージ内のメインボタンに関する情報を格納する InAppMessageButton オブジェクトです。	
SecondaryBtn	メッセージ内のメインボタンに関する情報を格納する InAppMessageButton オブジェクトです。アプリケーション内のメッセージのテンプレートでセカンダリボタンが指定されていない場合は存在しません。	

Schedule オブジェクト

Schedule オブジェクトには、次の属性が含まれます。

属性	説明	設定場所は
EndDate	キャンペーン終了の予定時刻 (ISO 8601 形式) です。	キャンペーン作成プロセスのステップ4 (コンソールでキャンペーンを作成した場合) または Schedule オブジェクト (API または SDK を使用してキャンペーンを作成した場合)。
EventFilter	アプリケーション内のメッセージが表示される要因となったイベントに関する情報です。Amazon Pinpoint アプリケーション内のキャンペーンに一致するイベントを生成	

属性	説明	設定場所は
	すると、メッセージが表示されます。	

InAppMessageButton オブジェクト

InAppMessageButton オブジェクトには、次の属性が含まれます。

属性	説明	設定場所は
DefaultConfig	アプリケーション内のメッセージに配置されたボタンのデフォルト設定に関する情報を格納する DefaultButtonConfig オブジェクトです。	キャンペーンに指定されたアプリケーション内メッセージテンプレートの内容に基づきます。
Android	Android 端末でのボタンの動作を指定する OverrideButtonConfig オブジェクトです。これは、DefaultConfig オブジェクトで詳述されているデフォルトのボタン設定を上書きします。	
IOS	iOS 端末でのボタンの動作を指定する OverrideButtonConfig オブジェクトです。これは、DefaultConfig オブジェクトで詳述されているデフォルトのボタン設定を上書きします。	
Web	ウェブアプリケーションでのボタンの動作を指定する OverrideButtonConfig オブジェクトです。これ	

属性	説明	設定場所は
	は、DefaultConfig オブジェクトで詳述されているデフォルトのボタン設定を上書きします。	

DefaultButtonConfig オブジェクト

DefaultButtonConfig オブジェクトには、次の属性が含まれます。

属性	説明	設定場所は
BackgroundColor	ボタンの背景色。16 進数のカラーコードを格納する文字列で表されます (例: 黒の場合は「#000000」)。	キャンペーンに指定されたアプリケーション内メッセージテンプレートの内容に基づきます。
BorderRadius	ボタンの境界線の半径をピクセル単位で整数値で指定します。数値が大きいほど、角が丸くなります。	
ButtonAction	受信者がアプリケーション内のメッセージに配置されたボタンを選択したときに発生するアクションを説明する文字列です。可能な値は以下のとおりです。 • LINK — ウェブ上の移動先へのリンク。 • DEEP_LINK — アプリケーション内の特定のページへのリンク。 • CLOSE – メッセージを閉じます。	

属性	説明	設定場所は
Link	ボタンの移動先の URL。ButtonAction が CLOSE であるボタンには存在しません。	
Text	ボタンに表示されるテキスト。	
TextColor	ボタンテキストの色。16 進数のカラーコードを含む文字列で表示されます (例: 黒の場合は「#000000」)。	

OverrideButtonConfig オブジェクト

OverrideButtonConfig オブジェクトは、アプリケーション内のメッセージテンプレートがオーバーライドボタンを使用している場合にのみ存在します。オーバーライドボタンとは、iOS デバイス、Android デバイス、ウェブブラウザなど、特定のデバイスの種類に応じた設定を持つボタンのことです。

OverrideButtonConfig オブジェクトには、次の属性が含まれます。

属性	説明	設定場所は
ButtonAction	アプリケーション内のメッセージで受信者がボタンを選択した際に発生するアクションです。可能な値は以下のとおりです。 • LINK — ウェブ上の移動先へのリンク。 • DEEP_LINK — アプリケーション内の特定のページへのリンク。	キャンペーンに指定されたアプリケーション内メッセージテンプレートの内容に基づきます。

属性	説明	設定場所は
	• CLOSE – メッセージを閉じます。	
Link	ボタンの移動先の URL。ButtonAction が CLOSE であるボタンには存在しません。	
Text	ボタンに表示されるテキスト。	
TextColor	ボタンテキストの色。16 進数のカラーコードを含む文字列で表示されます (例: 黒の場合は「#000000」)。	

Amazon Pinpoint での電話番号の検証

Amazon Pinpoint には、電話番号が有効かどうかを判断したり、電話番号自体に関する追加情報を入手するために使用できる電話番号検証サービスが含まれています。例えば、電話番号検証サービスを使用すると、次のような情報が返されます。

- この電話番号は E.164 形式です。
- 電話番号の種類 (携帯電話、固定電話、または VoIP など)。
- 電話番号の市および国または。
- 電話番号に関連付けられているサービスプロバイダー。

電話番号検証サービスの使用には追加料金が発生します。詳細については、「Amazon Pinpoint pricing」を参照してください。

> **Important**
>
> 米国とカナダの電話番号発信の場合、電話番号検証 API は `City`、`County`、`Timezone`、`ZipCode` のデータを返しません。

電話番号検証のユースケース

電話番号検証サービスを使用すると、以下のようないくつかのユースケースを有効にできます。

- [Verifying phone numbers provided on a web form] - ウェブベースのフォームを使用してお客様の連絡先情報を収集する場合は、フォームを送信する前にお客様が提供する電話番号を検証します。ウェブサイトのバックエンドを使用して、Amazon Pinpoint API を使用して番号を検証します。API レスポンスは、番号が無効であるかどうか (例えば、電話番号が正しくフォーマットされていないなど) を示します。お客様が提供した電話番号が無効であると判断した場合、ウェブフォームはお客様に別の番号を提供するように促すことができます。
- [Cleansing your existing contact database] - お客様の電話番号のデータベースがある場合は、各電話番号を検証して、調査結果に基づいてデータベースを更新することができます。例えば、SMS メッセージを受信できない電話番号を持つエンドポイントを見つけた場合は、そのエンドポイントの `ChannelType` プロパティを `SMS` から `VOICE` に変更できます。最初に電話番号を確認してから、Amazon Pinpoint へエンドポイントを追加する (単一のエンドポイントの場合) または エンド

ポイントのバッチを Amazon Pinpoint に追加する (複数のエンドポイント場合) の指示に従って、新規または既存のエンドポイントの ChannelType プロパティを更新できます。

- [Choosing the right channel before you send a message] - SMS メッセージを送信しようとしているが送信先番号が無効であると判断した場合は、別のチャネルを介して受信者にメッセージを送信できます。例えば、エンドポイントが、SMS メッセージを受信できない場合は、代わりに音声メッセージを送信できます。

電話番号検証サービスを使用する

次の例は、 を使用して電話番号を検証する方法を示しています AWS CLI。詳細については、 コマンドリファレンスphone-number-validateの「」を参照してください。 AWS CLI 検証レスポンスの例については、「」を参照してください電話番号検証のレスポンス。の設定の詳細については AWS CLI、「 ユーザーガイド」の「 の設定 AWS CLIAWS Command Line Interface」を参照してください。

を使用して電話番号検証サービスを使用するには AWS CLI

- コマンドラインで以下のコマンドを入力します。

```
aws pinpoint phone-number-validate --number-validate-request
 PhoneNumber=+442079460881,IsoCountryCode=GB
```

前述のコマンドで、*+442079460881* を検証したい電話番号に置き換え、*GB* を 2 桁の ISO 国または地域コードに置き換えます。

> **Note**
>
> 電話番号を電話番号検証サービスに入力するときは、必ず国コードを含める必要があります。国コードを含めないと、サービスは別の国の電話番号の情報を返す場合があります。電話番号にダッシュを付けることができます。例えば、*+44-207-946-0881* です。

電話番号検証のレスポンス

電話番号検証サービスが提供する情報は、入力した電話番号に使用できるデータによって多少異なります。このセクションには、電話番号検証サービスが返すレスポンスの例が含まれています。

> **Note**
>
> 電話番号検証サービスによって提供されるデータは、電気通信プロバイダーおよび世界中の他のエンティティによって提供される情報に基づいています。一部の国のプロバイダーは、他の国のプロバイダーほど頻繁にこの情報を更新しない場合があります。例えば、携帯電話番号を検証するリクエストを発行し、指定した番号がある携帯電話会社から別の携帯電話会社に移動した場合、電話番号検証サービスからのレスポンスには、現在の携帯電話会社ではなく、元の携帯電話会社の名前が含まれます。

有効な携帯電話番号

電話番号検証サービスにリクエストを送信したときに、その電話番号が有効な携帯電話番号である場合は、次の例のような情報が返されます。

```
{
    "NumberValidateResponse": {
        "Carrier": "ExampleCorp Mobile",
        "City": "Seattle",
        "CleansedPhoneNumberE164": "+12065550142",
        "CleansedPhoneNumberNational": "2065550142",
        "Country": "United States",
        "CountryCodeIso2": "US",
        "CountryCodeNumeric": "1",
        "OriginalPhoneNumber": "+12065550142",
        "PhoneType": "MOBILE",
        "PhoneTypeCode": 0,
        "Timezone": "America/Los_Angeles",
        "ZipCode": "98101"
    }
}
```

有効な固定電話番号

リクエストに有効な固定電話番号が含まれている場合、電話番号検証サービスは次の例のような情報を返します。

```
{
    "CountryCodeIso2": "US",
    "CountryCodeNumeric": "1",
    "Country": "United States",
```

```
    "City": "Santa Clara",
    "ZipCode": "95037",
    "Timezone": "America/Los_Angeles",
    "CleansedPhoneNumberNational": "4085550101",
    "CleansedPhoneNumberE164": "14085550101",
    "Carrier": "AnyCompany",
    "PhoneTypeCode": 1,
    "PhoneType": "LANDLINE",
    "OriginalPhoneNumber": "+14085550101"
}
```

有効な VoIP 電話番号

リクエストに有効なボイスオーバーインターネットプロトコル (VoIP) 電話番号が含まれている場合、電話番号検証サービスは次の例のような情報を返します。

```
{
    "NumberValidateResponse": {
        "Carrier": "ExampleCorp",
        "City": "Countrywide",
        "CleansedPhoneNumberE164": "+441514960001",
        "CleansedPhoneNumberNational": "1514960001",
        "Country": "United Kingdom",
        "CountryCodeIso2": "GB",
        "CountryCodeNumeric": "44",
        "OriginalPhoneNumber": "+441514960001",
        "PhoneType": "VOIP",
        "PhoneTypeCode": 2
    }
}
```

無効な電話番号

リクエストに無効な電話番号が含まれている場合、電話番号検証サービスは次の例のような情報を返します。

```
{
    "NumberValidateResponse": {
        "CleansedPhoneNumberE164": "+44163296076",
        "CleansedPhoneNumberNational": "163296076",
        "Country": "United Kingdom",
        "CountryCodeIso2": "GB",
```

```
        "CountryCodeNumeric": "44",
        "OriginalPhoneNumber": "+440163296076",
        "PhoneType": "INVALID",
        "PhoneTypeCode": 3
    }
}
```

このレスポンスの PhoneType プロパティは、この電話番号が INVALID であること、および電話番号に関連付けられた携帯電話会社または場所に関する情報が含まれていないことを示しています。PhoneType が INVALID である電話番号に SMS または音声メッセージを送信することは避けてください。これらの番号は実際の受信者に属しているとは考えられないためです。

その他の電話番号

場合によっては、電話番号検証サービスからのレスポンスに OTHER の PhoneType 値が含まれています。次のような状況では、サービスはこのようなレスポンスを返す場合があります。

- この電話番号は通話料無料の番号です。
- この電話番号は、555 で始まる北米の電話番号など、テレビ番組や映画での使用にリザーブされています。
- この電話番号には、北米の999 市外局番など、現在使用されていない市外局番が含まれています。
- この電話番号は他の目的のためにリザーブされています。

以下の例は、リクエストに架空の北米の電話番号が含まれている場合に電話番号検証サービスが提供するレスポンスを示しています。

```
{
    "NumberValidateResponse": {
        "Carrier": "Multiple OCN Listing",
        "CleansedPhoneNumberE164": "+14255550199",
        "CleansedPhoneNumberNational": "4255550199",
        "Country": "United States",
        "CountryCodeIso2": "US",
        "CountryCodeNumeric": "1",
        "OriginalPhoneNumber": "+14255550199",
        "PhoneType": "OTHER",
        "PhoneTypeCode": 4,
        "Timezone": "America/Los_Angeles"
    }
}
```

プリペイド電話の番号

リクエストに有効なプリペイド電話の番号が含まれている場合、電話番号検証サービスは次の例のような情報を返します。

```
{
    "NumberValidateResponse": {
        "Carrier": "ExampleCorp",
        "City": "Countrywide",
        "CleansedPhoneNumberE164": "+14255550199",
        "CleansedPhoneNumberNational": "4255550199",
        "Country": "United States",
        "CountryCodeIso2": "US",
        "CountryCodeNumeric": "1",
        "OriginalPhoneNumber": "+14255550199",
        "PhoneType": "PREPAID",
        "PhoneTypeCode": 5
    }
}
```

これらの応答に含まれる情報の詳細については、「Amazon Pinpoint API リファレンス」の「電話番号の検証」を参照してください。

アプリケーションからトランザクションメッセージを送信する

Amazon Pinpoint API および AWS SDK を使用して、アプリケーションから直接トランザクションメッセージを送信できます。トランザクションメッセージは、セグメントに送信するメッセージとは反対に、特定の受診者に送るメッセージです。キャンペーンベースのメッセージではなくトランザクションメッセージを送信する理由はいくつかあります。たとえば、顧客が注文を確定したとき、注文を確認するために E メールを送信します。サービスのアカウントの作成プロセスを完了するために顧客が使用する一時パスワードを、SMS または音声で送ることもできます。

このセクションには、トランザクションの E メール、SMS メッセージ、音声メッセージの送信を開始できる、いくつかのプログラミング言語のコードの例が含まれます。

このセクションのトピック:

- トランザクション E メールメッセージの送信
- SMS メッセージの送信
- 音声メッセージの送信
- プッシュ通知の送信

トランザクション E メールメッセージの送信

このセクションでは、Amazon Pinpoint を介してトランザクション E メールメッセージを送信するのに使用できる完全なコードの例を提供します。

- Amazon Pinpoint API の SendMessages オペレーションを使用して、Amazon Pinpoint API の SendMessagesオペレーションを使用して、プッシュ通知、SMS、音声、E メールチャネルなど、Amazon Pinpoint がサポートするすべてのチャネルでメッセージを送信できます。

  このオペレーションを使用する利点は、すべてのチャンネルにおいてメッセージ送信のリクエスト構文が非常に類似することです。このため、既存のコードの再利用が容易になります。SendMessages オペレーションを使用すると、E メールメッセージのコンテンツを置換して、特定の E メールアドレスではなく Amazon Pinpoint エンドポイント ID に E メールを送信することもできます。

このセクションには、トランザクションの E メールの送信を開始できる、いくつかのプログラミング言語のコードの例が含まれます。

このセクションのトピック:

E メール送信方法の選択

トランザクション E メールの送信に使用する最適な方法は、ユースケースによって異なります。例えば、サードパーティーのアプリケーションを使用して E メールを送信する必要がある場合や、プログラミング言語で使用できる AWS SDK がない場合は、SMTP インターフェイスを使用する必要がある場合があります。Amazon Pinpoint がサポートする他のチャンネルでメッセージを送信する場合、これらのリクエストを一貫したコードで作成するには、Amazon Pinpoint API の `SendMessages` オペレーションを使用してください。

Amazon Pinpoint または Amazon Simple Email Service (SES) の選択

購入確認やパスワードリセットメッセージなどの大量のトランザクション E メールを送信する場合は、Amazon SES の使用を検討してください。Amazon SES には API と SMTP インターフェイスがあり、どちらもアプリケーションやサービスから E メールを送信するのに適しています。また、E メールの受信機能、設定セット、送信承認機能など、そのほかにも E メールの機能があります。

Amazon SES には、Salesforce などの顧客関係管理 (CRM) サービスを含む既存のサードパーティーアプリケーションと統合できる SMTP インターフェイスもあります。Amazon SES を使用した E メールの送信の詳細については、『Amazon Simple Email Service デベロッパーガイド』を参照してください。

Amazon Pinpoint API を使用して E メールを送信する

このセクションでは、 AWS SDK を使用して Amazon Pinpoint API 経由で E メールを送信するために使用できる完全なコード例を示します。

C#

この例を使用して、AWS SDK for .NET を使用して E メールを送信します。この例は、 AWS SDK for .NETがすでにインストールされ、設定されていることを前提としています。詳細については、『AWS SDK for .NET デベロッパーガイド』の「AWS SDK for .NETの使用開始」を参照してください。

この例では、共有認証情報ファイルを使用して、既存のユーザーのアクセスキーとシークレットアクセスキーを指定することを前提としています。詳細については、AWS SDK for .NET デベロッパーガイドの「AWS 認証情報の設定」を参照してください。

このコード例は、 AWS SDK for .NET バージョン 3.3.29.13 および .NET Core ランタイムバージョン 2.1.2 を使用してテストされました。

```
using Amazon;
using Amazon.Pinpoint;
using Amazon.Pinpoint.Model;
using Microsoft.Extensions.Configuration;

namespace SendEmailMessage;

public class SendEmailMainClass
{
    public static async Task Main(string[] args)
    {
        var configuration = new ConfigurationBuilder()
        .SetBasePath(Directory.GetCurrentDirectory())
        .AddJsonFile("settings.json") // Load test settings from .json file.
        .AddJsonFile("settings.local.json",
            true) // Optionally load local settings.
        .Build();

        // The AWS Region that you want to use to send the email. For a list of
        // AWS Regions where the Amazon Pinpoint API is available, see
        // https://docs.aws.amazon.com/pinpoint/latest/apireference/
        string region = "us-east-1";

        // The "From" address. This address has to be verified in Amazon Pinpoint
        // in the region you're using to send email.
        string senderAddress = configuration["SenderAddress"]!;

        // The address on the "To" line. If your Amazon Pinpoint account is in
```

```
        // the sandbox, this address also has to be verified.
        string toAddress = configuration["ToAddress"]!;

        // The Amazon Pinpoint project/application ID to use when you send this
 message.
        // Make sure that the SMS channel is enabled for the project or application
        // that you choose.
        string appId = configuration["AppId"]!;

        try
        {
            await SendEmailMessage(region, appId, toAddress, senderAddress);
        }
        catch (Exception ex)
        {
            Console.WriteLine("The message wasn't sent. Error message: " +
 ex.Message);
        }
    }

    public static async Task<MessageResponse> SendEmailMessage(
        string region, string appId, string toAddress, string senderAddress)
    {
        var client = new
 AmazonPinpointClient(RegionEndpoint.GetBySystemName(region));

        // The subject line of the email.
        string subject = "Amazon Pinpoint Email test";

        // The body of the email for recipients whose email clients don't
        // support HTML content.
        string textBody = @"Amazon Pinpoint Email Test (.NET)"
                          + "\n---------------------------------"
                          + "\nThis email was sent using the Amazon Pinpoint API
 using the AWS SDK for .NET.";

        // The body of the email for recipients whose email clients support
        // HTML content.
        string htmlBody = @"<html>"
                          + "\n<head></head>"
                          + "\n<body>"
                          + "\n  <h1>Amazon Pinpoint Email Test (AWS SDK for .NET)</
h1>"
                          + "\n  <p>This email was sent using the "
```

```
                            + "\n    <a href='https://aws.amazon.com/pinpoint/'>Amazon
 Pinpoint</a> API "
                            + "\n    using the <a href='https://aws.amazon.com/sdk-
for-net/'>AWS SDK for .NET</a>"
                            + "\n  </p>"
                            + "\n</body>"
                            + "\n</html>";

        // The character encoding the you want to use for the subject line and
        // message body of the email.
        string charset = "UTF-8";

        var sendRequest = new SendMessagesRequest
        {
            ApplicationId = appId,
            MessageRequest = new MessageRequest
            {
                Addresses = new Dictionary<string, AddressConfiguration>
                {
                    {
                        toAddress,
                        new AddressConfiguration
                        {
                            ChannelType = ChannelType.EMAIL
                        }
                    }
                },
                MessageConfiguration = new DirectMessageConfiguration
                {
                    EmailMessage = new EmailMessage
                    {
                        FromAddress = senderAddress,
                        SimpleEmail = new SimpleEmail
                        {
                            HtmlPart = new SimpleEmailPart
                            {
                                Charset = charset,
                                Data = htmlBody
                            },
                            TextPart = new SimpleEmailPart
                            {
                                Charset = charset,
                                Data = textBody
                            },
```

```
                            Subject = new SimpleEmailPart
                            {
                                Charset = charset,
                                Data = subject
                            }
                        }
                    }
                }
            }
        };
        Console.WriteLine("Sending message...");
        SendMessagesResponse response = await client.SendMessagesAsync(sendRequest);
        Console.WriteLine("Message sent!");
        return response.MessageResponse;
    }
}
```

Java

この例を使用して、AWS SDK for Java を使用して E メールを送信します。この例は、 AWS SDK for Java 2.xがすでにインストールされ、設定されていることを前提としています。詳細については、『AWS SDK for Java 2.x デベロッパーガイド』の「使用開始」を参照してください。

この例では、共有認証情報ファイルを使用して、既存のユーザーのアクセスキーとシークレットアクセスキーを指定することを前提としています。詳細については、『AWS SDK for Java デベロッパーガイド』の「デフォルトの認証情報およびリージョンの設定」を参照してください。

このコード例は、 AWS SDK for Java バージョン 2.3.1 と OpenJDK バージョン 11.0.1 を使用してテストされました。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.AddressConfiguration;
import software.amazon.awssdk.services.pinpoint.model.ChannelType;
import software.amazon.awssdk.services.pinpoint.model.SimpleEmailPart;
import software.amazon.awssdk.services.pinpoint.model.SimpleEmail;
import software.amazon.awssdk.services.pinpoint.model.EmailMessage;
import software.amazon.awssdk.services.pinpoint.model.DirectMessageConfiguration;
import software.amazon.awssdk.services.pinpoint.model.MessageRequest;
import software.amazon.awssdk.services.pinpoint.model.SendMessagesRequest;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import software.amazon.awssdk.services.pinpointemail.PinpointEmailClient;
```

```
import software.amazon.awssdk.services.pinpointemail.model.Body;
import software.amazon.awssdk.services.pinpointemail.model.Content;
import software.amazon.awssdk.services.pinpointemail.model.Destination;
import software.amazon.awssdk.services.pinpointemail.model.EmailContent;
import software.amazon.awssdk.services.pinpointemail.model.Message;
import software.amazon.awssdk.services.pinpointemail.model.SendEmailRequest;

import java.util.HashMap;
import java.util.Map;
```

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.AddressConfiguration;
import software.amazon.awssdk.services.pinpoint.model.ChannelType;
import software.amazon.awssdk.services.pinpoint.model.SimpleEmailPart;
import software.amazon.awssdk.services.pinpoint.model.SimpleEmail;
import software.amazon.awssdk.services.pinpoint.model.EmailMessage;
import software.amazon.awssdk.services.pinpoint.model.DirectMessageConfiguration;
import software.amazon.awssdk.services.pinpoint.model.MessageRequest;
import software.amazon.awssdk.services.pinpoint.model.SendMessagesRequest;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import software.amazon.awssdk.services.pinpointemail.PinpointEmailClient;
import software.amazon.awssdk.services.pinpointemail.model.Body;
import software.amazon.awssdk.services.pinpointemail.model.Content;
import software.amazon.awssdk.services.pinpointemail.model.Destination;
import software.amazon.awssdk.services.pinpointemail.model.EmailContent;
import software.amazon.awssdk.services.pinpointemail.model.Message;
import software.amazon.awssdk.services.pinpointemail.model.SendEmailRequest;

import java.util.HashMap;
import java.util.Map;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class SendEmailMessage {

        // The character encoding the you want to use for the subject line and
```

```
        // message body of the email.
        public static String charset = "UTF-8";

    // The body of the email for recipients whose email clients support HTML
content.
    static final String body = """
        Amazon Pinpoint test (AWS SDK for Java 2.x)

        This email was sent through the Amazon Pinpoint Email API using the AWS SDK
for Java 2.x

        """;

        public static void main(String[] args) {
                final String usage = """

                                Usage:    <subject> <appId> <senderAddress>
<toAddress>

            Where:
               subject - The email subject to use.
               senderAddress - The from address. This address has to be verified in
Amazon Pinpoint in the region you're using to send email\s
               toAddress - The to address. This address has to be verified in Amazon
Pinpoint in the region you're using to send email\s
            """;

        if (args.length != 3) {
            System.out.println(usage);
            System.exit(1);
        }

        String subject = args[0];
        String senderAddress = args[1];
        String toAddress = args[2];
        System.out.println("Sending a message");
        PinpointEmailClient pinpoint = PinpointEmailClient.builder()
            .region(Region.US_EAST_1)
            .build();

        sendEmail(pinpoint, subject, senderAddress, toAddress);
        System.out.println("Email was sent");
        pinpoint.close();
    }
```

```
    public static void sendEmail(PinpointEmailClient pinpointEmailClient, String
 subject, String senderAddress, String toAddress) {
        try {
            Content content = Content.builder()
                .data(body)
                .build();

            Body messageBody = Body.builder()
                .text(content)
                .build();

            Message message = Message.builder()
                .body(messageBody)
                .subject(Content.builder().data(subject).build())
                .build();

            Destination destination = Destination.builder()
                .toAddresses(toAddress)
                .build();

            EmailContent emailContent = EmailContent.builder()
                .simple(message)
                .build();

            SendEmailRequest sendEmailRequest = SendEmailRequest.builder()
                .fromEmailAddress(senderAddress)
                .destination(destination)
                .content(emailContent)
                .build();

            pinpointEmailClient.sendEmail(sendEmailRequest);
            System.out.println("Message Sent");

        } catch (PinpointException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

SDK の完全な例については、「」のSendEmailMessage「.java」を参照してくださいGitHub。

JavaScript (Node.js)

この例を使用して、NodeAWS .js JavaScript の SDK for を使用して E メールを送信します。この例では、Node.js JavaScript で の SDK を既にインストールして設定していることを前提としています。詳細については、「 SDK for Node.js https://docs.aws.amazon.com/AWSJavaScriptSDK/guide/node-intro.htmlデベロッパーガイド」の「開始方法」を参照してください。 AWS JavaScript

この例では、共有認証情報ファイルを使用して既存のユーザーのアクセスキーとシークレットアクセスキーを指定することを前提としています。詳細については、「 SDK for Node.js https://docs.aws.amazon.com/sdk-for-javascript/latest/developer-guide/setting-credentials.htmlデベロッパーガイド」の「認証情報の設定」を参照してください。 AWS JavaScript

このコード例は、Node.js JavaScript バージョン 2.388.0 および Node.js バージョン 11.7.0 の SDK for を使用してテストされました。

```
"use strict";

const AWS = require("aws-sdk");

// The AWS Region that you want to use to send the email. For a list of
// AWS Regions where the Amazon Pinpoint API is available, see
// https://docs.aws.amazon.com/pinpoint/latest/apireference/
const aws_region = "us-west-2";

// The "From" address. This address has to be verified in Amazon Pinpoint
// in the region that you use to send email.
const senderAddress = "sender@example.com";

// The address on the "To" line. If your Amazon Pinpoint account is in
// the sandbox, this address also has to be verified.
var toAddress = "recipient@example.com";

// The Amazon Pinpoint project/application ID to use when you send this message.
// Make sure that the SMS channel is enabled for the project or application
// that you choose.
const appId = "ce796be37f32f178af652b26eexample";

// The subject line of the email.
var subject = "Amazon Pinpoint (AWS SDK for JavaScript in Node.js)";
```

```
// The email body for recipients with non-HTML email clients.
var body_text = `Amazon Pinpoint Test (SDK for JavaScript in Node.js)
-----------------------------------------------------
This email was sent with Amazon Pinpoint using the AWS SDK for JavaScript in
 Node.js.
For more information, see https:\/\/aws.amazon.com/sdk-for-node-js/`;

// The body of the email for recipients whose email clients support HTML content.
var body_html = `<html>
<head></head>
<body>
  <h1>Amazon Pinpoint Test (SDK for JavaScript in Node.js)</h1>
  <p>This email was sent with
    <a href='https://aws.amazon.com/pinpoint/'>the Amazon Pinpoint API</a> using the
    <a href='https://aws.amazon.com/sdk-for-node-js/'>
      AWS SDK for JavaScript in Node.js</a>.</p>
</body>
</html>`;

// The character encoding the you want to use for the subject line and
// message body of the email.
var charset = "UTF-8";

// Specify that you're using a shared credentials file.
var credentials = new AWS.SharedIniFileCredentials({ profile: "default" });
AWS.config.credentials = credentials;

// Specify the region.
AWS.config.update({ region: aws_region });

//Create a new Pinpoint object.
var pinpoint = new AWS.Pinpoint();

// Specify the parameters to pass to the API.
var params = {
  ApplicationId: appId,
  MessageRequest: {
    Addresses: {
      [toAddress]: {
        ChannelType: "EMAIL",
      },
    },
    MessageConfiguration: {
      EmailMessage: {
```

```
        FromAddress: senderAddress,
        SimpleEmail: {
          Subject: {
            Charset: charset,
            Data: subject,
          },
          HtmlPart: {
            Charset: charset,
            Data: body_html,
          },
          TextPart: {
            Charset: charset,
            Data: body_text,
          },
        },
      },
    },
  },
};

//Try to send the email.
pinpoint.sendMessages(params, function (err, data) {
  // If something goes wrong, print an error message.
  if (err) {
    console.log(err.message);
  } else {
    console.log(
      "Email sent! Message ID: ",
      data["MessageResponse"]["Result"][toAddress]["MessageId"]
    );
  }
});
```

Python

この例を使用して、AWS SDK for Python (Boto3) を使用して E メールを送信します。この例は、SDK for Python (Boto3) がすでにインストールされ、設定されていることを前提としています。詳細については、「AWS SDK for Python (Boto3) API リファレンス」の「クイックスタート」を参照してください。

```
import logging
```

```
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)


def send_email_message(
    pinpoint_client,
    app_id,
    sender,
    to_addresses,
    char_set,
    subject,
    html_message,
    text_message,
):
    """
    Sends an email message with HTML and plain text versions.

    :param pinpoint_client: A Boto3 Pinpoint client.
    :param app_id: The Amazon Pinpoint project ID to use when you send this message.
    :param sender: The "From" address. This address must be verified in
                   Amazon Pinpoint in the AWS Region you're using to send email.
    :param to_addresses: The addresses on the "To" line. If your Amazon Pinpoint
 account
                         is in the sandbox, these addresses must be verified.
    :param char_set: The character encoding to use for the subject line and message
                     body of the email.
    :param subject: The subject line of the email.
    :param html_message: The body of the email for recipients whose email clients
 can
                         display HTML content.
    :param text_message: The body of the email for recipients whose email clients
                         don't support HTML content.
    :return: A dict of to_addresses and their message IDs.
    """
    try:
        response = pinpoint_client.send_messages(
            ApplicationId=app_id,
            MessageRequest={
                "Addresses": {
                    to_address: {"ChannelType": "EMAIL"} for to_address in
 to_addresses
                },
```

```
                "MessageConfiguration": {
                    "EmailMessage": {
                        "FromAddress": sender,
                        "SimpleEmail": {
                            "Subject": {"Charset": char_set, "Data": subject},
                            "HtmlPart": {"Charset": char_set, "Data": html_message},
                            "TextPart": {"Charset": char_set, "Data": text_message},
                        },
                    }
                },
            },
        )
    except ClientError:
        logger.exception("Couldn't send email.")
        raise
    else:
        return {
            to_address: message["MessageId"]
            for to_address, message in response["MessageResponse"]["Result"].items()
        }


def main():
    app_id = "ce796be37f32f178af652b26eexample"
    sender = "sender@example.com"
    to_address = "recipient@example.com"
    char_set = "UTF-8"
    subject = "Amazon Pinpoint Test (SDK for Python (Boto3))"
    text_message = """Amazon Pinpoint Test (SDK for Python)
    -----------------------------------
    This email was sent with Amazon Pinpoint using the AWS SDK for Python (Boto3).
    For more information, see https://aws.amazon.com/sdk-for-python/
                """
    html_message = """<html>
    <head></head>
    <body>
      <h1>Amazon Pinpoint Test (SDK for Python (Boto3)</h1>
      <p>This email was sent with
        <a href='https://aws.amazon.com/pinpoint/'>Amazon Pinpoint</a> using the
        <a href='https://aws.amazon.com/sdk-for-python/'>
          AWS SDK for Python (Boto3)</a>.</p>
    </body>
    </html>
                """
```

```
    print("Sending email.")
    message_ids = send_email_message(
        boto3.client("pinpoint"),
        app_id,
        sender,
        [to_address],
        char_set,
        subject,
        html_message,
        text_message,
    )
    print(f"Message sent! Message IDs: {message_ids}")


if __name__ == "__main__":
    main()
```

次の例のように、メッセージテンプレートを使用して E メールメッセージを送信することもできます。

```
import logging
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)


def send_templated_email_message(
    pinpoint_client, project_id, sender, to_addresses, template_name,
 template_version
):
    """
    Sends an email message with HTML and plain text versions.

    :param pinpoint_client: A Boto3 Pinpoint client.
    :param project_id: The Amazon Pinpoint project ID to use when you send this
 message.
    :param sender: The "From" address. This address must be verified in
                   Amazon Pinpoint in the AWS Region you're using to send email.
    :param to_addresses: The addresses on the "To" line. If your Amazon Pinpoint
                         account is in the sandbox, these addresses must be
 verified.
```

```
    :param template_name: The name of the email template to use when sending the
 message.
    :param template_version: The version number of the message template.

    :return: A dict of to_addresses and their message IDs.
    """
    try:
        response = pinpoint_client.send_messages(
            ApplicationId=project_id,
            MessageRequest={
                "Addresses": {
                    to_address: {"ChannelType": "EMAIL"} for to_address in
 to_addresses
                },
                "MessageConfiguration": {"EmailMessage": {"FromAddress": sender}},
                "TemplateConfiguration": {
                    "EmailTemplate": {
                        "Name": template_name,
                        "Version": template_version,
                    }
                },
            },
        )
    except ClientError:
        logger.exception("Couldn't send email.")
        raise
    else:
        return {
            to_address: message["MessageId"]
            for to_address, message in response["MessageResponse"]["Result"].items()
        }


def main():
    project_id = "296b04b342374fceb661bf494example"
    sender = "sender@example.com"
    to_addresses = ["recipient@example.com"]
    template_name = "My_Email_Template"
    template_version = "1"

    print("Sending email.")
    message_ids = send_templated_email_message(
        boto3.client("pinpoint"),
        project_id,
```

```
        sender,
        to_addresses,
        template_name,
        template_version,
    )
    print(f"Message sent! Message IDs: {message_ids}")


if __name__ == "__main__":
    main()
```

これらの例では、共有認証情報ファイルを使用して既存のユーザーのアクセスキーとシークレットアクセスキーを指定することを前提としています。詳細については、『AWS SDK for Python (Boto3) API Reference』の「認証情報」を参照してください。

サブスクリプション解除ヘッダーを含む E メールの送信

> **Note**
>
> E メールヘッダーを使用する前に、キャンペーンまたはジャーニーから E メールを送信する場合は、E メールオーケストレーション送信ロールを設定する必要があります。直接送信 E メールの場合は、 `ses:SendEmail`および のアクセス許可が必要です`ses:SendRawEmail`。詳細については、Amazon Pinpoint ユーザーガイド」の「E メールオーケストレーション送信ロールの作成」を参照してください。

E メールに登録解除リンクを含めることはベストプラクティスであり、一部の国ではそれが法的要件となっています。ワンクリックのサブスクリプション解除リンクを追加するには、次のヘッダーを追加します。

1. ヘッダー名を に設定`List-Unsubscribe`し、値をサブスクリプション解除リンクに設定します。リンクは、受信者のサブスクリプション解除リクエストを処理するための HTTP POST リクエストをサポートしている必要があります。
2. ヘッダー名を に設定`List-Unsubscribe-Post`し、値を に設定します`List-Unsubscribe=One-Click`。

E メールメッセージには最大 15 個のヘッダーを追加できます。サポートされているヘッダーのリストについては、Amazon SES ヘッダーフィールド」を参照してください。 https://docs.aws.amazon.com/ses/latest/DeveloperGuide/

次の例は、 を使用してサブスクリプション解除ヘッダーを含む E メールメッセージを送信する方法を示しています AWS Command Line Interface。の設定の詳細については AWS CLI、「 ユーザーガイド」の「 の設定 AWS CLIAWS Command Line Interface」を参照してください。

次のコマンドで、次の操作を行います。

- をアプリケーション ID *AppId*に置き換えます。
- *richard_roe@example.com* を受信者の E メールアドレスに置き換えます。
- *https://example.com/unsub* をサブスクリプション解除リンクに置き換えます。
- *example123456* を受信者の一意の識別子に置き換えます。

```
aws pinpoint send-messages --application-id AppId --message-request '{
 "Addresses": {
   "richard_roe@example.com": {
     "ChannelType": "EMAIL"
   }
 },
 "MessageConfiguration": {
   "EmailMessage": {
     "Substitutions": {
       "url": [
         "https://example.com/unsub"
       ],
        "id1": [
          "/example123456"
        ]
     },
     "SimpleEmail": {
       "TextPart": {
         "Data": "Sample email message with an subscribe header",
         "Charset": "UTF-8"
       },
       "Subject": {
         "Data": "Hello",
         "Charset": "UTF-8"
       },
       "Headers": [
```

```
          {
            "Name": "List-Unsubscribe",
            "Value": "{{url}}{{id1}}"
          },
          {
            "Name": "List-Unsubscribe-Post",
            "Value": "List-Unsubscribe=One-Click"
          }
        ]
      }
    }
  }
}'
```

SMS メッセージの送信

Amazon Pinpoint API を使用して、特定の電話番号やエンドポイント ID に SMS メッセージ (テキストメッセージ) を送信できます。このセクションでは、 AWS SDK を使用して Amazon Pinpoint API 経由で SMS メッセージを送信するために使用できる完全なコード例を示します。

C#

この例を使用して、AWS SDK for .NET を使用して SMS メッセージを送信します。この例は、AWS SDK for .NETがすでにインストールされ、設定されていることを前提としています。詳細については、『AWS SDK for .NET デベロッパーガイド』の「使用開始」を参照してください。

この例では、共有認証情報ファイルを使用して、既存の IAM ユーザーのアクセスキーとシークレットアクセスキーを指定することを前提としています。詳細については、『AWS SDK for .NET デベロッパーガイド』の「AWS 認証情報の設定」を参照してください。

```
using Amazon;
using Amazon.Pinpoint;
using Amazon.Pinpoint.Model;
using Microsoft.Extensions.Configuration;

namespace SendSmsMessage;

public class SendSmsMessageMainClass
{
    public static async Task Main(string[] args)
    {
```

```
        var configuration = new ConfigurationBuilder()
            .SetBasePath(Directory.GetCurrentDirectory())
            .AddJsonFile("settings.json") // Load test settings from .json file.
            .AddJsonFile("settings.local.json",
                true) // Optionally load local settings.
            .Build();

        // The AWS Region that you want to use to send the message. For a list of
        // AWS Regions where the Amazon Pinpoint API is available, see
        // https://docs.aws.amazon.com/pinpoint/latest/apireference/
        string region = "us-east-1";

        // The phone number or short code to send the message from. The phone number
        // or short code that you specify has to be associated with your Amazon
 Pinpoint
        // account. For best results, specify long codes in E.164 format.
        string originationNumber = configuration["OriginationNumber"]!;

        // The recipient's phone number.  For best results, you should specify the
        // phone number in E.164 format.
        string destinationNumber = configuration["DestinationNumber"]!;

        // The Pinpoint project/ application ID to use when you send this message.
        // Make sure that the SMS channel is enabled for the project or application
        // that you choose.
        string appId = configuration["AppId"]!;

        // The type of SMS message that you want to send. If you plan to send
        // time-sensitive content, specify TRANSACTIONAL. If you plan to send
        // marketing-related content, specify PROMOTIONAL.
        MessageType messageType = MessageType.TRANSACTIONAL;

        // The registered keyword associated with the originating short code.
        string? registeredKeyword = configuration["RegisteredKeyword"];

        // The sender ID to use when sending the message. Support for sender ID
        // varies by country or region. For more information, see
        // https://docs.aws.amazon.com/pinpoint/latest/userguide/channels-sms-
countries.html
        string? senderId = configuration["SenderId"];

        try
        {
            var response = await SendSmsMessage(region, appId, destinationNumber,
```

```
                originationNumber, registeredKeyword, senderId, messageType);
            Console.WriteLine($"Message sent to
{response.MessageResponse.Result.Count} recipient(s).");
            foreach (var messageResultValue in
                     response.MessageResponse.Result.Select(r => r.Value))
            {
                Console.WriteLine($"{messageResultValue.MessageId} Status:
{messageResultValue.DeliveryStatus}");
            }
        }
        catch (Exception ex)
        {
            Console.WriteLine("The message wasn't sent. Error message: " +
ex.Message);
        }
    }

    public static async Task<SendMessagesResponse> SendSmsMessage(
        string region, string appId, string destinationNumber, string
originationNumber,
        string? keyword, string? senderId, MessageType messageType)
    {

        // The content of the SMS message.
        string message = "This message was sent through Amazon Pinpoint using" +
                         " the AWS SDK for .NET. Reply STOP to opt out.";


        var client = new
AmazonPinpointClient(RegionEndpoint.GetBySystemName(region));

        SendMessagesRequest sendRequest = new SendMessagesRequest
        {
            ApplicationId = appId,
            MessageRequest = new MessageRequest
            {
                Addresses =
                    new Dictionary<string, AddressConfiguration>
                    {
                        {
                            destinationNumber,
                            new AddressConfiguration { ChannelType =
ChannelType.SMS }
                        }
```

```
                    },
                MessageConfiguration = new DirectMessageConfiguration
                {
                    SMSMessage = new SMSMessage
                    {
                        Body = message,
                        MessageType = MessageType.TRANSACTIONAL,
                        OriginationNumber = originationNumber,
                        SenderId = senderId,
                        Keyword = keyword
                    }
                }
            }
        };
        SendMessagesResponse response = await client.SendMessagesAsync(sendRequest);
        return response;
    }
}
```

Java

この例を使用して、AWS SDK for Java を使用して SMS メッセージを送信します。この例は、SDK for Java がすでにインストールされ、設定されていることを前提としています。詳細については、『AWS SDK for Java デベロッパーガイド』の「使用開始」を参照してください。

この例では、共有認証情報ファイルを使用して、既存の IAM ユーザーのアクセスキーとシークレットアクセスキーを指定することを前提としています。詳細については、『AWS SDK for Java デベロッパーガイド』の「デフォルトの認証情報およびリージョンの設定」を参照してください。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.DirectMessageConfiguration;
import software.amazon.awssdk.services.pinpoint.model.SMSMessage;
import software.amazon.awssdk.services.pinpoint.model.AddressConfiguration;
import software.amazon.awssdk.services.pinpoint.model.ChannelType;
import software.amazon.awssdk.services.pinpoint.model.MessageRequest;
import software.amazon.awssdk.services.pinpoint.model.SendMessagesRequest;
import software.amazon.awssdk.services.pinpoint.model.SendMessagesResponse;
import software.amazon.awssdk.services.pinpoint.model.MessageResponse;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import java.util.HashMap;
```

```
import java.util.Map;
```

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.DirectMessageConfiguration;
import software.amazon.awssdk.services.pinpoint.model.SMSMessage;
import software.amazon.awssdk.services.pinpoint.model.AddressConfiguration;
import software.amazon.awssdk.services.pinpoint.model.ChannelType;
import software.amazon.awssdk.services.pinpoint.model.MessageRequest;
import software.amazon.awssdk.services.pinpoint.model.SendMessagesRequest;
import software.amazon.awssdk.services.pinpoint.model.SendMessagesResponse;
import software.amazon.awssdk.services.pinpoint.model.MessageResponse;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import java.util.HashMap;
import java.util.Map;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class SendMessage {

        // The type of SMS message that you want to send. If you plan to send
        // time-sensitive content, specify TRANSACTIONAL. If you plan to send
        // marketing-related content, specify PROMOTIONAL.
        public static String messageType = "TRANSACTIONAL";

        // The registered keyword associated with the originating short code.
        public static String registeredKeyword = "myKeyword";

        // The sender ID to use when sending the message. Support for sender ID
        // varies by country or region. For more information, see
        // https://docs.aws.amazon.com/pinpoint/latest/userguide/channels-sms-
countries.html
        public static String senderId = "MySenderID";

        public static void main(String[] args) {
                final String usage = """
```

```
                                Usage:   <message> <appId> <originationNumber>
<destinationNumber>\s

                                Where:
                                  message - The body of the message to send.
                                  appId - The Amazon Pinpoint project/application ID
to use when you send this message.
                                  originationNumber - The phone number or short code
that you specify has to be associated with your Amazon Pinpoint account. For best
results, specify long codes in E.164 format (for example, +1-555-555-5654).
                                  destinationNumber - The recipient's phone number.
For best results, you should specify the phone number in E.164 format (for example,
+1-555-555-5654).\s
                                  """;

              if (args.length != 4) {
                      System.out.println(usage);
                      System.exit(1);
              }

              String message = args[0];
              String appId = args[1];
              String originationNumber = args[2];
              String destinationNumber = args[3];
              System.out.println("Sending a message");
              PinpointClient pinpoint = PinpointClient.builder()
                              .region(Region.US_EAST_1)
                              .build();

              sendSMSMessage(pinpoint, message, appId, originationNumber,
destinationNumber);
              pinpoint.close();
       }

       public static void sendSMSMessage(PinpointClient pinpoint, String message,
String appId,
                      String originationNumber,
                      String destinationNumber) {
              try {
                      Map<String, AddressConfiguration> addressMap = new
HashMap<String, AddressConfiguration>();
                      AddressConfiguration addConfig =
AddressConfiguration.builder()
                                      .channelType(ChannelType.SMS)
```

```
                                        .build();

                        addressMap.put(destinationNumber, addConfig);
                        SMSMessage smsMessage = SMSMessage.builder()
                                        .body(message)
                                        .messageType(messageType)
                                        .originationNumber(originationNumber)
                                        .senderId(senderId)
                                        .keyword(registeredKeyword)
                                        .build();

                        // Create a DirectMessageConfiguration object.
                        DirectMessageConfiguration direct =
 DirectMessageConfiguration.builder()
                                        .smsMessage(smsMessage)
                                        .build();

                        MessageRequest msgReq = MessageRequest.builder()
                                        .addresses(addressMap)
                                        .messageConfiguration(direct)
                                        .build();

                        // create a SendMessagesRequest object
                        SendMessagesRequest request = SendMessagesRequest.builder()
                                        .applicationId(appId)
                                        .messageRequest(msgReq)
                                        .build();

                        SendMessagesResponse response =
 pinpoint.sendMessages(request);
                        MessageResponse msg1 = response.messageResponse();
                        Map map1 = msg1.result();

                        // Write out the result of sendMessage.
                        map1.forEach((k, v) -> System.out.println((k + ":" + v)));

                } catch (PinpointException e) {
                        System.err.println(e.awsErrorDetails().errorMessage());
                        System.exit(1);
                }
        }
}
```

SDK の完全な例については、「」のSendMessage「.java」を参照してくださいGitHub。

JavaScript (Node.js)

この例では、 AWS SDK for in JavaScript Node.js を使用して SMS メッセージを送信します。この例では、Node.js JavaScript で の SDK を既にインストールして設定していることを前提としています。詳細については、Node.js https://docs.aws.amazon.com/AWSJavaScriptSDK/guide/node-intro.htmlデベロッパーガイドの SDK for の開始方法を参照してください。 AWS JavaScript

この例では、共有認証情報ファイルを使用して、既存の IAM ユーザーのアクセスキーとシークレットアクセスキーを指定することを前提としています。詳細については、「 SDK for Node.js https://docs.aws.amazon.com/sdk-for-javascript/latest/developer-guide/setting-credentials.htmlデベロッパーガイド」の「認証情報の設定」を参照してください。 AWS JavaScript

```
"use strict";

var AWS = require("aws-sdk");

// The AWS Region that you want to use to send the message. For a list of
// AWS Regions where the Amazon Pinpoint API is available, see
// https://docs.aws.amazon.com/pinpoint/latest/apireference/.
var aws_region = "us-east-1";

// The phone number or short code to send the message from. The phone number
// or short code that you specify has to be associated with your Amazon Pinpoint
// account. For best results, specify long codes in E.164 format.
var originationNumber = "+12065550199";

// The recipient's phone number.  For best results, you should specify the
// phone number in E.164 format.
var destinationNumber = "+14255550142";

// The content of the SMS message.
var message =
  "This message was sent through Amazon Pinpoint " +
  "using the AWS SDK for JavaScript in Node.js. Reply STOP to " +
  "opt out.";

// The Amazon Pinpoint project/application ID to use when you send this message.
// Make sure that the SMS channel is enabled for the project or application
// that you choose.
var applicationId = "ce796be37f32f178af652b26eexample";
```

```
// The type of SMS message that you want to send. If you plan to send
// time-sensitive content, specify TRANSACTIONAL. If you plan to send
// marketing-related content, specify PROMOTIONAL.
var messageType = "TRANSACTIONAL";

// The registered keyword associated with the originating short code.
var registeredKeyword = "myKeyword";

// The sender ID to use when sending the message. Support for sender ID
// varies by country or region. For more information, see
// https://docs.aws.amazon.com/pinpoint/latest/userguide/channels-sms-countries.html
var senderId = "MySenderID";

// Specify that you're using a shared credentials file, and optionally specify
// the profile that you want to use.
var credentials = new AWS.SharedIniFileCredentials({ profile: "default" });
AWS.config.credentials = credentials;

// Specify the region.
AWS.config.update({ region: aws_region });

//Create a new Pinpoint object.
var pinpoint = new AWS.Pinpoint();

// Specify the parameters to pass to the API.
var params = {
  ApplicationId: applicationId,
  MessageRequest: {
    Addresses: {
      [destinationNumber]: {
        ChannelType: "SMS",
      },
    },
    MessageConfiguration: {
      SMSMessage: {
        Body: message,
        Keyword: registeredKeyword,
        MessageType: messageType,
        OriginationNumber: originationNumber,
        SenderId: senderId,
      },
    },
  },
```

```
};

//Try to send the message.
pinpoint.sendMessages(params, function (err, data) {
  // If something goes wrong, print an error message.
  if (err) {
    console.log(err.message);
    // Otherwise, show the unique ID for the message.
  } else {
    console.log(
      "Message sent! " +
        data["MessageResponse"]["Result"][destinationNumber]["StatusMessage"]
    );
  }
});
```

Python

この例を使用して、AWS SDK for Python (Boto3) を使用して SMS メッセージを送信します。この例は、SDK for Python がすでにインストールされ、設定されていることを前提としています。詳細については、 SDK for Python (Boto3) 入門ガイドの「クイックスタート」を参照してください。 AWS

```
import logging
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)


def send_sms_message(
    pinpoint_client,
    app_id,
    origination_number,
    destination_number,
    message,
    message_type,
):
    """
    Sends an SMS message with Amazon Pinpoint.
```

```
    :param pinpoint_client: A Boto3 Pinpoint client.
    :param app_id: The Amazon Pinpoint project/application ID to use when you send
                   this message. The SMS channel must be enabled for the project or
                   application.
    :param destination_number: The recipient's phone number in E.164 format.
    :param origination_number: The phone number to send the message from. This phone
                               number must be associated with your Amazon Pinpoint
                               account and be in E.164 format.
    :param message: The content of the SMS message.
    :param message_type: The type of SMS message that you want to send. If you send
                         time-sensitive content, specify TRANSACTIONAL. If you send
                         marketing-related content, specify PROMOTIONAL.
    :return: The ID of the message.
    """
    try:
        response = pinpoint_client.send_messages(
            ApplicationId=app_id,
            MessageRequest={
                "Addresses": {destination_number: {"ChannelType": "SMS"}},
                "MessageConfiguration": {
                    "SMSMessage": {
                        "Body": message,
                        "MessageType": message_type,
                        "OriginationNumber": origination_number,
                    }
                },
            },
        )
    except ClientError:
        logger.exception("Couldn't send message.")
        raise
    else:
        return response["MessageResponse"]["Result"][destination_number]
["MessageId"]


def main():
    app_id = "ce796be37f32f178af652b26eexample"
    origination_number = "+12065550199"
    destination_number = "+14255550142"
    message = (
        "This is a sample message sent from Amazon Pinpoint by using the AWS SDK for
 "
        "Python (Boto 3)."
```

```
    )
    message_type = "TRANSACTIONAL"

    print("Sending SMS message.")
    message_id = send_sms_message(
        boto3.client("pinpoint"),
        app_id,
        origination_number,
        destination_number,
        message,
        message_type,
    )
    print(f"Message sent! Message ID: {message_id}.")


if __name__ == "__main__":
    main()
```

次の例のように、メッセージテンプレートを使用して SMS メッセージを送信することもできます。

```
import logging
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)


def send_templated_sms_message(
    pinpoint_client,
    project_id,
    destination_number,
    message_type,
    origination_number,
    template_name,
    template_version,
):
    """
    Sends an SMS message to a specific phone number using a pre-defined template.

    :param pinpoint_client: A Boto3 Pinpoint client.
    :param project_id: An Amazon Pinpoint project (application) ID.
    :param destination_number: The phone number to send the message to.
```

```
    :param message_type: The type of SMS message (promotional or transactional).
    :param origination_number: The phone number that the message is sent from.
    :param template_name: The name of the SMS template to use when sending the
 message.
    :param template_version: The version number of the message template.

    :return The ID of the message.
    """
    try:
        response = pinpoint_client.send_messages(
            ApplicationId=project_id,
            MessageRequest={
                "Addresses": {destination_number: {"ChannelType": "SMS"}},
                "MessageConfiguration": {
                    "SMSMessage": {
                        "MessageType": message_type,
                        "OriginationNumber": origination_number,
                    }
                },
                "TemplateConfiguration": {
                    "SMSTemplate": {"Name": template_name, "Version":
 template_version}
                },
            },
        )

    except ClientError:
        logger.exception("Couldn't send message.")
        raise
    else:
        return response["MessageResponse"]["Result"][destination_number]
["MessageId"]


def main():
    region = "us-east-1"
    origination_number = "+18555550001"
    destination_number = "+14255550142"
    project_id = "7353f53e6885409fa32d07cedexample"
    message_type = "TRANSACTIONAL"
    template_name = "My_SMS_Template"
    template_version = "1"
    message_id = send_templated_sms_message(
        boto3.client("pinpoint", region_name=region),
```

```
        project_id,
        destination_number,
        message_type,
        origination_number,
        template_name,
        template_version,
    )
    print(f"Message sent! Message ID: {message_id}.")


if __name__ == "__main__":
    main()
```

これらの例では、共有認証情報ファイルを使用して既存の IAM ユーザーのアクセスキーとシークレットアクセスキーを指定することを前提としています。詳細については、『AWS SDK for Python (Boto3) API Reference』の「認証情報」を参照してください。

音声メッセージの送信

Amazon Pinpoint API を使用して、特定の電話番号に音声メッセージを送信できます。このセクションには、AWS SDK を使用して Amazon Pinpoint SMS および 音声 API を介して音声メッセージを送信するのに使用できる、完全なコードの例を含みます。

Java

この例を使用して、AWS SDK for Java を使用して音声メッセージを送信します。この例は、SDK for Java がすでにインストールされ、設定されていることを前提としています。詳細については、『AWS SDK for Java デベロッパーガイド』の「使用開始」を参照してください。

この例では、共有認証情報ファイルを使用して既存のユーザーのアクセスキーとシークレットアクセスキーを指定することを前提としています。詳細については、『AWS SDK for Java デベロッパーガイド』の「開発用の AWS 認証情報およびリージョンのセットアップ」を参照してください。

```
import software.amazon.awssdk.core.client.config.ClientOverrideConfiguration;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpointsmsvoice.PinpointSmsVoiceClient;
import software.amazon.awssdk.services.pinpointsmsvoice.model.SSMLMessageType;
import software.amazon.awssdk.services.pinpointsmsvoice.model.VoiceMessageContent;
import
 software.amazon.awssdk.services.pinpointsmsvoice.model.SendVoiceMessageRequest;
```

```
import
 software.amazon.awssdk.services.pinpointsmsvoice.model.PinpointSmsVoiceException;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
```

```
import software.amazon.awssdk.core.client.config.ClientOverrideConfiguration;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpointsmsvoice.PinpointSmsVoiceClient;
import software.amazon.awssdk.services.pinpointsmsvoice.model.SSMLMessageType;
import software.amazon.awssdk.services.pinpointsmsvoice.model.VoiceMessageContent;
import
 software.amazon.awssdk.services.pinpointsmsvoice.model.SendVoiceMessageRequest;
import
 software.amazon.awssdk.services.pinpointsmsvoice.model.PinpointSmsVoiceException;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class SendVoiceMessage {

        // The Amazon Polly voice that you want to use to send the message. For a
 list
        // of voices, see https://docs.aws.amazon.com/polly/latest/dg/voicelist.html
        static final String voiceName = "Matthew";

        // The language to use when sending the message. For a list of supported
        // languages, see
        // https://docs.aws.amazon.com/polly/latest/dg/SupportedLanguage.html
        static final String languageCode = "en-US";

        // The content of the message. This example uses SSML to customize and
 control
```

```
        // certain aspects of the message, such as by adding pauses and changing
        // phonation. The message can't contain any line breaks.
        static final String ssmlMessage = "<speak>This is a test message sent from "
                        + "<emphasis>Amazon Pinpoint</emphasis> "
                        + "using the <break strength='weak'/>AWS "
                        + "SDK for Java. "
                        + "<amazon:effect phonation='soft'>Thank "
                        + "you for listening.</amazon:effect></speak>";

        public static void main(String[] args) {

                final String usage = """

                                    Usage:   <originationNumber> <destinationNumber>\s

                                    Where:
                                      originationNumber - The phone number or short code
that you specify has to be associated with your Amazon Pinpoint account. For best
results, specify long codes in E.164 format (for example, +1-555-555-5654).
                                      destinationNumber - The recipient's phone number.
For best results, you should specify the phone number in E.164 format (for example,
+1-555-555-5654).\s
                                      """;

                if (args.length != 2) {
                        System.out.println(usage);
                        System.exit(1);
                }

                String originationNumber = args[0];
                String destinationNumber = args[1];
                System.out.println("Sending a voice message");

                // Set the content type to application/json.
                List<String> listVal = new ArrayList<>();
                listVal.add("application/json");
                Map<String, List<String>> values = new HashMap<>();
                values.put("Content-Type", listVal);

                ClientOverrideConfiguration config2 =
ClientOverrideConfiguration.builder()
                                    .headers(values)
                                    .build();
```

```
                PinpointSmsVoiceClient client = PinpointSmsVoiceClient.builder()
                                .overrideConfiguration(config2)
                                .region(Region.US_EAST_1)
                                .build();

                sendVoiceMsg(client, originationNumber, destinationNumber);
                client.close();
        }

        public static void sendVoiceMsg(PinpointSmsVoiceClient client, String
 originationNumber,
                        String destinationNumber) {
                try {
                        SSMLMessageType ssmlMessageType = SSMLMessageType.builder()
                                        .languageCode(languageCode)
                                        .text(ssmlMessage)
                                        .voiceId(voiceName)
                                        .build();

                        VoiceMessageContent content = VoiceMessageContent.builder()
                                        .ssmlMessage(ssmlMessageType)
                                        .build();

                        SendVoiceMessageRequest voiceMessageRequest =
 SendVoiceMessageRequest.builder()
                                        .destinationPhoneNumber(destinationNumber)
                                        .originationPhoneNumber(originationNumber)
                                        .content(content)
                                        .build();

                        client.sendVoiceMessage(voiceMessageRequest);
                        System.out.println("The message was sent successfully.");

                } catch (PinpointSmsVoiceException e) {
                        System.err.println(e.awsErrorDetails().errorMessage());
                        System.exit(1);
                }
        }
}
```

SDK のサンプルについては、「GitHub」の「SendVoiceMessage.java」を参照してください。

JavaScript (Node.js)

この例では、AWS SDK for JavaScript in Node.js を使用して音声メッセージを送信します。この例は、SDK for JavaScript in Node.js がすでにインストールされ、設定されていることを前提としています。

この例では、共有認証情報ファイルを使用して既存のユーザーのアクセスキーとシークレットアクセスキーを指定することを前提としています。詳細については、『AWS SDK for JavaScript in Node.js デベロッパーガイド』の「認証情報の設定」を参照してください。

```
"use strict";

var AWS = require("aws-sdk");

// The AWS Region that you want to use to send the voice message. For a list of
// AWS Regions where the Amazon Pinpoint SMS and Voice API is available, see
// https://docs.aws.amazon.com/pinpoint-sms-voice/latest/APIReference/
var aws_region = "us-east-1";

// The phone number that the message is sent from. The phone number that you
// specify has to be associated with your Amazon Pinpoint account. For best results,
 you
// should specify the phone number in E.164 format.
var originationNumber = "+12065550110";

// The recipient's phone number. For best results, you should specify the phone
// number in E.164 format.
var destinationNumber = "+12065550142";

// The language to use when sending the message. For a list of supported
// languages, see https://docs.aws.amazon.com/polly/latest/dg/SupportedLanguage.html
var languageCode = "en-US";

// The Amazon Polly voice that you want to use to send the message. For a list
// of voices, see https://docs.aws.amazon.com/polly/latest/dg/voicelist.html
var voiceId = "Matthew";

// The content of the message. This example uses SSML to customize and control
// certain aspects of the message, such as the volume or the speech rate.
// The message can't contain any line breaks.
var ssmlMessage =
  "<speak>" +
```

```
  "This is a test message sent from <emphasis>Amazon Pinpoint</emphasis> " +
  "using the <break strength='weak'/>AWS SDK for JavaScript in Node.js. " +
  "<amazon:effect phonation='soft'>Thank you for listening." +
  "</amazon:effect>" +
  "</speak>";

// The phone number that you want to appear on the recipient's device. The phone
// number that you specify has to be associated with your Amazon Pinpoint account.
var callerId = "+12065550199";

// The configuration set that you want to use to send the message.
var configurationSet = "ConfigSet";

// Specify that you're using a shared credentials file, and optionally specify
// the profile that you want to use.
var credentials = new AWS.SharedIniFileCredentials({ profile: "default" });
AWS.config.credentials = credentials;

// Specify the region.
AWS.config.update({ region: aws_region });

//Create a new Pinpoint object.
var pinpointsmsvoice = new AWS.PinpointSMSVoice();

var params = {
  CallerId: callerId,
  ConfigurationSetName: configurationSet,
  Content: {
    SSMLMessage: {
      LanguageCode: languageCode,
      Text: ssmlMessage,
      VoiceId: voiceId,
    },
  },
  DestinationPhoneNumber: destinationNumber,
  OriginationPhoneNumber: originationNumber,
};

//Try to send the message.
pinpointsmsvoice.sendVoiceMessage(params, function (err, data) {
  // If something goes wrong, print an error message.
  if (err) {
    console.log(err.message);
    // Otherwise, show the unique ID for the message.
```

```
  } else {
    console.log("Message sent! Message ID: " + data["MessageId"]);
  }
});
```

Python

この例を使用して、AWS SDK for Python (Boto3) を使用して音声メッセージを送信します。この例は、SDK for Python (Boto3) がすでにインストールされ、設定されていることを前提としています。

この例では、共有認証情報ファイルを使用して既存のユーザーのアクセスキーとシークレットアクセスキーを指定することを前提としています。詳細については、『AWS SDK for Python (Boto3) API Reference』の「認証情報」を参照してください。

```
import logging
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)


def send_voice_message(
    sms_voice_client,
    origination_number,
    caller_id,
    destination_number,
    language_code,
    voice_id,
    ssml_message,
):
    """
    Sends a voice message using speech synthesis provided by Amazon Polly.

    :param sms_voice_client: A Boto3 PinpointSMSVoice client.
    :param origination_number: The phone number that the message is sent from.
                               The phone number must be associated with your Amazon
                               Pinpoint account and be in E.164 format.
    :param caller_id: The phone number that you want to appear on the recipient's
                      device. The phone number must be associated with your Amazon
                      Pinpoint account and be in E.164 format.
```

```
    :param destination_number: The recipient's phone number. Specify the phone
                               number in E.164 format.
    :param language_code: The language to use when sending the message.
    :param voice_id: The Amazon Polly voice that you want to use to send the
 message.
    :param ssml_message: The content of the message. This example uses SSML to
 control
                         certain aspects of the message, such as the volume and the
                         speech rate. The message must not contain line breaks.
    :return: The ID of the message.
    """
    try:
        response = sms_voice_client.send_voice_message(
            DestinationPhoneNumber=destination_number,
            OriginationPhoneNumber=origination_number,
            CallerId=caller_id,
            Content={
                "SSMLMessage": {
                    "LanguageCode": language_code,
                    "VoiceId": voice_id,
                    "Text": ssml_message,
                }
            },
        )
    except ClientError:
        logger.exception(
            "Couldn't send message from %s to %s.",
            origination_number,
            destination_number,
        )
        raise
    else:
        return response["MessageId"]


def main():
    origination_number = "+12065550110"
    caller_id = "+12065550199"
    destination_number = "+12065550142"
    language_code = "en-US"
    voice_id = "Matthew"
    ssml_message = (
        "<speak>"
        "This is a test message sent from <emphasis>Amazon Pinpoint</emphasis> "
```

```
        "using the <break strength='weak'/>AWS SDK for Python (Boto3). "
        "<amazon:effect phonation='soft'>Thank you for listening."
        "</amazon:effect>"
        "</speak>"
    )
    print(f"Sending voice message from {origination_number} to
 {destination_number}.")
    message_id = send_voice_message(
        boto3.client("pinpoint-sms-voice"),
        origination_number,
        caller_id,
        destination_number,
        language_code,
        voice_id,
        ssml_message,
    )
    print(f"Message sent!\nMessage ID: {message_id}")


if __name__ == "__main__":
    main()
```

プッシュ通知の送信

Amazon Pinpoint API は、トランザクションプッシュ通知を特定のデバイス識別子に送信できます。このセクションには、AWS SDK を使用して Amazon Pinpoint API を介してプッシュ通知を送信するのに使用できる、完全なコードの例を含みます。

これらの例を使用して、Amazon Pinpoint がサポートするプッシュ通知サービスを通じてプッシュ通知を送信できます。現在、Amazon Pinpoint は Firebase Cloud Messaging (FCM)、Apple プッシュ通知サービス (APNS)、Baidu Cloud Push、Amazon Device Messaging (ADM) の各チャネルをサポートしています。

> **Note**
>
> Firebase Cloud Messaging (FCM) サービスを介してプッシュ通知を送信する場合は、Amazon Pinpoint API への呼び出しでサービス名 `GCM` を使用します。Google Cloud Messaging (GCM) サービスは、2018 年 4 月 10 日に Google によって停止されました。ただし、Amazon Pinpoint API は、GCM サービスの停止前に記述された API コードとの互換性を

維持するために、FCM サービスを介して送信するメッセージに GCM サービス名を使用します。

JavaScript (Node.js)

この例では、AWS SDK for JavaScript in Node.js を使用してプッシュ通知を送信しています。この例は、SDK for JavaScript in Node.js がすでにインストールされ、設定されていることを前提としています。

この例では、共有認証情報ファイルを使用して、既存の ユーザーのアクセスキーとシークレットアクセスキーを指定するものと想定しています。詳細については、『AWS SDK for JavaScript in Node.js デベロッパーガイド』の「認証情報の設定」を参照してください。

```
'use strict';

const AWS = require('aws-sdk');

// The AWS Region that you want to use to send the message. For a list of
// AWS Regions where the Amazon Pinpoint API is available, see
// https://docs.aws.amazon.com/pinpoint/latest/apireference/
const region = 'us-east-1';

// The title that appears at the top of the push notification.
var title = 'Test message sent from Amazon Pinpoint.';

// The content of the push notification.
var message = 'This is a sample message sent from Amazon Pinpoint by using the '
            + 'AWS SDK for JavaScript in Node.js';

// The Amazon Pinpoint project ID that you want to use when you send this
// message. Make sure that the push channel is enabled for the project that
// you choose.
var applicationId = 'ce796be37f32f178af652b26eexample';

// An object that contains the unique token of the device that you want to send
// the message to, and the push service that you want to use to send the message.
var recipient = {
  'token': 'a0b1c2d3e4f5g6h7i8j9k0l1m2n3o4p5q6r7s8t9u0v1w2x3y4z5a6b7c8d8e9f0',
  'service': 'GCM'
  };
```

```
// The action that should occur when the recipient taps the message. Possible
// values are OPEN_APP (opens the app or brings it to the foreground),
// DEEP_LINK (opens the app to a specific page or interface), or URL (opens a
// specific URL in the device's web browser.)
var action = 'URL';

// This value is only required if you use the URL action. This variable contains
// the URL that opens in the recipient's web browser.
var url = 'https://www.example.com';

// The priority of the push notification. If the value is 'normal', then the
// delivery of the message is optimized for battery usage on the recipient's
// device, and could be delayed. If the value is 'high', then the notification is
// sent immediately, and might wake a sleeping device.
var priority = 'normal';

// The amount of time, in seconds, that the push notification service provider
// (such as FCM or APNS) should attempt to deliver the message before dropping
// it. Not all providers allow you specify a TTL value.
var ttl = 30;

// Boolean that specifies whether the notification is sent as a silent
// notification (a notification that doesn't display on the recipient's device).
var silent = false;

function CreateMessageRequest() {
  var token = recipient['token'];
  var service = recipient['service'];
  if (service == 'GCM') {
    var messageRequest = {
      'Addresses': {
        [token]: {
          'ChannelType' : 'GCM'
        }
      },
      'MessageConfiguration': {
        'GCMMessage': {
          'Action': action,
          'Body': message,
          'Priority': priority,
          'SilentPush': silent,
          'Title': title,
          'TimeToLive': ttl,
          'Url': url
```

```
        }
      }
    };
  } else if (service == 'APNS') {
    var messageRequest = {
      'Addresses': {
        [token]: {
          'ChannelType' : 'APNS'
        }
      },
      'MessageConfiguration': {
        'APNSMessage': {
          'Action': action,
          'Body': message,
          'Priority': priority,
          'SilentPush': silent,
          'Title': title,
          'TimeToLive': ttl,
          'Url': url
        }
      }
    };
  } else if (service == 'BAIDU') {
    var messageRequest = {
      'Addresses': {
        [token]: {
          'ChannelType' : 'BAIDU'
        }
      },
      'MessageConfiguration': {
        'BaiduMessage': {
          'Action': action,
          'Body': message,
          'SilentPush': silent,
          'Title': title,
          'TimeToLive': ttl,
          'Url': url
        }
      }
    };
  } else if (service == 'ADM') {
    var messageRequest = {
      'Addresses': {
        [token]: {
```

```
          'ChannelType' : 'ADM'
        }
      },
      'MessageConfiguration': {
        'ADMMessage': {
          'Action': action,
          'Body': message,
          'SilentPush': silent,
          'Title': title,
          'Url': url
        }
      }
    };
  }

  return messageRequest
}

function ShowOutput(data){
  if (data["MessageResponse"]["Result"][recipient["token"]]["DeliveryStatus"]
      == "SUCCESSFUL") {
    var status = "Message sent! Response information: ";
  } else {
    var status = "The message wasn't sent. Response information: ";
  }
  console.log(status);
  console.dir(data, { depth: null });
}

function SendMessage() {
  var token = recipient['token'];
  var service = recipient['service'];
  var messageRequest = CreateMessageRequest();

  // Specify that you're using a shared credentials file, and specify the
  // IAM profile to use.
  var credentials = new AWS.SharedIniFileCredentials({ profile: 'default' });
  AWS.config.credentials = credentials;

  // Specify the AWS Region to use.
  AWS.config.update({ region: region });

  //Create a new Pinpoint object.
  var pinpoint = new AWS.Pinpoint();
```

```
  var params = {
    "ApplicationId": applicationId,
    "MessageRequest": messageRequest
  };

  // Try to send the message.
  pinpoint.sendMessages(params, function(err, data) {
    if (err) console.log(err);
    else     ShowOutput(data);
  });
}

SendMessage()
```

Python

AWS SDK for Python (Boto3) を使用してプッシュ通知を送信するには、この例を使用します。この例は、SDK for Python (Boto3) がすでにインストールされ、設定されていることを前提としています。

この例では、共有認証情報ファイルを使用して、既存の ユーザーのアクセスキーとシークレットアクセスキーを指定するものと想定しています。詳細については、『AWS SDK for Python (Boto3) API Reference』の「認証情報」を参照してください。

```
import json
import boto3
from botocore.exceptions import ClientError

# The AWS Region that you want to use to send the message. For a list of
# AWS Regions where the Amazon Pinpoint API is available, see
# https://docs.aws.amazon.com/pinpoint/latest/apireference/
region = "us-east-1"

# The title that appears at the top of the push notification.
title = "Test message sent from Amazon Pinpoint."

# The content of the push notification.
message = ("This is a sample message sent from Amazon Pinpoint by using the "
           "AWS SDK for Python (Boto3).")

# The Amazon Pinpoint project/application ID to use when you send this message.
# Make sure that the push channel is enabled for the project or application
# that you choose.
```

```
application_id = "ce796be37f32f178af652b26eexample"

# A dictionary that contains the unique token of the device that you want to send
 the
# message to, and the push service that you want to use to send the message.
recipient = {
    "token": "a0b1c2d3e4f5g6h7i8j9k0l1m2n3o4p5q6r7s8t9u0v1w2x3y4z5a6b7c8d8e9f0",
    "service": "GCM"
    }

# The action that should occur when the recipient taps the message. Possible
# values are OPEN_APP (opens the app or brings it to the foreground),
# DEEP_LINK (opens the app to a specific page or interface), or URL (opens a
# specific URL in the device's web browser.)
action = "URL"

# This value is only required if you use the URL action. This variable contains
# the URL that opens in the recipient's web browser.
url = "https://www.example.com"

# The priority of the push notification. If the value is 'normal', then the
# delivery of the message is optimized for battery usage on the recipient's
# device, and could be delayed. If the value is 'high', then the notification is
# sent immediately, and might wake a sleeping device.
priority = "normal"

# The amount of time, in seconds, that the push notification service provider
# (such as FCM or APNS) should attempt to deliver the message before dropping
# it. Not all providers allow you specify a TTL value.
ttl = 30

# Boolean that specifies whether the notification is sent as a silent
# notification (a notification that doesn't display on the recipient's device).
silent = False

# Set the MessageType based on the values in the recipient variable.
def create_message_request():

    token = recipient["token"]
    service = recipient["service"]

    if service == "GCM":
        message_request = {
            'Addresses': {
```

```
                token: {
                    'ChannelType': 'GCM'
                }
            },
            'MessageConfiguration': {
                'GCMMessage': {
                    'Action': action,
                    'Body': message,
                    'Priority' : priority,
                    'SilentPush': silent,
                    'Title': title,
                    'TimeToLive': ttl,
                    'Url': url
                }
            }
        }
    elif service == "APNS":
        message_request = {
            'Addresses': {
                token: {
                    'ChannelType': 'APNS'
                }
            },
            'MessageConfiguration': {
                'APNSMessage': {
                    'Action': action,
                    'Body': message,
                    'Priority' : priority,
                    'SilentPush': silent,
                    'Title': title,
                    'TimeToLive': ttl,
                    'Url': url
                }
            }
        }
    elif service == "BAIDU":
        message_request = {
            'Addresses': {
                token: {
                    'ChannelType': 'BAIDU'
                }
            },
            'MessageConfiguration': {
                'BaiduMessage': {
```

```
                    'Action': action,
                    'Body': message,
                    'SilentPush': silent,
                    'Title': title,
                    'TimeToLive': ttl,
                'Url': url
                }
            }
        }
    elif service == "ADM":
        message_request = {
            'Addresses': {
                token: {
                    'ChannelType': 'ADM'
                }
            },
            'MessageConfiguration': {
                'ADMMessage': {
                    'Action': action,
                    'Body': message,
                    'SilentPush': silent,
                    'Title': title,
                    'Url': url
                }
            }
        }
    else:
        message_request = None

    return message_request

# Show a success or failure message, and provide the response from the API.
def show_output(response):
    if response['MessageResponse']['Result'][recipient["token"]]['DeliveryStatus']
 == "SUCCESSFUL":
        status = "Message sent! Response information:\n"
    else:
        status = "The message wasn't sent. Response information:\n"
    print(status, json.dumps(response,indent=4))

# Send the message through the appropriate channel.
def send_message():

    token = recipient["token"]
```

```
    service = recipient["service"]
    message_request = create_message_request()

    client = boto3.client('pinpoint',region_name=region)

    try:
        response = client.send_messages(
            ApplicationId=application_id,
            MessageRequest=message_request
        )
    except ClientError as e:
        print(e.response['Error']['Message'])
    else:
        show_output(response)

send_message()
```

Amazon Pinpoint でカスタムチャンネルの作成

Amazon Pinpoint には、プッシュ通知、E メール、SMS、音声チャンネルによるメッセージの送信の組み込みサポートが含まれています。カスタムチャンネルを作成することで、Amazon Pinpoint が他のチャンネル経由でメッセージを送信するように設定することもできます。Amazon Pinpoint のカスタムチャンネルを使用すると、サードパーティーサービスを含む、API を持つサービスを通じてメッセージを送信できます。Webhook を使用するか、AWS Lambda 関数を呼び出して API を操作できます。

カスタムチャンネルキャンペーンを送信するセグメントには、すべてのタイプのエンドポイント (つまり、ChannelType 属性の値が EMAL、VOICE、SMS、CUSTOM、またはさまざまなプッシュ通知エンドポイントタイプの 1 つであるエンドポイント) を含めることができます。

カスタムチャンネル経由でメッセージを送信するキャンペーンの作成

Lambda 関数または Webhook を個別のキャンペーンに割り当てるには、Amazon Pinpoint API を使用して Campaign オブジェクトを作成または更新します。

キャンペーン内の MessageConfiguration オブジェクトには CustomMessage オブジェクトも含まれている必要があります。このオブジェクトには、1 つのメンバー (Data) があります。Data の値は、カスタムチャンネルに送信するメッセージペイロードを含む JSON 文字列です。

キャンペーンには CustomDeliveryConfiguration オブジェクトが含まれている必要があります。CustomDeliveryConfiguration オブジェクト内で、以下を指定します。

- EndpointTypes – カスタムチャンネルキャンペーンの送信先となるすべてのエンドポイントタイプを含む配列。このチャンネルには、次のチャンネルタイプのいずれか、またはすべてを含めることができます。
  - ADM
  - APNS
  - APNS_SANDBOX
  - APNS_VOIP
  - APNS_VOIP_SANDBOX
  - BAIDU

- CUSTOM
- EMAIL
- GCM
- SMS
- VOICE

- DeliveryUri – エンドポイントの送信先。以下のいずれかを 1 つだけ指定できます。
  - キャンペーンの実行時に実行する Lambda 関数の Amazon リソースネーム (ARN)。
  - キャンペーンの実行時にエンドポイントデータを送信する Webhook の URL。

> **Note**
>
> Campaign オブジェクトには、Hook オブジェクトを含めることもできます。このオブジェクトは、キャンペーンが実行されたときに Lambda 関数によってカスタマイズされるセグメントを作成するためにのみ使用されます。詳細については、「AWS Lambda を使用したセグメントのカスタマイズ」を参照してください。

Amazon Pinpoint がカスタムチャンネルに送信するイベントデータについて

カスタムチャンネル経由でメッセージを送信する Lambda 関数を作成する前に、Amazon Pinpoint が送信するデータについて理解しておく必要があります。Amazon Pinpoint キャンペーンがカスタムチャンネル経由でメッセージを送信すると、次の例のようなターゲット Lambda 関数にペイロードが送信されます。

```
{
  "Message":{},
  "Data":"The payload that's provided in the CustomMessage object in
 MessageConfiguration",
  "ApplicationId":"3a9b1f4e6c764ba7b031e7183example",
  "CampaignId":"13978104ce5d6017c72552257example",
  "TreatmentId":"0",
  "ActivityId":"575cb1929d5ba43e87e2478eeexample",
  "ScheduledTime":"2020-04-08T19:00:16.843Z",
  "Endpoints":{
    "1dbcd396df28ac6cf8c1c2b7fexample":{
```

```
      "ChannelType":"EMAIL",
      "Address":"mary.major@example.com",
      "EndpointStatus":"ACTIVE",
      "OptOut":"NONE",
      "Location":{
        "City":"Seattle",
        "Country":"USA"
      },
      "Demographic":{
        "Make":"OnePlus",
        "Platform":"android"
      },
      "EffectiveDate":"2020-04-01T01:05:17.267Z",
      "Attributes":{
        "CohortId":[
          "42"
        ]
      },
      "CreationDate":"2020-04-01T01:05:17.267Z"
    }
  }
}
```

イベントデータは次の属性を提供します。

- `ApplicationId` – キャンペーンが属する Amazon Pinpoint プロジェクトの ID。
- `CampaignId` – Lambda 関数を呼び出した Amazon Pinpoint キャンペーンの ID。
- `TreatmentId` – キャンペーンバリアントの ID。標準キャンペーンを作成した場合、この値は常に 0 です。A/B テストキャンペーンを作成した場合、この値は 0 ~ 4 の整数です。
- `ActivityId` – キャンペーンによって実行中のアクティビティの ID。
- `ScheduledTime` – Amazon Pinpoint がキャンペーンを実行した時刻。ISO 8601 形式で表示されます。
- `Endpoints` – キャンペーンのターゲットとなったエンドポイントのリスト。各ペイロードには、最大 50 のエンドポイントを含めることができます。キャンペーンの送信先セグメントに 50 以上のエンドポイントが含まれている場合、Amazon Pinpoint はすべてのエンドポイントが処理されるまで、繰り返し関数を呼び出します (最大で一度に 50 のエンドポイント)。

このサンプルデータは、カスタムチャンネルの Lambda 関数を作成およびテストするときに使用できます。

Webhook の設定

Webhook を使用してカスタムチャンネルメッセージを送信する場合、Webhook の URL は「https://」で始まる必要があります。Webhook URL には、英数字と次の記号のみを含めることができます。ハイフン (-)、ピリオド (.)、アンダースコア (_)、チルダ (~)、疑問符 (?)、スラッシュ (/)、ポンドまたはハッシュ記号 (#)、セミコロン (:)。URL は RFC3986 に準拠している必要があります。

Webhook URL を指定するキャンペーンを作成すると、Amazon Pinpoint はその URL に HTTP HEAD を発行します。HEAD リクエストに対する応答には、`X-Amz-Pinpoint-AccountId` というヘッダーが含まれている必要があります。このヘッダーの値は、AWS アカウント ID と同じである必要があります。

Lambda 関数オプションの設定

このセクションでは、カスタムチャンネル経由でメッセージを送信する Lambda 関数を作成するときに必要な手順の概要を説明します。まず、関数を作成します。その後、関数に実行ポリシーを追加します。このポリシーにより、Amazon Pinpoint はキャンペーンの実行時にポリシーを実行できます。

Lambda 関数の作成の概要については、『AWS Lambda デベロッパーガイド』の「Building Lambda functions」を参照してください。

Lambda 関数の例

次のコード例では、ペイロードを処理し、CloudWatch の各エンドポイントタイプのエンドポイントの数を記録します。

```
import boto3
import random
import pprint
import json
import time

cloudwatch = boto3.client('cloudwatch')

def lambda_handler(event, context):
    customEndpoints = 0
    smsEndpoints = 0
    pushEndpoints = 0
    emailEndpoints = 0
    voiceEndpoints = 0
```

```
    numEndpoints = len(event['Endpoints'])

    print("Payload:\n", event)
    print("Endpoints in payload: " + str(numEndpoints))

    for key in event['Endpoints'].keys():
        if event['Endpoints'][key]['ChannelType'] == "CUSTOM":
            customEndpoints += 1
        elif event['Endpoints'][key]['ChannelType'] == "SMS":
            smsEndpoints += 1
        elif event['Endpoints'][key]['ChannelType'] == "EMAIL":
            emailEndpoints += 1
        elif event['Endpoints'][key]['ChannelType'] == "VOICE":
            voiceEndpoints += 1
        else:
            pushEndpoints += 1

    response = cloudwatch.put_metric_data(
        MetricData = [
            {
                'MetricName': 'EndpointCount',
                'Dimensions': [
                    {
                        'Name': 'CampaignId',
                        'Value': event['CampaignId']
                    },
                    {
                        'Name': 'ApplicationId',
                        'Value': event['ApplicationId']
                    }
                ],
                'Unit': 'None',
                'Value': len(event['Endpoints'])
            },
            {
                'MetricName': 'CustomCount',
                'Dimensions': [
                    {
                        'Name': 'CampaignId',
                        'Value': event['CampaignId']
                    },
                    {
                        'Name': 'ApplicationId',
                        'Value': event['ApplicationId']
```

```
                    }
                ],
                'Unit': 'None',
                'Value': customEndpoints
            },
            {
                'MetricName': 'SMSCount',
                'Dimensions': [
                    {
                        'Name': 'CampaignId',
                        'Value': event['CampaignId']
                    },
                    {
                        'Name': 'ApplicationId',
                        'Value': event['ApplicationId']
                    }
                ],
                'Unit': 'None',
                'Value': smsEndpoints
            },
            {
                'MetricName': 'EmailCount',
                'Dimensions': [
                    {
                        'Name': 'CampaignId',
                        'Value': event['CampaignId']
                    },
                    {
                        'Name': 'ApplicationId',
                        'Value': event['ApplicationId']
                    }
                ],
                'Unit': 'None',
                'Value': emailEndpoints
            },
            {
                'MetricName': 'VoiceCount',
                'Dimensions': [
                    {
                        'Name': 'CampaignId',
                        'Value': event['CampaignId']
                    },
                    {
                        'Name': 'ApplicationId',
```

```
                    'Value': event['ApplicationId']
                }
            ],
            'Unit': 'None',
            'Value': voiceEndpoints
        },
        {
            'MetricName': 'PushCount',
            'Dimensions': [
                {
                    'Name': 'CampaignId',
                    'Value': event['CampaignId']
                },
                {
                    'Name': 'ApplicationId',
                    'Value': event['ApplicationId']
                }
            ],
            'Unit': 'None',
            'Value': pushEndpoints
        },
        {
            'MetricName': 'EndpointCount',
            'Dimensions': [
            ],
            'Unit': 'None',
            'Value': len(event['Endpoints'])
        },
        {
            'MetricName': 'CustomCount',
            'Dimensions': [
            ],
            'Unit': 'None',
            'Value': customEndpoints
        },
        {
            'MetricName': 'SMSCount',
            'Dimensions': [
            ],
            'Unit': 'None',
            'Value': smsEndpoints
        },
        {
            'MetricName': 'EmailCount',
```

```
                'Dimensions': [
                ],
                'Unit': 'None',
                'Value': emailEndpoints
            },
            {
                'MetricName': 'VoiceCount',
                'Dimensions': [
                ],
                'Unit': 'None',
                'Value': voiceEndpoints
            },
            {
                'MetricName': 'PushCount',
                'Dimensions': [
                ],
                'Unit': 'None',
                'Value': pushEndpoints
            }
        ],
        Namespace = 'PinpointCustomChannelExecution'
    )
    print("cloudwatchResponse:\n",response)
```

Amazon Pinpoint キャンペーンがこの Lambda 関数を実行すると、Amazon Pinpoint はセグメントメンバーのリストを関数に送信します。この関数は、それぞれの `ChannelType` のエンドポイントの数をカウントします。その後、そのデータを Amazon CloudWatch に送信します。CloudWatch コンソールでこれらのメトリクスを表示することもできます。メトリクスは、PinPointCustomChannelExecution 名前空間で使用できます。

このコード例を変更して、このサービスを通じてメッセージを送信するために、外部サービスの API に接続するようにもできます。

Amazon Pinpoint の Lambda 関数レスポンス形式

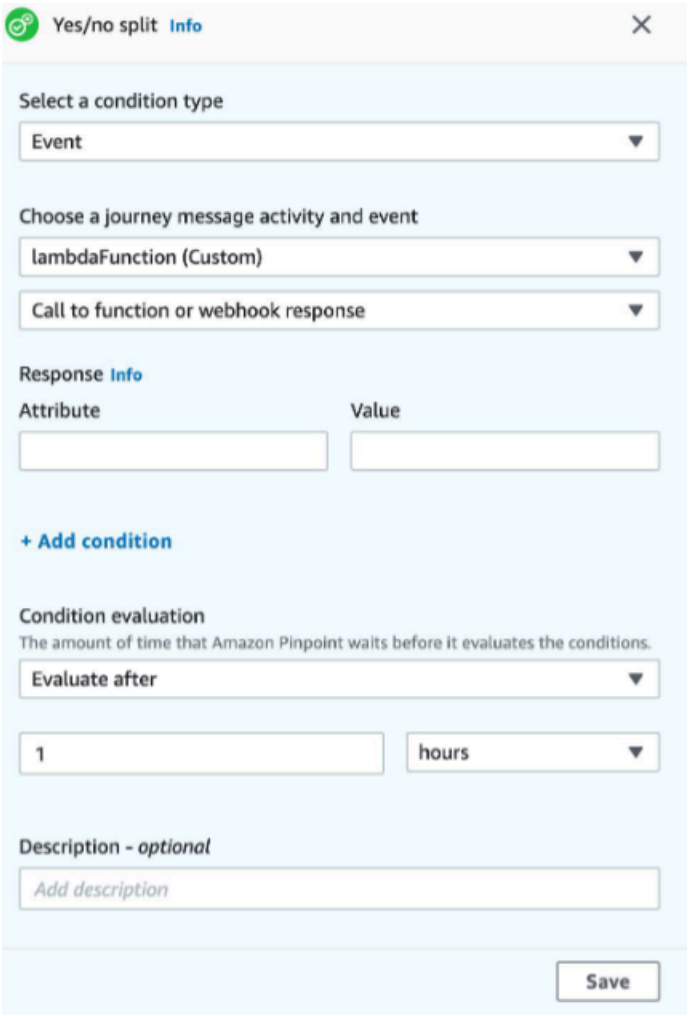
カスタムのチャンネルアクティビティの後にジャーニー多変量分割または、はい/いいえ分割を使用してエンドポイントパスを決定するには、Lambda 関数レスポンスを Amazon Pinpoint で認識可能な形式に構成し、エンドポイントを正しいパスに送信する必要があります。

レスポンスの構造は次の形式にする必要があります。

```
{
```

```
    <Endpoint ID 1>:{
        EventAttributes: {
            <Key1>: <Value1>,
            <Key2>: <Value2>,
            ...
        }
    },
    <Endpoint ID 2>:{
        EventAttributes: {
            <Key1>: <Value1>,
            <Key2>: <Value2>,
            ...
        }
    },
...
}
```

これにより、エンドポイントパスを決定するためのキーと値を選択できます。



Lambda 関数を呼び出すための Amazon Pinpoint アクセス許可を付与する

AWS Command Line Interface (AWS CLI) を使用して、Lambda 関数に割り当てられた Lambda 関数ポリシーにアクセス許可を追加できます。Amazon Pinpoint に関数の呼び出しを許可するには、次の例に示すように Lambda の add-permission コマンドを使用します。

```
aws lambda add-permission \
--function-name myFunction \
--statement-id sid0 \
--action lambda:InvokeFunction \
--principal pinpoint.us-east-1.amazonaws.com \
--source-arn arn:aws:mobiletargeting:us-east-1:111122223333:apps/*
--source-account 111122223333
```

上記のコマンドで、次の操作を行います。

- *myFunction* を Lambda 関数の名前に置き換えます。
- *us-east-1* を、Amazon Pinpoint を使用する AWS リージョンに置き換えます。
- *111122223333* を自分の AWS アカウント ID に置き換えます。

add-permission のコマンドを実行すると、Lambda により以下のような出力が表示されます。

```
{
  "Statement": "{\"Sid\":\"sid\",
    \"Effect\":\"Allow\",
    \"Principal\":{\"Service\":\"pinpoint.us-east-1.amazonaws.com\"},
    \"Action\":\"lambda:InvokeFunction\",
    \"Resource\":\"arn:aws:lambda:us-east-1:111122223333:function:myFunction\",
    \"Condition\":
      {\"ArnLike\":
        {\"AWS:SourceArn\":
         \"arn:aws:mobiletargeting:us-east-1:111122223333:apps/*\"}},
      {\"StringEquals\":
        {\"AWS:SourceAccount\":
          \"111122223333\"}}}
}
```

Statement 値は Lambda 関数ポリシーに追加されたステートメントの JSON 文字列バージョンです。

実行ポリシーのその他の制限

特定の Amazon Pinpoint プロジェクトに制限することで、実行ポリシーを変更できます。これを行うには、前の例にある * をプロジェクトの一意の ID に置き換えます。ポリシーを特定のキャンペーンに限定することで、ポリシーをさらに制限できます。例えば、プロジェクト ID 95fee4cd1d7f5cd67987c1436example のプロジェクトで、キャンペーン ID dbaf6ec2226f0a9a8615e3ea5example を持つキャンペーンのみを許可するようにポリシーを制限するには、source-arn 属性に次の値を使用します。

```
arn:aws:mobiletargeting:us-east-1:111122223333:apps/dbaf6ec2226f0a9a8615e3ea5example/
campaigns/95fee4cd1d7f5cd67987c1436example
```

> **Note**
>
> Lambda 関数の実行を特定のキャンペーンに制限する場合は、まず、制限の少ないポリシーを使用して関数を作成する必要があります。次に、Amazon Pinpoint でキャンペーンを作成し、機能を選択する必要があります。最後に、指定したキャンペーンを参照するように実行ポリシーを更新する必要があります。

Amazon Pinpoint のイベントを Kinesis にストリーミングする

Amazon Pinpoint では、イベントは、ユーザーがアプリケーションの 1 つを操作した場合、お客様がキャンペーンまたはジャーニーからメッセージを送信した場合、お客様がトランザクション SMS または E メールメッセージを送信した場合に発生するアクションです。たとえば、E メールメッセージを送信すると、いくつかのイベントが発生します。

- メッセージを送信すると、send イベントが発生します。
- メッセージが受信者の受信ボックスに到達すると、delivered イベントが発生します。
- 受信者がメッセージを開くと、open イベントが発生します。

イベントに関する情報を Amazon Kinesis に送信するように Amazon Pinpoint を設定できます。Kinesis プラットフォームは、 のサービスからデータをリアルタイムで収集、処理、分析するために使用できる AWS サービスを提供します。Amazon Pinpoint は、イベントデータを Firehose に送信できます。Firehose は、このデータを Amazon S3 や Amazon Redshift などの AWS データストアにストリーミングします。Amazon Pinpoint は、Kinesis Data Streams にデータを流すこともできます。Kinesis Data Streams は、分析アプリケーションで処理するために複数のデータストリームを取り込み、保存します。

Amazon Pinpoint イベントストリームには、Amazon Pinpoint に接続するアプリケーション (アプリ) のユーザー操作に関する情報が含まれています。また、キャンペーン、任意のチャネル、ジャーニーから送信するすべてのメッセージに関する情報も含まれています。定義した任意のカスタムイベントを含めることができます。最後に、送信するすべてのトランザクション E メールおよび SMS メッセージに関する情報が含まれています。

> **Note**
>
> Amazon Pinpoint は、トランザクションプッシュ通知や音声メッセージに関する情報をストリーミングしません。

この章では、イベントデータを Kinesis にストリーミングするための Amazon Pinpoint の設定に関する情報を提供します。また、Amazon Pinpoint がストリーミングするイベントデータの例も含まれています。

トピック

イベントストリーミングのセットアップ

イベントデータを Amazon Kinesis ストリームまたは Amazon Data Firehose 配信ストリームに送信するように Amazon Pinpoint を設定できます。 Amazon Kinesis Amazon Pinpoint は、キャンペーン、ジャーニー、トランザクション用の Eメールや SMS メッセージのイベントデータを送信することができます。

このセクションには、プログラムによるイベントストリーミングのセットアップに関する情報が含まれています。Amazon Pinpoint コンソールを使用して、イベントストリーミングをセットアップすることもできます。Amazon Pinpoint コンソールを使用したイベントストリーミングのセットアップの詳細については、『Amazon Pinpoint ユーザーガイド』の「イベントストリーム設定」を参照してください。

前提条件

このセクションの例には、次の入力が必要です。

- Amazon Pinpoint およびレポートイベントと統合されているアプリケーションのアプリケーション ID。統合する方法については、「Amazon Pinpoint とアプリケーションの統合」を参照してください。
- AWS アカウントの Kinesis ストリームまたは Firehose 配信ストリームの Amazon リソースネーム (ARN)。これらのリソースの作成については、Amazon Kinesis Data Streams デベロッパーガイドの「ストリームの作成と管理」または Amazon Data Firehose デベロッパーガイドの「Amazon Data Firehose 配信ストリームの作成」を参照してください。 Amazon Kinesis
- Amazon Pinpoint がストリームにデータを送信することを許可する AWS Identity and Access Management (IAM) ロールの ARN。ロールの作成の詳細については、「Kinesis にイベントをストリーミングするための IAM ロール」を参照してください。

AWS CLI

次の AWS CLI 例では、 put-event-stream コマンドを使用します。このコマンドは、イベントを Kinesis ストリームに送信するように Amazon Pinpoint を設定します。

```
aws pinpoint put-event-stream \
--application-id projectId \
--write-event-stream DestinationStreamArn=streamArn,RoleArn=roleArn
```

AWS SDK for Java

次の Java の例では、イベントを Kinesis ストリームに送信するように Amazon Pinpoint を設定します。

```
public PutEventStreamResult createEventStream(AmazonPinpoint pinClient,
        String appId, String streamArn, String roleArn) {

    WriteEventStream stream = new WriteEventStream()
            .withDestinationStreamArn(streamArn)
            .withRoleArn(roleArn);

    PutEventStreamRequest request = new PutEventStreamRequest()
            .withApplicationId(appId)
            .withWriteEventStream(stream);

    return pinClient.putEventStream(request);
}
```

この例では、Kinesis ストリームの ARN と IAM ロールを保存する `WriteEventStream` オブジェクトを構築します。`WriteEventStream` オブジェクトは、`PutEventStreamRequest` オブジェクトに渡され、特定のアプリケーションのためにイベントをストリームするように Amazon Pinpoint を設定します。`PutEventStreamRequest` オブジェクトは Amazon Pinpoint クライアントの `putEventStream` メソッドに渡されます。

Kinesis ストリームは複数のアプリケーションに割り当てることができます。これを行うと、Amazon Pinpoint は各アプリケーションからストリームに base64 でエンコードされたイベントデータを送信するので、データをコレクションとして分析できます。以下の例のメソッドでは、アプリケーション (アプリ) ID のリストを受け取り、前の例のメソッドである `createEventStream` を使用して各アプリにストリームを割り当てます。

```
public List<PutEventStreamResult> createEventStreamFromAppList(
        AmazonPinpoint pinClient, List<String> appIDs,
        String streamArn, String roleArn) {

    return appIDs.stream()
            .map(appId -> createEventStream(pinClient, appId, streamArn,
                    roleArn))
            .collect(Collectors.toList());
}
```

1 つのストリームを複数のアプリケーションに割り当てることができますが、複数のストリームを 1 つのアプリケーションに割り当てることはできません。

イベントストリーミングの無効化

Kinesis ストリームをアプリケーションに割り当てた場合、そのアプリケーションのイベントストリーミングを無効にできます。Amazon Pinpoint は Kinesis へのイベントのストリーミングを停止しますが、Amazon Pinpoint コンソールを使用することでイベント分析を表示することができます。

AWS CLI

delete-event-streamコマンドを使用します。

```
aws pinpoint delete-event-stream --application-id application-id
```

AWS SDK for Java

Amazon Pinpoint クライアントの deleteEventStreamメソッドを使用します。

```
pinClient.deleteEventStream(new DeleteEventStreamRequest().withApplicationId(appId));
```

アプリケーションイベント

アプリケーション (アプリ) を Amazon Pinpoint と統合した後、Amazon Pinpoint によって、アプリケーションのユーザーアクティビティとメッセージ配信に関するイベントデータがストリーミング可能になります。

例

アプリイベントの JSON オブジェクトには以下の例に示されているデータが含まれています。

```
{
  "event_type": "_session.stop",
  "event_timestamp": 1487973802507,
  "arrival_timestamp": 1487973803515,
  "event_version": "3.0",
  "application": {
    "app_id": "a1b2c3d4e5f6g7h8i9j0k1l2m3n4o5p6",
    "cognito_identity_pool_id": "us-east-1:a1b2c3d4-e5f6-g7h8-i9j0-k1l2m3n4o5p6",
    "package_name": "main.page",
    "sdk": {
      "name": "aws-sdk-mobile-analytics-js",
      "version": "0.9.1:2.4.8"
    },
    "title": "title",
    "version_name": "1.0",
    "version_code": "1"
  },
  "client": {
    "client_id": "m3n4o5p6-a1b2-c3d4-e5f6-g7h8i9j0k1l2",
    "cognito_id": "us-east-1:i9j0k1l2-m3n4-o5p6-a1b2-c3d4e5f6g7h8"
  },
  "device": {
    "locale": {
      "code": "en_US",
      "country": "US",
      "language": "en"
    },
    "make": "generic web browser",
    "model": "Unknown",
    "platform": {
      "name": "android",
      "version": "10.10"
    }
  },
  "session": {
    "session_id": "f549dea9-1090-945d-c3d1-e4967example",
    "start_timestamp": 1487973202531,
    "stop_timestamp": 1487973802507
  },
  "attributes": {},
  "metrics": {}
}
```

アプリイベントの属性

このセクションでは、アプリケーションのイベントストリームに含まれる属性を定義します。

属性	説明
event_type	イベントのタイプ。可能な値は以下のとおりです。 • _session.start – エンドポイントが新しいセッションを開始しました。 • _session.stop – エンドポイントがセッションを終了しました。 • _userauth.sign_in – アプリケーションにログインしているエンドポイント。 • _userauth.sign_up – 新しいエンドポイントがアプリケーションの登録プロセスを完了しました。 • _userauth.auth_fail – エンドポイントがアプリケーションにサインインしようとしましたが、プロセスを完了できませんでした。 • _monetization.purchase – エンドポイントがアプリケーションで購入されました。 • _session.pause – エンドポイントはセッションを一時停止しました。一時停止したセッションは再開できるため、まったく新しいセッションを開始しなくてもメトリクスの収集を継続できます。 • _session.resume – エンドポイントはセッションを再開しました。
event_timestamp	イベントが報告された時刻。Unix 時間 (ミリ秒単位) として表示されます。

属性	説明
arrival_timestamp	イベントが Amazon Pinpoint によって受信された時刻が、Unix 時間 (ミリ秒単位) として表示されます。
event_version	イベントの JSON スキーマのバージョン。 **Tip** イベント処理アプリケーションでこのバージョンをチェックし、スキーマの更新に合わせてアプリケーションを更新する時期を把握します。
application	イベントに関連付けられた Amazon Pinpoint プロジェクトに関する情報。詳細については、表「アプリケーション」を参照してください。
client	イベントを報告したエンドポイントに関する情報。詳細については、表「クライアント」を参照してください。
device	イベントを報告したデバイスに関する情報。詳細については、表「デバイス」を参照してください。
session	イベントを生成したセッションに関する情報。詳細については、表「セッション」を参照してください。
attributes	イベントに関連付けられている属性。アプリケーションによって報告されるイベントの場合、このオブジェクトにはユーザーが定義したカスタム属性が含まれます。

属性	説明
metrics	イベントに関連するメトリクス。オプションで、カスタムメトリクスを Amazon Pinpoint に送信するようにアプリケーションを設定できます。

アプリケーション

イベントが関連付けられている Amazon Pinpoint プロジェクトに関する情報が含まれています。

属性	説明
app_id	イベントを報告した Amazon Pinpoint プロジェクトの一意の ID。
cognito_identity_pool_id	エンドポイントが関連付けられている Amazon Cognito ID プールの ID。
package_name	アプリパッケージの名前 (com.example.my_app など)。
sdk	イベントを報告するために使用された SDK に関する情報。詳細については、表「SDK」を参照してください。
title	アプリケーションの名前。
version_name	アプリケーションのバージョンの名前 (V2.5 など)。
version_code	アプリケーションのバージョン番号 (3 など)。

SDK

イベントを報告するために使用された SDK に関する情報を含めます。

属性	説明
name	イベントを報告するために使用された SDK の名前。
version	SDK のバージョン。

クライアント

イベントを生成したエンドポイントに関する情報が含まれています。

属性	説明
client_id	エンドポイントの ID。
cognito_id	エンドポイントに関連付けられている Amazon Cognito ID トークン。

デバイス

イベントを生成したエンドポイントのデバイスに関する情報が含まれています。

属性	説明
locale	エンドポイントのデバイスの言語とリージョンの設定に関する情報。詳細については、表「ロケール」を参照してください。
make	エンドポイントのデバイスの製造元。
model	エンドポイントのデバイスのモデル識別子。
platform	エンドポイントのデバイス上のオペレーティングシステムに関する情報。詳細については、表「プラットフォーム」を参照してください。

[Locale] (国)

エンドポイントのデバイスの言語とリージョンの設定に関する情報を含めます。

属性	説明
code	デバイスに関連付けられているロケール識別子。
country	デバイスのロケールに関連付けられている国またはリージョン。
language	デバイスのロケールに関連付けられている言語。

プラットフォーム

エンドポイントのデバイス上のオペレーティングシステムに関する情報を含めます。

属性	説明
name	デバイス上のオペレーティングシステムの名前。
version	デバイス上のオペレーティングシステムのバージョン。

セッション

イベントを生成したセッションに関する情報が含まれています。

属性	説明
session_id	セッションを識別する一意の ID。
start_timestamp	セッションが開始された日時。Unix 時間 (ミリ秒単位) として表示されます。

属性	説明
stop_timestamp	セッションが終了した日時。Unix 時間 (ミリ秒単位) として表示されます。

キャンペーンイベント

Amazon Pinpoint を使用して任意のチャンネルを介してキャンペーンを送信する場合、Amazon Pinpoint はそれらのキャンペーンに関するイベントデータをストリーミングできます。これには、キャンペーンから送信した E メールまたは SMS メッセージのイベントデータが含まれます。これらのタイプのメッセージに対して Amazon Pinpoint がストリーミングするデータの詳細については、「the section called “E メールイベント”」および「the section called “SMS イベント”」を参照してください。

イベント例

キャンペーンイベントの JSON オブジェクトには以下のサンプルに示されているデータが含まれています。

```
{
  "event_type": "_campaign.send",
  "event_timestamp": 1562109497426,
  "arrival_timestamp": 1562109497494,
  "event_version": "3.1",
  "application": {
    "app_id": "a1b2c3d4e5f6g7h8i9j0k1l2m3n4o5p6",
    "sdk": {}
  },
  "client": {
    "client_id": "d8dcf7c5-e81a-48ae-8313-f540cexample"
  },
  "device": {
    "platform": {}
  },
  "session": {},
  "attributes": {
    "treatment_id": "0",
    "campaign_activity_id": "5473285727f04865bc673e527example",
    "delivery_type": "GCM",
    "campaign_id": "4f8d6097c2e8400fa3081d875example",
```

```
    "campaign_send_status": "SUCCESS"
  },
  "client_context": {
    "custom": {
      "endpoint": "{\"ChannelType\":\"GCM\",\"EndpointStatus\":\"ACTIVE\",
          #\"OptOut\":\"NONE\",\"RequestId\":\"ec229696-9d1e-11e9-8bf1-85d0aexample\",
          #\"EffectiveDate\":\"2019-07-02T23:12:54.836Z\",\"User\":{}}"
    }
  },
  "awsAccountId": "123456789012"
}
```

キャンペーンイベント属性

このセクションでは、キャンペーンイベントストリームに含まれる属性を定義します。

属性	説明
event_type	イベントのタイプ。可能な値は以下のとおりです。 • _campaign.send – Amazon Pinpoint がキャンペーンを実行しました。 • _campaign.opened_notification – プッシュ通知キャンペーンの場合、このイベントタイプは受信者が通知をタップして開いたことを示します。 • _campaign.received_foreground – プッシュ通知キャンペーンの場合、このイベントタイプは受信者がメッセージをフォアグラウンド通知として受け取ったことを示します。 • _campaign.received_background – プッシュ通知キャンペーンの場合、このイベントタイプは受信者がメッセージをバックグラウンド通知として受け取ったことを示します。

属性	説明
	Note _campaign.opened_notification、_campaign.received_foreground、_campaign.received_background は AWS Amplify を使用した場合のみ返されます。詳細については、「アプリケーションを AWS Amplify と統合する」を参照してください。「AWS Amplify を使用してフロントエンドアプリケーションを Amazon Pinpoint に接続する」を参照してください。
`event_timestamp`	イベントが報告された時刻。Unix 時間 (ミリ秒単位) として表示されます。
`arrival_timestamp`	イベントが Amazon Pinpoint によって受信された時刻が、Unix 時間 (ミリ秒単位) として表示されます。
`event_version`	イベントの JSON スキーマのバージョン。 **Tip** イベント処理アプリケーションでこのバージョンをチェックし、スキーマの更新に合わせてアプリケーションを更新する時期を把握します。
`application`	イベントに関連付けられた Amazon Pinpoint プロジェクトに関する情報。詳細については、表「アプリケーション」を参照してください。

属性	説明
client	イベントが関連付けられているエンドポイントに関する情報。詳細については、表「クライアント」を参照してください。
device	イベントを報告したデバイスに関する情報。キャンペーンメッセージとトランザクションメッセージの場合、このオブジェクトは空です。
session	イベントを生成したセッションに関する情報。キャンペーンの場合、このオブジェクトは空です。
attributes	イベントに関連付けられている属性。アプリケーションの 1 つによって報告されるイベントの場合、このオブジェクトにはアプリケーションによって定義されたカスタム属性を含めることができます。キャンペーンを送信したときに作成されたイベントの場合、このオブジェクトにはキャンペーンに関連付けられた属性が含まれています。トランザクションメッセージを送信するときに生成されるイベントの場合、このオブジェクトにはメッセージ自体に関連する情報が含まれます。 詳細については、表「属性」を参照してください。
client_context	endpoint プロパティを格納した custom オブジェクトが含まれています。endpoint プロパティには、キャンペーンが送信されたエンドポイントのエンドポイントレコードの内容が含まれています。
awsAccountId	メッセージの送信に使用された AWS アカウントの ID。

アプリケーション

イベントが関連付けられている Amazon Pinpoint プロジェクトに関する情報が含まれています。

属性	説明
app_id	イベントを報告した Amazon Pinpoint プロジェクトの一意の ID。
sdk	イベントを報告するために使用された SDK。

属性

イベントを生成したキャンペーンに関する情報が含まれています。

属性	説明
treatment_id	メッセージが A/B テストキャンペーンを使用して送信された場合、この値はメッセージの処理番号を表します。標準キャンペーンの場合、この値は 0 です。
campaign_activity_id	イベントが発生したときに Amazon Pinpoint が生成する一意の ID。
delivery_type	キャンペーンの配信方法。この属性を、client_context の endpoint プロパティで指定された ChannelType フィールドと混同しないでください。ChannelType フィールドは通常、メッセージの送信先のエンドポイントに基づいています。 1 つのエンドポイントタイプのみをサポートするチャンネルの場合、delivery_type および ChannelType フィールドの値は同じになります。例えば、E メールチャンネルの場合、delivery_type および ChannelType フィールドの値は同じ EMAIL になります。

属性	説明
	ただし、カスタムチャンネルなど、さまざまなエンドポイントタイプをサポートするチャンネルでは、この条件は必ずしも当てはまりません。EMAIL、SMS、CUSTOM など、さまざまなエンドポイントに対してカスタムチャンネルを使用できます。この場合、`delivery_type` はカスタム配信イベントの CUSTOM を特定し、`ChannelType` はキャンペーンの送信先のエンドポイントのタイプ (EMAIL、SMS、CUSTOM など) を指定します。カスタムチャンネルの作成の詳細については、「カスタムチャンネルの作成」を参照してください。 可能な値は以下のとおりです。 • EMAIL • SMS • ADM • APNS • APNS_SANDBOX • APNS_VOIP • APNS_VOIP_SANDBOX • VOICE • GCM • BAIDU • PUSH • CUSTOM
`campaign_id`	メッセージの送信元のキャンペーンの一意の ID。

属性	説明
campaign_send_status	ターゲットエンドポイントのキャンペーンのステータスを示します。可能な値は以下のとおりです。 • SUCCESS – キャンペーンはエンドポイントに正常に送信されました。 • FAILURE – キャンペーンはエンドポイントに送信されませんでした。 • DAILY_CAP – エンドポイントにすでに 1 日あたりの最大数のメッセージを送信しているため、キャンペーンはエンドポイントに送信されませんでした。 • EXPIRED – キャンペーンの最大期間または送信レート設定を超えるため、キャンペーンはエンドポイントに送信されませんでした。 • QUIET_TIME – クワイエットタイム制限のため、キャンペーンはエンドポイントに送信されませんでした。 • HOLDOUT – エンドポイントがホールドアウトグループのメンバーであるため、キャンペーンはエンドポイントに送信されませんでした。 • DUPLICATE_ADDRESS – セグメントに重複するエンドポイントアドレスが含まれています。キャンペーンはエンドポイントアドレスに 1 回送信されました。 • QUIET_TIME – クワイエットタイム制限のため、キャンペーンはエンドポイントに送信されませんでした。 • CAMPAIGN_CAP – キャンペーンからエンドポイントに既に最大数のメッセージが送信されているため、キャンペーンはエンドポイントに送信されませんでした。

属性	説明
	• FAILURE_PERMANENT – エンドポイントへの送信時に永続的な障害が発生しました。 • TRANSIENT_FAILURE – エンドポイントへの送信時に一時的な障害が発生しました。 • THROTTLED – 送信がスロットリングされました。 • UNKNOWN – 不明な障害です。 • HOOK_FAILURE – キャンペーンフックが失敗しました。 • CUSTOM_DELIVERY_FAILURE – カスタム配信が失敗しました。 • RECOMMENDATION_FAILURE – レコメンダーが失敗しました。 • UNSUPPORTED_CHANNEL – チャンネルはサポートされていません。

クライアント

キャンペーンの対象となったエンドポイントに関する情報が含まれています。

属性	説明
`client_id`	キャンペーンが送信されたエンドポイントのエンドポイント ID。

ジャーニーイベント

ジャーニーを発行する場合、Amazon Pinpoint はジャーニーのイベントデータをストリーミングできます。これには、ジャーニーから送信する E メール、SMS、プッシュ、カスタムメッセージのイベントデータも含まれます。

Amazon Pinpoint がストリーミングするデータについては、以下を参照してください。

- E メールメッセージについては、「the section called “E メールイベント”」を参照してください。
- SMS メッセージについては、「SMS イベント」を参照してください。

イベント例

ジャーニーイベントの JSON オブジェクトには以下のサンプルに示されているデータが含まれています。

```
{
   "event_type":"_journey.send",
   "event_timestamp":1572989078843,
   "arrival_timestamp":1572989078843,
   "event_version":"3.1",
   "application":{
      "app_id":"a1b2c3d4e5f6g7h8i9j0k1l2m3n4o5p6",
      "sdk":{

      }
   },
   "client":{
      "client_id":"d8dcf7c5-e81a-48ae-8313-f540cexample"
   },
   "device":{
      "platform":{

      }
   },
   "session":{

   },
   "attributes":{
      "journey_run_id":"edc9a0b577164d1daf72ebd15example",
      "journey_send_status":"SUCCESS",
      "journey_id":"546401670c5547b08811ac6a9example",
      "journey_activity_id":"0yKexample",
      "journey_activity_type": "EMAIL",
      "journey_send_status_message": "200",
      "journey_send_status_code": "200"
   },
   "client_context":{
      "custom":{
```

```
         "endpoint":"{\"ChannelType\":\"EMAIL\",\"EndpointStatus\":\"ACTIVE\",\"OptOut
\":\"NONE\",\"Demographic\":{\"Timezone\":\"America/Los_Angeles\"}}"
      }
   },
   "awsAccountId":"123456789012"
}
```

ジャーニーイベントの属性

このセクションでは、Amazon Pinpoint がジャーニーについて生成するイベントストリームデータに含まれる属性を定義します。

属性	説明
event_type	イベントのタイプ。ジャーニーイベントの場合、この属性の値は常に _journey.send であり、Amazon Pinpoint がジャーニーを実行したことを示します。
event_timestamp	イベントが報告された時刻。Unix 時間 (ミリ秒単位) として表示されます。
arrival_timestamp	イベントが Amazon Pinpoint によって受信された時刻が、Unix 時間 (ミリ秒単位) として表示されます。
event_version	イベントの JSON スキーマのバージョン。 **Tip** イベント処理アプリケーションでこのバージョンをチェックし、スキーマの更新に合わせてアプリケーションを更新する時期を把握します。
application	イベントに関連付けられた Amazon Pinpoint プロジェクトに関する情報。詳細については、表「アプリケーション」を参照してください。

属性	説明
client	イベントに関連付けられているエンドポイントについての情報。詳細については、表「クライアント」を参照してください。
device	イベントを報告したデバイスに関する情報。ジャーニーの場合、このオブジェクトは空です。
session	イベントを生成したセッションに関する情報。ジャーニーの場合、このオブジェクトは空です。
attributes	イベントを生成したジャーニーおよびジャーニーアクティビティに関連付けられている属性。詳細については、表「属性」を参照してください。
client_context	endpoint プロパティを格納した custom オブジェクトが含まれています。endpoint プロパティには、イベントに関連付けられているエンドポイントのエンドポイントレコードの内容が含まれています。
awsAccountId	ジャーニーの実行に使用された AWS アカウントの ID。

アプリケーション

イベントが関連付けられている Amazon Pinpoint プロジェクトに関する情報が含まれています。

属性	説明
app_id	イベントを報告した Amazon Pinpoint プロジェクトの一意の ID。
sdk	イベントを報告するために使用された SDK。

クライアント

イベントが関連付けられているエンドポイントに関する情報が含まれています。

属性	説明
client_id	エンドポイントの ID。

属性

イベントを生成したジャーニーに関する情報が含まれています。

属性	説明
journey_run_id	イベントを生成したジャーニーの一意の ID。Amazon Pinpoint はジャーニーの新しい実行ごとに自動的にこの ID を生成し、割り当てます。
journey_send_status	イベントに関連付けられているメッセージの配信ステータスを示します。可能な値は以下のとおりです: • SUCCESS – メッセージはエンドポイントに正常に送信されました。 • FAILURE – エラーが発生したため、メッセージはエンドポイントに送信されませんでした。 • CUSTOM_DELIVERY_FAILURE – カスタム配信が失敗しました。 • FAILURE_PERMANENT – エンドポイントへの送信時に永続的な障害が発生しました。 Tip FAILURE_PERMANENT ステータスのイベントをフィルタリングし、

属性	説明
	を 403 journey_send_statu s_code に設定して、アクセスポリシーとロール違反があるかどうかを判断できます。音声によるアウトバウンドキャンペーンの場合、これらの例外は、Amazon Connect キャンペーンへの Amazon Pinpoint ジャーニーをバインドする接続キャンペーン実行ロールが、処理中のジャーニー実行のために誤って削除された場合に発生するのが一般的です。 • THROTTLED – 送信がスロットリングされました。 • UNSUPPORTED_CHANNEL – チャネルはサポートされていません。 • DAILY_CAP – 24 時間内にジャーニーまたはプロジェクトが 1 つのエンドポイントに送信できるメッセージの最大数を超えるため、メッセージはエンドポイントに送信されませんでした。 • QUIET_TIME – ジャーニーまたはプロジェクトのクワイエットタイム制限のため、メッセージは送信されませんでした。 • QUIET_TIME_MISSING_TIMEZONE – タイムゾーンの推定ではエンドポイントのタイムゾーンを推定できず、クワイエットタイムが有効になっているため、メッセージは送信されませんでした。
journey_id	イベントを生成したジャーニーの一意の ID。
journey_activity_id	イベントを生成したジャーニーアクティビティの一意の ID。

属性	説明
journey_activity_type	イベントのジャーニーアクティビティタイプです。これは、EMAIL、SMS、PUSH、CONTACT_CENTER、または CUSTOM です。 **Note** 音声は、サポートされているジャーニーアクティビティタイプではありません。
journey_send_status_message	送信イベントのステータスの説明。
journey_send_status_code	リクエストの HTTP ステータスコード。

E メールイベント

メールメッセージを送信すると、Amazon Pinpoint はそれらのメッセージに対応して、以下のタイプのイベントに関する追加情報を提供するデータをストリーミングできます。

- 送信数
- 配信数
- バウンス
- 苦情
- 開封数
- クリック数
- 拒否数
- サブスクリプション解除数
- レンダリングの失敗

前述のリストのイベントタイプについては、「E メールイベント属性」で詳しく説明します。

E メールメッセージの送信に使用する API および設定によっては、追加のイベントタイプや異なるデータが表示される場合があります。例えば、Amazon Simple Email Service (Amazon SES) が提供する設定セットなど、Amazon Kinesis にイベントデータを発行する設定セットを使用してメッセージを送信する場合、データにはテンプレートのレンダリングの失敗に関するイベントが含まれることもあります。データの詳細については、『Amazon Simple Email Service デベロッパーガイド』の「Amazon SES イベント発行を使用したモニタリング」を参照してください。

イベント例

E メール送信

E メール送信イベントの JSON オブジェクトには以下の例に示されているデータが含まれています。

```
{
  "event_type": "_email.send",
  "event_timestamp": 1564618621380,
  "arrival_timestamp": 1564618622025,
  "event_version": "3.1",
  "application": {
    "app_id": "a1b2c3d4e5f6g7h8i9j0k1l2m3n4o5p6",
    "sdk": {}
  },
  "client": {
    "client_id": "9a311b17-6f8e-4093-be61-4d0bbexample"
  },
  "device": {
    "platform": {}
  },
  "session": {},
  "attributes": {
    "feedback": "received"
  },
  "awsAccountId": "123456789012",
  "facets": {
    "email_channel": {
      "mail_event": {
        "mail": {
          "message_id": "0200000073rnbmd1-mbvdg3uo-q8ia-m3ku-ibd3-ms77kexample-000000",
          "message_send_timestamp": 1564618621380,
          "from_address": "sender@example.com",
          "destination": ["recipient@example.com"],
```

```
          "headers_truncated": false,
          "headers": [{
            "name": "From",
            "value": "sender@example.com"
          }, {
            "name": "To",
            "value": "recipient@example.com"
          }, {
            "name": "Subject",
            "value": "Amazon Pinpoint Test"
          }, {
            "name": "MIME-Version",
            "value": "1.0"
          }, {
            "name": "Content-Type",
            "value": "multipart/alternative;  boundary=\"----=_Part_314159_271828\""
          }],
          "common_headers": {
            "from": "sender@example.com",
            "to": ["recipient@example.com"],
            "subject": "Amazon Pinpoint Test"
          }
        },
        "send": {}
      }
    }
  }
}
```

E メール配信済み

E メール配信イベントの JSON オブジェクトには以下の例に示されているデータが含まれています。

```
{
  "event_type": "_email.delivered",
  "event_timestamp": 1564618621380,
  "arrival_timestamp": 1564618622690,
  "event_version": "3.1",
  "application": {
    "app_id": "a1b2c3d4e5f6g7h8i9j0k1l2m3n4o5p6",
    "sdk": {}
  },
```

```
  "client": {
    "client_id": "e9a3000d-daa2-40dc-ac47-1cd34example"
  },
  "device": {
    "platform": {}
  },
  "session": {},
  "attributes": {
    "feedback": "delivered"
  },
  "awsAccountId": "123456789012",
  "facets": {
    "email_channel": {
      "mail_event": {
        "mail": {
          "message_id": "0200000073rnbmd1-mbvdg3uo-q8ia-m3ku-ibd3-ms77kexample-000000",
          "message_send_timestamp": 1564618621380,
          "from_address": "sender@example.com",
          "destination": ["recipient@example.com"],
          "headers_truncated": false,
          "headers": [{
            "name": "From",
            "value": "sender@example.com"
          }, {
            "name": "To",
            "value": "recipient@example.com"
          }, {
            "name": "Subject",
            "value": "Amazon Pinpoint Test"
          }, {
            "name": "MIME-Version",
            "value": "1.0"
          }, {
            "name": "Content-Type",
            "value": "multipart/alternative;  boundary=\"----=_Part_314159_271828\""
          }],
          "common_headers": {
            "from": "sender@example.com",
            "to": ["recipient@example.com"],
            "subject": "Amazon Pinpoint Test"
          }
        },
        "delivery": {
          "smtp_response": "250 ok:  Message 82080542 accepted",
```

```
          "reporting_mta": "a8-53.smtp-out.amazonses.com",
          "recipients": ["recipient@example.com"],
          "processing_time_millis": 1310
        }
      }
    }
  }
}
```

E メールクリック

E メールクリックイベントの JSON オブジェクトには以下の例に示されているデータが含まれています。

```
{
  "event_type": "_email.click",
  "event_timestamp": 1564618621380,
  "arrival_timestamp": 1564618713751,
  "event_version": "3.1",
  "application": {
    "app_id": "a1b2c3d4e5f6g7h8i9j0k1l2m3n4o5p6",
    "sdk": {}
  },
  "client": {
    "client_id": "49c1413e-a69c-46dc-b1c4-6470eexample"
  },
  "device": {
    "platform": {}
  },
  "session": {},
  "attributes": {
    "feedback": "https://aws.amazon.com/pinpoint/"
  },
  "awsAccountId": "123456789012",
  "facets": {
    "email_channel": {
      "mail_event": {
        "mail": {
          "message_id": "0200000073rnbmd1-mbvdg3uo-q8ia-m3ku-ibd3-ms77kexample-000000",
          "message_send_timestamp": 1564618621380,
          "from_address": "sender@example.com",
          "destination": ["recipient@example.com"],
          "headers_truncated": false,
```

```
          "headers": [{
            "name": "From",
            "value": "sender@example.com"
          }, {
            "name": "To",
            "value": "recipient@example.com"
          }, {
            "name": "Subject",
            "value": "Amazon Pinpoint Test"
          }, {
            "name": "MIME-Version",
            "value": "1.0"
          }, {
            "name": "Content-Type",
            "value": "multipart/alternative;  boundary=\"----=_Part_314159_271828\""
          }, {
            "name": "Message-ID",
            "value": "null"
          }],
          "common_headers": {
            "from": "sender@example.com",
            "to": ["recipient@example.com"],
            "subject": "Amazon Pinpoint Test"
          }
        },
        "click": {
          "ip_address": "72.21.198.67",
          "user_agent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6)
 AppleWebKit/605.1.15 (KHTML, like Gecko) Version/12.1.2 Safari/605.1.15",
          "link": "https://aws.amazon.com/pinpoint/"
        }
      }
    }
  }
}
```

E メール開封

E メール開封イベントの JSON オブジェクトには以下の例に示されているデータが含まれています。

```
{
  "event_type": "_email.open",
```

```
  "event_timestamp": 1564618621380,
  "arrival_timestamp": 1564618712316,
  "event_version": "3.1",
  "application": {
    "app_id": "a1b2c3d4e5f6g7h8i9j0k1l2m3n4o5p6",
    "sdk": {}
  },
  "client": {
    "client_id": "8dc1f651-b3ec-46fc-9b67-2a050example"
  },
  "device": {
    "platform": {}
  },
  "session": {},
  "attributes": {
    "feedback": "opened"
  },
  "awsAccountId": "123456789012",
  "facets": {
    "email_channel": {
      "mail_event": {
        "mail": {
          "message_id": "0200000073rnbmd1-mbvdg3uo-q8ia-m3ku-ibd3-ms77kexample-000000",
          "message_send_timestamp": 1564618621380,
          "from_address": "sender@example.com",
          "destination": ["recipient@example.com"],
          "headers_truncated": false,
          "headers": [{
            "name": "From",
            "value": "sender@example.com"
          }, {
            "name": "To",
            "value": "recipient@example.com"
          }, {
            "name": "Subject",
            "value": "Amazon Pinpoint Test"
          }, {
            "name": "MIME-Version",
            "value": "1.0"
          }, {
            "name": "Content-Type",
            "value": "multipart/alternative;  boundary=\"----=_Part_314159_271828\""
          }, {
            "name": "Message-ID",
```

```
            "value": "null"
          }],
          "common_headers": {
            "from": "sender@example.com",
            "to": ["recipient@example.com"],
            "subject": "Amazon Pinpoint Test"
          }
        },
        "open": {
          "ip_address": "72.21.198.67",
          "user_agent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6)
 AppleWebKit/605.1.15 (KHTML, like Gecko)"
        }
      }
    }
  }
}
```

E メールイベント属性

このセクションでは、E メールメッセージの送信時に Amazon Pinpoint が生成するイベントストリームデータに含まれる属性を定義します。

属性	説明
event_type	イベントのタイプ。可能な値は以下のとおりです。 • _email.send – Amazon Pinpoint はメッセージを受け入れて、受信者に配信しようとしました。 • _email.delivered – メッセージが受信者に配信されました。 • _email.rejected – Amazon Pinpoint は、メッセージにマルウェアが含まれていると判断し、送信を試みませんでした。 • _email.hardbounce – 恒久的な問題により、Amazon Pinpoint がメッセージを配信でき

属性	説明
	ませんでした。Amazon Pinpoint は再度メッセージを配信しようとはしません。 • _email.softbounce – 一時的な問題により、Amazon Pinpoint はメッセージを配信することができませんでした。Amazon Pinpoint は、一定時間、再度配信を試みます。それでも配信できない場合、それ以上の再試行は行いません。このとき、メールの最終的なステータスは SOFTBAUNCE になります。 • _email.complaint — 受信者がメッセージを受信し、そのメッセージをスパムとしてメールプロバイダーに報告しました (例えば、メールクライアントの「スパム報告」機能を使用した場合)。 • _email.open – 受信者がメッセージを受信して開封しました。 • _email.click – 受信者がメッセージを受信し、メッセージ内のリンクをクリックしました。 • _email.click – 受信者がメッセージを受信し、メッセージ内のリンクをクリックしました。 • email.rendering_failure — レンダリングが失敗したため E メールが送信されませんでした。これは、テンプレートデータが見つからない場合や、テンプレートのパラメータとデータが一致しない場合に発生します。
`event_timestamp`	メッセージが送信された時刻。Unix 時間 (ミリ秒単位) として表示されます。この値は通常、メッセージに対して生成されるすべてのイベントで同じです。
`arrival_timestamp`	イベントが Amazon Pinpoint によって受信された時刻が、Unix 時間 (ミリ秒単位) として表示されます。

属性	説明
event_version	イベントの JSON スキーマのバージョン。 **Tip** イベント処理アプリケーションでこのバージョンをチェックし、スキーマの更新に合わせてアプリケーションを更新する時期を把握します。
application	イベントに関連付けられた Amazon Pinpoint プロジェクトに関する情報。詳細については、表「アプリケーション」を参照してください。
client	イベントをレポートしたデバイスにインストールされているアプリクライアントに関する情報。詳細については、表「クライアント」を参照してください。
device	イベントを報告したデバイスに関する情報。詳細については、表「デバイス」を参照してください。 E メールイベントの場合、このオブジェクトは空です。
session	E メールイベントの場合、このオブジェクトは空です。

属性	説明
attributes	イベントに関連付けられている属性。詳細については、表「属性」を参照してください。 アプリケーションの 1 つによって報告されるイベントの場合、このオブジェクトにはアプリケーションによって定義されたカスタム属性を含めることができます。キャンペーンまたはジャーニーからメッセージを送信するときに作成されるイベントの場合、このオブジェクトにはキャンペーンまたはジャーニーに関連付けられた属性が含まれます。トランザクションメッセージを送信するときに生成されるイベントの場合、このオブジェクトにはメッセージ自体に関連する情報が含まれます。
client_context	E メールイベントの場合、このオブジェクトには legacy_identifier 属性を格納した custom オブジェクトが含まれています。legacy_identifier 属性の値は、メッセージの送信元のプロジェクトの ID です。
facets	E メールヘッダーなど、メッセージに関する追加情報。詳細については、表「ファセット」を参照してください。
awsAccountId	メッセージの送信に使用された AWS アカウントの ID。

アプリケーション

イベントが関連付けられている Amazon Pinpoint プロジェクトに関する情報が含まれています。

属性	説明
app_id	イベントを報告した Amazon Pinpoint プロジェクトの一意の ID。
sdk	イベントを報告するために使用された SDK。Amazon Pinpoint API を直接呼び出すか、Amazon Pinpoint コンソールを使用してトランザクション E メールメッセージを送信する場合、このオブジェクトは空です。

属性

イベントを生成したキャンペーンまたはジャーニーに関する情報が含まれています。

Campaign

イベントを生成したキャンペーンに関する情報が含まれています。

属性	説明
feedback	_email.click イベントの場合、この属性の値は、受取人がイベントを生成するためにメッセージ内でクリックしたリンクの URL です。他のイベントの場合、この値はイベントタイプ (received、opened、clicked など) を表します。
treatment_id	メッセージが A/B テストキャンペーンを使用して送信された場合、この値はメッセージの処理番号を表します。標準キャンペーンおよびトランザクション E メールメッセージの場合、この値は 0 です。
campaign_activity_id	イベントが発生したときに Amazon Pinpoint が生成する一意の ID。

属性	説明
campaign_id	メッセージを送信した キャンペーンの一意の ID。

ジャーニー

イベントを生成したジャーニーに関する情報が含まれています。

属性	説明
journey_run_id	メッセージを送信したジャーニーの一意の ID。Amazon Pinpoint はジャーニーの新しい実行ごとに自動的にこの ID を生成し、割り当てます。
feedback	_email.click イベントの場合、この属性の値は、受取人がイベントを生成するためにメッセージ内でクリックしたリンクの URL です。他のイベントの場合、この値はイベントタイプ (received、delivered 、opened など) を表します。
journey_id	メッセージを送信したジャーニーの一意の ID。
journey_activity_id	メッセージを送信したジャーニーアクティビティの一意の ID。

クライアント

キャンペーンまたはジャーニーのターゲットとなったクライアントの一意の識別子。

属性	説明
client_id	クライアントの ID。値はキャンペーンとジャーニーのエンドポイント ID で、トランザクション送信の場合は UUID です。

ファセット

メッセージおよびイベントタイプに関する情報が含まれています。

属性	説明
email_channel	mail_event オブジェクトが含まれています。このオブジェクトは、2 つのオブジェクト (mail、およびイベントタイプに対応するオブジェクト) を格納しています。

Mail

E メールメッセージのコンテンツに関する情報と、メッセージに関するメタデータが含まれています。

属性	説明
message_id	メッセージの ID。Amazon Pinpoint は、メッセージを受け入れる際に、このIDを自動的に生成します。
message_send_timestamp	メッセージが送信された日付と時刻。RFC 822 で指定された形式で示されます。
from_address	メッセージの送信元の E メールアドレス。
destination	メッセージの送信先の E メールアドレスを含む配列。

属性	説明
headers_truncated	E メールヘッダーが切り捨てられたかどうかを示すブール値。
headers	メッセージのヘッダーに対応する複数の名前と値のペアを含むオブジェクト。このオブジェクトには通常、次のヘッダーに関する情報が含まれています。 • From – 送信者の E メールアドレス。 • To – 受信者の E メールアドレス。 • Subject – E メールの件名。 Tip キャンペーンの _email.send イベントには Subject ヘッダーは含まれません。 • MIME-Version – メッセージが MIME 形式であることを示します。このヘッダーが存在する場合、値は常に 1.0 です。 • Content-Type – メッセージコンテンツの MIME メディアタイプ。
common_headers	E メールメッセージの複数の共通ヘッダーに関する情報が含まれています。情報には、メッセージが送信された日付、メッセージの宛先、送信元、および件名が含まれます。

SMS イベント

SMS チャンネルがプロジェクトに対して有効になっている場合、Amazon Pinpoint はプロジェクトの SMS メッセージ配信に関するイベントデータをストリーミングできます。通信事業者が生成した SMS イベントは、受信されるまでに最大 72 時間かかることがあるため、送信メッセージの配

信に遅延があるかどうかを判断するのには使用しないでください。72 時間が経過しても、Amazon Pinpoint が通信事業者から最終イベントを受信していない場合、そのメッセージに何が起こったのかわからないため、サービスは自動的に UNKNOWN のrecord_status を返します。

例

SMS イベントの JSON オブジェクトには以下の例に示されているデータが含まれています。

```
{
  "event_type": "_SMS.SUCCESS",
  "event_timestamp": 1553104954322,
  "arrival_timestamp": 1553104954064,
  "event_version": "3.1",
  "application": {
    "app_id": "a1b2c3d4e5f6g7h8i9j0k1l2m3n4o5p6",
    "sdk": {}
  },
  "client": {
    "client_id": "123456789012"
  },
  "device": {
    "platform": {}
  },
  "session": {},
  "attributes": {
    "sender_request_id": "565d4425-4b3a-11e9-b0a5-example",
    "campaign_activity_id": "cbcfc3c5e3bd48a8ae2b9cb41example",
    "origination_phone_number": "+12065550142",
    "destination_phone_number": "+14255550199",
    "record_status": "DELIVERED",
    "iso_country_code": "US",
    "treatment_id": "0",
    "number_of_message_parts": "1",
    "message_id": "1111-2222-3333",
    "message_type": "Transactional",
    "campaign_id": "52dc44b35c4742c98c5935269example"
  },
  "metrics": {
    "price_in_millicents_usd": 645.0
  },
  "awsAccountId": "123456789012"
}
```

SMS イベントの属性

このセクションでは、SMS メッセージの送信時に Amazon Pinpoint が生成するイベントストリームデータに含まれる属性を定義します。

イベント

属性	説明
`event_type`	イベントのタイプ。可能な値は以下のとおりです。 • _SMS.BUFFERED – メッセージはまだ受信者に配信処理中です。 • _SMS.SUCCESS – メッセージが受信者に正常に配信されました。 • _SMS.FAILURE – Amazon Pinpoint がメッセージを受信者に配信できませんでした。メッセージを配信できない原因となったエラーの詳細については、「`attributes.record_status`」を参照してください。 • _SMS.OPTOUT – 顧客がメッセージを受信し、オプトアウトキーワード (通常は「STOP」) を送信することで返信しました。
`event_timestamp`	イベントが報告された時刻。Unix 時間 (ミリ秒単位) として表示されます。
`arrival_timestamp`	イベントが Amazon Pinpoint によって受信された時刻が、Unix 時間 (ミリ秒単位) として表示されます。
`event_version`	イベントの JSON スキーマのバージョン。

属性	説明
	Tip イベント処理アプリケーションでこのバージョンをチェックし、スキーマの更新に合わせてアプリケーションを更新する時期を把握します。
`application`	イベントに関連付けられた Amazon Pinpoint プロジェクトに関する情報。詳細については、表「アプリケーション」を参照してください。
`client`	イベントをレポートしたデバイスにインストールされているアプリクライアントに関する情報。詳細については、表「クライアント」を参照してください。
`device`	イベントを報告したデバイスに関する情報。詳細については、表「デバイス」を参照してください。 SMS イベントの場合、このオブジェクトは空です。
`session`	SMS イベントの場合、このオブジェクトは空です。

属性	説明
attributes	イベントに関連付けられている属性。アプリケーションの 1 つによって報告されるイベントの場合、このオブジェクトにはアプリケーションによって定義されたカスタム属性を含めることができます。キャンペーンを送信したときに作成されたイベントの場合、このオブジェクトにはキャンペーンに関連付けられた属性が含まれています。トランザクションメッセージを送信するときに生成されるイベントの場合、このオブジェクトにはメッセージ自体に関連する情報が含まれます。 詳細については、表「属性」を参照してください。
metrics	イベントに関連付けられている追加のメトリクス。詳細については、表「メトリクス」を参照してください。
awsAccountId	メッセージの送信に使用された AWS アカウントの ID。

アプリケーション

イベントが関連付けられている Amazon Pinpoint プロジェクトと、イベントを報告するために使用された SDK (該当する場合) に関する情報が含まれています。

属性	説明
app_id	イベントを報告した Amazon Pinpoint プロジェクトの一意の ID。
sdk	イベントを報告するために使用された SDK。Amazon Pinpoint API を直接呼び出すか、Amazon Pinpoint コンソールを使用してト

属性	説明
	ランザクション SMS メッセージを送信する場合、このオブジェクトは空です。

属性

イベントに関連付けられている属性に関する情報が含まれています。

属性	説明
`sender_request_id`	SMS メッセージを送信するリクエストに関連付けられている一意の ID。
`campaign_activity_id`	キャンペーン内のアクティビティの一意の ID。
`origination_phone_number`	メッセージの送信元の電話番号。
`destination_phone_number`	メッセージを送信しようとした電話番号。
`record_status`	メッセージのステータスに関する追加情報。可能な値は以下のとおりです。 • SUCCESSFUL/DELIVERED – メッセージが正常に配信されました。 • PENDING – メッセージはまだ受信者のデバイスに配信されていません。 • INVALID – 送信先の電話番号が無効です。 • UNREACHABLE – 受信者のデバイスが現在到達できないか、利用できません。例えば、デバイスの電源がオフになっているか、ネットワークから切断されている可能性があります。後でメッセージの送信を再試行できます。 • UNKNOWN – メッセージの配信を妨げるエラーが発生しました。このエラーは通常一時

属性	説明
	的なものであるため、後でもう一度メッセージを送信できます。 • BLOCKED – 受信者のデバイスが発信元番号からの SMS メッセージをブロックしています。 • CARRIER_UNREACHABLE – 受信者のモバイルネットワークの問題により、メッセージを配信できませんでした。このエラーは通常一時的なものであるため、後でもう一度メッセージを送信できます。 • SPAM – 受信者のモバイルキャリアがメッセージのコンテンツをスパムとして識別し、メッセージの配信をブロックしました。 • INVALID_MESSAGE – SMS メッセージの本文が無効であるため、配信できません。 • CARRIER_BLOCKED – 受信者のキャリアがこのメッセージの配信をブロックしています。これは、多くの場合、キャリアがメッセージの内容を未承諾または悪意のあるものとして識別した場合に発生します。 • TTL_EXPIRED – 特定の期間内に SMS メッセージを配信できませんでした。このエラーは通常一時的なものであるため、後でもう一度メッセージを送信できます。 • MAX_PRICE_EXCEEDED – メッセージを送信すると、アカウントの月間の SMS 使用クォータを超えた課金が発生しました。Amazon Pinpoint ユーザーガイドの毎月の SMS クォータの増額を申請するの手続きを行うことで、この枠の増額を申請することができます。

属性	説明
	• OPTED_OUT – 受信者があなたからのメッセージの受信をオプトアウトしたため、SMS メッセージは送信されませんでした。 • NO_QUOTA_LEFT_ON_ACCOUNT – メッセージを送信するのに十分な使用クォータがアカウントに残っていません。Amazon Pinpoint ユーザーガイドの毎月の SMS クォータの増額を申請するの手続きを行うことで、この枠の増額を申請することができます。 • NO_ORIGINATION_IDENTITY_AVAILABLE_TO_SEND – アカウントには、宛先へのメッセージの送信に使用できる電話番号が含まれていません。 • DESTINATION_COUNTRY_NOT_SUPPORTED – 送信先の国がブロックされています。サポートされている国については、「Supported countries and regions (SMS channel)」を参照してください。 • ACCOUNT_IN_SANDBOX – アカウントはサンドボックス内にあり、検証済みの宛先番号にのみ送信できます。Amazon Pinpoint コンソールで宛先番号を検証するか、アカウントをサンドボックス外に移動するプロセスを開始できます。「Moving from the Amazon Pinpoint SMS sandbox to production」を参照してください。 • RATE_EXCEEDED – メッセージの送信が速すぎたため、スロットリングされました。コールレートを下げる必要があります。制限の詳細については、「Message Parts per Second (MPS) limits」を参照してください。 • INVALID_ORIGINATION_IDENTITY – 指定された送信元 ID は無効です。

属性	説明
	• ORIGINATION_IDENTITY_DOES_NOT_EXIST – 指定された送信元 ID は存在しません。 • INVALID_DLT_PARAMETERS – 無効な DLT パラメータ (インドの宛先には必須) が指定されました。 • INVALID_PARAMETERS – 無効なパラメータが提供されました。 • ACCESS_DENIED – アカウントによるメッセージの送信はブロックされています。カスタマーサポートに連絡して原因を突き止め、問題を解決してください。 • INVALID_KEYWORD – 指定されたキーワードは無効です。キーワードの形式が間違っているか、キーワードがアカウントで設定されていない可能性があります。 • INVALID_SENDER_ID – 指定された送信者 ID は無効です。送信者 ID の形式または長さが正しくない可能性があります。 • INVALID_POOL_ID – 指定されたプール ID は無効です。プール ID の形式が間違っているか、プール ID がアカウントに属していない可能性があります。 • SENDER_ID_NOT_SUPPORTED_FOR_DESTINATION – 送信先の国は送信者 ID をサポートしていません。送信には、電話番号または別の送信元 ID を使用する必要があります。 • INVALID_PHONE_NUMBER – 指定された発信元の電話番号は無効です。電話番号の形式または長さが正しくない可能性があります。

属性	説明
iso_country_code	受信者の電話番号に関連付けられている国 (ISO 3166-1 alpha-2 形式で表示)。
treatment_id	メッセージが A/B キャンペーンで送信された場合のメッセージ処理の ID。
treatment_id	メッセージが A/B テストキャンペーンを使用して送信された場合、この値はメッセージの処理番号を表します。トランザクション SMS メッセージの場合、この値は 0 です。
number_of_message_parts	メッセージを送信するために Amazon Pinpoint が作成したメッセージパートの数。 通常、SMS メッセージには GSM-7 文字 160 文字または非 GSM 文字 67 文字のみを含めることができますが、これらの制限は国によって異なる場合があります。これらの制限を超えるメッセージを送信すると、Amazon Pinpoint は自動的にメッセージを小さなパートに分割します。送信したメッセージパートの数に基づいて請求されます。
message_id	メッセージを受け入れるときに Amazon Pinpoint が生成する一意の ID。
message_type	メッセージのタイプ。指定できる値は、Promotional および Transactionalです。この値は、キャンペーンの作成時、または Amazon Pinpoint API の SendMessages オペレーションを使用してトランザクションメッセージを送信する際に指定します。
campaign_id	メッセージを送信した Amazon Pinpoint キャンペーンの一意の ID。

クライアント

イベントをレポートしたデバイスにインストールされているアプリクライアントに関する情報が含まれています。

属性	説明
client_id	アプリによって生成されたイベントの場合、この値はデバイスにインストールされたアプリクライアントの一意の ID です。この ID は、AWS Mobile SDK for iOS および AWS Mobile SDK for Android で自動的に生成されます。 キャンペーンやトランザクションメッセージの送信時に生成されるイベントの場合、この値はメッセージの送信先のエンドポイントの ID と同じです。
cognito_id	アプリケーションが使用する Amazon Cognito ID プールのアプリクライアントに割り当てられた一意の ID。

デバイス

イベントを報告したデバイスに関する情報が含まれています。

属性	説明
locale	デバイスロケール。
make	デバイスのメーカー (Apple、Samsung など)。
model	デバイスモデル (iPhone など)。
platform	デバイスのプラットフォーム (ios、android など)。

メトリクス

イベントに関連付けられているメトリクスに関する情報が含まれています。

属性	説明
price_in_millicents_usd	メッセージの送信に課金された金額。この価格は、米ドルセントの 1000 分の 1 で示されます。例えば、この属性の値が 645 の場合、メッセージの送信に 0.645¢ が課金されます (645 / 1000 = 0.645¢ = $0.00645)。 **Note** このプロパティは、event_type が _SMS.BUFFERED のメッセージには表示されません。

Amazon Pinpoint 分析データのクエリを実行する

Amazon Pinpoint コンソールの分析ページの使用に加えて、 Analytics API を使用して、ユーザーエンゲージメント、キャンペーンアウトリーチなどに関連する傾向についてのインサイトを提供する、標準メトリクスのサブセットの分析データをクエリします。重要業績評価指標 (KPI) とも呼ばれるこれらのメトリクスは、プロジェクト、キャンペーン、およびジャーニーのパフォーマンスを監視および評価する際に役立つ測定可能な値です。

API を使用して分析データのクエリを実行する場合、お客様が選択したレポート作成ツールを使用してデータを分析することができます。その場合、Amazon Pinpoint コンソールにサインインしたり、Amazon Kinesis Streams などのソースからの未加工のイベントデータを分析したりすることが不要になります。例えば、毎週のキャンペーン結果の表示またはキャンペーンの配信率に関する詳細分析を提供する、カスタムダッシュボードを作成することができます。

Amazon Pinpoint REST API、 AWS Command Line Interface （ AWS CLI ）、または AWS SDK を使用してデータをクエリできます。データのクエリは、Amazon Pinpoint API にリクエストを送信し、サポートされているパラメータを使用して、必要なデータと適用したいフィルターを指定します。クエリを送信すると、Amazon Pinpoint はクエリ結果を JSON レスポンスで返します。その後、結果を別のサービスまたはアプリケーションに渡して、より詳細な分析、保存、またはレポートを作成できます。

サポートされるメトリクス

Amazon Pinpoint では、次の数種類の標準メトリクスの分析データに、プログラムによりアクセスすることができます。

- アプリケーションメトリクス – これらのメトリクスは、プロジェクトと関連付けられているすべてのキャンペーンおよびトランザクションメッセージの傾向についてのインサイトを提供します。例えば、アプリケーションメトリクスを使用して、プロジェクトに関連付けられている各キャンペーンについて受信者によって開かれたメッセージの数の内訳を入手できます。これらのメトリクスのデータにアクセスするには、Amazon Pinpoint API のアプリケーションメトリクスリソースを使用します。
- キャンペーンメトリクス – これらのメトリクスは、個々のキャンペーンのパフォーマンスについてのインサイトを提供します。例えば、キャンペーンメトリクスを使用して、キャンペーンメッセージが送信されたエンドポイントの数を確定できます。これらのメトリクスのデータにアクセスするには、Amazon Pinpoint API のキャンペーンメトリクスのリソースを使用します。

- ジャーニーエンゲージメントメトリクス – これらのメトリクスは、個々のジャーニーのパフォーマンスに関するインサイトを提供します。例えば、ジャーニーのエンゲージメントメトリクスを使用して、ジャーニーの各アクティビティで参加者が開封したメッセージ数の内訳を取得できます。これらのメトリクスのデータにアクセスするには、Amazon Pinpoint API のジャーニーエンゲージメントメトリクスのリソースを使用します。
- ジャーニー実行メトリクス – これらのメトリクスは、個々のジャーニーの参加傾向に関するインサイトを提供します。例えば、ジャーニー実行メトリクスを使用して、ジャーニー内のアクティビティを進めている参加者の数を判断できます。ジャーニー実行メトリクスのデータにアクセスするには、Amazon Pinpoint API のジャーニー実行メトリクスのリソースを使用します。
- ジャーニーアクティビティ実行メトリクス – これらのメトリクスは、ジャーニーにおける個々のアクティビティの参加傾向についてのインサイトを提供します。例えば、ジャーニーアクティビティ実行メトリクスを使用して、アクティビティを完了した参加者数を判断できます。ジャーニーアクティビティメトリクスのデータにアクセスするには、Amazon Pinpoint API のジャーニーアクティビティメトリクスのリソースを使用します。

プログラミングでクエリを実行できるすべての標準メトリクスについては、「標準メトリクス」を参照してください。

Amazon Pinpoint は、お客様のすべてのプロジェクト、キャンペーン、およびジャーニーにおいてサポートされているすべてのメトリクスのデータを、自動的に収集して集計します。さらに、データは継続的に更新されるため、データのレイテンシー期間は、約 2 時間に制限されています。ただし、特定のメトリクスに対して追加のデータレイテンシーが発生する場合があることに注意してください。これは、一部のメトリクスのデータが、受取人の E メールプロバイダーから受け取った情報に基づいているためです。プロバイダーによっては、この情報を直ちに当社に送信するものもあれば、あまり頻繁には送信しないものもあります。

Amazon Pinpoint は 90 日間データを保存します。90 日以上データを保存したり、未加工の分析データにリアルタイムでアクセスしたりするには、Amazon Pinpoint プロジェクトを設定して、イベントデータを Amazon Kinesis Data Streams または Amazon Data Firehose にストリーミングできます。イベントストリームの設定方法については、「Amazon Pinpoint のイベントを Kinesis にストリーミングする」を参照してください。

クエリのベーシック

メトリクスのデータのクエリを実行するには、Amazon Pinpoint API の適切なメトリクスリソースに `get` リクエストを送信します。リクエストでは、次のクエリコンポーネントでサポートされているパラメータを使用して、クエリを定義します。

- Project – `application-id` パラメータの値としてプロジェクト ID を入力して、プロジェクトを指定します。このパラメータは、すべてのメトリクスに必須です。
- Campaign – `campaign-id` パラメータの値としてキャンペーン ID を入力して、キャンペーンを指定します。このパラメータは、キャンペーンメトリクスにのみ必要です。
- Journey – `journey-id` パラメータの値としてジャーニー ID を指定して、ジャーニーを指定します。このパラメータは、ジャーニーのエンゲージメントと実行メトリクス、およびジャーニーアクティビティ実行メトリクスにのみ必要です。
- Journey activity – `journey-activity-id` パラメータの値としてジャーニーアクティビティ ID を指定して、ジャーニーアクティビティを指定します。このパラメータは、ジャーニーアクティビティ実行メトリクスにのみ必要です。
- Date range – データを日付範囲でフィルタリングするには、サポートされている開始時刻と終了時刻パラメータを使用して、日付範囲の最初と最後の日時を指定します。値は拡張 ISO 8601 形式で、協定世界時 (UTC) を使用する必要があります (例：`2019-07-19T20:00:00Z` は、2019年7月19日午後8時 (UTC))。

  日付範囲は包括的であり、31 日以内に制限する必要があります。また、最初の日付と時刻は、現在の日付から 90 日未満である必要があります。日付範囲を指定しない場合、Amazon Pinpoint は過去 31 暦日のデータを返します。日付範囲パラメータは、ジャーニー実行メトリクスおよびジャーニーアクティビティ実行メトリクスを除くすべてのメトリクスでサポートされます。
- Metric – `kpi-name` パラメータの値としてメトリクス名を提供することで、メトリクスを指定します。この値は、関連するメトリクスを記述し、ハイフンで区切られた小文字の英数字で構成される 2 つ以上の用語で構成されます。例は、`email-open-rate` および `successful-delivery-rate` です。このパラメータは、ジャーニー実行メトリクスおよびジャーニーアクティビティ実行メトリクスを除くすべてのメトリクスに必要です。サポートされているすべてのメトリクスと各メトリクスに使用する `kpi-name` 値については、「標準メトリクス」を参照してください。

クエリを送信すると、Amazon Pinpoint はクエリ結果を JSON レスポンスで返します。レスポンスにおける結果の構造は、クエリしたメトリクスによって異なります。

一部のメトリクスでは、例えばキャンペーンによって配信されたメッセージ数など、1 つの値 — のみが提供されます。その他のメトリクスは、関連するフィールド — によってグループ化された複数の値を提供します。例えば、キャンペーンの実行ごとに配信されたメッセージ数で、キャンペーンの実行ごとグループ化されたものなどが含まれます。メトリクスが複数の値を提供し、グループ化する場合、JSON レスポンスにはデータのグループ化に使用されたフィールドを示すためのフィールドが含まれます。クエリ結果の構造の詳細については、「クエリ結果の使用」を参照してください。

Amazon Pinpoint 分析データのクエリ実行用の IAM ポリシー

Amazon Pinpoint API を使用して、標準メトリクスのサブセットの分析データをクエリできます。重要業績評価指標標 (KPI) Amazon Pinpoint プロジェクト、キャンペーン、およびジャーニーに適用されます。標準メトリクスは、プロジェクト、キャンペーン、およびジャーニーのパフォーマンスを監視および評価する際に役立ちます。

データへのアクセスは、データのアクセス権を持つ ロールまたはユーザーの権限を定義するための AWS Identity and Access Management (IAM) ポリシーを作成することで、管理することができます。データアクセス時の管理をきめ細かくサポートできるように、Amazon Pinpoint には、IAM ポリシーで指定できる個別のアクションが提供されています。Amazon Pinpoint コンソールで分析データを表示するための特定のアクションが Amazon Pinpoint コンソール (`mobiletargeting:GetReports`) に提供されています。Amazon Pinpoint API を使用して、プログラムにより分析データにアクセスするためのアクションも提供されています。

分析データへのアクセスを管理する IAM ポリシーを作成するには、AWS Management Console、AWS CLI、または IAM API を使用します。なお、AWS Management Console の [Visual editor] タブには、現在 Amazon Pinpoint の分析データを表示したり、問い合わせたりするためのアクションは含まれていません。ただし、コンソールの [JSON] タブを使用して、必要なアクションを IAM ポリシーに手動で追加できます。

例えば、次のポリシーでは、すべての AWS リージョンにおける、お客様のすべてのプロジェクト、キャンペーン、およびジャーニーのすべての分析データに、プログラムによりアクセスすることができます。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "QueryAllAnalytics",
            "Effect": "Allow",
            "Action": [
                "mobiletargeting:GetApplicationDateRangeKpi",
                "mobiletargeting:GetCampaignDateRangeKpi",
                "mobiletargeting:GetJourneyDateRangeKpi",
                "mobiletargeting:GetJourneyExecutionMetrics",
                "mobiletargeting:GetJourneyExecutionActivityMetrics"
            ],
            "Resource": [
                "arn:aws:mobiletargeting:*:accountId:apps/*/kpis/*",
```

```
                "arn:aws:mobiletargeting:*:accountId:apps/*/campaigns/*/kpis/*",
                "arn:aws:mobiletargeting:*:accountId:apps/*/journeys/*/kpis/*",
                "arn:aws:mobiletargeting:*:accountId:apps/*/journeys/*/execution-
metrics",
                "arn:aws:mobiletargeting:*:accountId:apps/*/journeys/*/activities/*/
execution-metrics"
            ]
        }
    ]
}
```

accountId はお客様の AWS アカウント ID です。

ただしベストプラクティスとして、最小特権の原則に準拠したポリシーを作成してください。別の言い方をすると、特定タスクの実行にのみ必要とされる権限のみが含まれたポリシーを作成してください。権限の制限をサポートし、よりきめ細かい管理を行うために、プログラムによる分析データへのアクセスを、特定の AWS リージョンの特定のプロジェクトにのみに制限します。以下の例をご確認ください。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "QueryProjectAnalytics",
            "Effect": "Allow",
            "Action": [
                "mobiletargeting:GetApplicationDateRangeKpi",
                "mobiletargeting:GetCampaignDateRangeKpi",
                "mobiletargeting:GetJourneyDateRangeKpi",
                "mobiletargeting:GetJourneyExecutionMetrics",
                "mobiletargeting:GetJourneyExecutionActivityMetrics"
            ],
            "Resource": [
                "arn:aws:mobiletargeting:region:accountId:apps/projectId/kpis/*",
                "arn:aws:mobiletargeting:region:accountId:apps/projectId/campaigns/*/
kpis/*",
                "arn:aws:mobiletargeting:region:accountId:apps/projectId/journeys/*/
kpis/*",
                "arn:aws:mobiletargeting:region:accountId:apps/projectId/journeys/*/
execution-metrics",
                "arn:aws:mobiletargeting:region:accountId:apps/projectId/journeys/*/
activities/*/execution-metrics"
            ]
```

```
        }
    ]
}
```

実行する条件は以下のとおりです。

- *region* は、プロジェクトをホストする AWS リージョンの名前です。
- *accountId* は、お客様の AWS アカウント ID です。
- *projectId* は、アクセス権限を提供したいプロジェクトの識別子です。

同様に、次のポリシーの例は、特定のキャンペーンの分析データにのみ、プログラムによるアクセスを許可します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "QueryCampaignAnalytics",
            "Effect": "Allow",
            "Action": "mobiletargeting:GetCampaignDateRangeKpi",
            "Resource": "arn:aws:mobiletargeting:region:accountId:apps/projectId/
campaigns/campaignId/kpis/*"
        }
    ]
}
```

実行する条件は以下のとおりです。

- *region* は、プロジェクトをホストする AWS リージョンの名前です。
- *accountId*は、お客様の AWS アカウント ID です。
- *projectId* は、キャンペーンに関連付けられているプロジェクトの識別子です。
- *campaignId* は、アクセス権限を提供したいキャンペーンの識別子です。

また、次のポリシー例では、特定のジャーニーとそのジャーニーを構成するアクティビティについて、エンゲージメントデータと実行データの両方のすべての分析データへのプログラムによるアクセスを許可します。

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "QueryJourneyAnalytics",
            "Effect": "Allow",
            "Action": [
                "mobiletargeting:GetJourneyDateRangeKpi",
                "mobiletargeting:GetJourneyExecutionMetrics",
                "mobiletargeting:GetJourneyExecutionActivityMetrics"
            ],
            "Resource": [
                "arn:aws:mobiletargeting:region:accountId:apps/projectId/
journeys/journeyId/kpis/*",
                "arn:aws:mobiletargeting:region:accountId:apps/projectId/
journeys/journeyId/execution-metrics",
                "arn:aws:mobiletargeting:region:accountId:apps/projectId/
journeys/journeyId/activities/*/execution-metrics"
            ]
        }
    ]
}
```

実行する条件は以下のとおりです。

- *region* は、プロジェクトをホストする AWS リージョンの名前です。
- *accountId* は、お客様の AWS アカウント ID です。
- *projectId* は、ジャーニーに関連付けられているプロジェクトの識別子です。
- *journeyId* は、アクセス権限を提供したいジャーニーの識別子です。

IAM ポリシーで使用できる Amazon Pinpoint API アクションの全リストについては、IAM ポリシーの Amazon Pinpointアクション をご参照ください。IAM ポリシーの作成と管理の詳細については、『IAM ユーザーガイド』をご参照ください。

Amazon Pinpoint の標準的な分析メトリクス

Amazon Pinpoint Analytics API を使用して、Amazon Pinpoint プロジェクト、キャンペーン、およびジャーニーに適用される標準メトリクスのサブセットの分析データをクエリできます。これらのメトリクスは、主要業績評価指標 (KPI) とも呼ばれ、プロジェクト、キャンペーン、およびジャーニーの成果を監視および評価するのに役立つ測定可能な値です。

Amazon Pinpoint では、次の数種類の標準メトリクスの分析データに、プログラムによりアクセスすることができます。

- アプリケーションメトリクス – これらのメトリクスは、アプリケーションとも呼ばれる、プロジェクトと関連付けられているすべてのキャンペーンおよびトランザクションメッセージの傾向についてのインサイトを提供します。例えば、アプリケーションメトリクスを使用して、プロジェクトに関連付けられている各キャンペーンについて受信者によって開かれたメッセージの数の内訳を入手できます。
- キャンペーンメトリクス – これらのメトリクスは、個々のキャンペーンのパフォーマンスについてのインサイトを提供します。例えば、キャンペーンメトリクスを使用して、キャンペーンメッセージが送信されたエンドポイントの数、またはエンドポイントに配信されたメッセージの数を特定できます。
- ジャーニーエンゲージメントメトリクス – これらのメトリクスは、個々のジャーニーのパフォーマンスに関するインサイトを提供します。例えば、ジャーニーのエンゲージメントメトリクスを使用して、ジャーニーの各アクティビティで参加者が開封したメッセージ数の内訳を取得できます。
- ジャーニー実行メトリクス – これらのメトリクスは、個々のジャーニーの参加傾向に関するインサイトを提供します。例えば、ジャーニー実行メトリクスを使用して、ジャーニーを開始した参加者の数を判断できます。
- ジャーニーアクティビティ実行メトリクス – これらのメトリクスは、ジャーニーにおける個々のアクティビティの参加傾向についてのインサイトを提供します。例えば、ジャーニーアクティビティ実行メトリクスを使用して、アクティビティを開始した参加者数と、アクティビティの各パスを完了した参加者数を判断できます。

このセクションのトピックでは、メトリクスのタイプごとにクエリを実行できる個々のメトリクスをリストし、説明します。

トピック

- ジャーニーおよびキャンペーン実行メトリクス

キャンペーンのアプリケーションメトリクス

次の表に、Amazon Pinpoint プロジェクトに関連付けられているすべてのキャンペーンのパフォーマンスを評価するためにクエリできる標準アプリケーションメトリクスのリストと説明を示します。これらのメトリクスのデータをクエリするには、Amazon Pinpoint API のアプリケーションメトリクスリソースを使用します。表の kpi-name 列は、クエリの `kpi-name` パラメータに使用する値を示しています。

メトリクス	Kpi-name	説明
配信率	`successful-delivery-rate`	プロジェクトに関連付けられているすべてのキャンペーンで、受信者に配信されたメッセージの割合。 このメトリクスは、プロジェクトのすべてのキャンペーンによって送信され、受信者に配信されたメッセージ数を、それらすべてのキャンペーンによって送信されたメッセージ数で割ったものとして計算されます。
日付別にグループ化された配信率	`successful-delivery-rate-grouped-by-date`	プロジェクトに関連付けられているすべてのキャンペーンについて、指定された日付範囲内で毎日、受信者に配信されたメッセージの割合。 このメトリクスは、プロジェクトのすべてのキャンペーンによって送信され、受信者に配信されたメッセージの数を、指定された日付範囲内

メトリクス	Kpi-name	説明
		の日ごとのそれらのすべてのキャンペーンによって送信されたメッセージの数で割ったものとして計算されます。 このメトリクスのクエリの結果は、拡張 ISO 8601 形式で、暦日ごとにグループ化されています。
E メールのオープン率	`email-open-rate`	プロジェクトに関連付けられているすべてのキャンペーンについて、受信者によって開かれた E メールメッセージの割合。 このメトリクスは、プロジェクトのすべてのキャンペーンによって送信され、受信者に配信された E メールメッセージの数を、それらすべてのキャンペーンによって送信された E メールメッセージの数で割ったものとして計算されます。

メトリクス	Kpi-name	説明
キャンペーンでグループ化された E メールのオープン率	`email-open-rate-grouped-by-campaign`	プロジェクトに関連付けられている各キャンペーンについて、受信者によって開かれた E メールメッセージの割合。 このメトリクスは、プロジェクトのすべてのキャンペーンによって送信され、受信者によって開かれた E メールメッセージの数を、キャンペーンによって送信され、受信者に配信された E メールメッセージの数で割ったものとして計算されます。 このメトリクスのクエリ結果はキャンペーン ID (`CampaignId`) でグループ化されますが、これはキャンペーンを一意に識別する文字列です。
エンドポイントの配信	`unique-deliveries`	プロジェクトに関連付けられているすべてのキャンペーンについて、メッセージが配信された一意のエンドポイントの数。

メトリクス	Kpi-name	説明
キャンペーンでグループ化された配信エンドポイント	`unique-deliveries-grouped-by-campaign`	プロジェクトに関連付けられている各キャンペーンについて、メッセージが配信された一意のエンドポイントの数 このメトリクスのクエリ結果はキャンペーン ID (`CampaignId`) でグループ化されますが、これはキャンペーンを一意に識別する文字列です。
日付別にグループ化されたされた配信エンドポイント	`unique-deliveries-grouped-by-date`	プロジェクトに関連付けられているすべてのキャンペーンについて、指定された日付範囲の各日にメッセージが配信された一意のエンドポイントの数。 このメトリクスのクエリの結果は、拡張 ISO 8601 形式で、暦日ごとにグループ化されています。

メトリクス	Kpi-name	説明
キャンペーンでグループ化された配信されたメッセージ	`successful-deliveries-grouped-by-campaign`	プロジェクトに関連付けられている各キャンペーンについて、受信者受信者に配信されたメッセージの数。 このメトリクスは、キャンペーンによって送信されたメッセージの数から、キャンペーンによって送信され、ハードバウンスのために受信者に配信できなかったメッセージの数を引いたものとして計算されます。 このメトリクスのクエリ結果はキャンペーン ID (`CampaignId`) でグループ化されますが、これはキャンペーンを一意に識別する文字列です。

メトリクス	Kpi-name	説明
プッシュオープン率	`push-open-rate`	プロジェクトに関連付けられているすべてのキャンペーンについて、受信者によって開かれたプッシュ通知の割合です。 このメトリクスは、プロジェクトのすべてのキャンペーンによって送信され、受信者によって開かれたプッシュ通知の数を、それらすべてのキャンペーンによって送信され、受信者に配信されたプッシュ通知の数で割ったものとして計算されます。

メトリクス	Kpi-name	説明
キャンペーンでグループ化されたプッシュオープン率	`push-open-rate-grouped-by-campaign`	プロジェクトに関連付けられている各キャンペーンについて、受信者によって開かれたプッシュ通知の割合です。 このメトリクスは、キャンペーンによって送信され、受信者によって開かれたプッシュ通知の数を、キャンペーンによって送信され、受信者に配信されたプッシュ通知の数で割ったものとして計算されます。 このメトリクスのクエリ結果はキャンペーン ID (`CampaignId`) でグループ化されますが、これはキャンペーンを一意に識別する文字列です。

トランザクション E メールメッセージのアプリケーションメトリクス

次の表に、Amazon Pinpoint プロジェクトに関連付けられているすべてのトランザクション E メールメッセージの傾向をモニタリングするためにクエリできる標準アプリケーションメトリクスのリストと説明を示します。これらのメトリクスのデータをクエリするには、Amazon Pinpoint API のアプリケーションメトリクスリソースを使用します。表の kpi-name 列は、クエリの `kpi-name` パラメータに使用する値を示しています。

これらのメトリクスは、キャンペーンによって送信された E メールメッセージに関するデータを提供しないことに注意してください。トランザクション E メールメッセージに関するデータのみを提供します。1 つ以上のキャンペーンによって送信されたメッセージのデータをクエリするには、キャンペーンのキャンペーンメトリクスまたはアプリケーションメトリクスを使用します。

メトリクス	Kpi-name	説明
クリック数	`txn-emails-clicked`	受信者がメッセージのリンクをクリックした回数。1 人の受信者がメッセージ内の複数のリンクをクリックするか、同じリンクを複数回クリックした場合は、それぞれのクリックはカウントされます。
日付別にグループ化されたクリック数	`txn-emails-clicked-grouped-by-date`	指定された日付範囲の各日に受信者がメッセージのリンクをクリックした回数。1 人の受信者がメッセージ内の複数のリンクをクリックするか、同じリンクを複数回クリックした場合は、それぞれのクリックはカウントされます。 このメトリクスのクエリの結果は、拡張 ISO 8601 形式で、暦日ごとにグループ化されています。
Complaint rate	`txn-emails-complaint-rate`	受信者によって未承諾または不要な E メールとして報告されたメッセージの割合。 このメトリクスは、受信者によって未承諾または不要な E メールとして報告されたメッセージの数を、送信されたメッセージ数で除算して計算されます。
日付別にグループ化された苦情率	`txn-emails-complaint-rate-grouped-by-date`	指定された日付範囲の各日に受信者によって未承諾または

メトリクス	Kpi-name	説明
		不要な E メールとして報告されたメッセージの割合。 このメトリクスは、指定された日付範囲の各日に受信者によって未承諾または不要な E メールとして報告されたメッセージの数を、送信されたメッセージ数で除算して計算されます。 このメトリクスのクエリの結果は、拡張 ISO 8601 形式で、暦日ごとにグループ化されています。
苦情	`txn-emails-with-complaints`	受信者によって未承諾または不要な E メールとして報告されたメッセージの数。
日付別にグループ化された苦情	`txn-emails-with-complaints-grouped-by-date`	受信者によって未承諾または不要な E メールとして報告されたメッセージのうち、指定した日付範囲の各日の数。 このメトリクスのクエリの結果は、拡張 ISO 8601 形式で、暦日ごとにグループ化されています。

メトリクス	Kpi-name	説明
配信数	`txn-emails-delivered`	受信者に配信されたメッセージの数。 このメトリクスは、送信されたメッセージ数から、ソフトバウンス、ハードバウンス、または拒否されたために配信できなかったメッセージの数を引いた数として計算されます。Amazon Pinpoint がメッセージにマルウェアが含まれていると判断した場合、メッセージは拒否されます。Amazon Pinpoint は、拒否されたメッセージを送信しません。

メトリクス	Kpi-name	説明
日付別にグループ化された配信数	`txn-emails-delivered-grouped-by-date`	指定された日付範囲の各日に受信者に配信されたメッセージの数。 このメトリクスは、指定された日付範囲の各日に、送信されたメッセージの数から、ソフトバウンス、ハードバウンス、または拒否されたために配信できなかったメッセージの数を引いた数として計算されます。Amazon Pinpoint がメッセージにマルウェアが含まれていると判断した場合、メッセージは拒否されます。Amazon Pinpoint は、拒否されたメッセージを送信しません。 このメトリクスのクエリの結果は、拡張 ISO 8601 形式で、暦日ごとにグループ化されています。
配信率	`txn-emails-delivery-rate`	受信者に配信されたメッセージの割合。 このメトリクスは、受信者に配信されたメッセージの数を、送信されたメッセージの数で割ったものとして計算されます。

メトリクス	Kpi-name	説明
日付別にグループ化された配信率	`txn-emails-delivery-rate-grouped-by-date`	指定された日付範囲の各日に受信者に配信されたメッセージの割合。 このメトリクスは、受信者に配信されたメッセージの数を、送信されたメッセージの数で割ったものとして計算されます。 このメトリクスのクエリの結果は、拡張 ISO 8601 形式で、暦日ごとにグループ化されています。
ハードバウンス	`txn-emails-hard-bounced`	ハードバウンスのために受信者に配信できなかったメッセージの数。ハードバウンスは、永続的な問題によってメッセージが配信されない場合に発生します。例えば、受信者の E メールアドレスが存在しない場合などに発生します。

メトリクス	Kpi-name	説明
日付別にグループ化されたハードバウンス	txn-emails-hard-bounced-grouped-by-date	指定された日付範囲の各日に、ハードバウンスのために受信者に配信できなかったメッセージの数。ハードバウンスは、永続的な問題によってメッセージが配信されない場合に発生します。例えば、受信者の E メールアドレスが存在しない場合などに発生します。 このメトリクスのクエリの結果は、拡張 ISO 8601 形式で、暦日ごとにグループ化されています。
開封数	txn-emails-opened	受信者によって開封されたメッセージの数。
日付別にグループ化された開封数	txn-emails-opened-grouped-by-date	指定された日付範囲の各日に、受信者によって開封されたメッセージの数。 このメトリクスのクエリの結果は、拡張 ISO 8601 形式で、暦日ごとにグループ化されています。
送信数	txn-emails-sent	送信されたメッセージの数。

メトリクス	Kpi-name	説明
日付別にグループ化された送信数	`txn-emails-sent-grouped-by-date`	指定された日付範囲の各日に送信されたメッセージの数。 このメトリクスのクエリの結果は、拡張 ISO 8601 形式で、暦日ごとにグループ化されています。
ソフトバウンス	`txn-emails-soft-bounced`	ソフトバウンスのために受信者に配信できなかったメッセージの数。ソフトバウンスは、一時的な問題によってメッセージが配信されない場合に発生します。例えば、受信者の受信トレイが満杯である場合や受信サーバーが一時的に使用できない場合などに発生します。
日付別にグループ化されたソフトバウンス	`txn-emails-soft-bounced-grouped-by-date`	指定された日付範囲の各日に、ソフトバウンスのために受信者に配信できなかったメッセージの数。ソフトバウンスは、一時的な問題によってメッセージが配信されない場合に発生します。例えば、受信者の受信トレイが満杯である場合や受信サーバーが一時的に使用できない場合などに発生します。 このメトリクスのクエリの結果は、拡張 ISO 8601 形式で、暦日ごとにグループ化されています。

メトリクス	Kpi-name	説明
一意のユーザークリックイベント	`txn-emails-unique-clicks`	メッセージ内のリンクをクリックした一意の受信者 (エンドポイント) の数。 [クリック数] メトリクスとは異なり、このメトリクスは発生したクリックイベントの数ではなく、リンクをクリックした一意の受信者の数を報告します。例えば、1 人の受信者が同じメッセージ内の複数のリンクをクリックした場合や、同じリンクを複数回クリックした場合、このメトリクスはその受信者に対して 1 つのクリックイベントのみを報告します。

メトリクス	Kpi-name	説明
日付別にグループ化された一意のユーザークリックイベント	`txn-emails-unique-clicks-grouped-by-date`	指定した日付範囲の各日にメッセージ内のリンクをクリックした一意の受信者 (エンドポイント) の数。 [日付別にグループ化されたクリック数] とは異なり、このメトリクスは、発生したクリックイベントの数ではなく、リンクをクリックした一意の受信者の数を報告します。例えば、1 人の受信者が同じメッセージ内の複数のリンクをクリックした場合や、同じリンクを複数回クリックした場合、このメトリクスはその受信者に対して 1 つのクリックイベントのみを報告します。 このメトリクスのクエリの結果は、拡張 ISO 8601 形式で、暦日ごとにグループ化されています。

メトリクス	Kpi-name	説明
一意のユーザー開封イベント	`txn-emails-unique-opens`	メッセージを開いた一意の受信者 (エンドポイント) の数。 [開封] メトリクスとは異なり、このメトリクスは発生した開封イベントの数ではなく、メッセージを開封した一意の受信者の数を報告します。例えば、1 人の受信者が同じメッセージを複数回開封した場合、このメトリクスはその受信者の開封イベントを 1 つだけ報告します。
日付別にグループ化された一意のユーザー開封イベント	`txn-emails-unique-opens-grouped-by-date`	指定した日付範囲の各日にメッセージを開封した一意の受信者 (エンドポイント) の数。 [日付別にグループ化された開封] メトリクスとは異なり、このメトリクスは発生した開封イベントの数ではなく、メッセージを開封した一意の受信者の数を報告します。例えば、1 人の受信者が同じメッセージを複数回開封した場合、このメトリクスはその受信者の開封イベントを 1 つだけ報告します。 このメトリクスのクエリの結果は、拡張 ISO 8601 形式で、暦日ごとにグループ化されています。

トランザクション SMS メッセージのアプリケーションメトリクス

次の表に、Amazon Pinpoint プロジェクトに関連付けられているすべてのトランザクション SMS メッセージの傾向をモニタリングするためにクエリできる標準アプリケーションメトリクスのリストと説明を示します。これらのメトリクスのデータをクエリするには、Amazon Pinpoint API のアプリケーションメトリクスリソースを使用します。表の kpi-name 列は、クエリの `kpi-name` パラメータに使用する値を示しています。

これらのメトリクスは、キャンペーンによって送信された SMS メッセージに関するデータは提供しないことに注意してください。トランザクション SMS メッセージに関するデータのみを提供します。1 つ以上のキャンペーンによって送信されたメッセージのデータをクエリするには、キャンペーンのキャンペーンメトリクスまたはアプリケーションメトリクスを使用します。

メトリクス	Kpi-name	説明
国別にグループ化された平均価格	`txn-sms-average-price-grouped-by-country`	各メッセージの送信先の国または地域ごとのメッセージ送信の平均コスト。価格は、米国セントの 1000 分の 1 で示されます。例えば、この属性の値が 645 の場合、メッセージの送信に 0.645¢ が課金されます (645 / 1000 = 0.645¢ = 0.00645 USD)。 このメトリクスは、各国または地域の受信者に送信されたすべてのメッセージの合計コストを、それぞれの国および地域の受信者に送信されたメッセージ数で割って計算されます。 このメトリクスのクエリ結果は、国または地域ごとに ISO 3166-1 alpha-2 形式でグループ化されます。

メトリクス	Kpi-name	説明
国別にグループ化されたメッセージパートあたりの平均価格	`txn-sms-average-price-by-parts-grouped-by-country`	各メッセージの送信先の国または地域ごとのメッセージパート送信の平均コスト。メッセージ部分は SMS メッセージの一部です。価格は、米国セントの 1000 分の 1 で示されます。例えば、この属性の値が 645 の場合、メッセージの送信に 0.645¢ が課金されます (645 / 1000 = 0.645¢ = 0.00645 USD)。 このメトリクスは、各国または地域の受信者に送信されたすべてのメッセージパートの合計コストを、それぞれの国および地域の受信者に送信されたメッセージ数で割って計算されます。 このメトリクスのクエリ結果は、国または地域ごとに ISO 3166-1 alpha-2 形式でグループ化されます。
配信数	`txn-sms-delivered`	受信者に配信されたメッセージの数。
国別にグループ化された配信数	`txn-sms-delivered-grouped-by-country`	メッセージの送信先の国または地域ごとに、受信者に配信されたメッセージの数。 このメトリクスのクエリ結果は、国または地域ごとに ISO 3166-1 alpha-2 形式でグループ化されます。

メトリクス	Kpi-name	説明
日付別にグループ化された配信数	`txn-sms-delivered-grouped-by-date`	指定された日付範囲の各日に受信者に配信されたメッセージの数。 このメトリクスのクエリの結果は、拡張 ISO 8601 形式で、暦日ごとにグループ化されています。
配信エラー数	`txn-sms-error-distribution`	発生したエラーのタイプごとに、メッセージの配信中にエラーが発生した回数。 このメトリクスのクエリ結果は、発生したエラーのタイプごとにエラーコードごとにグループ化されます。
配信率	`txn-sms-delivery-rate`	受信者に配信されたメッセージの割合。 このメトリクスは、受信者に配信されたメッセージの数を、送信されたメッセージの数で割ったものとして計算されます。

メトリクス	Kpi-name	説明
日付別にグループ化された配信率	`txn-sms-delivery-rate-grouped-by-date`	指定された日付範囲の各日に受信者に配信されたメッセージの割合。 このメトリクスは、受信者に配信されたメッセージの数を、送信されたメッセージの数で割ったものとして計算されます。 このメトリクスのクエリの結果は、拡張 ISO 8601 形式で、暦日ごとにグループ化されています。
配信されたメッセージパート	`txn-sms-delivered-by-parts`	参加者に配信されたメッセージパートの数。メッセージパートは、SMS メッセージの一部です。SMS のメッセージに使われている文字数が SMS プロトコルで許可されている文字数を超える場合、Amazon Pinpoint は送信可能な文字数でメッセージを分割し、受信者にメッセージを送信します。

メトリクス	Kpi-name	説明
国別にグループ化された配信されたメッセージパート	`txn-sms-delivered-by-parts-grouped-by-country`	メッセージの送信先の国または地域ごとに、受信者に配信されたメッセージパートの数。メッセージパートは、SMS メッセージの一部です。 このメトリクスのクエリ結果は、国または地域ごとに ISO 3166-1 alpha-2 形式でグループ化されます。
送信されたメッセージパート	`txn-sms-sent-by-parts`	送信されたメッセージパートの数。メッセージパートは、SMS メッセージの一部です。SMS のメッセージに使われている文字数が SMS プロトコルで許可されている文字数を超える場合、Amazon Pinpoint は送信可能な文字数でメッセージを分割し、受信者にメッセージを送信します。
国別にグループ化された送信済みメッセージパート	`txn-sms-sent-by-parts-grouped-by-country`	メッセージの送信先の国または地域ごとに、送信されたメッセージパートの数。メッセージパートは、SMS メッセージの一部です。 このメトリクスのクエリ結果は、国または地域ごとに ISO 3166-1 alpha-2 形式でグループ化されます。
送信されたメッセージ	`txn-sms-sent`	送信されたメッセージの数。

メトリクス	Kpi-name	説明
国別にグループ化された送信済みメッセージ	`txn-sms-sent-grouped-by-country`	メッセージの送信先の国または地域ごとに、送信されたメッセージの数。 このメトリクスのクエリ結果は、国または地域ごとに ISO 3166-1 alpha-2 形式でグループ化されます。
日付別にグループ化された送信済みメッセージ	`txn-sms-sent-grouped-by-date`	指定された日付範囲の各日に送信されたメッセージの数。 このメトリクスのクエリの結果は、拡張 ISO 8601 形式で、暦日ごとにグループ化されています。
国別にグループ化された合計価格	`txn-sms-total-price-grouped-by-country`	メッセージの送信先の国または地域ごとのメッセージ送信の合計コスト。価格は、米国セントの 1000 分の 1 で示されます。例えば、この属性の値が 645 の場合、メッセージの送信に 0.645¢ が課金されます (645 / 1000 = 0.645¢ = 0.00645 USD)。 このメトリクスのクエリ結果は、国または地域ごとに ISO 3166-1 alpha-2 形式でグループ化されます。

キャンペーンに関するメトリクス

次の表に、個々のキャンペーンのパフォーマンスを評価するためにクエリできる標準的なキャンペーンメトリクスのリストと説明を示します。これらのメトリクスのデータをクエリするには、Amazon Pinpoint API のキャンペーンメトリクスのリソースを使用います。表の kpi-name 列は、クエリの `kpi-name` パラメータに使用する値を示しています。

メトリクス	Kpi-name	説明
バウンス率	`hard-bounce-rate`	すべてのキャンペーンの実行において、受信者に配信できなかった E メールメッセージの割合。このメトリクスでは、ハードバウンスのみ計測されます。つまり、受信者の E メールアドレスに、メッセージを配信できない永続的な問題があるメッセージを指します。 このメトリクスは、すべてのキャンペーンの実行によって送信され、バウンスされた E メールメッセージの数を、それらすべてのキャンペーンの実行によって送信された E メールメッセージの数で割ったものとして計算されます。
キャンペーンの実行によってグループ化されたバウンス率	`hard-bounce-rate-grouped-by-campaign-activity`	各キャンペーンの実行において、受信者に配信できなかった E メールメッセージの割合。このメトリクスでは、ハードバウンスのみ計測されます。つまり、受信者の E メールアドレスに、メッセージを配信できない永続的な問

メトリクス	Kpi-name	説明
		題があるメッセージを指します。 このメトリクスは、キャンペーンの実行によって送信され、バウンスされた E メールメッセージの数を、キャンペーンの実行によって送信された E メールメッセージの数で割ったものとして計算されます。 このメトリクスのクエリ結果はキャンペーン アクティビティ ID (CampaignActivityId) でグループ化されますが、これはキャンペーンの実行を一意に識別する文字列です。
配信率	successful-delivery-rate	すべてのキャンペーンの実行において、受信者に配信されたメッセージの割合。 このメトリクスは、すべてのキャンペーンの実行によって送信され、受信者に配信されたメッセージの数を、それらすべてのキャンペーンの実行によって送信されたメッセージの数で割ったものとして計算されます。

メトリクス	Kpi-name	説明
キャンペーンの実行によってグループ化された配信率	`successful-delivery-rate-grouped-by-campaign-activity`	各キャンペーンの実行において、受信者に配信されたメッセージの割合。 このメトリクスは、キャンペーンの実行によって送信され、受信者に配信されたメッセージの数を、それらすべてのキャンペーンの実行によって送信されたメッセージの数で割ったものとして計算されます。 このメトリクスのクエリ結果はキャンペーン アクティビティ ID (`CampaignActivityId`) でグループ化されますが、これはキャンペーンの実行を一意に識別する文字列です。

メトリクス	Kpi-name	説明
日付別にグループ化された配信率	`successful-delivery-rate-grouped-by-date`	すべてのキャンペーンの実行について、指定された日付範囲の各日中に受信者に配信されたメッセージの割合。 このメトリクスは、すべてのキャンペーンの実行によって送信され、受信者に配信されたメッセージの数を、指定された日付範囲の日ごとのそれらのすべてのキャンペーンによって送信されたメッセージの数で割ったものとして計算されます。 このメトリクスのクエリの結果は、拡張 ISO 8601 形式で、暦日ごとにグループ化されています。
E メールのオープン率	`email-open-rate`	すべてのキャンペーンの実行において、受信者によって開かれた E メールメッセージの割合。 このメトリクスは、すべてのキャンペーンの実行によって送信され、受信者によって開かれた E メールメッセージの数を、それらすべてのキャンペーンの実行によって送信され、受信者に配信された E メールメッセージの数で割ったものとして計算されます。

メトリクス	Kpi-name	説明
キャンペーンの実行によってグループ化された E メールのオープン率	email-open-rate-grouped-by-campaign-activity	各キャンペーンの実行において、受信者によって開かれた E メールメッセージの割合。 このメトリクスは、キャンペーンの実行によって送信され、受信者によって開かれた E メールメッセージの数を、キャンペーンの実行によって送信され、受信者に配信された E メールメッセージの数で割ったものとして計算されます。 このメトリクスのクエリ結果はキャンペーン アクティビティ ID (CampaignActivityId) でグループ化されますが、これはキャンペーンの実行を一意に識別する文字列です。
キャンペーンの実行によってグループ化され、開かれた E メール	direct-email-opens-grouped-by-campaign-activity	各キャンペーンの実行において、受信者によって開かれた E メールメッセージの数 このメトリクスのクエリ結果はキャンペーン アクティビティ ID (CampaignActivityId) でグループ化されますが、これはキャンペーンの実行を一意に識別する文字列です。

メトリクス	Kpi-name	説明
エンドポイントの配信	unique-deliveries	すべてのキャンペーンの実行において、メッセージが配信された一意のエンドポイントの数。
キャンペーンの実行によってグループ化されたエンドポイント配信	unique-deliveries-grouped-by-campaign-activity	各キャンペーンの実行において、メッセージが配信された一意のエンドポイントの数。 このメトリクスのクエリ結果はキャンペーン アクティビティ ID (CampaignActivityId) でグループ化されますが、これはキャンペーンの実行を一意に識別する文字列です。
日付別にグループ化されたされた配信エンドポイント	unique-deliveries-grouped-by-date	すべてのキャンペーンの実行について、指定された日付範囲の各日にメッセージが配信された一意のエンドポイントの数。 このメトリクスのクエリの結果は、拡張 ISO 8601 形式で、暦日ごとにグループ化されています。

メトリクス	Kpi-name	説明
キャンペーンの実行によってグループ化され、クリックされたリンク	`clicks-grouped-by-campaign-activity`	各キャンペーンの実行において、受信者がE メールメッセージ内のリンクをクリックした回数。1 人の受信者がメッセージ内の複数のリンクをクリックするか、同じリンクを複数回クリックした場合は、それぞれのクリックはカウントされます。 このメトリクスのクエリ結果はキャンペーン アクティビティ ID (`CampaignActivityId`) でグループ化されますが、これはキャンペーンの実行を一意に識別する文字列です。

メトリクス	Kpi-name	説明
キャンペーンの実行によってグループ化され、配信されたメッセージ	`successful-deliveries-grouped-by-campaign-activity`	各キャンペーンの実行において、受信者に配信されたメッセージの数。 このメトリクスは、キャンペーンによって送信されたメッセージの数を、キャンペーンの実行によって送信されたメッセージの数から、キャンペーンによって送信され、ハードバウンスのために実行の受信者に配信できなかったメッセージの数を引いたものとして計算されます。 このメトリクスのクエリ結果はキャンペーン アクティビティ ID (`CampaignActivityId`) でグループ化されますが、これはキャンペーンの実行を一意に識別する文字列です。
キャンペーンの実行によってグループ化され、送信されたメッセージ	`attempted-deliveries-grouped-by-campaign-activity`	各キャンペーンの実行において、送信されたメッセージの数。 このメトリクスのクエリ結果はキャンペーン アクティビティ ID (`CampaignActivityId`) でグループ化されますが、これはキャンペーンの実行を一意に識別する文字列です。

メトリクス	Kpi-name	説明
プッシュオープン率	push-open-rate	すべてのキャンペーンの実行において、受信者によって開かれたプッシュ通知の割合。 このメトリクスは、すべてのキャンペーンの実行によって送信され、受信者によって開かれたプッシュ通知の数を、それらすべてのキャンペーンの実行によって送信され、受信者に配信されたプッシュ通知の数で割ったものとして計算されます。
キャンペーンの実行によってグループ化されたプッシュオープン率	push-open-rate-grouped-by-campaign-activity	各キャンペーンの実行において、受信者によって開かれたプッシュ通知の割合。 このメトリクスは、キャンペーンの実行によって送信され、受信者によって開かれたプッシュ通知の数を、キャンペーンの実行によって送信され、受信者に配信されたプッシュ通知の数で割ったものとして計算されます。 このメトリクスのクエリ結果はキャンペーン アクティビティ ID (CampaignActivityId) でグループ化されますが、これはキャンペーンの実行を一意に識別する文字列です。

メトリクス	Kpi-name	説明
キャンペーンの実行によってグループ化され、開かれたプッシュの合計	`direct-push-opens-grouped-by-campaign-activity`	各キャンペーンの実行において、受信者によって開かれたプッシュ通知の数。 このメトリクスのクエリ結果はキャンペーン アクティビティ ID (`CampaignActivityId`) でグループ化されますが、これはキャンペーンの実行を一意に識別する文字列です。
SMS 使用量の合計	`sms-spend`	すべてのキャンペーンで送信された SMS の合計金額 (ミリセント)。

ジャーニーエンゲージメントメトリクス

次の表は、ジャーニーによって送信されたすべての E メールメッセージの傾向を監視するためにクエリを実行できる標準的な Amazon Pinpoint ジャーニーのエンゲージメントメトリクスの一覧と説明です。これらのメトリクスのデータをクエリするには、Amazon Pinpoint API のジャーニーエンゲージメントメトリクスのリソースを使用います。表の kpi-name 列は、クエリの `kpi-name` パラメータに使用する値を示しています。

メトリクス	Kpi-name	説明
クリック数	`journey-emails-clicked`	参加者がメッセージのリンクをクリックした回数。1 人の参加者がメッセージ内の複数のリンクをクリックするか、同じリンクを複数回クリックした場合は、それぞれのクリックはカウントされます。

メトリクス	Kpi-name	説明
アクティビティ別にグループ化されたクリック数	`emails-clicked-grouped-by-journey-activity`	ジャーニーの各アクティビティについて、参加者がメッセージ内のリンクをクリックした回数。1 人の参加者がメッセージ内の複数のリンクをクリックするか、同じリンクを複数回クリックした場合は、それぞれのクリックはカウントされます。 このメトリクスのクエリ結果は、アクティビティ ID (`JourneyActivityId`) でグループ化されますが、これはアクティビティを一意に識別する文字列です。
苦情	`journey-emails-complained`	参加者によって未承諾または不要な E メールとして報告されたメッセージの数。
アクティビティ別にグループ化された苦情	`emails-complained-grouped-by-journey-activity`	ジャーニーの各アクティビティについて、参加者が未承諾または不要なメールとして報告されたメッセージの数。 このメトリクスのクエリ結果は、アクティビティ ID (`JourneyActivityId`) でグループ化されますが、これはアクティビティを一意に識別する文字列です。

メトリクス	Kpi-name	説明
配信数	`journey-emails-delivered`	参加者に配信されたメッセージの数。 このメトリクスは、送信されたメッセージ数から、ソフトバウンス、ハードバウンス、または拒否されたために配信できなかったメッセージの数を引いた数として計算されます。
アクティビティ別にグループ化された配信数	`emails-delivered-grouped-by-journey-activity`	ジャーニーの各アクティビティについて、参加者に配信されたメッセージの数。 このメトリクスは、ジャーニーの各アクティビティに、送信されたメッセージの数から、ソフトバウンス、ハードバウンス、または拒否されたために配信できなかったメッセージの数を引いた数として計算されます。 このメトリクスのクエリ結果は、アクティビティ ID (`JourneyActivityId`) でグループ化されますが、これはアクティビティを一意に識別する文字列です。

メトリクス	Kpi-name	説明
ハードバウンス	journey-emails-hardbounced	ハードバウンスのために参加者に配信できなかったメッセージの数。ハードバウンスは、永続的な問題によってメッセージが配信されない場合に発生します。例えば、参加者の E メールアドレスが存在しない場合などに発生します。
アクティビティ別にグループ化されたハードバウンス数	emails-hardbounced-grouped-by-journey-activity	ジャーニーの各アクティビティについて、ハードバウンスのために参加者に配信できなかったメッセージの数。ハードバウンスは、永続的な問題によってメッセージが配信されない場合に発生します。例えば、参加者の E メールアドレスが存在しない場合などに発生します。 このメトリクスのクエリ結果は、アクティビティ ID (JourneyActivityId) でグループ化されますが、これはアクティビティを一意に識別する文字列です。
開封数	journey-emails-opened	参加者によって開封されたメッセージの数。

メトリクス	Kpi-name	説明
アクティビティ別にグループ化された開封数	`emails-opened-grouped-by-journey-activity`	ジャーニーの各アクティビティについて、参加者によって開封されたメッセージの数。 このメトリクスのクエリ結果は、アクティビティ ID (`JourneyActivityId`) でグループ化されますが、これはアクティビティを一意に識別する文字列です。
拒否数	`journey-emails-rejected`	参加者が拒否されたために参加者に送信されなかったメッセージの数。Amazon Pinpoint がメッセージにマルウェアが含まれていると判断した場合、メッセージは拒否されます。Amazon Pinpoint は、拒否されたメッセージを送信しません。

メトリクス	Kpi-name	説明
アクティビティ別にグループ化された拒否数	`emails-rejected-grouped-by-journey-activity`	ジャーニーの各アクティビティについて、拒否されたために参加者に送信されなかったメッセージの数。Amazon Pinpoint がメッセージにマルウェアが含まれていると判断した場合、メッセージは拒否されます。Amazon Pinpoint は、拒否されたメッセージを送信しません。 このメトリクスのクエリ結果は、アクティビティ ID (`JourneyActivityId`) でグループ化されますが、これはアクティビティを一意に識別する文字列です。
送信数	`journey-emails-sent`	送信されたメッセージの数。
アクティビティ別にグループ化された送信数	`emails-sent-grouped-by-journey-activity`	ジャーニーの各アクティビティについて、送信されたメッセージの数。 このメトリクスのクエリ結果は、アクティビティ ID (`JourneyActivityId`) でグループ化されますが、これはアクティビティを一意に識別する文字列です。

メトリクス	Kpi-name	説明
ソフトバウンス	`journey-emails-softbounced`	ソフトバウンスのために参加者に配信できなかったメッセージの数。ソフトバウンスは、一時的な問題によってメッセージが配信されない場合に発生します。例えば、参加者の受信トレイが満杯である場合や受信サーバーが一時的に使用できない場合などに発生します。
アクティビティ別にグループ化されたソフトバウンス数	`emails-softbounced-grouped-by-journey-activity`	ジャーニーの各アクティビティについて、ソフトバウンスのために参加者に配信できなかったメッセージの数。ソフトバウンスは、一時的な問題によってメッセージが配信されない場合に発生します。例えば、参加者の受信トレイが満杯である場合や受信サーバーが一時的に使用できない場合などに発生します。 このメトリクスのクエリ結果は、アクティビティ ID (`JourneyActivityId`) でグループ化されますが、これはアクティビティを一意に識別する文字列です。

メトリクス	Kpi-name	説明
サブスクリプション解除数	`journey-emails-unsubscribed`	参加者がメッセージ内のサブスクリプション解除リンクをクリックした回数。1 人の参加者が同じサブスクリプション解除リンクを複数回クリックした場合、各クリックはカウントに含まれます。
アクティビティ別にグループ化されたサブスクリプション解除数	`emails-unsubscribed-grouped-by-journey-activity`	ジャーニーの各アクティビティについて、参加者がメッセージ内のサブスクリプション解除リンクをクリックした回数。1 人の参加者が同じサブスクリプション解除リンクを複数回クリックした場合、各クリックはカウントに含まれます。 このメトリクスのクエリ結果は、アクティビティ ID (`JourneyActivityId`) でグループ化されますが、これはアクティビティを一意に識別する文字列です。

ジャーニー実行メトリクス

次の表に、Amazon Pinpoint ジャーニー参加者のステータスを評価するためにクエリできる標準的な実行メトリクスのリストと説明を示します。これらのメトリクスのデータをクエリするには、Amazon Pinpoint API のジャーニー実行メトリクスのリソースを使用います。テーブルの [フィールド] 列には、各メトリクスのクエリ結果に表示されるフィールドの名前が表示されます。

メトリクス	フィールド	説明
アクティブな参加者	ENDPOINT_ACTIVE	ジャーニーのアクティビティを積極的に進めている参加者の数。 このメトリクスは、ジャーニーを開始した参加者数から、ジャーニーを離れた参加者数およびジャーニーから削除された参加者数を引いたものとして計算されます。
参加者のキャンセル	CANCELLED	ジャーニーがキャンセルされたためにジャーニーを完了しなかった参加者の数。
参加者の出発	ENDPOINT_LEFT	ジャーニーを離れた参加者の数。
参加者のエントリ	ENDPOINT_ENTERED	ジャーニーを開始した参加者の数。
参加者の例外、再エントリ制限	REENTRY_CAP_EXCEEDED	1 人の参加者がジャーニーに再エントリすることができる最大回数を超えたため、ジャーニーを完了しなかった参加者の数。
参加者の例外、拒否	ACTIVE_ENDPOINT_REJECTED	ジャーニーのすでにアクティブな参加者であるため、そのジャーニーを開始できない参加者の数。 参加者がジャーニーを開始し、その後、お客様がセグメント（セグメント条件に基づく）またはジャーニー（アク

メトリクス	フィールド	説明
		ティビティ条件に基づく) に参加者を含めるように、参加者のエンドポイント定義を更新した場合、参加者は拒否されます。

ジャーニーアクティビティ実行メトリクス

次の表に、Amazon Pinpoint ジャーニーの個々のアクティビティのタイプごとの参加者のステータスを評価するためにクエリできる標準的な実行メトリクスのリストと説明を示します。これらのメトリクスのデータをクエリするには、Amazon Pinpoint API のジャーニーアクティビティ実行メトリクスのリソースを使用います。表のメトリクス列には、アクティビティのタイプごとにクエリ結果に表示されるフィールドが一覧表示されます。また、各フィールドの簡単な説明も示します。

アクティビティタイプ	メトリクス
はい/いいえ分割 (CONDITIONAL_SPLIT)	メトリクスは次のとおりです。 • Branch_FALSE – アクティビティの条件を満たさず、「いいえ」パスのアクティビティに進んだ参加者の数。 • Branch_TRUE – アクティビティの条件を満たし、「はい」パスのアクティビティに進んだ参加者の数。 パスごとにアクティビティに追加のメトリクスを使用できます。これらのメトリクスについては、このテーブルのそのタイプのアクティビティの行を参照してください。
ホールドアウト (HOLDOUT)	メトリクスは次のとおりです。 • HOLDOUT – アクティビティのホールドアウト率の一部としてジャーニーから削除された参加者の数。

アクティビティタイプ	メトリクス
	• PASSED – ジャーニーの次のアクティビティに進んだ参加者の数。
E メール (MESSAGE)	メトリクスは次のとおりです。 • DAILY_CAP_EXCEEDED – 1 人の参加者が 24 時間内に受信できるメッセージの最大数を超えたために送信されなかったメッセージの数。 • FAILURE_PERMANENT – 永続的な問題のために送信されなかったメッセージの数。 • QUIET_TIME – 参加者のタイムゾーンのクワイエットタイム内に配信されたために送信されなかったメッセージの数。 • SERVICE_FAILURE – Amazon Pinpoint の問題により送信されなかったメッセージの数。 • SUCCESS – 参加者に正常に配信されたメッセージの数。 • THROTTLED – メッセージの送信時、Amazon Pinpoint アカウントの送信クォータを超えたため送信されなかったメッセージの数。 • TRANSIENT_FAILURE – 一時的な問題のために送信されなかったメッセージの数。 • UNKNOWN – 不明な問題が原因で送信されなかったメッセージの数。

アクティビティタイプ	メトリクス
多変量分割 (MULTI_CONDITIONAL_SPLIT)	アクティビティのパス別に、そのパスのアクティビティに進んだ参加者の数。 このメトリクスのクエリ結果は、パス Branch_# でグループ化されます。# はパスの数値識別子です。例えば、アクティビティの最初のパスは Branch_1 です。 パスごとにアクティビティに追加のメトリクスを使用できます。これらのメトリクスについては、このテーブルのそのタイプのアクティビティの行を参照してください。
ランダム分割 (RANDOM_SPLIT)	アクティビティのパス別に、そのパスのアクティビティに進んだ参加者の数。 このメトリクスのクエリ結果は、パス Branch_# でグループ化されます。# はパスの数値識別子です。例えば、アクティビティの最初のパスは Branch_1 です。 パスごとにアクティビティに追加のメトリクスを使用できます。これらのメトリクスについては、このテーブルのそのタイプのアクティビティの行を参照してください。

アクティビティタイプ	メトリクス
待機 (WAIT)	メトリクスは次のとおりです。 • WAIT_FINISHED – 指定した時間待機を終了した参加者の数。 • WAIT_SKIPPED – 指定した時間まで待機しなかった参加者の数。通常、アクティビティの予定終了時刻以降にアクティビティまたはジャーニーを開始したためです。 • WAIT_STARTED – 待機を開始し、指定された時間だけスキップしていないか、待機を終了していない参加者の数。

アクティビティタイプ	メトリクス
コンタクトセンター (CONTACT_CENTER)	メトリクスは次のとおりです。 • CALL_QUEUED – ダイヤルインして Amazon Connect のキューに入れられた参加者の数。リダイヤル試行が含まれます。 • CONTINUE_WAITING – ダイヤル試行が実行されるまで待ち続ける参加者の数。 • DIAL_FAILURE – ダイヤル試行が失敗した参加者の数。 • DROPPED – 送信時点で、以前のジャーニーアクティビティで定義された条件を満たさなくなっている参加者の数。 • TIMEOUT – 数回のダイヤル試行後に Amazon Connect 処理コードを受け取らなかった参加者の数。 • WAIT_FINISHED – 指定した時間待機を終了した参加者の数。 • WAIT_FOR_QUIET_HOURS – チャネルに配信するために、クワイエットタイムが終了するのを待っている参加者の数。 • WAIT_STARTED – 待機を開始し、指定された時間だけスキップしていないか、待機を終了していない参加者の数。

ジャーニーおよびキャンペーン実行メトリクス

Amazon Pinpoint ジャーニーまたはキャンペーンの個々のアクティビティのタイプごとに参加者のステータスを評価するために、標準的な実行メトリクスをクエリできます。これらのメトリクスのデータをクエリするには、Amazon Pinpoint API のジャーニーアクティビティ実行メトリクスまたはキャンペーンメトリクスのリソースを使用します。次の表は、アクティビティのタイプごとにクエリ結果に表示されるフィールドを示しています。

メトリクス名	ジャーニー、キャンペーン、またはその両方に適用	説明
ENDPOINT_PRODUCED	両方	フィルタリング前にセグメントまたはイベントから最初に生成されたエンドポイントの数。
ENDPOINTS_FROM_USER	両方	カスタマーがユーザー ID のみのセグメントを持っている場合は、そのユーザーのすべてのエンドポイントが追加されます。このメトリクスでは、この方法で追加されたエンドポイントの数が測定されます。
ENDPOINT_OPT_OUT	両方	エンドポイントはオプトアウトされており、キャンペーンやジャーニーにエントリしませんでした。
ENDPOINT_INACTIVE	両方	エンドポイントは非アクティブになっており、キャンペーンやジャーニーにエントリしませんでした。
FILTERED_OUT_BY_SEGMENT	両方	エンドポイントはセグメントフィルターに一致しておらず、キャンペーンやジャーニーにエントリしませんでした。
ENDPOINT_MISSING_ADDRESS	両方	エンドポイントに住所が見当たらず、キャンペーンやジャーニーにエントリしませんでした。

メトリクス名	ジャーニー、キャンペーン、またはその両方に適用	説明
ENDPOINT_MISSING_CHANNEL	両方	エンドポイントにチャネルが見当たらず、キャンペーンやジャーニーにエントリしませんでした。
ENDPOINT_MISSING_TIMEZONE	両方	エンドポイントにタイムゾーンの値が見当たらず、除外されました。これはタイムゾーンの値が必要な場合にのみ発生します。
ENDPOINT_TIMEZONE_MISMATCH	両方	エンドポイントは、その時点で実行対象に含まれていなかったタイムゾーンにありました。
ENDPOINT_CHANNEL_MISMATCH	キャンペーン	キャンペーンには、このエンドポイントのチャネルタイプ用のメッセージが設定されていません。
DUPLICATE_ENDPOINT	両方	重複するエンドポイントが見つかり、重複除外されました。
DUPLICATE_USER	両方	重複するユーザーが見つかり、ユーザー ID のみのセグメントから重複除外されました。ユーザー ID が同じ場合、メトリクス 1 が出力されます。
PAUSED	ジャーニー	ジャーニーが一時停止されたため、実行対象から削除されました。

メトリクス名	ジャーニー、キャンペーン、またはその両方に適用	説明
ENDED	ジャーニー	ジャーニーが終了したため、実行対象から削除されました。
TREATMENT_HOLDOUT	キャンペーン	これは、コホートが現在の処理と一致しないエンドポイントについて、A/B キャンペーンで出力されます。例えば、A/B を 50/50 に分割した場合、エンドポイントの 50% が各処理でこのメトリクスを出力します。
ENDPOINT_ESTIMATED_TIMEZONE	ジャーニー	タイムゾーン推定により、エンドポイントのタイムゾーンを推定できました。

Amazon Pinpoint 分析キャンペーンデータのクエリ

Amazon Pinpoint コンソールの分析ページを使用することに加えて、Amazon Pinpoint 分析 API を使用して、キャンペーンの配信とエンゲージメントの傾向についてのインサイトを提供する標準メトリクスのサブセットについて分析データをクエリできます。

これらの各メトリクスは測定可能な値で、重要業績評価指標 (KPI) とも呼ばれ、1 つ以上のキャンペーンのパフォーマンスを監視および評価するのに役立ちます。例えば、メトリクスを使用して、キャンペーンメッセージが送信されたエンドポイントの数、または目的のエンドポイントに配信されたメッセージの数を確認できます。

Amazon Pinpoint は、すべてのキャンペーンでこのデータを自動的に収集し、集計します。これは、90 日間データを保存します。AWS Mobile SDK を使用してモバイルアプリケーションを Amazon Pinpoint と統合した場合、Amazon Pinpoint はこのサポートを拡張して、受信者によって開かれたプッシュ通知の割合などの追加メトリクスを含めることができます。モバイルアプリケーションの統合については、「Amazon Pinpoint とアプリケーションの統合」を参照してください

Amazon Pinpoint 分析 API を使用してデータをクエリする場合、クエリの範囲、データ、グループ化、フィルターを定義するさまざまなオプションを選択できます。これを行うには、適用する日付ベースのフィルターに加えて、クエリするプロジェクト、キャンペーン、メトリクスを指定するパラメータを使用します。

このトピックでは、これらのオプションを選択し、1 つ以上のキャンペーンのデータをクエリする方法の例を説明します。

前提条件

1 つ以上のキャンペーンの分析データをクエリする前に、クエリの定義に使用する、次の情報を収集すると役立ちます。

- Project ID – 1 つまたは複数のキャンペーンに関連付けられたプロジェクトの一意な ID。Amazon Pinpoint API では、この値は `application-id` プロパティに保存されます。Amazon Pinpoint コンソールでは、この値はプロジェクト ID ですべてのプロジェクトページで表示されます。
- Campaign ID – キャンペーンの一意の識別子 (1 つのキャンペーンのデータのみをクエリする場合) 。Amazon Pinpoint API では、この値は `campaign-id` プロパティに保存されます。この値は、コンソールには表示されません。
- Date range – オプションで、データを問い合せる日付範囲の最初と最後の日付と時刻。日付範囲は包括的であり、31日以内に制限する必要があります。また、最初の日付と時刻は、現在の日付から 90 日未満である必要があります。日付範囲を指定しない場合、Amazon Pinpoint は過去 31 暦日のデータを自動的にクエリします。
- Metric type – クエリするメトリクスのタイプ。アプリケーションメトリクスとキャンペーンメトリクス の 2 つのタイプがあります。アプリケーションメトリクスは、プロジェクトに関連付けられているすべてのキャンペーンのデータを提供し、アプリケーションとも呼ばれます。キャンペーンメトリクスは、1 つのキャンペーンのみのデータを提供します。
- Metric – クエリするメトリクスの名前。具体的には、メトリクスの `kpi-name` 値。サポートされているメトリクスの完全なリストと各メトリクスの `kpi-name` 値については、「標準メトリクス」を参照してください。

また、関連するフィールドでデータをグループ化するかどうかを決定することもできます。する場合、データを自動的にグループ化するように設計されたメトリクスを選択することで、分析とレポート作成を簡素化できます。例えば、Amazon Pinpoint には、キャンペーンの受信者に配信されたメッセージの割合を報告する標準メトリクスがいくつか用意されています。これらのメトリクスの 1 つは、データを日付ごとに自動的にグループ化します (`successful-delivery-rate-grouped-`

by-date)。別のメトリクスでは、データがキャンペーンの実行ごとに自動的にグループ化されます (successful-delivery-rate-grouped-by-campaign-activity)。3 番目のメトリクスは、すべてのキャンペーンの実行によって受信者に配信されたメッセージの割合 (%) を返します (successful-delivery-rate)。

必要な方法でデータをグループ化する標準メトリクスが見つからない場合は、必要なデータを返す一連のクエリを開発できます。その後、クエリ結果を手動で分類したり、デザインするカスタムグループに結合できます。

最後に、クエリするデータにアクセスする権限があることを確認することが重要です。詳細については、「Amazon Pinpoint 分析データのクエリ実行用の IAM ポリシー」を参照してください。

1 つのキャンペーンのデータのクエリ

1 つのキャンペーンのデータをクエリするには、Campaign Metrics API を使用して、次の必須パラメータに値を指定します。

- application-id – キャンペーンに関連付けられているプロジェクトの一意の識別子であるプロジェクト ID。Amazon Pinpoint では、プロジェクトとアプリケーションという用語は同じ意味です。
- campaign-id – キャンペーンの一意の識別子。
- kpi-name – クエリするメトリクスの名前。この値は、関連するメトリクスを記述し、ハイフンで区切られた小文字の英数字で構成される 2 つ以上の用語で構成されます。サポートされているメトリクスの完全なリストと各メトリクスの kpi-name 値については、「標準メトリクス」を参照してください。

また、特定の日付範囲のデータをクエリするフィルターを適用することもできます。日付範囲を指定しない場合、Amazon Pinpoint は過去 31 暦日のデータを返します。異なる日付でデータをフィルタリングするには、サポートされている日付範囲パラメータを使用して、日付範囲の最初と最後の日時を指定します。値は拡張 ISO 8601 形式で、協定世界時 (UTC) を使用する必要があります (例: 2019-07-19T20:00:00Z は、2019年7月19日午後8時 (UTC))。日付範囲は包括的であり、31 日以内に制限する必要があります。また、最初の日付と時刻は、現在の日付から 90 日未満である必要があります。

次の例は、Amazon Pinpoint REST API、AWS CLI、および AWS SDK for Java を使用して、キャンペーンの分析データをクエリする方法を示しています。サポートされている任意の AWS SDK を使用して、キャンペーンの分析データをクエリできます。これらの AWS CLI の例は、Microsoft Windows 用にフォーマットされています。Unix、Linux、macOS の場合は、キャレット (^) 行継続文字をバックスラッシュ (\) に置き換えます。

REST API

Amazon Pinpoint REST API を使用してキャンペーンの分析データをクエリするには、HTTP (S) GET リクエストを Campaign Metrics URI に送信します。URI で、必要なパスパラメータに次の適切な値を指定します。

```
https://endpoint/v1/apps/application-id/campaigns/campaign-id/kpis/daterange/kpi-name
```

各パラメータの意味は次のとおりです。

- *endpoint* は、キャンペーンに関連付けられたプロジェクトをホストする AWS リージョンの Amazon Pinpoint エンドポイントです。
- *application-id* は、キャンペーンに関連付けられているプロジェクトの一意の識別子です。
- *campaign-id* は、キャンペーンの一意の識別子です。
- *kpi-name* は、クエリするメトリクスの kpi-name 値です。

すべてのパラメータは、URL エンコードする必要があります。

特定の日付範囲のデータをクエリするフィルターを適用するには、URI に start-time および end-time クエリパラメータと値を追加します。これらのパラメータを使用すると、データを取得する日付範囲の最初と最後の日付と時刻 (両端を含む) を、拡張 ISO 8601 形式で指定できます。パラメータを区切るには、アンパサンド (&) を使用します。

例えば、次のリクエストは、2019 年 7 月 19 日から 2019 年 7 月 26 日まで、キャンペーンのすべての実行によってメッセージが配信された一意のエンドポイントの数を取得します。

```
https://pinpoint.us-east-1.amazonaws.com/v1/apps/12345678901234567890123example/campaigns/80b8efd84042ff8d9c96ce2f8example/kpis/daterange/unique-deliveries?start-time=2019-07-19T00:00:00Z&end-time=2019-07-26T23:59:59Z
```

実行する条件は以下のとおりです。

- pinpoint.us-east-1.amazonaws.com は、プロジェクトをホストする AWS リージョンの Amazon Pinpoint エンドポイントです。
- 12345678901234567890123example は、キャンペーンに関連付けられているプロジェクトの一意の識別子です。

- 80b8efd84042ff8d9c96ce2f8example は、キャンペーンの一意の識別子です。
- unique-deliveriesは、エンドポイントの配信キャンペーンメトリクスの kpi-name 値です。これは、キャンペーンのすべての実行によって、メッセージが配信されたユニークなエンドポイントの数を報告するメトリックです。
- 2019-07-19T00:00:00Z は、端の値を含む日付範囲の一部として、データを取得する最初の日付と時刻です。
- 2019-07-26T23:59:59Z は、端の値を含む日付範囲の一部として、データを取得する最後の日付と時刻です。

AWS CLI

AWS CLI を使用してキャンペーンの分析データをクエリするには、get-campaign-date-range-kpi コマンドを使用して、必要なパラメータに適切な値を指定します。

```
C:\> aws pinpoint get-campaign-date-range-kpi ^
    --application-id application-id ^
    --campaign-id campaign-id ^
    --kpi-name kpi-name
```

実行する条件は以下のとおりです。

- *application-id* は、キャンペーンに関連付けられているプロジェクトの一意の識別子です。
- *campaign-id* は、キャンペーンの一意の識別子です。
- *kpi-name* は、クエリするメトリクスの kpi-name 値です。

特定の日付範囲のデータをクエリするフィルターを適用するには、start-time および end-time パラメータと値をクエリに追加します。これらのパラメータを使用すると、データを取得する日付範囲の最初と最後の日付と時刻（両端を含む）を、拡張 ISO 8601 形式で指定できます。例えば、次のリクエストは、2019 年 7 月 19 日から 2019 年 7 月 26 日まで、キャンペーンのすべての実行によってメッセージが配信された一意のエンドポイントの数を取得します。

```
C:\> aws pinpoint get-campaign-date-range-kpi ^
    --application-id 12345678901234567890123456example ^
    --campaign-id 80b8efd84042ff8d9c96ce2f8example ^
    --kpi-name unique-deliveries ^
```

```
    --start-time 2019-07-19T00:00:00Z ^
    --end-time 2019-07-26T23:59:59Z
```

実行する条件は以下のとおりです。

- 12345678901234567890123456example は、キャンペーンに関連付けられているプロジェクトの一意の識別子です。
- 80b8efd84042ff8d9c96ce2f8example は、キャンペーンの一意の識別子です。
- unique-deliveriesは、エンドポイントの配信キャンペーンメトリクスの kpi-name 値です。これは、キャンペーンのすべての実行によって、メッセージが配信されたユニークなエンドポイントの数を報告するメトリックです。
- 2019-07-19T00:00:00Z は、端の値を含む日付範囲の一部として、データを取得する最初の日付と時刻です。
- 2019-07-26T23:59:59Z は、端の値を含む日付範囲の一部として、データを取得する最後の日付と時刻です。

SDK for Java

AWS SDK for Java を使用してキャンペーンの分析データをクエリするには、Campaign Metrics API の GetCampaignDateRangeKpiRequest メソッドを使用します。必要なパラメータに適切な値を指定します。

```
GetCampaignDateRangeKpiRequest request = new GetCampaignDateRangeKpiRequest()
        .withApplicationId("applicationId")
        .withCampaignId("campaignId")
        .withKpiName("kpiName")
```

実行する条件は以下のとおりです。

- *applicationId* は、キャンペーンに関連付けられているプロジェクトの一意の識別子です。
- *campaignId* は、キャンペーンの一意の識別子です。
- *kpiName* は、クエリするメトリクスの kpi-name 値です。

特定の日付範囲のデータをクエリするフィルターを適用するには、startTime および endTime パラメータと値をクエリに含めます。これらのパラメータを使用すると、データを取得する日付範囲の最初と最後の日付と時刻（両端を含む）を、拡張 ISO 8601 形式で指定できます。例え

ば、次のリクエストは、2019 年 7 月 19 日から 2019 年 7 月 26 日まで、キャンペーンのすべての実行によってメッセージが配信された一意のエンドポイントの数を取得します。

```
GetCampaignDateRangeKpiRequest request = new GetCampaignDateRangeKpiRequest()
        .withApplicationId("12345678901234567890123456example")
        .withCampaignId("80b8efd84042ff8d9c96ce2f8example")
        .withKpiName("unique-deliveries")
        .withStartTime(Date.from(Instant.parse("2019-07-19T00:00:00Z")))
        .withEndTime(Date.from(Instant.parse("2019-07-26T23:59:59Z")));
```

実行する条件は以下のとおりです。

- 12345678901234567890123456example は、キャンペーンに関連付けられているプロジェクトの一意の識別子です。
- 80b8efd84042ff8d9c96ce2f8example は、キャンペーンの一意の識別子です。
- unique-deliveriesは、エンドポイントの配信キャンペーンメトリクスの kpi-name 値です。これは、キャンペーンのすべての実行によって、メッセージが配信されたユニークなエンドポイントの数を報告するメトリックです。
- 2019-07-19T00:00:00Z は、端の値を含む日付範囲の一部として、データを取得する最初の日付と時刻です。
- 2019-07-26T23:59:59Z は、端の値を含む日付範囲の一部として、データを取得する最後の日付と時刻です。

クエリを送信すると、Amazon Pinpoint はクエリ結果を JSON レスポンスで返します。結果の構造は、クエリしたメトリクスによって異なります。一部のメトリクスは 1 つの値しか返しません。例えば、前述の例で使用されたエンドポイント配信 (unique-deliveries) キャンペーンメトリックは、すべてのキャンペーンの実行により、1 つの値 (メッセージが配信されたユニークエンドポイントの数) を返します。この場合、JSON レスポンスは次のようになります。

```
{
    "CampaignDateRangeKpiResponse":{
        "ApplicationId":"12345678901234567890123456example",
        "CampaignId":"80b8efd84042ff8d9c96ce2f8example",
        "EndTime":"2019-07-26T23:59:59Z",
        "KpiName":"unique-deliveries",
        "KpiResult":{
            "Rows":[
                {
```

```
                    "Values":[
                        {
                            "Key":"UniqueDeliveries",
                            "Type":"Double",
                            "Value":"123.0"
                        }
                    ]
                }
            ]
        },
        "StartTime":"2019-07-19T00:00:00Z"
    }
}
```

他のメトリクスは複数の値を返し、関連するフィールドで値をグループ化します。メトリクスが複数の値を返す場合、JSON レスポンスにはデータのグループ化に使用されたフィールドを示すフィールドが含まれます。

クエリ結果の構造の詳細については、「クエリ結果の使用」を参照してください。

複数のキャンペーンのデータのクエリ

複数のキャンペーンのデータをクエリする方法は 2 つあります。最適な方法は、すべて同じプロジェクトに関連付けられているキャンペーンのデータをクエリするかどうかによって異なります。その場合は、すべてのキャンペーンのデータに対してクエリを実行するか、それらのキャンペーンのみまたはサブセットのどちらに対してクエリを実行するかによっても異なります。

異なるプロジェクトに関連付けられているキャンペーン、または同じプロジェクトに関連付けられているキャンペーンのサブセットのみのデータをクエリするには、データをクエリするキャンペーンごとに 1 つずつ、一連の個別のクエリを作成して実行するのが最善の方法です。前のセクションでは、1 つのキャンペーンのデータをクエリする方法について説明しています。

同じプロジェクトに関連付けられているすべてのキャンペーンのデータをクエリするには、アプリケーションメトリクス API を使用します。以下の必須のパラメータの値を指定します。

- application-id – プロジェクトの一意の識別子であるプロジェクト ID。Amazon Pinpoint では、プロジェクトとアプリケーションという用語は同じ意味です。
- kpi-name – クエリするメトリクスの名前。この値は、関連するメトリクスを記述し、ハイフンで区切られた小文字の英数字で構成される 2 つ以上の用語で構成されます。サポートされているメトリクスの完全なリストと各メトリクスの kpi-name 値については、「標準メトリクス」を参照してください。

日付範囲でデータをフィルタリングすることもできます。日付範囲を指定しない場合、Amazon Pinpoint は過去 31 暦日のデータを返します。異なる日付でデータをフィルタリングするには、サポートされている日付範囲パラメータを使用して、日付範囲の最初と最後の日時を指定します。値は拡張 ISO 8601 形式で、協定世界時 (UTC) を使用する必要があります (例: `2019-07-19T20:00:00Z` は、2019年7月19日午後8時 (UTC))。日付範囲は包括的であり、31 日以内に制限する必要があります。また、最初の日付と時刻は、現在の日付から 90 日未満である必要があります。

次の例は、Amazon Pinpoint REST API、AWS CLI、および AWS SDK for Java を使用して、キャンペーンの分析データをクエリする方法を示しています。サポートされている任意の AWS SDK を使用して、キャンペーンの分析データをクエリできます。これらの AWS CLI の例は、Microsoft Windows 用にフォーマットされています。Unix、Linux、macOS の場合は、キャレット (^) 行継続文字をバックスラッシュ (\) に置き換えます。

REST API

Amazon Pinpoint REST API を使用して複数のキャンペーンの分析データをクエリするには、HTTP(S) GET リクエストを Application Metrics URI に送信します。URI で、必要なパスパラメータに次の適切な値を指定します。

```
https://endpoint/v1/apps/application-id/kpis/daterange/kpi-name
```

各パラメータの意味は次のとおりです。

- *endpoint* は、キャンペーンに関連付けられたプロジェクトをホストする AWS リージョンの Amazon Pinpoint エンドポイントです。
- *application-id* は、キャンペーンに関連付けられているプロジェクトの一意の識別子です。
- *kpi-name* は、クエリするメトリクスの `kpi-name` 値です。

すべてのパラメータは、URL エンコードする必要があります。

特定の日付範囲のデータを取得するフィルターを適用するには、URI に `start-time` および `end-time` クエリパラメータと値を追加します。これらのパラメータを使用すると、データを取得する日付範囲の最初と最後の日付と時刻 (両端を含む) を、拡張 ISO 8601 形式で指定できます。パラメータを区切るには、アンパサンド (&) を使用します。

例えば、次のリクエストは、2019 年 7 月 19 日から 2019 年 7 月 26 日まで、プロジェクトの各キャンペーンによってメッセージが配信された一意のエンドポイントの数を取得します。

```
https://pinpoint.us-east-1.amazonaws.com/v1/apps/12345678901234567890123456789012345example/
kpis/daterange/unique-deliveries-grouped-by-campaign?start-
time=2019-07-19T00:00:00Z&end-time=2019-07-26T23:59:59Z
```

実行する条件は以下のとおりです。

- pinpoint.us-east-1.amazonaws.com は、プロジェクトをホストする AWS リージョンの Amazon Pinpoint エンドポイントです。
- 12345678901234567890123456789012345example は、キャンペーンに関連付けられているプロジェクトの一意の識別子です。
- unique-deliveries-grouped-by-campaign は、エンドポイント配信の *kpi-name* 値で、キャンペーンごとのアプリケーションメトリック (メッセージが配信されたユニークエンドポイントの数を返すメトリック) ごとにグループ化されています。
- 2019-07-19T00:00:00Z は、端の値を含む日付範囲の一部として、データを取得する最初の日付と時刻です。
- 2019-07-26T23:59:59Z は、端の値を含む日付範囲の一部として、データを取得する最後の日付と時刻です。

AWS CLI

AWS CLI を使用して複数のキャンペーンの分析データをクエリするには、get-application-date-range-kpi コマンドを使用して、必要なパラメータに適切な値を指定します。

```
C:\> aws pinpoint get-application-date-range-kpi ^
    --application-id application-id ^
    --kpi-name kpi-name
```

実行する条件は以下のとおりです。

- *application-id* は、キャンペーンに関連付けられているプロジェクトの一意の識別子です。
- *kpi-name* は、クエリするメトリクスの kpi-name 値です。

特定の日付範囲のデータを取得するフィルターを適用するには、start-time および end-time パラメータと値をクエリに含めます。これらのパラメータを使用すると、データを取得する日付範囲の最初と最後の日付と時刻 (両端を含む) を、拡張 ISO 8601 形式で指定できます。例えば、次のリクエストは、2019 年 7 月 19 日から 2019 年 7 月 26 日まで、プロジェクトの各キャンペーンによってメッセージが配信された一意のエンドポイントの数を取得します。

```
C:\> aws pinpoint get-application-date-range-kpi ^
    --application-id 1234567890123456789012345example ^
    --kpi-name unique-deliveries-grouped-by-campaign ^
    --start-time 2019-07-19T00:00:00Z ^
    --end-time 2019-07-26T23:59:59Z
```

実行する条件は以下のとおりです。

- 1234567890123456789012345example は、キャンペーンに関連付けられているプロジェクトの一意の識別子です。
- unique-deliveries-grouped-by-campaign は、エンドポイント配信の *kpi-name* 値で、キャンペーンごとのアプリケーションメトリック (メッセージが配信されたユニークエンドポイントの数を返すメトリック) ごとにグループ化されています。
- 2019-07-19T00:00:00Z は、端の値を含む日付範囲の一部として、データを取得する最初の日付と時刻です。
- 2019-07-26T23:59:59Z は、端の値を含む日付範囲の一部として、データを取得する最後の日付と時刻です。

SDK for Java

AWS SDK for Java を使用して複数のキャンペーンの解析データをクエリする場合は、Application Metrics API の GetApplicationDateRangeKpiRequest メソッドを使用します。必要なパラメータに適切な値を指定します。

```
GetApplicationDateRangeKpiRequest request = new GetApplicationDateRangeKpiRequest()
        .withApplicationId("applicationId")
        .withKpiName("kpiName")
```

実行する条件は以下のとおりです。

- *applicationId* は、キャンペーンに関連付けられているプロジェクトの一意の識別子です。
- *kpiName* は、クエリするメトリクスの kpi-name 値です。

特定の日付範囲のデータを取得するフィルターを適用するには、startTime および endTime パラメータと値をクエリに含めます。これらのパラメータを使用すると、データを取得する日付範囲の最初と最後の日付と時刻 (両端を含む) を、拡張 ISO 8601 形式で指定できます。例えば、次のリクエストは、2019 年 7 月 19 日から 2019 年 7 月 26 日まで、プロジェクトの各キャンペーンによってメッセージが配信された一意のエンドポイントの数を取得します。

```
GetApplicationDateRangeKpiRequest request = new GetApplicationDateRangeKpiRequest()
        .withApplicationId("12345678901234567890123456example")
        .withKpiName("unique-deliveries-grouped-by-campaign")
        .withStartTime(Date.from(Instant.parse("2019-07-19T00:00:00Z")))
        .withEndTime(Date.from(Instant.parse("2019-07-26T23:59:59Z")));
```

実行する条件は以下のとおりです。

- 12345678901234567890123456example は、キャンペーンに関連付けられているプロジェクトの一意の識別子です。
- unique-deliveries-grouped-by-campaign は、エンドポイント配信の *kpi-name* 値で、キャンペーンごとのアプリケーションメトリック (メッセージが配信されたユニークエンドポイントの数を返すメトリック) ごとにグループ化されています。
- 2019-07-19T00:00:00Z は、端の値を含む日付範囲の一部として、データを取得する最初の日付と時刻です。
- 2019-07-26T23:59:59Z は、端の値を含む日付範囲の一部として、データを取得する最後の日付と時刻です。

クエリを送信すると、Amazon Pinpoint はクエリ結果を JSON レスポンスで返します。結果の構造は、クエリしたメトリクスによって異なります。一部のメトリクスは 1 つの値しか返しません。その他のメトリクスは複数の値を返し、それらの値は関連するフィールドによってグループ化されます。メトリクスが複数の値を返す場合、JSON レスポンスにはデータのグループ化に使用されたフィールドを示すフィールドが含まれます。

例えば、前述の例で使用されているキャンペーンごとにグループ化されたエンドポイント配信 (unique-deliveries-grouped-by-campaign) アプリケーションメトリックは、プロジェクトに関連付けられた各キャンペーンについて、複数の値 (メッセージが配信されたユニークなエンドポイントの数) を返します。この場合、JSON レスポンスは次のようになります。

```
{
    "ApplicationDateRangeKpiResponse":{
        "ApplicationId":"12345678901234567890123456example",
```

```
        "EndTime":"2019-07-26T23:59:59Z",
        "KpiName":"unique-deliveries-grouped-by-campaign",
        "KpiResult":{
            "Rows":[
                {
                    "GroupedBys":[
                        {
                            "Key":"CampaignId",
                            "Type":"String",
                            "Value":"80b8efd84042ff8d9c96ce2f8example"
                        }
                    ],
                    "Values":[
                        {
                            "Key":"UniqueDeliveries",
                            "Type":"Double",
                            "Value":"123.0"
                        }
                    ]
                },
                {
                    "GroupedBys":[
                        {
                            "Key":"CampaignId",
                            "Type":"String",
                            "Value":"810c7aab86d42fb2b56c8c966example"
                        }
                    ],
                    "Values":[
                        {
                            "Key":"UniqueDeliveries",
                            "Type":"Double",
                            "Value":"456.0"
                        }
                    ]
                },
                {
                    "GroupedBys":[
                        {
                            "Key":"CampaignId",
                            "Type":"String",
                            "Value":"42d8c7eb0990a57ba1d5476a3example"
                        }
                    ],
```

```
                    "Values":[
                        {
                            "Key":"UniqueDeliveries",
                            "Type":"Double",
                            "Value":"789.0"
                        }
                    ]
                }
            ]
        },
        "StartTime":"2019-07-19T00:00:00Z"
    }
}
```

この場合、GroupedBys フィールドは値がキャンペーン ID (CampaignId) でグループ化されていることを示しています。

クエリ結果の構造の詳細については、「クエリ結果の使用」を参照してください。

トランザクションメッセージの Amazon Pinpoint 分析データのクエリ

Amazon Pinpoint コンソールの分析ページを使用することに加えて、Amazon Pinpoint Analytics API を使用して、プロジェクトに送信されたトランザクションメッセージの配信とエンゲージメントの傾向についてのインサイトを提供する標準メトリクスのサブセットについて分析データをクエリできます。

メトリクスは測定可能な値で、重要業績評価指標 (KPI) とも呼ばれ、トランザクションメッセージのパフォーマンスを監視および評価するのに役立ちます。例えば、メトリクスを使用して、送信した E メールのトランザクション数、送信した SMS メッセージ数、または受取人に配信されたメッセージ数を調べることができます。Amazon Pinpoint は、プロジェクトに送信するすべての E メールのトランザクションおよび SMS メッセージについて、このデータを自動的に収集し、集計します。これは、90 日間データを保存します。

Amazon Pinpoint 分析 API を使用してデータをクエリする場合、クエリの範囲、データ、グループ化、フィルターを定義するさまざまなオプションを選択できます。これを行うには、適用する日付ベースのフィルターに加えて、クエリするプロジェクトとメトリクスを指定するパラメータを使用します。

このトピックでは、これらのオプションを選択し、プロジェクトに関するトランザクションメッセージングデータをクエリする方法の例を説明します。

前提条件

トランザクションメッセージの分析データをクエリする前に、クエリの定義に使用する、次の情報を収集すると役立ちます。

- Project ID – メッセージの送信元のプロジェクトの一意の識別子。Amazon Pinpoint API では、この値は `application-id` プロパティに保存されます。Amazon Pinpoint コンソールでは、この値は すべてのプロジェクト ページでProject ID として表示されます。
- Date range – オプションで、データを問い合せる日付範囲の最初と最後の日付と時刻。日付範囲は包括的であり、31日以内に制限する必要があります。また、最初の日付と時刻は、現在の日付から 90 日未満である必要があります。日付範囲を指定しない場合、Amazon Pinpoint は過去 31 暦日のデータを自動的にクエリします。
- Metric – クエリするメトリクスの名前。具体的には、メトリクスの `kpi-name` 値。サポートされているメトリクスの完全なリストと各メトリクスの `kpi-name` 値については、「標準メトリクス」を参照してください。

また、関連するフィールドでデータをグループ化するかどうかを決定することもできます。する場合、データを自動的にグループ化するように設計されたメトリクスを選択することで、分析とレポート作成を簡素化できます。例えば、Amazon Pinpoint には、受信者に配信されたトランザクション SMS メッセージの数を報告する標準メトリクスがいくつか用意されています。これらのメトリクスの 1 つは、データを日付ごとに自動的にグループ化します (`txn-sms-delivered-grouped-by-date`)。別のメトリクスでは、データが国またはリージョンによって自動的にグループ化されます (`txn-sms-delivered-grouped-by-country`)。3 番目のメトリクスは、単純に一つの値、つまり受信者に配信されたメッセージングの数を返します (`txn-sms-delivered`)。必要な方法でデータをグループ化する標準メトリクスが見つからない場合は、必要なデータを返す一連のクエリを開発できます。その後、クエリ結果を手動で分類したり、デザインするカスタムグループに結合できます。

最後に、クエリするデータにアクセスする権限があることを確認することが重要です。詳細については、「Amazon Pinpoint 分析データのクエリ実行用の IAM ポリシー」を参照してください。

トランザクション E メールメッセージのデータのクエリ

プロジェクトに送信されたトランザクション E メールメッセージのデータをクエリするには、Application Metrics API を使用して、次の必須パラメータの値を指定します。

- application-id – プロジェクトの一意の識別子であるプロジェクト ID。Amazon Pinpoint では、プロジェクトとアプリケーションという用語は同じ意味です。
- kpi-name – クエリするメトリクスの名前。この値は、関連するメトリクスを記述し、ハイフンで区切られた小文字の英数字で構成される 2 つ以上の用語で構成されます。サポートされているメトリクスの完全なリストと各メトリクスの `kpi-name` 値については、「標準メトリクス」を参照してください。

また、特定の日付範囲のデータをクエリするフィルターを適用することもできます。日付範囲を指定しない場合、Amazon Pinpoint は過去 31 暦日のデータを返します。異なる日付でデータをフィルタリングするには、サポートされている日付範囲パラメータを使用して、日付範囲の最初と最後の日時を指定します。値は拡張 ISO 8601 形式で、協定世界時 (UTC) を使用する必要があります (例: `2019-09-06T20:00:00Z` は、2019 年 9 月 6 日午後 8 時 (UTC))。日付範囲は包括的であり、31 日以内に制限する必要があります。また、最初の日付と時刻は、現在の日付から 90 日未満である必要があります。

次の例は、Amazon Pinpoint REST API、AWS CLI、および AWS SDK for Java を使用して、トランザクション E メールメッセージの分析データをクエリする方法を示しています。任意のサポートされている AWS SDK を使用して、トランザクションメッセージの分析データをクエリできます。これらの AWS CLI の例は、Microsoft Windows 用にフォーマットされています。Unix、Linux、macOS の場合は、キャレット (^) 行継続文字をバックスラッシュ (\) に置き換えます。

REST API

Amazon Pinpoint REST API を使用してトランザクション E メールメッセージの分析データをクエリするには、HTTP(S) GET リクエストを Application Metrics URI に送信します。URI で、必要なパスパラメータに次の適切な値を指定します。

```
https://endpoint/v1/apps/application-id/kpis/daterange/kpi-name
```

各パラメータの意味は次のとおりです。

- *`endpoint`* は、プロジェクトをホストする AWS リージョンの Amazon Pinpoint エンドポイントです。
- *`application-id`* は、プロジェクトの一意の識別子です。
- *`kpi-name`* は、クエリするメトリクスの `kpi-name` 値です。

すべてのパラメータは、URL エンコードする必要があります。

特定の日付範囲のデータをクエリするフィルターを適用するには、URI に start-time および end-time クエリパラメータと値を追加します。これらのパラメータを使用すると、データを取得する日付範囲の最初と最後の日付と時刻 (両端を含む) を、拡張 ISO 8601 形式で指定できます。パラメータを区切るには、アンパサンド (&) を使用します。

例えば、次のリクエストは、2019 年 9 月 6 日から 2019 年 9 月 13 日までプロジェクトに送信されたトランザクション E メールメッセージの数を取得します。

```
https://pinpoint.us-east-1.amazonaws.com/v1/apps/12345678901234567890123456example/
kpis/daterange/txn-emails-sent?start-time=2019-09-06T00:00:00Z&end-
time=2019-09-13T23:59:59Z
```

実行する条件は以下のとおりです。

- pinpoint.us-east-1.amazonaws.com は、プロジェクトをホストする AWS リージョンの Amazon Pinpoint エンドポイントです。
- 12345678901234567890123456example は、プロジェクトの一意の識別子です。
- txn-emails-sent は、送信アプリケーションメトリクスのkpi-name 値で、プロジェクトのために送信されたトランザクション E メールメッセージの数を報告するメトリクスです。
- 2019-09-06T00:00:00Z は、端の値を含む日付範囲の一部として、データを取得する最初の日付と時刻です。
- 2019-09-13T23:59:59Z は、端の値を含む日付範囲の一部として、データを取得する最後の日付と時刻です。

AWS CLI

AWS CLI を使用してトランザクション E メールメッセージの分析データをクエリするには、get-application-date-range-kpi コマンドを使用して、必要なパラメータに適切な値を指定します。

```
C:\> aws pinpoint get-application-date-range-kpi ^
    --application-id application-id ^
    --kpi-name kpi-name
```

実行する条件は以下のとおりです。

- *application-id* は、プロジェクトの一意の識別子です。

- *kpi-name* は、クエリするメトリクスの kpi-name 値です。

特定の日付範囲のデータをクエリするフィルターを適用するには、start-time および end-time パラメータと値をクエリに追加します。これらのパラメータを使用すると、データを取得する日付範囲の最初と最後の日付と時刻 (両端を含む) を、拡張 ISO 8601 形式で指定できます。例えば、次のリクエストは、2019 年 9 月 6 日から 2019 年 9 月 13 日までプロジェクトに送信されたトランザクション E メールメッセージの数を取得します。

```
C:\> aws pinpoint get-application-date-range-kpi ^
    --application-id 12345678901234567890123456example ^
    --kpi-name txn-emails-sent ^
    --start-time 2019-09-06T00:00:00Z ^
    --end-time 2019-09-13T23:59:59Z
```

実行する条件は以下のとおりです。

- 12345678901234567890123456example は、プロジェクトの一意の識別子です。
- txn-emails-sent は、送信アプリケーションメトリクスのkpi-name 値で、プロジェクトのために送信されたトランザクション E メールメッセージの数を報告するメトリクスです。
- 2019-09-06T00:00:00Z は、端の値を含む日付範囲の一部として、データを取得する最初の日付と時刻です。
- 2019-09-13T23:59:59Z は、端の値を含む日付範囲の一部として、データを取得する最後の日付と時刻です。

SDK for Java

AWS SDK for Java を使用してトランザクション E メールメッセージの分析データを照会するには、Application Metrics API の GetApplicationDateRangeKpiRequest メソッドを使用します。必要なパラメータに適切な値を指定します。

```
GetApplicationDateRangeKpiRequest request = new GetApplicationDateRangeKpiRequest()
        .withApplicationId("applicationId")
        .withKpiName("kpiName")
```

実行する条件は以下のとおりです。

- *applicationId* は、プロジェクトの一意の識別子です。
- *kpiName* は、クエリするメトリクスの kpi-name 値です。

特定の日付範囲のデータをクエリするフィルターを適用するには、startTime および endTime パラメータと値をクエリに含めます。これらのパラメータを使用すると、データを取得する日付範囲の最初と最後の日付と時刻（両端を含む）を、拡張 ISO 8601 形式で指定できます。例えば、次のリクエストは、2019 年 9 月 6 日から 2019 年 9 月 13 日までプロジェクトに送信されたトランザクション E メールメッセージの数を取得します。

```
GetApplicationDateRangeKpiRequest request = new GetApplicationDateRangeKpiRequest()
        .withApplicationId("12345678901234567890123456789012345example")
        .withKpiName("txn-emails-sent")
        .withStartTime(Date.from(Instant.parse("2019-09-06T00:00:00Z")))
        .withEndTime(Date.from(Instant.parse("2019-09-13T23:59:59Z")));
```

実行する条件は以下のとおりです。

- 12345678901234567890123456789012345example は、プロジェクトの一意の識別子です。
- txn-emails-sent は、送信アプリケーションメトリクスのkpi-name 値で、プロジェクトのために送信されたトランザクション E メールメッセージの数を報告するメトリクスです。
- 2019-09-06T00:00:00Z は、端の値を含む日付範囲の一部として、データを取得する最初の日付と時刻です。
- 2019-09-13T23:59:59Z は、端の値を含む日付範囲の一部として、データを取得する最後の日付と時刻です。

クエリを送信すると、Amazon Pinpoint はクエリ結果を JSON レスポンスで返します。結果の構造は、クエリしたメトリクスによって異なります。一部のメトリクスは 1 つの値しか返しません。例えば、前述の例で使用されている、送信 (txn-emails-sent) アプリケーションメトリクスは、1 つの値 (プロジェクトから送信されたトランザクション E メールメッセージの数) を返します。この場合、JSON レスポンスは次のようになります。

```
{
    "ApplicationDateRangeKpiResponse":{
        "ApplicationId":"12345678901234567890123456789012345example",
        "EndTime":"2019-09-13T23:59:59Z",
        "KpiName":"txn-emails-sent",
        "KpiResult":{
            "Rows":[
                {
                    "Values":[
                        {
                            "Key":"TxnEmailsSent",
```

```
                        "Type":"Double",
                        "Value":"62.0"
                    }
                ]
            }
        ]
    },
    "StartTime":"2019-09-06T00:00:00Z"
  }
}
```

他のメトリクスは複数の値を返し、関連するフィールドで値をグループ化します。メトリクスが複数の値を返す場合、JSON レスポンスにはデータのグループ化に使用されたフィールドを示すフィールドが含まれます。

クエリ結果の構造の詳細については、「クエリ結果の使用」を参照してください。

トランザクション SMS メッセージのデータのクエリ

プロジェクトに送信されたトランザクション SMS メッセージのデータをクエリするには、Application Metrics API を使用して、次の必須パラメータの値を指定します。

- application-id – プロジェクトの一意の識別子であるプロジェクト ID。Amazon Pinpoint では、プロジェクトとアプリケーションという用語は同じ意味です。
- kpi-name – クエリするメトリクスの名前。この値は、関連するメトリクスを記述し、ハイフンで区切られた小文字の英数字で構成される 2 つ以上の用語で構成されます。サポートされているメトリクスの完全なリストと各メトリクスの `kpi-name` 値については、「標準メトリクス」を参照してください。

また、特定の日付範囲のデータをクエリするフィルターを適用することもできます。日付範囲を指定しない場合、Amazon Pinpoint は過去 31 暦日のデータを返します。異なる日付でデータをフィルタリングするには、サポートされている日付範囲パラメータを使用して、日付範囲の最初の日時と最後の日時を指定します。値は拡張 ISO 8601 形式で、協定世界時 (UTC) を使用する必要があります (例: `2019-09-06T20:00:00Z` は、2019 年 9 月 6 日午後 8 時 (UTC))。日付範囲は包括的であり、31 日以内に制限する必要があります。また、最初の日付と時刻は、現在の日付から 90 日未満である必要があります。

次の例は、Amazon Pinpoint REST API、AWS CLI 、および AWS SDK for Java を使用して、トランザクション SMS メッセージの分析データをクエリする方法を示しています。任意のサポートされて

いる AWS SDK を使用して、トランザクションメッセージの分析データをクエリできます。これらの AWS CLI の例は、Microsoft Windows 用にフォーマットされています。Unix、Linux、macOS の場合は、キャレット (^) 行継続文字をバックスラッシュ (\) に置き換えます。

REST API

Amazon Pinpoint REST API を使用してトランザクション SMS メッセージの分析データをクエリするには、HTTP(S) GET リクエストを Application Metrics URI に送信します。URI で、必要なパスパラメータに次の適切な値を指定します。

```
https://endpoint/v1/apps/application-id/kpis/daterange/kpi-name
```

各パラメータの意味は次のとおりです。

- *endpoint* は、プロジェクトをホストする AWS リージョンの Amazon Pinpoint エンドポイントです。
- *application-id* は、プロジェクトの一意の識別子です。
- *kpi-name* は、クエリするメトリクスの kpi-name 値です。

すべてのパラメータは、URL エンコードする必要があります。

特定の日付範囲のデータを取得するフィルターを適用するには、URI に start-time および end-time クエリパラメータと値を追加します。これらのパラメータを使用すると、データを取得する日付範囲の最初と最後の日付と時刻 (両端を含む) を、拡張 ISO 8601 形式で指定できます。パラメータを区切るには、アンパサンド (&) を使用します。

例えば、次のリクエストは、2019 年 9 月 6 日から 2019 年 9 月 8 日まで毎日送信されたトランザクション SMS メッセージの数を取得します。

```
https://pinpoint.us-east-1.amazonaws.com/v1/apps/12345678901234567890123456example/
kpis/daterange/txn-sms-sent-grouped-by-date?start-time=2019-09-06T00:00:00Z&end-
time=2019-09-08T23:59:59Z
```

実行する条件は以下のとおりです。

- pinpoint.us-east-1.amazonaws.com は、プロジェクトをホストする AWS リージョンの Amazon Pinpoint エンドポイントです。
- 12345678901234567890123456example は、プロジェクトの一意の識別子です。

- txn-sms-sent-grouped-by-date は、日付アプリケーションメトリクスでグループ化された送信の kpi-name 値で、日付範囲の各日に送信されたトランザクション SMS メッセージの数を返すメトリクスです。
- 2019-09-06T00:00:00Z は、端の値を含む日付範囲の一部として、データを取得する最初の日付と時刻です。
- 2019-09-08T23:59:59Z は、端の値を含む日付範囲の一部として、データを取得する最後の日付と時刻です。

AWS CLI

AWS CLI を使用してトランザクション SMS メッセージの分析データをクエリするには、get-application-date-range-kpi コマンドを使用して、必要なパラメータに適切な値を指定します。

```
C:\> aws pinpoint get-application-date-range-kpi ^
    --application-id application-id ^
    --kpi-name kpi-name
```

実行する条件は以下のとおりです。

- *application-id* は、プロジェクトの一意の識別子です。
- *kpi-name* は、クエリするメトリクスの kpi-name 値です。

特定の日付範囲のデータを取得するフィルターを適用するには、start-time および end-time パラメータと値をクエリに含めます。これらのパラメータを使用すると、データを取得する日付範囲の最初と最後の日付と時刻（両端を含む）を、拡張 ISO 8601 形式で指定できます。例えば、次のリクエストは、2019 年 9 月 6 日から 2019 年 9 月 8 日まで毎日送信されたトランザクション SMS メッセージの数を取得します。

```
C:\> aws pinpoint get-application-date-range-kpi ^
    --application-id 12345678901234567890123example ^
    --kpi-name txn-sms-sent-grouped-by-date ^
    --start-time 2019-09-06T00:00:00Z ^
    --end-time 2019-09-08T23:59:59Z
```

実行する条件は以下のとおりです。

- 12345678901234567890123example は、プロジェクトの一意の識別子です。

- txn-sms-sent-grouped-by-date は、日付アプリケーションメトリクスでグループ化された送信の kpi-name 値で、日付範囲の各日に送信されたトランザクション SMS メッセージの数を返すメトリクスです。
- 2019-09-06T00:00:00Z は、端の値を含む日付範囲の一部として、データを取得する最初の日付と時刻です。
- 2019-09-08T23:59:59Z は、端の値を含む日付範囲の一部として、データを取得する最後の日付と時刻です。

SDK for Java

AWS SDK for Java を使用してトランザクション SMS メッセージの分析データをクエリするには、Application Metrics API の GetApplicationDateRangeKpiRequest メソッドを使用して、必要なパラメータに適切な値を指定します。

```
GetApplicationDateRangeKpiRequest request = new GetApplicationDateRangeKpiRequest()
        .withApplicationId("applicationId")
        .withKpiName("kpiName")
```

実行する条件は以下のとおりです。

- *applicationId* は、プロジェクトの一意の識別子です。
- *kpiName* は、クエリするメトリクスの kpi-name 値です。

特定の日付範囲のデータを取得するフィルターを適用するには、startTime および endTime パラメータと値をクエリに含めます。これらのパラメータを使用すると、データを取得する日付範囲の最初と最後の日付と時刻 (両端を含む) を、拡張 ISO 8601 形式で指定できます。例えば、次のリクエストは、2019 年 9 月 6 日から 2019 年 9 月 8 日まで毎日送信されたトランザクション SMS メッセージの数を取得します。

```
GetApplicationDateRangeKpiRequest request = new GetApplicationDateRangeKpiRequest()
        .withApplicationId("12345678901234567890123456example")
        .withKpiName("txn-sms-sent-grouped-by-date")
        .withStartTime(Date.from(Instant.parse("2019-09-06T00:00:00Z")))
        .withEndTime(Date.from(Instant.parse("2019-09-08T23:59:59Z")));
```

実行する条件は以下のとおりです。

- 12345678901234567890123456example は、プロジェクトの一意の識別子です。

- txn-sms-sent-grouped-by-date は、日付アプリケーションメトリクスでグループ化された送信の kpi-name 値で、日付範囲の各日に送信されたトランザクション SMS メッセージの数を返すメトリクスです。
- 2019-09-06T00:00:00Z は、端の値を含む日付範囲の一部として、データを取得する最初の日付と時刻です。
- 2019-09-08T23:59:59Z は、端の値を含む日付範囲の一部として、データを取得する最後の日付と時刻です。

クエリを送信すると、Amazon Pinpoint はクエリ結果を JSON レスポンスで返します。結果の構造は、クエリしたメトリクスによって異なります。一部のメトリクスは 1 つの値しか返しません。他のメトリクスは複数の値を返し、関連するフィールドでそれらの値をグループ化します。メトリクスが複数の値を返す場合、JSON レスポンスにはデータのグループ化に使用されたフィールドを示すフィールドが含まれます。

例えば、前述の例で使用されている、日付 (txn-sms-sent-grouped-by-date) アプリケーションのメトリクスでグループ化された送信は、複数の値 (与えられた日付範囲の各日中に送信されたトランザクション SMS のメッセージ数) を返します。この場合、JSON レスポンスは次のようになります。

```
{
    "ApplicationDateRangeKpiResponse":{
        "ApplicationId":"12345678901234567890123456example",
        "EndTime":"2019-09-08T23:59:59Z",
        "KpiName":"txn-sms-sent-grouped-by-date",
        "KpiResult":{
            "Rows":[
                {
                    "GroupedBys":[
                        {
                            "Key":"Date",
                            "Type":"String",
                            "Value":"2019-09-06"
                        }
                    ],
                    "Values":[
                        {
                            "Key":"TxnSmsSent",
                            "Type":"Double",
                            "Value":"29.0"
                        }
```

```
                    ]
                },
                {
                    "GroupedBys":[
                        {
                            "Key":"Date",
                            "Type":"String",
                            "Value":"2019-09-07"
                        }
                    ],
                    "Values":[
                        {
                            "Key":"TxnSmsSent",
                            "Type":"Double",
                            "Value":"35.0"
                        }
                    ]
                },
                {
                    "GroupedBys":[
                        {
                            "Key":"Date",
                            "Type":"String",
                            "Value":"2019-09-08"
                        }
                    ],
                    "Values":[
                        {
                            "Key":"TxnSmsSent",
                            "Type":"Double",
                            "Value":"10.0"
                        }
                    ]
                }
            ]
        },
        "StartTime":"2019-09-06T00:00:00Z"
    }
}
```

この場合、GroupedBys フィールドは値が暦日 (Date) でグループ化されていることを示しています。これにより、以下のように処理されます。

- 29 件のメッセージが 2019 年 9 月 6 日に送信されました。
- 35 件のメッセージが 2019 年 9 月 7 日に送信されました。
- 10 件のメッセージが 2019 年 9 月 8 日に送信されました。

クエリ結果の構造の詳細については、「クエリ結果の使用」を参照してください。

Amazon Pinpoint 分析クエリ結果の使用

Amazon Pinpoint Analytics API を使用して分析データをクエリすると、Amazon Pinpoint は JSON レスポンスで結果を返します。アプリケーションメトリクス、キャンペーンメトリクス、およびジャーニーエンゲージメントメトリクスの場合、レスポンスのデータは、Amazon Pinpoint 分析データをレポートするための標準 JSON スキーマに準拠します。

つまり、選択したプログラミング言語またはツールを使用して、1 つ以上のこれらのメトリクスのデータをクエリし、各クエリの結果を取得して、結果をテーブル、オブジェクト、またはその他の場所に書き込むカスタムソリューションを実装できます。その後、別のサービスまたはアプリケーションを使用して、その場所でクエリ結果を操作できます。

例えば、以下のことが可能です。

- 一連のメトリクスを定期的にクエリし、好みのデータ可視化フレームワークを使用して結果を表示するカスタムダッシュボードを構築します。
- 適切なメトリクスをクエリし、その結果をグラフまたはユーザーが設計する他の種類のレポートに表示することで、エンゲージメント率を追跡するレポートを作成します。
- 解析データを解析して特定のストレージ形式に書き込み、結果を長期ストレージソリューションに移植します。

Amazon Pinpoint Analytics API は、Amazon Pinpoint プロジェクトや Amazon Pinpoint アカウントで後で読み込んだり使用したりできる永続オブジェクトを作成または保存するようには設計されていないことに注意してください。その代わりに、API は、分析データを取得し、そのデータを他のサービスやアプリケーションに転送して、さらに分析、保存、レポート作成を行うのに役立つように設計されています。これは、アプリケーションメトリクス、キャンペーンメトリクス、およびジャーニーエンゲージメントメトリクスに対してプログラミングによりクエリを実行できるすべての分析データに対して同じ JSON レスポンス構造とスキーマを使用して部分的に行います。

このトピックでは、アプリケーションメトリクス、キャンペーンメトリクス、またはジャーニーエンゲージメントメトリクスのクエリに対する JSON 応答の構造、オブジェクト、および項目について

説明します。ジャーニー実行メトリクスまたはジャーニーアクティビティ実行メトリクスのクエリに対する JSON レスポンスのフィールドについては、「Amazon Pinpoint の標準的な分析メトリクス」を参照してください。

JSON の構造

クエリ結果の解析と使用を支援するために、Amazon Pinpoint Analytics API では、アプリケーションメトリクス、キャンペーンメトリクス、およびジャーニーエンゲージメントメトリクスをプログラミングによりクエリを実行できるすべての Amazon Pinpoint 分析データに対して同じ JSON レスポンス構造を使用します。各 JSON レスポンスでは、プロジェクト ID (`ApplicationId`) など、クエリを定義した値を指定します。レスポンスには、1 つ (1 つだけ) の `KpiResult` オブジェクトも含まれます。`KpiResult` オブジェクトには、クエリの結果セット全体が含まれます。

各 `KpiResult` オブジェクトには `Rows` オブジェクトが含まれます。これは、クエリ結果とその結果の値に関する関連メタデータを含むオブジェクトの配列です。`Rows` オブジェクトの構造と内容には、次の一般的な特性があります。

- クエリ結果の各行は、`Rows` オブジェクト内にある `Values`という別個の JSON オブジェクトです。例えば、クエリが 3 つの値を返す場合、`Rows` オブジェクトには 3 つの `Values` オブジェクトが含まれます。各 `Values` オブジェクトには、クエリの個別の結果が含まれます。
- クエリ結果の各列は、適用される `Values` オブジェクトのプロパティです。列の名前は、`Values` オブジェクトの `Key` フィールドに保存されます。
- グループ化されたクエリ結果の場合、各 `Values` オブジェクトには関連付けられた `GroupedBys` オブジェクトがあります。`GroupedBys` オブジェクトは、結果をグループ化するために使用されたフィールドを示します。また、関連付けられた `Values` オブジェクトのグループ化値も提供します。
- メトリクスのクエリ結果が null の場合、`Rows` オブジェクトは空です。

これらの一般的な特性以外にも、`Rows` オブジェクトの構造と内容はメトリクスによって異なります。これは、Amazon Pinpoint が、単一値メトリクスと複数値メトリクスの 2 種類のメトリクスをサポートするためです。

単一値のメトリクスは、累積値を 1 つだけ提供します。例えば、キャンペーンのすべての実行によって受取人に配信されたメッセージの割合が挙げられます。複数値のメトリクスは、複数の値を提供し、それらの値を関連するフィールド別にグループ化します。例えば、キャンペーンの実行ごとに受取人に配信されたメッセージの割合を、キャンペーンの実行ごとにグループ化して示します。

メトリクスが単一値メトリクスか複数値メトリクスかは、メトリクスの名前を参照するとすばやく判断できます。名前に grouped-by が含まれていない場合は、単一値メトリクスです。含まれている場合は、複数値メトリクスです。プログラミングによりクエリを実行できるメトリクスの完全なリストについては、「Amazon Pinpoint の標準的な分析メトリクス」を参照してください。

単一値メトリクス

単一値メトリクスの場合、Rows オブジェクトには次の Values オブジェクトが含まれます。

- クエリされたメトリクスのフレンドリ名を指定します。
- クエリされたメトリクスに値を提供します。
- 返された値のデータ型を識別します。

例えば、次の JSON レスポンスには、単一値のメトリクスのクエリ結果が含まれます。このメトリクスは、2019 年 8 月 1 日から 2019 年 8 月 31 日まで、プロジェクトに関連付けられているすべてのキャンペーンによってメッセージが配信された一意のエンドポイントの数を示します。

```
{
    "ApplicationDateRangeKpiResponse":{
        "ApplicationId":"1234567890123456789012345example",
        "EndTime":"2019-08-31T23:59:59Z",
        "KpiName":"unique-deliveries",
        "KpiResult":{
            "Rows":[
                {
                    "Values":[
                        {
                            "Key":"UniqueDeliveries",
                            "Type":"Double",
                            "Value":"1368.0"
                        }
                    ]
                }
            ]
        },
        "StartTime":"2019-08-01T00:00:00Z"
    }
}
```

この例では、レスポンスは、2019 年 8 月 1 日から 2019 年 8 月 31 日までの間で、プロジェクトのすべてのキャンペーンで 1,368 個の一意のエンドポイントにメッセージを配信したことを示しています。

- Key は、値が Value フィールド (UniqueDeliveries) で指定されるメトリクスのフレンドリ名です。
- Type は、Value フィールド (Double) で指定された値のデータ型です。
- Value は、適用されたすべてのフィルター (1368.0) を含む、クエリされたメトリクスの実際の値です。

単一値メトリクスのクエリ結果が null (0 以上でない) の場合、Rows オブジェクトは空です。Amazon Pinpoint は、メトリクスに返すべきデータがない場合、メトリックに null 値を返します。例:

```
{
    "ApplicationDateRangeKpiResponse":{
        "ApplicationId":"2345678901234567890123456example",
        "EndTime":"2019-08-31T23:59:59Z",
        "KpiName":"unique-deliveries",
        "KpiResult":{
            "Rows":[

            ]
        },
        "StartTime":"2019-08-01T00:00:00Z"
    }
}
```

複数値メトリクス

複数値メトリクスの Rows オブジェクトの構造と内容は、単一値メトリクスとほとんど同じです。複数値のメトリクスの Rows オブジェクトには、Values オブジェクトも含まれます。Values オブジェクトは、クエリされたメトリクスのフレンドリ名を指定し、そのメトリクスの値を提供して、その値のデータ型を識別します。

ただし、複数値メトリクスの Rows オブジェクトには、1 つ以上の GroupedBy オブジェクトも含まれます。クエリ結果には、Values オブジェクトごとに 1 つの GroupedBy オブジェクトがあります。GroupedBy オブジェクトは、結果内のデータをグループ化するために使用されたフィールド

と、そのフィールドのデータ型を示します。また、そのフィールド (関連付けられた Values オブジェクト) のグループ化値も示します。

例えば、次の JSON レスポンスには、2019 年 8 月 1 日から 2019 年 8 月 31 日までの間で、プロジェクトに関連付けられた各キャンペーンについて、メッセージが配信された一意のエンドポイントの数を報告する複数値メトリクスのクエリ結果が含まれています。


```
{
    "ApplicationDateRangeKpiResponse":{
        "ApplicationId":"12345678901234567890123456example",
        "EndTime":"2019-08-31T23:59:59Z",
        "KpiName":"unique-deliveries-grouped-by-campaign",
        "KpiResult":{
            "Rows":[
                {
                    "GroupedBys":[
                        {
                            "Key":"CampaignId",
                            "Type":"String",
                            "Value":"80b8efd84042ff8d9c96ce2f8example"
                        }
                    ],
                    "Values":[
                        {
                            "Key":"UniqueDeliveries",
                            "Type":"Double",
                            "Value":"123.0"
                        }
                    ]
                },
                {
                    "GroupedBys":[
                        {
                            "Key":"CampaignId",
                            "Type":"String",
                            "Value":"810c7aab86d42fb2b56c8c966example"
                        }
                    ],
                    "Values":[
                        {
                            "Key":"UniqueDeliveries",
                            "Type":"Double",
                            "Value":"456.0"
```

```
                        }
                    ]
                },
                {
                    "GroupedBys":[
                        {
                            "Key":"CampaignId",
                            "Type":"String",
                            "Value":"42d8c7eb0990a57ba1d5476a3example"
                        }
                    ],
                    "Values":[
                        {
                            "Key":"UniqueDeliveries",
                            "Type":"Double",
                            "Value":"789.0"
                        }
                    ]
                }
            ]
        },
        "StartTime":"2019-08-01T00:00:00Z"
    }
}
```

この例では、2019 年 8 月 1 日から 2019 年 8 月 31 日までの間で、プロジェクトの 3 つのキャンペーンで一意のエンドポイントにメッセージを配信したという応答が示されます。これらのキャンペーンそれぞれの配信数の内訳は次のとおりです。

- キャンペーン 80b8efd84042ff8d9c96ce2f8example では、123 の一意のエンドポイントにメッセージを配信しました。
- キャンペーン 810c7aab86d42fb2b56c8c966example では、456 の一意のエンドポイントにメッセージを配信しました。
- キャンペーン 42d8c7eb0990a57ba1d5476a3example では、789 の一意のエンドポイントにメッセージを配信しました。

オブジェクトおよびフィールドの一般的な構造は次のとおりです。

- GroupedBys.Key – GroupedBys.Value フィールド (CampaignId) に指定されたグループ化値を保存するプロパティまたはフィールドの名前。

- GroupedBys.Type – GroupedBys.Value フィールド (String) で指定された値のデータ型。
- GroupedBys.Value – GroupedBys.Key フィールド (キャンペーン ID) で指定された、データのグループ化に使用されたフィールドの実際の値。
- Values.Key – 値が Values.Value フィールド (UniqueDeliveries) で指定されるメトリクスのフレンドリ名。
- Values.Type – Values.Value フィールド (Double) で指定された値のデータ型。
- Values.Value – 適用されたすべてのフィルターを含む、クエリされたメトリクスの実際の値。

特定のプロジェクト、キャンペーン、またはその他のリソースに対する、複数値メトリクスのクエリ結果が null (ゼロ以下) の場合、Amazon Pinpoint はリソースにオブジェクトまたはフィールドを返しません。複数値メトリクスのクエリ結果が、すべてのリソースで null の場合、Amazon Pinpoint は空の Rows オブジェクトを返します。

JSON オブジェクトとフィールド

プロジェクト ID (ApplicationId) などのクエリを定義した値を指定することに加えて、アプリケーションメトリクス、キャンペーンメトリック、またはジャーニーエンゲージメントメトリクスのクエリに対する各 JSON レスポンスには、KpiResult オブジェクトが含まれます。このオブジェクトには、クエリの全体的な結果セットが含まれ、この結果を解析して、別のサービスまたはアプリケーションに分析データを送信できます。各 KpiResult オブジェクトには、メトリクスにより、次の標準オブジェクトおよびフィールドの一部またはすべてが含まれます。

オブジェクトまたはフィールド	説明
Rows	クエリの結果セットを含むオブジェクトの配列。
Rows.GroupedBys	複数値メトリクスの場合、クエリ結果でデータをグループ化するために使用されたフィールドと値を定義するフィールドの配列。
Rows.GroupedBys.Key	複数値メトリクスの場合、GroupedBys.Value フィールドで指定された値を保存するプロパティまたはフィールドの名前。

オブジェクトまたはフィールド	説明
Rows.GroupedBys.Type	複数値メトリクスの場合、GroupedBys.Value フィールドで指定された値のデータ型。
Rows.GroupedBys.Value	複数値メトリクスの場合、クエリ結果のデータをグループ化するために使用されたフィールドの実際の値。この値は、関連付けられた Values オブジェクトに相関します。
Rows.Values	クエリ結果を含むフィールドの配列。
Rows.Values.Key	クエリされたメトリクスのフレンドリ名。メトリクスの値は Values.Value フィールドで指定されます。
Rows.Values.Type	Values.Value フィールドで指定された値のデータ型。
Rows.Values.Value	適用されたすべてのフィルターを含む、クエリされたメトリクスの実際の値。

ジャーニー実行メトリクスまたはジャーニーアクティビティ実行メトリクスのクエリに対する JSON レスポンスのフィールドについては、「Amazon Pinpoint の標準的な分析メトリクス」を参照してください。

を使用した Amazon Pinpoint API コールのログ記録 AWS CloudTrail

Amazon Pinpoint は AWS CloudTrail、Amazon Pinpoint の . CloudTrail captures API Amazon Pinpoint コールでユーザー、ロール、またはサービスによって実行されたアクションをイベントとして記録する AWS サービスである Amazon Pinpoint と統合されています。キャプチャされた呼び出しには、Amazon Pinpoint コンソールの呼び出しと、Amazon Pinpoint API オペレーションへのコード呼び出しが含まれます。

証跡を作成する場合は、Amazon Pinpoint の CloudTrail イベントなど、Amazon Simple Storage Service (Amazon S3) バケットへのイベントの継続的な配信を有効にすることができます。証跡を設定しない場合でも、 CloudTrail コンソールでイベント履歴を使用して最新のイベントを表示できます。で収集された情報を使用して CloudTrail、Amazon Pinpoint 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。

の設定と有効化の方法など 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。

の Amazon Pinpoint 情報 CloudTrail

CloudTrail AWS アカウントを作成すると、 がアカウントで有効になります。Amazon Pinpoint でサポートされているイベントアクティビティが発生すると、そのアクティビティは CloudTrail イベント履歴 の他の AWS サービスイベントとともにイベントに記録されます。 AWS アカウントで最近のイベントを表示、検索、ダウンロードできます。詳細については、「イベント履歴 を含む CloudTrail イベントの表示」を参照してください。

Amazon Pinpoint のイベントなど、 AWS アカウント内のイベントの継続的な記録については、証跡を作成します。証跡により CloudTrail 、 はログファイルを Amazon S3 バケットに配信できます。デフォルトでは、コンソールで証跡を作成すると、証跡はすべての AWS リージョンに適用されます。証跡は、 AWS パーティション内のすべてのリージョンからのイベントを口グに記録し、指定した Amazon S3 バケットにログファイルを配信します。さらに、 CloudTrail ログで収集されたイベントデータをさらに分析し、それに基づいて行動するように他の AWS サービスを設定できます。詳細については、次を参照してください:

- 「証跡作成の概要」
- CloudTrail がサポートするサービスと統合

- の Amazon SNS 通知の設定 CloudTrail
- 複数のリージョンからの CloudTrail ログファイルの受信と複数のアカウントからの CloudTrail ログファイルの受信

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。この ID 情報は以下のことを確認するのに役立ちます。

- リクエストがルート認証情報と AWS Identity and Access Management ユーザー認証情報のどちらを使用して行われたか。
- 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テンポラリ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 サービスによって行われたかどうか。

詳細については、「CloudTrail userIdentity 要素」を参照してください。

証跡を作成し、ログファイルを Amazon S3 バケットに必要な期間保存できます。また、Amazon S3 ライフサイクルのルールを定義して、自動的にログファイルをアーカイブまたは削除することもできます。デフォルトでは Amazon S3 のサーバー側の暗号化 (SSE) を使用して、ログファイルが暗号化されます。

ログファイルの配信を通知するには、新しいログファイルが配信されたときに Amazon SNS 通知を発行 CloudTrail するように を設定します。詳細については、「 の Amazon SNS 通知の設定 CloudTrail」を参照してください。

また、複数のリージョンと複数の AWS アカウントの Amazon Pinpoint ログファイル AWS を 1 つの Amazon S3 バケットに集約することもできます。詳細については、「複数のリージョンからの CloudTrail ログファイルの受信」および「複数のアカウントからの CloudTrail ログファイルの受信」を参照してください。

を使用して CloudTrail 、次の Amazon Pinpoint APIsのアクションをログに記録できます。

- Amazon Pinpoint API
- Amazon Pinpoint SMS および音声 API

によってログに記録できる Amazon Pinpoint API アクション CloudTrail

Amazon Pinpoint API は、以下のアクションをイベントとして CloudTrail ログファイルに記録することをサポートしています。

- CreateApp
- CreateCampaign
- CreateEmailTemplate
- CreateExportJob
- CreateImportJob
- CreateJourney
- CreatePushTemplate
- CreateRecommenderConfiguration
- CreateSegment
- CreateSmsTemplate
- CreateVoiceTemplate
- DeleteAdmChannel
- DeleteApnsChannel
- DeleteApnsSandboxChannel
- DeleteApnsVoipChannel
- DeleteApnsVoipSandboxChannel
- DeleteApp
- DeleteBaiduChannel
- DeleteCampaign
- DeleteEmailChannel
- DeleteEmailTemplate
- DeleteEndpoint
- DeleteEventStream
- DeleteGcmChannel

- DeleteJourney
- DeletePushTemplate
- DeleteRecommenderConfiguration
- DeleteSegment
- DeleteSmsChannel
- DeleteSmsTemplate
- DeleteUserEndpoints
- DeleteVoiceChannel
- DeleteVoiceTemplate
- GetAdmChannel
- GetApnsChannel
- GetApnsSandboxChannel
- GetApnsVoipChannel
- GetApnsVoipSandboxChannel
- GetApp
- GetApplicationDateRangeKpi
- GetApplicationSettings
- GetApps
- GetBaiduChannel
- GetCampaign
- GetCampaignActivities
- GetCampaignDateRangeKpi
- GetCampaignVersion
- GetCampaignVersions
- GetCampaigns
- GetChannels
- GetEmailChannel
- GetEmailTemplate

- GetEndpoint
- GetEventStream
- GetExportJob
- GetExportJobs
- GetGcmChannel
- GetImportJob
- GetImportJobs
- GetJourney
- GetJourneyDateRangeKpi
- GetJourneyExecutionActivityMetrics
- GetJourneyExecutionMetrics
- GetPushTemplate
- GetRecommenderConfiguration
- GetRecommenderConfigurations
- GetSegment
- GetSegmentExportJobs
- GetSegmentImportJobs
- GetSegmentVersion
- GetSegmentVersions
- GetSegments
- GetSmsChannel
- GetSmsTemplate
- GetUserEndpoints
- GetVoiceChannel
- GetVoiceTemplate
- ListJourneys
- ListTagsForResource
- ListTemplates

- ListTemplateVersions
- PhoneNumberValidate
- PutEventStream
- RemoveAttributes
- TagResource
- UntagResource
- UpdateAdmChannel
- UpdateApnsChannel
- UpdateApnsSandboxChannel
- UpdateApnsVoipChannel
- UpdateApnsVoipSandboxChannel
- UpdateApplicationSettings
- UpdateBaiduChannel
- UpdateCampaign
- UpdateEmailChannel
- UpdateEmailTemplate
- UpdateEndpoint
- UpdateEndpointsBatch
- UpdateGcmChannel
- UpdateJourney
- UpdateJourneyState
- UpdatePushTemplate
- UpdateRecommenderConfiguration
- UpdateSegment
- UpdateSmsChannel
- UpdateSmsTemplate
- UpdateTemplateActiveVersion
- UpdateVoiceChannel
- UpdateVoiceTemplate

次の Amazon Pinpoint API アクションは にログインされません CloudTrail。

- PutEvents
- SendMessages
- SendUsersMessages

によってログに記録できる Amazon Pinpoint E メール API アクション CloudTrail

Amazon Pinpoint Email API は、以下のアクションをイベントとして CloudTrail ログファイルに記録することをサポートしています。

- CreateConfigurationSet
- CreateConfigurationSetEventDestination
- CreateDedicatedIpPool
- CreateEmailIdentity
- DeleteConfigurationSet
- DeleteConfigurationSetEventDestination
- DeleteDedicatedIpPool
- DeleteEmailIdentity
- GetAccount
- GetConfigurationSet
- GetConfigurationSetEventDestinations
- GetDedicatedIp
- GetDedicatedIps
- GetEmailIdentity
- ListConfigurationSets
- ListDedicatedIpPools
- ListEmailIdentities
- PutAccountDedicatedIpWarmupAttributes
- PutAccountSendingAttributes

- PutConfigurationSetDeliveryOptions
- PutConfigurationSetReputationOptions
- PutConfigurationSetSendingOptions
- PutConfigurationSetTrackingOptions
- PutDedicatedIpInPool
- PutDedicatedIpWarmupAttributes
- PutEmailIdentityDkimAttributes
- PutEmailIdentityFeedbackAttributes
- PutEmailIdentityMailFromAttributes
- UpdateConfigurationSetEventDestination

次の Amazon Pinpoint Email API アクションは にログインされません CloudTrail。

- SendEmail

によって口グに記録できる Amazon Pinpoint SMS および音声 API バージョン 1 のアクション CloudTrail

Amazon Pinpoint SMS および音声バージョン 1 API は、以下のアクションをイベントとして CloudTrail 口グファイルに記録することをサポートしています。

- CreateConfigurationSet
- CreateConfigurationSetEventDestination
- DeleteConfigurationSet
- DeleteConfigurationSetEventDestination
- GetConfigurationSetEventDestinations
- UpdateConfigurationSetEventDestination

次の Amazon Pinpoint SMS および音声バージョン 1 API アクションは にログインされません CloudTrail。

- SendVoiceMessage

例: Amazon Pinpoint ログファイルエントリ

証跡は、指定した Amazon S3 バケットにイベントをログファイルとして配信できるようにする設定です。 CloudTrail ログファイルには 1 つ以上のログエントリが含まれます。イベント は、任意の送信元からの単一の要求を表します。これには、リクエストされたアクション、アクションの日時、リクエストパラメータなどに関する情報が含まれます。 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、Amazon Pinpoint API の `GetCampaigns`および `CreateCampaign`アクションを示す CloudTrail ログエントリを示しています。 Amazon Pinpoint

```
{
  "Records": [
    {
      "awsRegion": "us-east-1",
      "eventID": "example0-09a3-47d6-a810-c5f9fd2534fe",
      "eventName": "GetCampaigns",
      "eventSource": "pinpoint.amazonaws.com",
      "eventTime": "2018-02-03T00:56:48Z",
      "eventType": "AwsApiCall",
      "eventVersion": "1.05",
      "readOnly": true,
      "recipientAccountId": "123456789012",
      "requestID": "example1-b9bb-50fa-abdb-80f274981d60",
      "requestParameters": {
        "application-id": "example71dfa4c1aab66332a5839798f",
        "page-size": "1000"
      },
      "responseElements": null,
      "sourceIPAddress": "192.0.2.0",
      "userAgent": "Jersey/${project.version} (HttpUrlConnection 1.8.0_144)",
      "userIdentity": {
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "accountId": "123456789012",
        "arn": "arn:aws:iam::123456789012:root",
        "principalId": "123456789012",
        "sessionContext": {
          "attributes": {
            "creationDate": "2018-02-02T16:55:29Z",
            "mfaAuthenticated": "false"
          }
        },
```

```
        "type": "Root"
      }
    },
    {
      "awsRegion": "us-east-1",
      "eventID": "example0-09a3-47d6-a810-c5f9fd2534fe",
      "eventName": "CreateCampaign",
      "eventSource": "pinpoint.amazonaws.com",
      "eventTime": "2018-02-03T01:05:16Z",
      "eventType": "AwsApiCall",
      "eventVersion": "1.05",
      "readOnly": false,
      "recipientAccountId": "123456789012",
      "requestID": "example1-b9bb-50fa-abdb-80f274981d60",
      "requestParameters": {
        "Description": "***",
        "HoldoutPercent": 0,
        "IsPaused": false,
        "MessageConfiguration": "***",
        "Name": "***",
        "Schedule": {
          "Frequency": "ONCE",
          "IsLocalTime": true,
          "StartTime": "2018-02-03T00:00:00-08:00",
          "Timezone": "utc-08"
        },
        "SegmentId": "exampleda204adf991a80281aa0e591",
        "SegmentVersion": 1,
        "application-id": "example71dfa4c1aab66332a5839798f"
      },
      "responseElements": {
        "ApplicationId": "example71dfa4c1aab66332a5839798f",
        "CreationDate": "2018-02-03T01:05:16.425Z",
        "Description": "***",
        "HoldoutPercent": 0,
        "Id": "example54a654f80948680cbba240ede",
        "IsPaused": false,
        "LastModifiedDate": "2018-02-03T01:05:16.425Z",
        "MessageConfiguration": "***",
        "Name": "***",
        "Schedule": {
          "Frequency": "ONCE",
          "IsLocalTime": true,
          "StartTime": "2018-02-03T00:00:00-08:00",
```

```
          "Timezone": "utc-08"
        },
        "SegmentId": "example4da204adf991a80281example",
        "SegmentVersion": 1,
        "State": {
          "CampaignStatus": "SCHEDULED"
        },
        "Version": 1
      },
      "sourceIPAddress": "192.0.2.0",
      "userAgent": "aws-cli/1.14.9 Python/3.4.3 Linux/3.4.0+ botocore/1.8.34",
      "userIdentity": {
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "accountId": "123456789012",
        "arn": "arn:aws:iam::123456789012:user/userName",
        "principalId": "AIDAIHTHRCDA62EXAMPLE",
        "type": "IAMUser",
        "userName": "userName"
      }
    }
  ]
}
```

次の例は、Amazon Pinpoint SMS `CreateConfigurationSet` および音声 API の および `CreateConfigurationSetEventDestination`アクションを示す CloudTrail ログエントリを示しています。

```
{
  "Records": [
    {
      "eventVersion":"1.05",
      "userIdentity":{
        "type":"IAMUser",
        "principalId":"AIDAIHTHRCDA62EXAMPLE",
        "arn":"arn:aws:iam::111122223333:user/SampleUser",
        "accountId":"111122223333",
        "accessKeyId":"AKIAIOSFODNN7EXAMPLE",
        "userName":"SampleUser"
      },
      "eventTime":"2018-11-06T21:45:55Z",
      "eventSource":"sms-voice.amazonaws.com",
      "eventName":"CreateConfigurationSet",
      "awsRegion":"us-east-1",
```

```
      "sourceIPAddress":"192.0.0.1",
      "userAgent":"PostmanRuntime/7.3.0",
      "requestParameters":{
        "ConfigurationSetName":"MyConfigurationSet"
      },
      "responseElements":null,
      "requestID":"56dcc091-e20d-11e8-87d2-9994aexample",
      "eventID":"725843fc-8846-41f4-871a-7c52dexample",
      "readOnly":false,
      "eventType":"AwsApiCall",
      "recipientAccountId":"123456789012"
    },
    {
      "eventVersion":"1.05",
      "userIdentity":{
        "type":"IAMUser",
        "principalId":"AIDAIHTHRCDA62EXAMPLE",
        "arn":"arn:aws:iam::111122223333:user/SampleUser",
        "accountId":"111122223333",
        "accessKeyId":"AKIAIOSFODNN7EXAMPLE",
        "userName":"SampleUser"
      },
      "eventTime":"2018-11-06T21:47:08Z",
      "eventSource":"sms-voice.amazonaws.com",
      "eventName":"CreateConfigurationSetEventDestination",
      "awsRegion":"us-east-1",
      "sourceIPAddress":"192.0.0.1",
      "userAgent":"PostmanRuntime/7.3.0",
      "requestParameters":{
        "EventDestinationName":"CloudWatchEventDestination",
        "ConfigurationSetName":"MyConfigurationSet",
        "EventDestination":{
          "Enabled":true,
          "MatchingEventTypes":[
            "INITIATED_CALL",
            "INITIATED_CALL"
          ],
          "CloudWatchLogsDestination":{
            "IamRoleArn":"arn:aws:iam::111122223333:role/iamrole-01",
            "LogGroupArn":"arn:aws:logs:us-east-1:111122223333:log-
group:clientloggroup-01"
          }
        }
      },
```

```
      "responseElements":null,
      "requestID":"81de1e73-e20d-11e8-b158-d5536example",
      "eventID":"fcafc21f-7c93-4a3f-9e72-fca2dexample",
      "readOnly":false,
      "eventType":"AwsApiCall",
      "recipientAccountId":"111122223333"
    }
  ]
}
```

Amazon Pinpoint のリソースにタグを付ける

タグは、特定のタイプの AWS リソースなど、Amazon Pinpoint リソースをオプションで定義および関連付けるラベルです。タグを使用することで、目的、所有者、環境、その他の条件など、さまざまな方法でリソースを分類および管理できます。例えば、タグを使用してポリシーや自動化を適用したり、特定のコンプライアンス要件の対象となるリソースを識別したりできます。以下のタイプの Amazon Pinpoint リソースにタグを追加できます。

- キャンペーン
- メッセージテンプレート
- プロジェクト (アプリケーション)
- セグメント

リソースには、最大 50 個のタグを含めることができます。

タグの管理

タグはそれぞれ、1 つの必須タグキーとオプションの 1 つのタグ値で構成されており、どちらもお客様側が定義します。タグキーは、より具体的なタグ値のカテゴリのように動作する、一般的なラベルです。タグ値は、タグキーの記述子として機能します。

タグキーには最大 128 文字を含めることができます。タグ値は最大 256 文字を含めることができます。Unicode 文字、数字、空白、または次の記号のいずれか 1 つを使用できます。_ 。: / = + - さらに、タグには以下のような制限が追加されます。

- タグのキーと値は大文字と小文字が区別されます。
- タグキーは、関連付けられたリソースごとにそれぞれ一意である必要があります。また、設定できる値は 1 つのみです。
- `aws:` プレフィックスは AWS で使用されるためにリザーブされています。定義した任意のタグキーや値で使用することはできません。さらに、このプレフィックスを使用するタグキーまたは値は編集または削除できません。このプレフィックスを使用するタグは、リソースあたりのタグ数のクォータ (50 個) にはカウントされません。
- タグのみに基づいて、リソースを更新または削除することはできません。使用する操作に応じて、Amazon リソースネーム (ARN) またはリソース ID も指定する必要があります。

- パブリックリソースまたは共有リソースにタグを関連付けることができます。ただし、タグは自分の AWS アカウントにのみ使用でき、リソースを共有する他のアカウントには使用できません。さらに、AWS アカウント用に指定された AWS リージョンにあるリソースにのみタグを使用できます。

Amazon Pinpoint リソースからタグキーと値を追加、表示、更新、削除するには、AWS Command Line Interface (AWS CLI)、Amazon Pinpoint API、AWS Resource Groups タグ付け API、または AWS SDK を使用できます。AWS アカウントに指定された AWS リージョンにあるすべての AWS リソース (Amazon Pinpoint リソースなど) にわたってタグキーおよび値を管理するには、AWS Resource Groups タグ付け API を使用します。

IAMポリシーでタグを使用する

タグの実装を開始した後、タグベースのリソースレベルのアクセス許可を AWS Identity and Access Management (IAM) ポリシーと API オペレーションに適用できます。これには、リソースの作成時にリソースへのタグの追加をサポートするオペレーションが含まれます。このようにタグを使用することで、AWS アカウントのどのグループとユーザーにリソースの作成とタグ付けのアクセス許可があって、どのグループとユーザーにタグの作成、更新、削除のアクセス許可があるかを、広範囲にきめ細かく制御できます。

例えば、リソースの `Owner` タグの値がユーザー名となっているすべての Amazon Pinpoint リソースに対して、ユーザーにフルアクセスを許可するポリシーを作成できます。

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "ModifyResourceIfOwner",
         "Effect": "Allow",
         "Action": "mobiletargeting:*",
         "Resource": "*",
         "Condition": {
            "StringEqualsIgnoreCase": {
               "aws:ResourceTag/Owner": "${aws:username}"
            }
         }
      }
   ]
}
```

タグをベースにしてリソースレベルでアクセス許可を定義した場合、そのアクセス許可は即座に反映されます。つまり、リソースが作成されるとすぐにリソースの安全性が増し、新しいリソースにタグの使用をすぐに強制できるようになります。リソースレベルのアクセス許可を使用して、新しいリソースと既存のリソースに、どのタグキーと値を関連付けるかを制御することもできます。詳細については、AWS IAM ユーザーガイドの「タグを使用したアクセス制御」を参照してください。

リソースにタグを追加する

次の例は、AWS CLI と Amazon Pinpoint REST API を使用して、Amazon Pinpoint リソースにタグを追加する方法を示しています。サポートされている AWS SDK を使用して、リソースにタグを追加することもできます。

1 回の操作で複数の Amazon Pinpoint リソースにタグを追加するには、AWS CLI のリソースグループのタグ付け操作か、AWS Resource Groups タグ付け API を使用します。

API を使用したタグの追加

Amazon Pinpoint REST API を使用して新しいリソースを作成し、タグを追加するには、適切なリソース URI に POST リクエストを送信します。リクエストの本文で tags パラメータと値を含めます。次の例は、新しいプロジェクトを作成するときにタグを指定する方法を示しています。

```
POST /v1/apps HTTP/1.1
Host: pinpoint.us-east-1.amazonaws.com
Content-Type: application/x-www-form-urlencoded
Accept: application/json
Cache-Control: no-cache

{
   "Name":"MyProject",
   "tags":{
      "key1":"value1"
   }
}
```

既存のリソースにタグを追加するには、 タグ URI に POST リクエストを送信します。URI にリソースの Amazon リソースネーム (ARN) を含めます。ARN は URL エンコードする必要があります。リクエストの本文には、次の例に示すように、tags パラメータと値を含めます。

```
POST /v1/tags/resource-arn HTTP/1.1
```

```
Host: pinpoint.us-east-1.amazonaws.com
Content-Type: application/json
Accept: application/json
Cache-Control: no-cache

{
   "tags":{
      "key1":"value1"
   }
}
```

AWS CLI を使用したタグの追加

AWS CLI を使用して新しいリソースを作成し、タグを追加するには、そのリソースに適切な `create` コマンドを使用します。`tags` パラメータと値を含めます。次の例は、新しいプロジェクトを作成するときにタグを指定する方法を示しています。

Linux, macOS, or Unix

```
$ aws pinpoint create-app \
  --create-application-request '{
    "Name":"MyProject",
    "tags": {
      "key1":"value1",
      "key2":"value2"
    }
  }'
```

Windows Command prompt

```
C:\> aws pinpoint create-app ^
     --create-application-request Name=MyProject,tags={key1=value1,key2=value2}
```

前述の例で、以下を実行します。

- *MyProject* を、プロジェクトに付ける名前に置き換えます。
- *key1* と *key2* を、リソースに追加するタグのキーに置き換えます。
- *value1* と *value2* を、それぞれのキーに追加するタグの値に置き換えます。

Amazon Pinpoint リソースの作成に使用できるコマンドの詳細については、AWS CLI コマンドリファレンスを参照してください。

既存のリソースにタグを追加するには、tag-resource コマンドを使用して、必須のパラメータに適切な値を指定します。

Linux, macOS, or Unix

```
$ aws pinpoint tag-resource \
  --resource-arn resource-arn \
  --tags-model '{
    "tags": {
      "key1":"value1",
      "key2":"value2"
    }
  }'
```

Windows Command Prompt

```
C:\> aws pinpoint tag-resource ^
     --resource-arn resource-arn ^
     --tags-model tags={key1=value1,key2=value2}
```

前述の例で、以下を実行します。

- *resource-arn* を、タグを追加するリソースの Amazon リソースネーム (ARN) に置き換えます。
- *key1* と *key2* を、リソースに追加するタグのキーに置き換えます。
- *value1* と *value2* を、それぞれのキーに追加するタグの値に置き換えます。

リソースのタグを表示する

以下の例では、AWS CLI および Amazon Pinpoint REST API を使用して、 リソースに関連付けられているすべてのタグ (キーと値) を一覧表示する方法を示します。サポートされている AWS SDK を使用して、リソースに関連付けられているタグを表示することもできます。

API を使用したタグの表示

Amazon Pinpoint REST API を使用して、特定のリソースに関連付けられているすべてのタグを表示するには、GET リクエストを Tags URI に送信し、その URI にリソースの Amazon リソースネーム (ARN) を含めます。ARN は URL エンコードする必要があります。例えば、以下のリクエストは、指定されたキャンペーン (*resource-arn*) に関連付けられているすべてのタグを取得します。

```
GET /v1/tags/resource-arn HTTP/1.1
Host: pinpoint.us-east-1.amazonaws.com
Content-Type: application/json
Accept: application/json
Cache-Control: no-cache
```

リクエストへの JSON レスポンスには、tags オブジェクトが含まれます。tags オブジェクトで、キャンペーンに関連付けられているすべてのタグキーと値が一覧表示されます。

同じタイプの複数のリソースに関連付けられているタグをすべて表示するには、そのタイプのリソースの適切な URI に GET リクエストを送信します。例えば、以下のリクエストは、特定のプロジェクト (*application-id*) にあるすべてのキャンペーンに関する情報を取得します。

```
GET /v1/apps/application-id/campaigns HTTP/1.1
Host: pinpoint.us-east-1.amazonaws.com
Content-Type: application/json
Accept: application/json
Cache-Control: no-cache
```

リクエストへの JSON レスポンスで、そのプロジェクトにあるすべてのキャンペーンが一覧表示されます。各キャンペーンの tags オブジェクトにより、そのキャンペーンに関連付けられたすべてのタグキーと値が一覧表示されます。

AWS CLI を使用したタグの表示

AWS CLI を使用して、特定のリソースに関連付けられているタグのリストを表示するには、次の例に示すように、list-tags-for-resource コマンドを実行し、resource-arn パラメータにリソースの Amazon リソースネーム (ARN) を指定します。

Linux, macOS, or Unix

```
$ aws pinpoint list-tags-for-resource \
```

```
  --resource-arn resource-arn
```

Windows Command Prompt

```
C:\> aws pinpoint list-tags-for-resource ^
     --resource-arn resource-arn
```

タグの付いたすべての Amazon Pinpoint リソースと、それらのリソースのそれぞれに関連付けられたすべてのタグのリストを表示するには、AWS Resource Groups タグ付け API の get-resources コマンドを使用します。次のコード例に示すように、`resource-type-filters` パラメータ を `mobiletargeting` に設定します。

Linux, macOS, or Unix

```
$ aws resourcegroupstaggingapi get-resources \
     --resource-type-filters "mobiletargeting"
```

Windows Command Prompt

```
C:\> aws resourcegroupstaggingapi get-resources ^
     --resource-type-filters "mobiletargeting"
```

コマンドのアウトプットは、タグを持つすべての Amazon Pinpoint リソースの ARN のリストです。リストには、各リソースに関連付けられたタグキーと値がすべて含まれます。

リソースのタグを更新する

Amazon Pinpoint リソースのタグを更新する (上書きする) には、いくつかの方法があります。タグを更新する最適な方法は、次の要素に応じて異なります。

- 更新するタグのリソースのタイプ。
- 1 つのリソースのタグを更新するか、または複数のリソースのタグを同時に更新するか。
- タグキーまたはタグ値を更新するのか、またはその両方を更新するのか。

Amazon Pinpoint プロジェクトのタグを更新するか、複数のリソースのタグを同時に更新するには、AWS CLI または AWS Resource Groups タグ付け API のリソースグループのタグ付けオペレー

ションを使用します。現在、Amazon Pinpoint API はこれらのタスクについて、直接はサポートしていません。

あるリソースのタグを更新するには、Amazon Pinpoint API を使用して現在のタグを削除し、新しいタグを追加します。

リソースからのタグの削除

次の例は、AWS CLI と Amazon Pinpoint REST API を使用して、Amazon Pinpoint リソースにタグ (キーと値) を削除する方法を示しています。また、サポートされている AWS SDK を使用して、リソースからタグを削除することもできます。

1 回のオペレーションで複数の Amazon Pinpoint リソースからタグを削除するには、AWS CLI のリソースグループのタグ付け操作か、AWS Resource Groups タグ付け API を使用します。タグキーではなく、特定のタグ値のみをリソースから削除するには、そのリソースのタグを更新します。

API を使用したタグの削除

Amazon Pinpoint REST API を使用してリソースからタグを削除するには、DELETE リクエストを Tags URI に送信します。この URI では、タグを削除するリソースの Amazon リソースネーム (ARN) を含め、その後に `tagKeys` パラメータと削除するタグを続けます。例:

```
https://endpoint/v1/tags/resource-arn?tagKeys=key
```

実行する条件は以下のとおりです。

- *`endpoint`* は、リソースをホストする AWS リージョンの Amazon Pinpoint エンドポイントです。
- *`resource-arn`* は、タグを削除するリソースの ARN です。
- *`key`* は、リソースから削除するタグです。

すべてのパラメータは、URL エンコードする必要があります。

複数のタグキーとそれらに関連付けられた値の両方をリソースから削除するには、削除するさらなるタグそれぞれの `tagKeys` パラメータと引数を、アンパサンド (&) で区切って追加します。例:

```
https://endpoint/v1/tags/resource-arn?tagKeys=key1&tagKeys=key2
```

すべてのパラメータは、URL エンコードする必要があります。

AWS CLI を使用したタグの削除

AWS CLI を使用してリソースからタグを削除するには、`untag-resource` コマンドを実行します。次の例に示すように、`tag-keys` パラメータと引数を含めます。

Linux, macOS, or Unix

```
$ aws pinpoint untag-resource \
  --resource-arn resource-arn \
  --tag-keys key1 key2
```

Windows Command Prompt

```
C:\> aws pinpoint untag-resource ^
     --resource-arn resource-arn ^
     --tag-keys key1 key2
```

上の例に、以下の変更を加えます。

- *resource-arn* を、タグを削除するリソースの ARN に置き換えます。
- *key1* と *key2* を、リソースから削除するタグのキーに置き換えます。

関連情報

Amazon Pinpoint リソースの管理に使用できる CLI コマンドの詳細については、「AWS CLI コマンドリファレンス」の「Amazon Pinpoint」セクションを参照してください。

サポート対象の HTTP(S) メソッド、パラメータ、スキーマなど、Amazon Pinpoint API のリソースに関する詳細については、「Amazon Pinpoint API リファレンス」を参照してください。

AWS Lambda を使用した推奨事項のカスタマイズ

Amazon Pinpoint では、推奨モデルからパーソナライズされた推奨事項を取得して、キャンペーンおよびジャーニーから送信するメッセージに追加できます。推奨モデルは、データ内のパターンを検出し、検出されたパターンに基づいて予測と推奨事項を生成する機械学習 (ML) モデルの一種です。これにより、特定のユーザーが製品や商品の特定のセットから好むものが予測され、その情報がユーザーに対する推奨事項のセットとして提供されます。

Amazon Pinpoint で推奨モデルを使用して、各受信者の属性と動作に基づいてパーソナライズされた推奨事項をメッセージ受信者に送信できます。また、AWS Lambdaでは、これらの推奨事項をカスタマイズおよび強化することもできます。たとえば、推奨事項を 1 つのテキスト値 (製品名や ID など) からより洗練されたコンテンツ (製品名、説明、イメージ画像など) に動的に変換できます。さらに、Amazon Pinpoint がメッセージを送信するときにその変換をリアルタイムで実行できます。

この機能は以下の AWS リージョンで利用できます。米国東部 (バージニア北部)、米国西部 (オレゴン)、アジアパシフィック (ムンバイ)、アジアパシフィック (シドニー)、欧州 (アイルランド)

トピック

メッセージで推奨事項を使用する

Amazon Pinpoint で推奨モデルを使用するには、まず Amazon Personalize ソリューションを作成し、そのソリューションを Amazon Personalize キャンペーンとしてデプロイします。次に、Amazon Pinpoint で推奨モデルの構成を作成します。構成では、Amazon Personalize キャンペーンから推奨事項データを取得して処理する方法を決定する設定を指定します。これには、取得したデータの追加処理を実行する AWS Lambda 関数を呼び出すかどうかが含まれます。

Amazon Personalize は、アプリケーションを利用する顧客に対して、リアルタイムでパーソナライズされた推奨事項を提供するMLモデルの作成を支援するための AWS サービスです。Amazon Personalize では、ML モデルを作成してトレーニングし、Amazon Personalize キャンペーンとしてモデルを準備してデプロイするプロセスを説明します。その後、キャンペーンからパーソナライズさ

れたリアルタイムの推奨事項を取得できます。Amazon Personalize の詳細については、『Amazon Personalize デベロッパーガイド』を参照してください。。

AWS Lambda はサーバーをプロビジョニングしたり管理しなくてもコードを実行するために使用できるコンピューティングサービスです。コードをパッケージ化し、[Lambda 関数] として AWS Lambda にアップロードします。関数が呼び出されると AWS Lambda は関数を実行します。関数は、ユーザーが手動で呼び出したり、イベントに応答して自動的に呼び出したり、Amazon Pinpoint などのアプリケーションやサービスからのリクエストに応答したりすることができます。Lambda 関数の作成および呼び出しについては、『AWS Lambda デベロッパーガイド』を参照してください。

推奨モデルの Amazon Pinpoint 構成を作成した後、キャンペーンおよびジャーニーから送信するメッセージにモデルからの推奨事項を追加できます。これを行うには、推奨属性のメッセージ変数を含むメッセージテンプレートを使用します。推奨属性は、推奨事項データを保存するよう設計された動的エンドポイントまたはユーザー属性です。これらの属性は、推奨モデルの構成を作成するときに定義します。

次のタイプのメッセージテンプレートでは、推奨属性の変数を使用できます。

- メールテンプレート。キャンペーンまたはジャーニーから送信する E メールメッセージ用。
- プッシュ通知テンプレート。キャンペーンから送信するプッシュ通知用。
- SMS テンプレート。キャンペーンから送信する SMS テキストメッセージ用。

Amazon Pinpoint での推奨モデルの使用に関する詳細は、『Amazon Pinpoint ユーザーガイド』の「機械学習モデル」を参照してください。

推奨事項データを処理する Lambda 関数を呼び出すよう Amazon Pinpoint を設定した場合、Amazon Pinpoint は、キャンペーンまたはジャーニーのメッセージでパーソナライズされた推奨事項を送信するたびに、次の一般的なタスクを実行します。

1. メッセージおよびメッセージテンプレートの構成設定と内容を評価して処理します。
2. メッセージテンプレートが推奨モデルに接続されていることを判断します。
3. モデルに接続してそれを使用するための構成設定を評価します。これらは、モデルの「推奨モデル」リソースによって定義されます。
4. モデルの構成設定で定義されている推奨属性のメッセージ変数を 1 つ以上検出します。
5. モデルの構成設定で指定された Amazon Personalize キャンペーンから推奨事項データを取得します。このタスクを実行するために、Amazon Personalize Runtime API のGetRecommendations オペレーションを使用します。

6. 各メッセージ受信者の動的な推奨属性 (RecommendationItems) に適切な推奨事項データを追加します。
7. Lambda 関数を呼び出して、処理のために各受信者の推奨事項データをその関数に送信します。

   データは、各受信者のエンドポイント定義を含む JSON オブジェクトとして送信されます。各エンドポイント定義には、1～5 個の値の順序付き配列を含む RecommendationItems フィールドが含まれます。配列の値の数は、モデルの構成設定によって異なります。
8. Lambda 関数がデータを処理して結果を返すのを待機します。

   結果は、各受信者の更新されたエンドポイント定義を含む JSON オブジェクトです。更新された各エンドポイント定義には、新しい Recommendations オブジェクトが含まれます。このオブジェクトは、モデルの構成設定で定義したカスタム推奨属性ごとに 1～10 個のフィールドを含んでいます。これらの各フィールドには、エンドポイントの強化された推奨事項データが保存されます。
9. 各受信者の更新されたエンドポイント定義を使用して、各メッセージ変数をその受信者の適切な値に置き換えます。
10. メッセージ受信者ごとにパーソナライズされた推奨事項を含むメッセージのバージョンを送信します。

この方法で推奨事項をカスタマイズおよび強化するには、最初に、Amazon Pinpoint から送信されたエンドポイント定義を処理し、更新されたエンドポイント定義を返す Lambda 関数を作成します。次に、Lambda 関数ポリシーを関数に割り当てて、関数を呼び出すことを Amazon Pinpoint に許可します。その後、Amazon Pinpoint で推奨モデルを設定します。モデルを設定する際に、呼び出す関数を指定し、使用する推奨属性を定義します。

Lambda 関数の作成

Lambda 関数の作成方法については、『AWS Lambda デベロッパーガイド』の「開始方法」を参照してください。関数を設計および開発する際には、次の要件とガイドラインに留意してください。

入力イベントデータ

Amazon Pinpoint が推奨モデルの Lambda 関数を呼び出すと、メッセージを送信しているキャンペーンまたはジャーニーの構成およびその他の設定を含むペイロードが送信されます。ペイロードには、Endpoints オブジェクトが含まれます。このオブジェクトは、エンドポイント ID をメッセージ受信者のエンドポイント定義に関連付けるマップです。

エンドポイント定義は、Amazon Pinpoint API のエンドポイントリソースで定義された構造を使用します。ただし、RecommendationItems という名前の動的な推奨属性のフィールドも含まれます。この RecommendationItems フィールドには、Amazon Personalize キャンペーンから返された、エンドポイントに対する推奨商品が 1 つ以上含まれます。このフィールドの値は、1~5 の推奨項目の順序付き配列(文字列)です。配列内の商品数は、エンドポイントまたはユーザーごとに取得するよう Amazon Pinpoint を設定した推奨商品の数によって異なります。

例:

```
"Endpoints": {
    "endpointIDexample-1":{
        "ChannelType":"EMAIL",
        "Address":"sofiam@example.com",
        "EndpointStatus":"ACTIVE",
        "OptOut":"NONE",
        "EffectiveDate":"2020-02-26T18:56:24.875Z",
        "Attributes":{
            "AddressType":[
                "primary"
            ]
        },
        "User":{
            "UserId":"SofiaMartínez",
            "UserAttributes":{
                "LastName":[
                    "Martínez"
                ],
                "FirstName":[
                    "Sofia"
                ],
                "Neighborhood":[
                    "East Bay"
                ]
            }
        },
        "RecommendationItems":[
            "1815",
            "2009",
            "1527"
        ],
        "CreationDate":"2020-02-26T18:56:24.875Z"
    },
    "endpointIDexample-2":{
```

```
        "ChannelType":"EMAIL",
        "Address":"alejandror@example.com",
        "EndpointStatus":"ACTIVE",
        "OptOut":"NONE",
        "EffectiveDate":"2020-02-26T18:56:24.897Z",
        "Attributes":{
            "AddressType":[
                "primary"
            ]
        },
        "User":{
            "UserId":"AlejandroRosalez",
            "UserAttributes":{
                "LastName ":[
                    "Rosalez"
                ],
                "FirstName":[
                    "Alejandro"
                ],
                "Neighborhood":[
                    "West Bay"
                ]
            }
        },
        "RecommendationItems":[
            "1210",
            "6542",
            "4582"
        ],
        "CreationDate":"2020-02-26T18:56:24.897Z"
    }
}
```

上記の例では、関連する Amazon Pinpoint 設定は次のとおりです。

- 推奨モデルは、エンドポイントまたはユーザーごとに 3 つの推奨商品を取得するよう設定されています (RecommendationsPerMessage プロパティの値は 3 に設定されます)。この設定では、Amazon Pinpoint は、各エンドポイントまたはユーザーの 1 番目、2 番目、および 3 番目の推奨商品のみを取得して追加します。
- プロジェクトは、各ユーザーの名、姓、およびユーザーの住んでいる近辺を保存するカスタムユーザー属性を使用するよう設定されます (UserAttributes オブジェクトには、これらの属性の値が含まれます)。

- プロジェクトは、エンドポイントがプロジェクトからメッセージを受信するためのユーザーの優先アドレス (チャネル) かどうかを示すカスタムエンドポイント属性 (`AddressType`) を使用するよう設定されます (`Attributes` オブジェクトには、この属性の値が含まれます)。

Amazon Pinpoint が Lambda 関数を呼び出し、このペイロードをイベントデータとして送信すると、AWS Lambda は処理のためにそのデータを Lambda 関数に渡します。

各ペイロードには、最大 50 のエンドポイントのデータを含めることができます。セグメントに 50 以上のエンドポイントが含まれる場合、Amazon Pinpoint はすべてのデータが処理されるまで、繰り返し関数を呼び出します (最大で一度に 50 のエンドポイント)。

レスポンスデータと要件

Lambda 関数を設計および開発する際には、「機械学習モデルのクォータ」に注意してください。関数がこれらのクォータで定義された条件を満たしていない場合、Amazon Pinpoint はメッセージを処理して送信できません。

また、次の要件にも注意してください。

- 関数は、更新されたエンドポイント定義を、入力イベントデータによって提供されたものと同じ形式で返す必要があります。
- 更新された各エンドポイント定義には、エンドポイントまたはユーザーに対する 1 ~ 10 のカスタム推奨属性を含めることができます。これらの属性の名前は、Amazon Pinpoint で推奨モデルを設定するときに指定した属性名と一致する必要があります。
- すべてのカスタム推奨属性は、エンドポイントまたはユーザーごとに 1 つの `Recommendations` オブジェクトに返す必要があります。この要件は、名前の競合が発生しないようにするのに役立ちます。`Recommendations` オブジェクトは、エンドポイント定義の任意の場所に追加できます。
- 各カスタム推奨属性の値は、文字列 (1 つの値) または文字列の配列 (複数の値) である必要があります。値が文字列の配列である場合は、 `RecommendationItems` フィールドに示されているように、Amazon Personalize が返した推奨商品の順序を維持することをお勧めします。そうでない場合は、エンドポイントまたはユーザーに対するモデルの予測がコンテンツに反映されないことがあります。
- 関数は、エンドポイントまたはユーザーの他の属性値を含む、イベントデータ内の他の要素を変更しないようにします。カスタム推奨属性の値のみを追加し、返すようにする必要があります。Amazon Pinpoint は、関数のレスポンスに含まれる他の値の更新を受け付けません。

- 関数は、関数を呼び出している Amazon Pinpoint プロジェクトと同じ AWS リージョンでホストされている必要があります。関数とプロジェクトが同じリージョンにない場合、Amazon Pinpoint はイベントデータを関数に送信できません。

上記の要件のいずれかが満たされていない場合、Amazon Pinpoint はメッセージを処理して 1 つ以上のエンドポイントに送信できません。これにより、キャンペーンやジャーニーアクティビティが失敗する可能性があります。

最後に、関数に対して 256 回の同時実行数を予約することをお勧めします。

全体として、Lambda 関数は Amazon Pinpoint によって送信されたイベントデータを処理し、変更されたエンドポイント定義を返す必要があります。関数は、これを行うために Endpoints オブジェクト内の各エンドポイントを反復処理し、各エンドポイントに対して、使用するカスタム推奨属性の値を作成して設定します。Python で記述され、前述の入力イベントデータの例を続行する次の例のハンドラーで、これを示します。

```
import json
import string

def lambda_handler(event, context):
    print("Received event: " + json.dumps(event))
    print("Received context: " +  str(context))
    segment_endpoints = event["Endpoints"]
    new_segment = dict()
    for endpoint_id in segment_endpoints.keys():
        endpoint = segment_endpoints[endpoint_id]
        if supported_endpoint(endpoint):
            new_segment[endpoint_id] = add_recommendation(endpoint)

    print("Returning endpoints: " + json.dumps(new_segment))
    return new_segment

def supported_endpoint(endpoint):
    return True

def add_recommendation(endpoint):
    endpoint["Recommendations"] = dict()

    customTitleList = list()
    customGenreList = list()
    for i,item in enumerate(endpoint["RecommendationItems"]):
```

```
        item = int(item)
        if item == 1210:
            customTitleList.insert(i, "Hanna")
            customGenreList.insert(i, "Action")
        elif item == 1527:
            customTitleList.insert(i, "Catastrophe")
            customGenreList.insert(i, "Comedy")
        elif item == 1815:
            customTitleList.insert(i, "Fleabag")
            customGenreList.insert(i, "Comedy")
        elif item == 2009:
            customTitleList.insert(i, "Late Night")
            customGenreList.insert(i, "Drama")
        elif item == 4582:
            customTitleList.insert(i, "Agatha Christie\'s The ABC Murders")
            customGenreList.insert(i, "Crime")
        elif item == 6542:
            customTitleList.insert(i, "Hunters")
            customGenreList.insert(i, "Drama")

    endpoint["Recommendations"]["Title"] = customTitleList
    endpoint["Recommendations"]["Genre"] = customGenreList

    return endpoint
```

前述の例では、AWS Lambda はイベントデータを event パラメータとしてハンドラーに渡します。ハンドラーは、Endpoints オブジェクトの各エンドポイントを反復処理し、Recommendations.Title および Recommendations.Genre という名前のカスタム推奨属性の値を設定します。return ステートメントは、更新された各エンドポイント定義を Amazon Pinpoint に返します。

前述の入力イベントデータの例を続行して、更新されたエンドポイント定義は次のとおりです。

```
"Endpoints":{
    "endpointIDexample-1":{
        "ChannelType":"EMAIL",
        "Address":"sofiam@example.com",
        "EndpointStatus":"ACTIVE",
        "OptOut":"NONE",
        "EffectiveDate":"2020-02-26T18:56:24.875Z",
        "Attributes":{
            "AddressType":[
                "primary"
```

```
            ]
        },
        "User":{
            "UserId":"SofiaMartínez",
            "UserAttributes":{
                "LastName":[
                    "Martínez"
                ],
                "FirstName":[
                    "Sofia"
                ],
                "Neighborhood":[
                    "East Bay"
                ]
            }
        },
        "RecommendationItems":[
            "1815",
            "2009",
            "1527"
        ],
        "CreationDate":"2020-02-26T18:56:24.875Z",
        "Recommendations":{
            "Title":[
                "Fleabag",
                "Late Night",
                "Catastrophe"
            ],
            "Genre":[
                "Comedy",
                "Comedy",
                "Comedy"
            ]
        }
    },
    "endpointIDexample-2":{
        "ChannelType":"EMAIL",
        "Address":"alejandror@example.com",
        "EndpointStatus":"ACTIVE",
        "OptOut":"NONE",
        "EffectiveDate":"2020-02-26T18:56:24.897Z",
        "Attributes":{
            "AddressType":[
                "primary"
```

```
            ]
        },
        "User":{
            "UserId":"AlejandroRosalez",
            "UserAttributes":{
                "LastName ":[
                    "Rosalez"
                ],
                "FirstName":[
                    "Alejandro"
                ],
                "Neighborhood":[
                    "West Bay"
                ]
            }
        },
        "RecommendationItems":[
            "1210",
            "6542",
            "4582"
        ],
        "CreationDate":"2020-02-26T18:56:24.897Z",
        "Recommendations":{
            "Title":[
                "Hanna",
                "Hunters",
                "Agatha Christie\'s The ABC Murders"
            ],
            "Genre":[
                "Action",
                "Drama",
                "Crime"
            ]
        }
    }
}
```

前述の例では、関数は受け取った Endpoints オブジェクトを変更し、結果を返しています。各エンドポイントの Endpoint オブジェクトに、Title フィールドと Genre フィールドを含む新しい Recommendations オブジェクトが含まれています。これらの各フィールドには、3 つの値の順序付き配列 (文字列として) が保存されます。各値により、RecommendationItems フィールド内の対応する推奨商品の強化されたコンテンツが提供されます。

Lambda 関数ポリシーの割り当て

Lambda 関数を使用して推奨事項データを処理する前に、Amazon Pinpoint に関数の呼び出しを許可する必要があります。呼び出しのアクセス許可を付与するには、関数に Lambda 関数ポリシーを割り当てます。Lambda 関数ポリシーはリソースベースのアクセス許可ポリシーで、関数を使用できるエンティティと、それらのエンティティが実行できるアクションを指定します。詳細については、『AWS Lambda デベロッパーガイド』の「AWS Lambda のリソースベースのポリシーを使用する」を参照してください。

次のポリシーの例では、Amazon Pinpoint サービスプリンシパルが、特定の Amazon Pinpoint プロジェクト (*projectId*) 内の特定の Amazon Pinpoint キャンペーン (*campaignId*) に対して `lambda:InvokeFunction` アクションを使用することを許可しています。

```
{
  "Sid": "sid",
  "Effect": "Allow",
  "Principal": {
    "Service": "pinpoint.us-east-1.amazonaws.com"
  },
  "Action": "lambda:InvokeFunction",
  "Resource": "{arn:aws:lambda:us-east-1:accountId:function:function-name}",
  "Condition": {
    "ArnLike": {
      "AWS:SourceArn": "arn:aws:mobiletargeting:us-east-1:accountId:recommenders/*"
    }
  }
}
```

関数ポリシーは、`AWS:SourceArn` キーを含む `Condition` ブロックを必要とします。このキーは、関数を呼び出すことができるリソースを指定します。前述の例では、ポリシーにより、1 つの特定のキャンペーンで関数を呼び出すことが許可されています。

また、特定のAmazon Pinpointプロジェクト (*projectId*) 内のすべてのキャンペーンとジャーニーに対して、Amazon Pinpointサービスプリンシパルが `lambda:InvokeFunction` アクションを使用できるようにするポリシーを記述することができます。次のポリシー例でこれを示します。

```
{
  "Sid": "sid",
  "Effect": "Allow",
  "Principal": {
```

```
    "Service": "pinpoint.us-east-1.amazonaws.com"
  },
  "Action": "lambda:InvokeFunction",
  "Resource": "{arn:aws:lambda:us-east-1:accountId:function:function-name}",
  "Condition": {
    "ArnLike": {
      "AWS:SourceArn": "arn:aws:mobiletargeting:us-east-1:accountId:recommenders/*"
    }
  }
}
```

最初の例とは異なり、この例の Condition ブロックの AWS:SourceArn キーは、1 つの特定のプロジェクトで関数を呼び出すことができます。この許可は、プロジェクト内のすべてのキャンペーンおよびジャーニーに適用されます。

より一般的なポリシーを記述するには、複数文字に一致するワイルドカード (*) を使用します。たとえば、次の Condition ブロックを使用して、任意の Amazon Pinpoint プロジェクトで関数を呼び出すことができます。

```
"Condition": {
  "ArnLike": {
    "AWS:SourceArn": "arn:aws:mobiletargeting:us-east-1:accountId:recommenders/*"
  }
}
```

Amazon Pinpoint アカウントのすべてのプロジェクトで Lambda 関数を使用する場合は、上記の方法でポリシーの Condition ブロックを設定することをお勧めします。ただし、ベストプラクティスとして、特定のリソースに対して特定のアクションを実行するために必要なアクセス許可のみを含むポリシーを作成する必要があります。

Amazon Pinpoint に対する関数の呼び出しの許可

Lambda 関数ポリシーを関数に割り当てた後、Amazon Pinpoint が特定のプロジェクト、キャンペーン、またはジャーニーの関数を呼び出すことができるアクセス許可を追加できます。これを行うには、 AWS Command Line Interface (AWS CLI) および Lambda add-permission コマンドを使用します。次の例は、特定のプロジェクト (*projectId*) に対してこれを行う方法を示しています。

```
$ aws lambda add-permission \
--function-name function-name \
```

```
--statement-id sid \
--action lambda:InvokeFunction \
--principal pinpoint.us-east-1.amazonaws.com \
--source-arn arn:aws:mobiletargeting:us-east-1:accountId:recommenders/*
```

前述の例は、Unix、Linux、および macOS 用にフォーマットされています。Microsoft Windows の場合、バックスラッシュ (\) の行連結文字をキャレット (^) に置き換えます。

コマンドが正常に実行された場合は、次のような出力が表示されます。

```
{
  "Statement": "{\"Sid\":\"sid\",
    \"Effect\":\"Allow\",
    \"Principal\":{\"Service\":\"pinpoint.us-east-1.amazonaws.com\"},
    \"Action\":\"lambda:InvokeFunction\",
    \"Resource\":\"arn:aws:lambda:us-east-1:111122223333:function:function-name\",
    \"Condition\":
      {\"ArnLike\":
        {\"AWS:SourceArn\":
         \"arn:aws:mobiletargeting:us-east-1:111122223333:recommenders/*\"}}}"
}
```

`Statement` 値は、Lambda 関数ポリシーに追加されたステートメントの JSON 文字列バージョンです。

推奨モデルの設定

推奨モデルの Lambda 関数を呼び出すよう Amazon Pinpoint を設定するには、モデルに対して次の Lambda 固有の構成設定を指定します。

- `RecommendationTransformerUri` – このプロパティは、Lambda 関数の名前または Amazon リソースネーム (ARNs) を指定します。
- `Attributes` – このオブジェクトは、関数が各エンドポイント定義に追加するカスタム推奨属性を定義するマップです。これらの各属性は、メッセージテンプレートのメッセージ変数として使用できます。

これらの設定は、Amazon Pinpoint APIの推奨モデルリソース (モデルの設定を作成する場合)、またはAmazon Pinpoint API の推奨モデルリソース (モデルの設定を更新する場合) を使用して指定することができます。これらの設定は、Amazon Pinpoint コンソールを使用して定義することもできます。

Amazon Pinpoint での推奨モデルの使用に関する詳細は、『Amazon Pinpoint ユーザーガイド』の「機械学習モデル」を参照してください。

Amazon Pinpoint からデータを削除する

使用方法によっては、個人用と見なされる可能性のあるデータが Amazon Pinpoint に保存される場合があります。例えば、Amazon Pinpoint のエンドポイントには、そのユーザーの E メールアドレスまたは携帯電話番号などのエンドユーザーの連絡先情報が含まれています。

コンソールまたは Amazon Pinpoint API を使用して、個人データを完全に削除できます。このトピックには、個人用と見なされる可能性があるさまざまなタイプのデータを削除する手順が含まれています。

エンドポイントの削除

エンドポイントは、顧客の 1 人に連絡する単一のメソッドを表します。各エンドポイントは、顧客の E メールアドレス、モバイルデバイス識別子、電話番号、またはメッセージを送信できる他のタイプの宛先を参照できます。多くの管轄区域では、この種の情報は、個人情報と見なされます。

特定のエンドポイントのすべてのデータを削除するには、Amazon Pinpoint API を使用してエンドポイントを削除します。次の手順は、 AWS CLI を使用して Amazon Pinpoint API と対話することにより、エンドポイントを削除する方法を示しています。この手順は、 AWS CLIがすでにインストールされ、設定されていることを前提としています。詳細については、「AWS Command Line Interface ユーザーガイド」の「AWS CLIのインストール」を参照してください。

を使用してエンドポイントを削除するには AWS CLI

- コマンドラインで以下のコマンドを入力します。

  ```
  aws pinpoint delete-endpoint --application-id 810c7aab86d42fb2b56c8c966example --
  endpoint-id ad015a3bf4f1b2b0b82example
  ```

  上記のコマンドで、[*810c7aab86d42fb2b56c8c966example*] を、エンドポイントが関連付けられているプロジェクトの ID に置き換えます。また、[*ad015a3bf4f1b2b0b82example*] をエンドポイント自体の ID に置き換えます。

特定のエンドポイントのエンドポイント ID を見つけるには、エンドポイントが属するセグメントを特定し、そのセグメントを Amazon Pinpoint からエクスポートします。エクスポートされたデータには、各エンドポイントのエンドポイント ID が含まれます。セグメントをファイルにエクスポートするには、Amazon Pinpoint コンソールを使用します。方法については、『Amazon Pinpoint ユー

ザーガイド』の「Exporting Segments」を参照してください。Amazon Pinpoint API を使用して、セグメントを Amazon Simple Storage Service (Amazon S3) バケットにエクスポートすることもできます。方法については、このガイドの「Exporting Endpoints」を参照してください。

Amazon S3 に保存されているセグメントとエンドポイントデータの削除

Amazon Pinpoint コンソールまたは API を使用して、Amazon S3 バケットに保存されているファイルからセグメントをインポートすることができます。アプリケーション、セグメント、エンドポイントデータを Amazon Pinpoint から Amazon S3 バケットにエクスポートすることもできます。インポートおよびエクスポートされた両方のファイルには、E メールアドレス、携帯電話番号、エンドポイントの物理的な場所に関する情報を含む個人データが含まれています。これらのファイルを、Amazon S3 から削除することができます。

Amazon S3 バケットに配信されるコンテンツには、カスタマーコンテンツが含まれている場合があります。機密データの削除の詳細については、「How Do I Empty an S3 Bucket?」または「How Do I Delete an S3 Bucket?」を参照してください。

すべてのプロジェクトデータの削除

Amazon Pinpoint プロジェクトに保存したすべてのデータを完全に削除することができます。これは、プロジェクトを削除すれば可能です。

> **Warning**
>
> プロジェクトを削除すると、Amazon Pinpoint がそのプロジェクト固有の設定とデータをすべて削除します。この情報を復元することはできません。

プロジェクトを削除すると、Amazon Pinpoint はプッシュ通知および双方向 SMS メッセージのプロジェクト固有のすべての設定を削除します。また、すべてのセグメント、キャンペーン、ジャーニー、および Amazon Pinpoint に保存されたプロジェクト固有の分析データを削除します。これには次のようなものがあります。

- セグメント – すべてのセグメント設定およびデータ。ダイナミックな設定では、これに定義したセグメントグループおよびフィルターが含まれます。インポートした設定では、これにエンドポイント、ユーザー ID、インポートした他のデータ、適用したすべてのフィルターが含まれます。

- キャンペーン – すべてのメッセージ、メッセージ処理と変数、分析データ、スケジュール、およびその他の設定。
- ジャーニー – すべてのアクティビティ、分析データ、スケジュール、およびその他の設定。
- 分析 – キャンペーンとジャーニーついて送受信されたメッセージ数などのすべてのエンゲージメントメトリクスおよびすべてのジャーニー実行メトリクスのデータ。モバイルアプリとウェブアプリの場合、Amazon Kinesis などの別の AWS サービスにストリーミングされなかったすべてのイベントデータ、すべてのファネル、およびアプリケーションの使用状況、収益、属性メトリクスのデータ。プロジェクトを削除する前に、このデータを別の場所にエクスポートすることが推奨されます。

プロジェクトを削除するには、Amazon Pinpoint コンソールを使用します。詳細については、『Amazon Pinpoint ユーザーガイド』の「Deleting a Project」を参照してください。Amazon Pinpoint API の アプリケーション リソースを利用して、プログラム的にプロジェクトを削除することもできます。

AWS アカウントを閉鎖してすべての AWS データを削除する

AWS アカウントを解約することにより、Amazon Pinpoint に保存されたすべての個人データを削除することもできます。ただし、このアクションでは、他のすべての AWS サービスに保存した個人データまたは非個人データなど、他のすべてのデータも削除されます。閉鎖後期間を過ぎると、は AWS アカウント AWS を完全に閉鎖し、再度開くことはできなくなります。削除していなかったコンテンツは削除され、停止していなかった AWS サービスは停止されます。詳細については、「 AWS Account Management リファレンスガイド」の AWS 「アカウントの閉鎖」を参照してください。

> ⚠ Warning
>
> 次の手順では、すべての AWS のサービスと AWS リージョンで、アカウント AWS に保存されているすべてのデータを完全に削除します。

を使用して AWS アカウントを解約できます AWS Management Console。

AWS アカウントを解約するには

1. https://console.aws.amazon.com AWS Management Console で を開きます。

2. https://console.aws.amazon.com/billing/home?#/account の [Account Settings] (アカウント設定) ページに移動します。

   > **Warning**
   >
   > 次の手順では、すべての AWS リージョンのすべての AWS サービスに保存したすべてのデータを完全に削除します。

3. 「アカウントを閉鎖する」で、 AWS アカウントを閉鎖した結果を説明する免責事項を読んでください。条項に同意する場合、チェックボックスをオンにして、[Close Account] を選択します。
4. 確認ダイアログボックスで [Close Account] を選択します。

AWS SDK を使用した Amazon Pinpoint のコード例

次のコードの例は、AWS ソフトウェア開発キット (SDK) による Amazon Pinpoint の使用方法を示しています。

AWS SDK デベロッパーガイドとコード例の詳細なリストについては、「AWS SDK での Amazon Pinpoint の使用」を参照してください。このトピックには、使用開始方法に関する情報と、以前の SDK バージョンの詳細も含まれています。

コード例

- SDK を使用した Amazon Pinpoint のコード例 AWS SDKs
  - SDK を使用した Amazon Pinpoint のアクション AWS SDKs
    - AWS SDK または CLI CreateAppで を使用する
    - AWS SDK または CLI CreateCampaignで を使用する
    - AWS SDK または CLI CreateExportJobで を使用する
    - AWS SDK または CLI CreateImportJobで を使用する
    - AWS SDK または CLI CreateSegmentで を使用する
    - AWS SDK または CLI DeleteAppで を使用する
    - AWS SDK または CLI DeleteEndpointで を使用する
    - AWS SDK または CLI GetEndpointで を使用する
    - AWS SDK または CLI GetSegmentsで を使用する
    - AWS SDK または CLI GetSmsChannelで を使用する
    - AWS SDK または CLI GetUserEndpointsで を使用する
    - AWS SDK または CLI SendMessagesで を使用する
    - AWS SDK または CLI UpdateEndpointで を使用する
- SDK を使用した Amazon Pinpoint SMS および音声 API のコード例 AWS SDKs
  - SDK を使用した Amazon Pinpoint SMS および音声 API のアクション AWS SDKs
    - AWS SDK または CLI SendVoiceMessageで を使用する

SDK を使用した Amazon Pinpoint のコード例 AWS SDKs

次のコード例は、 AWS Software Development Kit (SDK) で Amazon Pinpoint を使用する方法を示しています。

アクションはより大きなプログラムからのコードの抜粋であり、コンテキスト内で実行する必要があります。アクションは個々のサービス機能を呼び出す方法を示していますが、関連するシナリオやサービス間の例ではアクションのコンテキストが確認できます。

AWS SDK デベロッパーガイドとコード例の完全なリストについては、「」を参照してくださいAWS SDK での Amazon Pinpoint の使用。このトピックには、使用開始方法に関する情報と、以前の SDK バージョンの詳細も含まれています。

コードの例

- SDK を使用した Amazon Pinpoint のアクション AWS SDKs
  - AWS SDK または CLI CreateAppで を使用する
  - AWS SDK または CLI CreateCampaignで を使用する
  - AWS SDK または CLI CreateExportJobで を使用する
  - AWS SDK または CLI CreateImportJobで を使用する
  - AWS SDK または CLI CreateSegmentで を使用する
  - AWS SDK または CLI DeleteAppで を使用する
  - AWS SDK または CLI DeleteEndpointで を使用する
  - AWS SDK または CLI GetEndpointで を使用する
  - AWS SDK または CLI GetSegmentsで を使用する
  - AWS SDK または CLI GetSmsChannelで を使用する
  - AWS SDK または CLI GetUserEndpointsで を使用する
  - AWS SDK または CLI SendMessagesで を使用する
  - AWS SDK または CLI UpdateEndpointで を使用する

SDK を使用した Amazon Pinpoint のアクション AWS SDKs

次のコード例は、 AWS SDKs を使用して個々の Amazon Pinpoint アクションを実行する方法を示しています。これらの抜粋は Amazon Pinpoint API を呼び出すもので、コンテキスト内で実行する必要がある大規模なプログラムからのコードの抜粋です。各例には へのリンクが含まれており GitHub、コードの設定と実行の手順を確認できます。

以下の例には、最も一般的に使用されるアクションのみ含まれています。詳細な一覧については、「Amazon Pinpoint API Reference」を参照してください。

例

- AWS SDK または CLI CreateAppで を使用する
- AWS SDK または CLI CreateCampaignで を使用する
- AWS SDK または CLI CreateExportJobで を使用する
- AWS SDK または CLI CreateImportJobで を使用する
- AWS SDK または CLI CreateSegmentで を使用する
- AWS SDK または CLI DeleteAppで を使用する
- AWS SDK または CLI DeleteEndpointで を使用する
- AWS SDK または CLI GetEndpointで を使用する
- AWS SDK または CLI GetSegmentsで を使用する
- AWS SDK または CLI GetSmsChannelで を使用する
- AWS SDK または CLI GetUserEndpointsで を使用する
- AWS SDK または CLI SendMessagesで を使用する
- AWS SDK または CLI UpdateEndpointで を使用する

AWS SDK または CLI **CreateApp**で を使用する

以下のコード例は、CreateApp の使用方法を示しています。

CLI

AWS CLI

例 1: アプリケーションを作成するには

次の create-app の例は、新しいアプリケーション (プロジェクト) を作成します。

```
aws pinpoint create-app \
    --create-application-request Name=ExampleCorp
```

出力:

```
{
    "ApplicationResponse": {
        "Arn": "arn:aws:mobiletargeting:us-
west-2:AIDACKCEVSQ6C2EXAMPLE:apps/810c7aab86d42fb2b56c8c966example",
        "Id": "810c7aab86d42fb2b56c8c966example",
        "Name": "ExampleCorp",
```

```
        "tags": {}
    }
}
```

例 2: タグが付けられたアプリケーションを作成するには

次の create-app の例は、新しいアプリケーション (プロジェクト) を作成し、タグ (キーと値) をアプリケーションに関連付けます。

```
aws pinpoint create-app \
    --create-application-request Name=ExampleCorp,tags={"Stack"="Test"}
```

出力:

```
{
    "ApplicationResponse": {
        "Arn": "arn:aws:mobiletargeting:us-
west-2:AIDACKCEVSQ6C2EXAMPLE:apps/810c7aab86d42fb2b56c8c966example",
        "Id": "810c7aab86d42fb2b56c8c966example",
        "Name": "ExampleCorp",
        "tags": {
            "Stack": "Test"
        }
    }
}
```

- API の詳細については、「 コマンドリファレンスCreateApp」の「」を参照してください。AWS CLI

Java

SDK for Java 2.x

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

```
import software.amazon.awssdk.regions.Region;
```

```
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.CreateAppRequest;
import software.amazon.awssdk.services.pinpoint.model.CreateAppResponse;
import software.amazon.awssdk.services.pinpoint.model.CreateApplicationRequest;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class CreateApp {
    public static void main(String[] args) {
        final String usage = """

                Usage:  <appName>

                Where:
                 appName - The name of the application to create.

               """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }
        String appName = args[0];
        System.out.println("Creating an application with name: " + appName);

        PinpointClient pinpoint = PinpointClient.builder()
                .region(Region.US_EAST_1)
                .build();

        String appID = createApplication(pinpoint, appName);
        System.out.println("App ID is: " + appID);
        pinpoint.close();
    }

    public static String createApplication(PinpointClient pinpoint, String
 appName) {
```

```
        try {
            CreateApplicationRequest appRequest =
 CreateApplicationRequest.builder()
                    .name(appName)
                    .build();

            CreateAppRequest request = CreateAppRequest.builder()
                    .createApplicationRequest(appRequest)
                    .build();

            CreateAppResponse result = pinpoint.createApp(request);
            return result.applicationResponse().id();

        } catch (PinpointException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }
}
```

- API の詳細については、「 API リファレンスCreateApp」の「」を参照してください。AWS SDK for Java 2.x

Kotlin

SDK for Kotlin

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

```
suspend fun createApplication(applicationName: String?): String? {

    val createApplicationRequestOb = CreateApplicationRequest {
        name = applicationName
    }
```

```
    PinpointClient { region = "us-west-2" }.use { pinpoint ->
        val result = pinpoint.createApp(
            CreateAppRequest {
                createApplicationRequest = createApplicationRequestOb
            }
        )
        return result.applicationResponse?.id
    }
}
```

- API の詳細については、 AWS SDK for Kotlin API リファレンスCreateAppの「」を参照してください。

AWS SDK デベロッパーガイドとコード例の完全なリストについては、「」を参照してくださいAWS SDK での Amazon Pinpoint の使用。このトピックには、使用開始方法に関する情報と、以前の SDK バージョンの詳細も含まれています。

AWS SDK または CLI **CreateCampaign**で を使用する

以下のコード例は、CreateCampaign の使用方法を示しています。

Java

SDK for Java 2.x

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

キャンペーンの作成

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.CampaignResponse;
import software.amazon.awssdk.services.pinpoint.model.Message;
import software.amazon.awssdk.services.pinpoint.model.Schedule;
import software.amazon.awssdk.services.pinpoint.model.Action;
import software.amazon.awssdk.services.pinpoint.model.MessageConfiguration;
```

```
import software.amazon.awssdk.services.pinpoint.model.WriteCampaignRequest;
import software.amazon.awssdk.services.pinpoint.model.CreateCampaignResponse;
import software.amazon.awssdk.services.pinpoint.model.CreateCampaignRequest;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class CreateCampaign {
    public static void main(String[] args) {

        final String usage = """

                Usage:   <appId> <segmentId>

                Where:
                  appId - The ID of the application to create the campaign in.
                  segmentId - The ID of the segment to create the campaign from.
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String appId = args[0];
        String segmentId = args[1];
        PinpointClient pinpoint = PinpointClient.builder()
                .region(Region.US_EAST_1)
                .build();

        createPinCampaign(pinpoint, appId, segmentId);
        pinpoint.close();
    }

    public static void createPinCampaign(PinpointClient pinpoint, String appId,
 String segmentId) {
        CampaignResponse result = createCampaign(pinpoint, appId, segmentId);
```

```
        System.out.println("Campaign " + result.name() + " created.");
        System.out.println(result.description());
    }

    public static CampaignResponse createCampaign(PinpointClient client, String
appID, String segmentID) {

        try {
            Schedule schedule = Schedule.builder()
                    .startTime("IMMEDIATE")
                    .build();

            Message defaultMessage = Message.builder()
                    .action(Action.OPEN_APP)
                    .body("My message body.")
                    .title("My message title.")
                    .build();

            MessageConfiguration messageConfiguration =
MessageConfiguration.builder()
                    .defaultMessage(defaultMessage)
                    .build();

            WriteCampaignRequest request = WriteCampaignRequest.builder()
                    .description("My description")
                    .schedule(schedule)
                    .name("MyCampaign")
                    .segmentId(segmentID)
                    .messageConfiguration(messageConfiguration)
                    .build();

            CreateCampaignResponse result =
client.createCampaign(CreateCampaignRequest.builder()
                    .applicationId(appID)
                    .writeCampaignRequest(request).build());

            System.out.println("Campaign ID: " + result.campaignResponse().id());
            return result.campaignResponse();

        } catch (PinpointException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
```

```
        return null;
    }
}
```

- API の詳細については、「 API リファレンスCreateCampaign」の「」を参照してください。 AWS SDK for Java 2.x

Kotlin

SDK for Kotlin

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

```
suspend fun createPinCampaign(appId: String, segmentIdVal: String) {

    val scheduleOb = Schedule {
        startTime = "IMMEDIATE"
    }

    val defaultMessageOb = Message {
        action = Action.OpenApp
        body = "My message body"
        title = "My message title"
    }

    val messageConfigurationOb = MessageConfiguration {
        defaultMessage = defaultMessageOb
    }

    val writeCampaign = WriteCampaignRequest {
        description = "My description"
        schedule = scheduleOb
        name = "MyCampaign"
        segmentId = segmentIdVal
        messageConfiguration = messageConfigurationOb
    }
```

```
    PinpointClient { region = "us-west-2" }.use { pinpoint ->
        val result: CreateCampaignResponse = pinpoint.createCampaign(
            CreateCampaignRequest {
                applicationId = appId
                writeCampaignRequest = writeCampaign
            }
        )
        println("Campaign ID is ${result.campaignResponse?.id}")
    }
}
```

- API の詳細については、 AWS SDK for Kotlin API リファレンスCreateCampaignの「」を参照してください。

AWS SDK デベロッパーガイドとコード例の完全なリストについては、「」を参照してくださいAWS SDK での Amazon Pinpoint の使用。このトピックには、使用開始方法に関する情報と、以前の SDK バージョンの詳細も含まれています。

AWS SDK または CLI **CreateExportJob**で を使用する

次の例は、CreateExportJob を使用する方法を説明しています。

Java

SDK for Java 2.x

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

エンドポイントをエクスポートする

```
import software.amazon.awssdk.core.ResponseBytes;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.ExportJobRequest;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
```

```
import software.amazon.awssdk.services.pinpoint.model.CreateExportJobRequest;
import software.amazon.awssdk.services.pinpoint.model.CreateExportJobResponse;
import software.amazon.awssdk.services.pinpoint.model.GetExportJobResponse;
import software.amazon.awssdk.services.pinpoint.model.GetExportJobRequest;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.GetObjectRequest;
import software.amazon.awssdk.services.s3.model.ListObjectsV2Request;
import software.amazon.awssdk.services.s3.model.ListObjectsV2Response;
import software.amazon.awssdk.services.s3.model.S3Object;
import software.amazon.awssdk.services.s3.model.GetObjectResponse;
import software.amazon.awssdk.services.s3.model.S3Exception;
import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.OutputStream;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Date;
import java.util.List;
import java.util.concurrent.TimeUnit;
import java.util.stream.Collectors;

/**
 * To run this code example, you need to create an AWS Identity and Access
 * Management (IAM) role with the correct policy as described in this
 * documentation:
 * https://docs.aws.amazon.com/pinpoint/latest/developerguide/audience-data-
export.html
 *
 * Also, set up your development environment, including your credentials.
 *
 * For information, see this documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class ExportEndpoints {
    public static void main(String[] args) {
        final String usage = """

                This program performs the following steps:

                1. Exports the endpoints to an Amazon S3 bucket.
```

```
                2. Downloads the exported endpoints files from Amazon S3.
                3. Parses the endpoints files to obtain the endpoint IDs and
prints them.
                Usage: ExportEndpoints <applicationId> <s3BucketName>
<iamExportRoleArn> <path>

                Where:
                  applicationId - The ID of the Amazon Pinpoint application that
has the endpoint.
                  s3BucketName - The name of the Amazon S3 bucket to export the
JSON file to.\s
                  iamExportRoleArn - The ARN of an IAM role that grants Amazon
Pinpoint write permissions to the S3 bucket.  path - The path where the files
downloaded from the Amazon S3 bucket are written (for example, C:/AWS/).
                """;

       if (args.length != 4) {
           System.out.println(usage);
           System.exit(1);
       }

       String applicationId = args[0];
       String s3BucketName = args[1];
       String iamExportRoleArn = args[2];
       String path = args[3];
       System.out.println("Deleting an application with ID: " + applicationId);

       Region region = Region.US_EAST_1;
       PinpointClient pinpoint = PinpointClient.builder()
               .region(region)
               .build();

       S3Client s3Client = S3Client.builder()
               .region(region)
               .build();

       exportAllEndpoints(pinpoint, s3Client, applicationId, s3BucketName, path,
iamExportRoleArn);
       pinpoint.close();
       s3Client.close();
   }

   public static void exportAllEndpoints(PinpointClient pinpoint,
           S3Client s3Client,
```

```
            String applicationId,
            String s3BucketName,
            String path,
            String iamExportRoleArn) {

        try {
            List<String> objectKeys = exportEndpointsToS3(pinpoint, s3Client,
 s3BucketName, iamExportRoleArn,
                    applicationId);
            List<String> endpointFileKeys = objectKeys.stream().filter(o ->
 o.endsWith(".gz"))
                    .collect(Collectors.toList());
            downloadFromS3(s3Client, path, s3BucketName, endpointFileKeys);

        } catch (PinpointException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static List<String> exportEndpointsToS3(PinpointClient pinpoint,
 S3Client s3Client, String s3BucketName,
            String iamExportRoleArn, String applicationId) {

        SimpleDateFormat dateFormat = new SimpleDateFormat("yyyy-MM-dd-
HH_mm:ss.SSS_z");
        String endpointsKeyPrefix = "exports/" + applicationId + "_" +
 dateFormat.format(new Date());
        String s3UrlPrefix = "s3://" + s3BucketName + "/" + endpointsKeyPrefix +
 "/";
        List<String> objectKeys = new ArrayList<>();
        String key;

        try {
            // Defines the export job that Amazon Pinpoint runs.
            ExportJobRequest jobRequest = ExportJobRequest.builder()
                    .roleArn(iamExportRoleArn)
                    .s3UrlPrefix(s3UrlPrefix)
                    .build();

            CreateExportJobRequest exportJobRequest =
 CreateExportJobRequest.builder()
                    .applicationId(applicationId)
                    .exportJobRequest(jobRequest)
```

```
                    .build();

            System.out.format("Exporting endpoints from Amazon Pinpoint
application %s to Amazon S3 " +
                    "bucket %s . . .\n", applicationId, s3BucketName);

            CreateExportJobResponse exportResult =
pinpoint.createExportJob(exportJobRequest);
            String jobId = exportResult.exportJobResponse().id();
            System.out.println(jobId);
            printExportJobStatus(pinpoint, applicationId, jobId);

            ListObjectsV2Request v2Request = ListObjectsV2Request.builder()
                    .bucket(s3BucketName)
                    .prefix(endpointsKeyPrefix)
                    .build();

            // Create a list of object keys.
            ListObjectsV2Response v2Response = s3Client.listObjectsV2(v2Request);
            List<S3Object> objects = v2Response.contents();
            for (S3Object object : objects) {
                key = object.key();
                objectKeys.add(key);
            }

            return objectKeys;

        } catch (PinpointException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return null;
    }

    private static void printExportJobStatus(PinpointClient pinpointClient,
            String applicationId,
            String jobId) {

        GetExportJobResponse getExportJobResult;
        String status;

        try {
            // Checks the job status until the job completes or fails.
            GetExportJobRequest exportJobRequest = GetExportJobRequest.builder()
```

```
                    .jobId(jobId)
                    .applicationId(applicationId)
                    .build();

            do {
                getExportJobResult =
pinpointClient.getExportJob(exportJobRequest);
                status =
getExportJobResult.exportJobResponse().jobStatus().toString().toUpperCase();
                System.out.format("Export job %s . . .\n", status);
                TimeUnit.SECONDS.sleep(3);

            } while (!status.equals("COMPLETED") && !status.equals("FAILED"));

            if (status.equals("COMPLETED")) {
                System.out.println("Finished exporting endpoints.");
            } else {
                System.err.println("Failed to export endpoints.");
                System.exit(1);
            }

        } catch (PinpointException | InterruptedException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }

    // Download files from an Amazon S3 bucket and write them to the path
location.
    public static void downloadFromS3(S3Client s3Client, String path, String
s3BucketName, List<String> objectKeys) {

        String newPath;
        try {
            for (String key : objectKeys) {
                GetObjectRequest objectRequest = GetObjectRequest.builder()
                        .bucket(s3BucketName)
                        .key(key)
                        .build();

                ResponseBytes<GetObjectResponse> objectBytes =
s3Client.getObjectAsBytes(objectRequest);
                byte[] data = objectBytes.asByteArray();
```

```
                // Write the data to a local file.
                String fileSuffix = new
 SimpleDateFormat("yyyyMMddHHmmss").format(new Date());
                newPath = path + fileSuffix + ".gz";
                File myFile = new File(newPath);
                OutputStream os = new FileOutputStream(myFile);
                os.write(data);
            }
            System.out.println("Download finished.");

        } catch (S3Exception | NullPointerException | IOException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
}
```

- API の詳細については、「 API リファレンスCreateExportJob」の「」を参照してください。 AWS SDK for Java 2.x

AWS SDK デベロッパーガイドとコード例の完全なリストについては、「」を参照してくださいAWS SDK での Amazon Pinpoint の使用。このトピックには、使用開始方法に関する情報と、以前の SDK バージョンの詳細も含まれています。

AWS SDK または CLI **CreateImportJob**で を使用する

次の例は、CreateImportJob を使用する方法を説明しています。

Java

SDK for Java 2.x

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

セグメントのインポート

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.CreateImportJobRequest;
import software.amazon.awssdk.services.pinpoint.model.ImportJobResponse;
import software.amazon.awssdk.services.pinpoint.model.ImportJobRequest;
import software.amazon.awssdk.services.pinpoint.model.Format;
import software.amazon.awssdk.services.pinpoint.model.CreateImportJobResponse;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class ImportSegment {
    public static void main(String[] args) {
        final String usage = """

                Usage:   <appId> <bucket> <key> <roleArn>\s

                Where:
                  appId - The application ID to create a segment for.
                  bucket - The name of the Amazon S3 bucket that contains the
 segment definitons.
                  key - The key of the S3 object.
                  roleArn - ARN of the role that allows Amazon
 Pinpoint to access S3. You need to set trust management for this
 to work. See https://docs.aws.amazon.com/IAM/latest/UserGuide/
reference_policies_elements_principal.html
                  """;

        if (args.length != 4) {
            System.out.println(usage);
            System.exit(1);
        }

        String appId = args[0];
        String bucket = args[1];
        String key = args[2];
```

```
        String roleArn = args[3];

        PinpointClient pinpoint = PinpointClient.builder()
                .region(Region.US_EAST_1)
                .build();

        ImportJobResponse response = createImportSegment(pinpoint, appId, bucket,
key, roleArn);
        System.out.println("Import job for " + bucket + " submitted.");
        System.out.println("See application " + response.applicationId() + " for
import job status.");
        System.out.println("See application " + response.jobStatus() + " for
import job status.");
        pinpoint.close();
    }

    public static ImportJobResponse createImportSegment(PinpointClient client,
            String appId,
            String bucket,
            String key,
            String roleArn) {

        try {
            ImportJobRequest importRequest = ImportJobRequest.builder()
                    .defineSegment(true)
                    .registerEndpoints(true)
                    .roleArn(roleArn)
                    .format(Format.JSON)
                    .s3Url("s3://" + bucket + "/" + key)
                    .build();

            CreateImportJobRequest jobRequest = CreateImportJobRequest.builder()
                    .importJobRequest(importRequest)
                    .applicationId(appId)
                    .build();

            CreateImportJobResponse jobResponse =
client.createImportJob(jobRequest);
            return jobResponse.importJobResponse();

        } catch (PinpointException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
```

```
        return null;
    }
}
```

- API の詳細については、「 API リファレンスCreateImportJob」の「」を参照してください。 AWS SDK for Java 2.x

AWS SDK デベロッパーガイドとコード例の完全なリストについては、「」を参照してくださいAWS SDK での Amazon Pinpoint の使用。このトピックには、使用開始方法に関する情報と、以前の SDK バージョンの詳細も含まれています。

AWS SDK または CLI **CreateSegment**で を使用する

以下のコード例は、CreateSegment の使用方法を示しています。

Java

SDK for Java 2.x

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.AttributeDimension;
import software.amazon.awssdk.services.pinpoint.model.SegmentResponse;
import software.amazon.awssdk.services.pinpoint.model.AttributeType;
import software.amazon.awssdk.services.pinpoint.model.RecencyDimension;
import software.amazon.awssdk.services.pinpoint.model.SegmentBehaviors;
import software.amazon.awssdk.services.pinpoint.model.SegmentDemographics;
import software.amazon.awssdk.services.pinpoint.model.SegmentLocation;
import software.amazon.awssdk.services.pinpoint.model.SegmentDimensions;
import software.amazon.awssdk.services.pinpoint.model.WriteSegmentRequest;
import software.amazon.awssdk.services.pinpoint.model.CreateSegmentRequest;
import software.amazon.awssdk.services.pinpoint.model.CreateSegmentResponse;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import java.util.HashMap;
```

```
import java.util.Map;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class CreateSegment {
        public static void main(String[] args) {
                final String usage = """

                                Usage:   <appId>

                                Where:
                                  appId - The application ID to create a segment
 for.

                                """;

                if (args.length != 1) {
                        System.out.println(usage);
                        System.exit(1);
                }

                String appId = args[0];
                PinpointClient pinpoint = PinpointClient.builder()
                                .region(Region.US_EAST_1)
                                .build();

                SegmentResponse result = createSegment(pinpoint, appId);
                System.out.println("Segment " + result.name() + " created.");
                System.out.println(result.segmentType());
                pinpoint.close();
        }

        public static SegmentResponse createSegment(PinpointClient client, String
 appId) {
                try {
                        Map<String, AttributeDimension> segmentAttributes = new
 HashMap<>();
```

```
                        segmentAttributes.put("Team",
AttributeDimension.builder()
                                        .attributeType(AttributeType.INCLUSIVE)
                                        .values("Lakers")
                                        .build());

                        RecencyDimension recencyDimension =
RecencyDimension.builder()
                                        .duration("DAY_30")
                                        .recencyType("ACTIVE")
                                        .build();

                        SegmentBehaviors segmentBehaviors =
SegmentBehaviors.builder()
                                        .recency(recencyDimension)
                                        .build();

                        SegmentDemographics segmentDemographics =
SegmentDemographics
                                        .builder()
                                        .build();

                        SegmentLocation segmentLocation = SegmentLocation
                                        .builder()
                                        .build();

                        SegmentDimensions dimensions = SegmentDimensions
                                        .builder()
                                        .attributes(segmentAttributes)
                                        .behavior(segmentBehaviors)
                                        .demographic(segmentDemographics)
                                        .location(segmentLocation)
                                        .build();

                        WriteSegmentRequest writeSegmentRequest =
WriteSegmentRequest.builder()
                                        .name("MySegment")
                                        .dimensions(dimensions)
                                        .build();

                        CreateSegmentRequest createSegmentRequest =
CreateSegmentRequest.builder()
                                        .applicationId(appId)
                                        .writeSegmentRequest(writeSegmentRequest)
```

```
                                        .build();

                        CreateSegmentResponse createSegmentResult =
 client.createSegment(createSegmentRequest);
                        System.out.println("Segment ID: " +
 createSegmentResult.segmentResponse().id());
                        System.out.println("Done");
                        return createSegmentResult.segmentResponse();

                } catch (PinpointException e) {
                        System.err.println(e.awsErrorDetails().errorMessage());
                        System.exit(1);
                }
                return null;
        }
}
```

- API の詳細については、「 API リファレンスCreateSegment」の「」を参照してください。 AWS SDK for Java 2.x

Kotlin

SDK for Kotlin

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

```
suspend fun createPinpointSegment(applicationIdVal: String?): String? {

    val segmentAttributes = mutableMapOf<String, AttributeDimension>()
    val myList = mutableListOf<String>()
    myList.add("Lakers")

    val atts = AttributeDimension {
        attributeType = AttributeType.Inclusive
        values = myList
    }
```

```
    segmentAttributes["Team"] = atts
    val recencyDimension = RecencyDimension {
        duration = Duration.fromValue("DAY_30")
        recencyType = RecencyType.fromValue("ACTIVE")
    }

    val segmentBehaviors = SegmentBehaviors {
        recency = recencyDimension
    }

    val segmentLocation = SegmentLocation {}
    val dimensionsOb = SegmentDimensions {
        attributes = segmentAttributes
        behavior = segmentBehaviors
        demographic = SegmentDemographics {}
        location = segmentLocation
    }

    val writeSegmentRequestOb = WriteSegmentRequest {
        name = "MySegment101"
        dimensions = dimensionsOb
    }

    PinpointClient { region = "us-west-2" }.use { pinpoint ->
        val createSegmentResult: CreateSegmentResponse = pinpoint.createSegment(
            CreateSegmentRequest {
                applicationId = applicationIdVal
                writeSegmentRequest = writeSegmentRequestOb
            }
        )
        println("Segment ID is ${createSegmentResult.segmentResponse?.id}")
        return createSegmentResult.segmentResponse?.id
    }
}
```

- API の詳細については、 AWS SDK for Kotlin API リファレンスCreateSegmentの「」を参照してください。

AWS SDK デベロッパーガイドとコード例の完全なリストについては、「」を参照してくださいAWS SDK での Amazon Pinpoint の使用。このトピックには、使用開始方法に関する情報と、以前の SDK バージョンの詳細も含まれています。

AWS SDK または CLI **DeleteApp**で を使用する

以下のコード例は、DeleteApp の使用方法を示しています。

CLI

AWS CLI

アプリケーションを削除するには

次の delete-app の例は、アプリケーション (プロジェクト) を削除します。

```
aws pinpoint delete-app \
    --application-id 810c7aab86d42fb2b56c8c966example
```

出力:

```
{
    "ApplicationResponse": {
        "Arn": "arn:aws:mobiletargeting:us-
west-2:AIDACKCEVSQ6C2EXAMPLE:apps/810c7aab86d42fb2b56c8c966example",
        "Id": "810c7aab86d42fb2b56c8c966example",
        "Name": "ExampleCorp",
        "tags": {}
    }
}
```

- API の詳細については、「 コマンドリファレンスDeleteApp」の「」を参照してください。AWS CLI

Java

SDK for Java 2.x

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

アプリケーションを削除します。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.DeleteAppRequest;
import software.amazon.awssdk.services.pinpoint.model.DeleteAppResponse;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DeleteApp {
    public static void main(String[] args) {
        final String usage = """

                Usage:  <appId>

                Where:
                 appId - The ID of the application to delete.

                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }
```

```
        String appId = args[0];
        System.out.println("Deleting an application with ID: " + appId);
        PinpointClient pinpoint = PinpointClient.builder()
                .region(Region.US_EAST_1)
                .build();

        deletePinApp(pinpoint, appId);
        System.out.println("Done");
        pinpoint.close();
    }

    public static void deletePinApp(PinpointClient pinpoint, String appId) {
        try {
            DeleteAppRequest appRequest = DeleteAppRequest.builder()
                    .applicationId(appId)
                    .build();

            DeleteAppResponse result = pinpoint.deleteApp(appRequest);
            String appName = result.applicationResponse().name();
            System.out.println("Application " + appName + " has been deleted.");

        } catch (PinpointException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- API の詳細については、「 API リファレンスDeleteApp」の「」を参照してください。 AWS SDK for Java 2.x

Kotlin

SDK for Kotlin

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

```
suspend fun deletePinApp(appId: String?) {

    PinpointClient { region = "us-west-2" }.use { pinpoint ->
        val result = pinpoint.deleteApp(
            DeleteAppRequest {
                applicationId = appId
            }
        )
        val appName = result.applicationResponse?.name
        println("Application $appName has been deleted.")
    }
}
```

- API の詳細については、 AWS SDK for Kotlin API リファレンスDeleteAppの「」を参照してください。

AWS SDK デベロッパーガイドとコード例の完全なリストについては、「」を参照してくださいAWS SDK での Amazon Pinpoint の使用。このトピックには、使用開始方法に関する情報と、以前の SDK バージョンの詳細も含まれています。

AWS SDK または CLI **DeleteEndpoint**で を使用する

以下のコード例は、DeleteEndpoint の使用方法を示しています。

Java

SDK for Java 2.x

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

エンドポイントを削除します。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.DeleteEndpointRequest;
import software.amazon.awssdk.services.pinpoint.model.DeleteEndpointResponse;
```

```
import software.amazon.awssdk.services.pinpoint.model.PinpointException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DeleteEndpoint {
    public static void main(String[] args) {
        final String usage = """

                Usage:   <appName> <endpointId >

                Where:
                  appId - The id of the application to delete.
                  endpointId - The id of the endpoint to delete.
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String appId = args[0];
        String endpointId = args[1];
        System.out.println("Deleting an endpoint with id: " + endpointId);
        PinpointClient pinpoint = PinpointClient.builder()
                .region(Region.US_EAST_1)
                .build();

        deletePinEncpoint(pinpoint, appId, endpointId);
        pinpoint.close();
    }

    public static void deletePinEncpoint(PinpointClient pinpoint, String appId,
 String endpointId) {
        try {
            DeleteEndpointRequest appRequest = DeleteEndpointRequest.builder()
                    .applicationId(appId)
                    .endpointId(endpointId)
```

```
                    .build();

            DeleteEndpointResponse result = pinpoint.deleteEndpoint(appRequest);
            String id = result.endpointResponse().id();
            System.out.println("The deleted endpoint id  " + id);

        } catch (PinpointException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        System.out.println("Done");
    }
}
```

- API の詳細については、「 API リファレンスDeleteEndpoint」の「」を参照してください。AWS SDK for Java 2.x

Kotlin

SDK for Kotlin

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

```
suspend fun deletePinEncpoint(appIdVal: String?, endpointIdVal: String?) {

    val deleteEndpointRequest = DeleteEndpointRequest {
        applicationId = appIdVal
        endpointId = endpointIdVal
    }

    PinpointClient { region = "us-west-2" }.use { pinpoint ->
        val result = pinpoint.deleteEndpoint(deleteEndpointRequest)
        val id = result.endpointResponse?.id
        println("The deleted endpoint is  $id")
    }
}
```

- API の詳細については、 AWS SDK for Kotlin API リファレンスDeleteEndpointの「」を参照してください。

AWS SDK デベロッパーガイドとコード例の完全なリストについては、「」を参照してくださいAWS SDK での Amazon Pinpoint の使用。このトピックには、使用開始方法に関する情報と、以前の SDK バージョンの詳細も含まれています。

AWS SDK または CLI **GetEndpoint**で を使用する

以下のコード例は、GetEndpoint の使用方法を示しています。

CLI

AWS CLI

アプリケーションの特定のエンドポイントに関する設定と属性の情報を取得するには

次の get-endpoint の例は、アプリケーションの指定されたエンドポイントの設定と属性に関する情報を取得します。

```
aws pinpoint get-endpoint \
    --application-id 611e3e3cdd47474c9c1399a505665b91 \
    --endpoint-id testendpoint \
    --region us-east-1
```

出力:

```
{
    "EndpointResponse": {
        "Address": "+11234567890",
        "ApplicationId": "611e3e3cdd47474c9c1399a505665b91",
        "Attributes": {},
        "ChannelType": "SMS",
        "CohortId": "63",
        "CreationDate": "2019-01-28T23:55:11.534Z",
        "EffectiveDate": "2021-08-06T00:04:51.763Z",
        "EndpointStatus": "ACTIVE",
        "Id": "testendpoint",
        "Location": {
            "Country": "USA"
```

```
        },
        "Metrics": {
            "SmsDelivered": 1.0
        },
        "OptOut": "ALL",
        "RequestId": "a204b1f2-7e26-48a7-9c80-b49a2143489d",
        "User": {
            "UserAttributes": {
                "Age": [
                    "24"
                ]
            },
        "UserId": "testuser"
        }
    }
}
```

- API の詳細については、「 コマンドリファレンスGetEndpoint」の「」を参照してください。 AWS CLI

Java

SDK for Java 2.x

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

```
import com.google.gson.FieldNamingPolicy;
import com.google.gson.Gson;
import com.google.gson.GsonBuilder;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.EndpointResponse;
import software.amazon.awssdk.services.pinpoint.model.GetEndpointResponse;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import software.amazon.awssdk.services.pinpoint.model.GetEndpointRequest;

/**
```

```
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class LookUpEndpoint {
    public static void main(String[] args) {
        final String usage = """

                Usage:   <appId> <endpoint>

                Where:
                  appId - The ID of the application to delete.
                  endpoint - The ID of the endpoint.\s
                  """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String appId = args[0];
        String endpoint = args[1];
        System.out.println("Looking up an endpoint point with ID: " + endpoint);
        PinpointClient pinpoint = PinpointClient.builder()
                .region(Region.US_EAST_1)
                .build();

        lookupPinpointEndpoint(pinpoint, appId, endpoint);
        pinpoint.close();
    }

    public static void lookupPinpointEndpoint(PinpointClient pinpoint, String
 appId, String endpoint) {
        try {
            GetEndpointRequest appRequest = GetEndpointRequest.builder()
                    .applicationId(appId)
                    .endpointId(endpoint)
                    .build();

            GetEndpointResponse result = pinpoint.getEndpoint(appRequest);
```

```
            EndpointResponse endResponse = result.endpointResponse();

            // Uses the Google Gson library to pretty print the endpoint JSON.
            Gson gson = new GsonBuilder()
                    .setFieldNamingPolicy(FieldNamingPolicy.UPPER_CAMEL_CASE)
                    .setPrettyPrinting()
                    .create();

            String endpointJson = gson.toJson(endResponse);
            System.out.println(endpointJson);

        } catch (PinpointException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        System.out.println("Done");
    }
}
```

- API の詳細については、「 API リファレンスGetEndpoint」の「」を参照してください。AWS SDK for Java 2.x

Kotlin

SDK for Kotlin

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

```
suspend fun lookupPinpointEndpoint(appId: String?, endpoint: String?) {

    PinpointClient { region = "us-west-2" }.use { pinpoint ->
        val result = pinpoint.getEndpoint(
            GetEndpointRequest {
                applicationId = appId
                endpointId = endpoint
            }
```

```
        )
        val endResponse = result.endpointResponse

        // Uses the Google Gson library to pretty print the endpoint JSON.
        val gson: com.google.gson.Gson = GsonBuilder()
            .setFieldNamingPolicy(FieldNamingPolicy.UPPER_CAMEL_CASE)
            .setPrettyPrinting()
            .create()

        val endpointJson: String = gson.toJson(endResponse)
        println(endpointJson)
    }
}
```

- API の詳細については、 AWS SDK for Kotlin API リファレンスGetEndpointの「」を参照してください。

AWS SDK デベロッパーガイドとコード例の完全なリストについては、「」を参照してくださいAWS SDK での Amazon Pinpoint の使用。このトピックには、使用開始方法に関する情報と、以前の SDK バージョンの詳細も含まれています。

AWS SDK または CLI **GetSegments**で を使用する

以下のコード例は、GetSegments の使用方法を示しています。

Java

SDK for Java 2.x

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

セグメントを一覧表示します。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.GetSegmentsRequest;
```

```
import software.amazon.awssdk.services.pinpoint.model.GetSegmentsResponse;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import software.amazon.awssdk.services.pinpoint.model.SegmentResponse;
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class ListSegments {
    public static void main(String[] args) {
        final String usage = """

                Usage:   <appId>

                Where:
                  appId - The ID of the application that contains a segment.

                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String appId = args[0];
        PinpointClient pinpoint = PinpointClient.builder()
                .region(Region.US_EAST_1)
                .build();

        listSegs(pinpoint, appId);
        pinpoint.close();
    }

    public static void listSegs(PinpointClient pinpoint, String appId) {
        try {
            GetSegmentsRequest request = GetSegmentsRequest.builder()
                    .applicationId(appId)
                    .build();
```

```
            GetSegmentsResponse response = pinpoint.getSegments(request);
            List<SegmentResponse> segments = response.segmentsResponse().item();
            for (SegmentResponse segment : segments) {
                System.out
                        .println("Segement " + segment.id() + " " +
 segment.name() + " " + segment.lastModifiedDate());
            }

        } catch (PinpointException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- API の詳細については、「 API リファレンスGetSegments」の「」を参照してください。 AWS SDK for Java 2.x

Kotlin

SDK for Kotlin

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

```
suspend fun listSegs(appId: String?) {

    PinpointClient { region = "us-west-2" }.use { pinpoint ->

        val response = pinpoint.getSegments(
            GetSegmentsRequest {
                applicationId = appId
            }
        )
        response.segmentsResponse?.item?.forEach { segment ->
            println("Segement id is ${segment.id}")
```

```
        }
    }
}
```

- API の詳細については、 AWS SDK for Kotlin API リファレンスGetSegmentsの「」を参照してください。

AWS SDK デベロッパーガイドとコード例の完全なリストについては、「」を参照してくださいAWS SDK での Amazon Pinpoint の使用。このトピックには、使用開始方法に関する情報と、以前の SDK バージョンの詳細も含まれています。

AWS SDK または CLI **GetSmsChannel**で を使用する

以下のコード例は、GetSmsChannel の使用方法を示しています。

CLI

AWS CLI

アプリケーションの SMS チャネルのステータスおよび設定に関する情報を取得するには

次の get-sms-channel の例は、アプリケーションの SMS チャネルのステータスと設定を取得します。

```
aws pinpoint get-sms-channel \
    --application-id 6e0b7591a90841d2b5d93fa11143e5a7 \
    --region us-east-1
```

出力:

```
{
    "SMSChannelResponse": {
        "ApplicationId": "6e0b7591a90841d2b5d93fa11143e5a7",
        "CreationDate": "2019-10-08T18:39:18.511Z",
        "Enabled": true,
        "Id": "sms",
        "IsArchived": false,
        "LastModifiedDate": "2019-10-08T18:39:18.511Z",
        "Platform": "SMS",
        "PromotionalMessagesPerSecond": 20,
        "TransactionalMessagesPerSecond": 20,
```

```
        "Version": 1
    }
}
```

- API の詳細については、「 コマンドリファレンスGetSmsChannel」の「」を参照してください。 AWS CLI

Java

SDK for Java 2.x

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.SMSChannelResponse;
import software.amazon.awssdk.services.pinpoint.model.GetSmsChannelRequest;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import software.amazon.awssdk.services.pinpoint.model.SMSChannelRequest;
import software.amazon.awssdk.services.pinpoint.model.UpdateSmsChannelRequest;
import software.amazon.awssdk.services.pinpoint.model.UpdateSmsChannelResponse;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class UpdateChannel {
    public static void main(String[] args) {
        final String usage = """

                Usage: CreateChannel <appId>
```

```
                Where:
                  appId - The name of the application whose channel is updated.

                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String appId = args[0];
        PinpointClient pinpoint = PinpointClient.builder()
                .region(Region.US_EAST_1)
                .build();

        SMSChannelResponse getResponse = getSMSChannel(pinpoint, appId);
        toggleSmsChannel(pinpoint, appId, getResponse);
        pinpoint.close();
    }

    private static SMSChannelResponse getSMSChannel(PinpointClient client, String
appId) {
        try {
            GetSmsChannelRequest request = GetSmsChannelRequest.builder()
                    .applicationId(appId)
                    .build();

            SMSChannelResponse response =
client.getSmsChannel(request).smsChannelResponse();
            System.out.println("Channel state is " + response.enabled());
            return response;

        } catch (PinpointException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return null;
    }

    private static void toggleSmsChannel(PinpointClient client, String appId,
SMSChannelResponse getResponse) {
        boolean enabled = !getResponse.enabled();
        try {
            SMSChannelRequest request = SMSChannelRequest.builder()
```

```
                    .enabled(enabled)
                    .build();

            UpdateSmsChannelRequest updateRequest =
 UpdateSmsChannelRequest.builder()
                    .smsChannelRequest(request)
                    .applicationId(appId)
                    .build();

            UpdateSmsChannelResponse result =
 client.updateSmsChannel(updateRequest);
            System.out.println("Channel state: " +
 result.smsChannelResponse().enabled());

        } catch (PinpointException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- API の詳細については、「 API リファレンスGetSmsChannel」の「」を参照してください。 AWS SDK for Java 2.x

AWS SDK デベロッパーガイドとコード例の完全なリストについては、「」を参照してくださいAWS SDK での Amazon Pinpoint の使用。このトピックには、使用開始方法に関する情報と、以前の SDK バージョンの詳細も含まれています。

AWS SDK または CLI **GetUserEndpoints**で を使用する

次の例は、GetUserEndpoints を使用する方法を説明しています。

Java

SDK for Java 2.x

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.EndpointResponse;
import software.amazon.awssdk.services.pinpoint.model.GetUserEndpointsRequest;
import software.amazon.awssdk.services.pinpoint.model.GetUserEndpointsResponse;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class ListEndpointIds {
    public static void main(String[] args) {
        final String usage = """

                Usage:    <applicationId> <userId>

                Where:
                   applicationId - The ID of the Amazon Pinpoint application that
 has the endpoint.
                   userId - The user id applicable to the endpoints""";

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String applicationId = args[0];
        String userId = args[1];
        PinpointClient pinpoint = PinpointClient.builder()
                .region(Region.US_EAST_1)
                .build();

        listAllEndpoints(pinpoint, applicationId, userId);
        pinpoint.close();
    }
```

```
    public static void listAllEndpoints(PinpointClient pinpoint,
            String applicationId,
            String userId) {

        try {
            GetUserEndpointsRequest endpointsRequest =
 GetUserEndpointsRequest.builder()
                    .userId(userId)
                    .applicationId(applicationId)
                    .build();

            GetUserEndpointsResponse response =
 pinpoint.getUserEndpoints(endpointsRequest);
            List<EndpointResponse> endpoints =
 response.endpointsResponse().item();

            // Display the results.
            for (EndpointResponse endpoint : endpoints) {
                System.out.println("The channel type is: " +
 endpoint.channelType());
                System.out.println("The address is  " + endpoint.address());
            }

        } catch (PinpointException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- API の詳細については、「 API リファレンスGetUserEndpoints」の「」を参照してください。 AWS SDK for Java 2.x

AWS SDK デベロッパーガイドとコード例の完全なリストについては、「」を参照してくださいAWS SDK での Amazon Pinpoint の使用。このトピックには、使用開始方法に関する情報と、以前の SDK バージョンの詳細も含まれています。

AWS SDK または CLI **SendMessages**で を使用する

以下のコード例は、SendMessages の使用方法を示しています。

.NET

AWS SDK for .NET

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

E メールメッセージを送信します。

```
using Amazon;
using Amazon.Pinpoint;
using Amazon.Pinpoint.Model;
using Microsoft.Extensions.Configuration;

namespace SendEmailMessage;

public class SendEmailMainClass
{
    public static async Task Main(string[] args)
    {
        var configuration = new ConfigurationBuilder()
        .SetBasePath(Directory.GetCurrentDirectory())
        .AddJsonFile("settings.json") // Load test settings from .json file.
        .AddJsonFile("settings.local.json",
            true) // Optionally load local settings.
        .Build();

        // The AWS Region that you want to use to send the email. For a list of
        // AWS Regions where the Amazon Pinpoint API is available, see
        // https://docs.aws.amazon.com/pinpoint/latest/apireference/
        string region = "us-east-1";

        // The "From" address. This address has to be verified in Amazon
 Pinpoint
        // in the region you're using to send email.
        string senderAddress = configuration["SenderAddress"]!;

        // The address on the "To" line. If your Amazon Pinpoint account is in
        // the sandbox, this address also has to be verified.
```

```
        string toAddress = configuration["ToAddress"]!;

        // The Amazon Pinpoint project/application ID to use when you send this
message.
        // Make sure that the SMS channel is enabled for the project or
application
        // that you choose.
        string appId = configuration["AppId"]!;

        try
        {
            await SendEmailMessage(region, appId, toAddress, senderAddress);
        }
        catch (Exception ex)
        {
            Console.WriteLine("The message wasn't sent. Error message: " +
ex.Message);
        }
    }

    public static async Task<MessageResponse> SendEmailMessage(
        string region, string appId, string toAddress, string senderAddress)
    {
        var client = new
AmazonPinpointClient(RegionEndpoint.GetBySystemName(region));

        // The subject line of the email.
        string subject = "Amazon Pinpoint Email test";

        // The body of the email for recipients whose email clients don't
        // support HTML content.
        string textBody = @"Amazon Pinpoint Email Test (.NET)"
                          + "\n---------------------------------"
                          + "\nThis email was sent using the Amazon Pinpoint API
using the AWS SDK for .NET.";

        // The body of the email for recipients whose email clients support
        // HTML content.
        string htmlBody = @"<html>"
                          + "\n<head></head>"
                          + "\n<body>"
                          + "\n  <h1>Amazon Pinpoint Email Test (AWS SDK
for .NET)</h1>"
                          + "\n  <p>This email was sent using the "
```

```
                            + "\n    <a href='https://aws.amazon.com/
pinpoint/'>Amazon Pinpoint</a> API "
                            + "\n    using the <a href='https://aws.amazon.com/sdk-
for-net/'>AWS SDK for .NET</a>"
                            + "\n  </p>"
                            + "\n</body>"
                            + "\n</html>";

        // The character encoding the you want to use for the subject line and
        // message body of the email.
        string charset = "UTF-8";

        var sendRequest = new SendMessagesRequest
        {
            ApplicationId = appId,
            MessageRequest = new MessageRequest
            {
                Addresses = new Dictionary<string, AddressConfiguration>
                {
                    {
                        toAddress,
                        new AddressConfiguration
                        {
                            ChannelType = ChannelType.EMAIL
                        }
                    }
                },
                MessageConfiguration = new DirectMessageConfiguration
                {
                    EmailMessage = new EmailMessage
                    {
                        FromAddress = senderAddress,
                        SimpleEmail = new SimpleEmail
                        {
                            HtmlPart = new SimpleEmailPart
                            {
                                Charset = charset,
                                Data = htmlBody
                            },
                            TextPart = new SimpleEmailPart
                            {
                                Charset = charset,
                                Data = textBody
                            },
```

```
                            Subject = new SimpleEmailPart
                            {
                                Charset = charset,
                                Data = subject
                            }
                        }
                    }
                }
            }
        };
        Console.WriteLine("Sending message...");
        SendMessagesResponse response = await
 client.SendMessagesAsync(sendRequest);
        Console.WriteLine("Message sent!");
        return response.MessageResponse;
    }
}
```

SMS メッセージを送信します。

```
using Amazon;
using Amazon.Pinpoint;
using Amazon.Pinpoint.Model;
using Microsoft.Extensions.Configuration;

namespace SendSmsMessage;

public class SendSmsMessageMainClass
{
    public static async Task Main(string[] args)
    {
        var configuration = new ConfigurationBuilder()
            .SetBasePath(Directory.GetCurrentDirectory())
            .AddJsonFile("settings.json") // Load test settings from .json file.
            .AddJsonFile("settings.local.json",
                true) // Optionally load local settings.
            .Build();

        // The AWS Region that you want to use to send the message. For a list of
        // AWS Regions where the Amazon Pinpoint API is available, see
```

```
        // https://docs.aws.amazon.com/pinpoint/latest/apireference/
        string region = "us-east-1";

        // The phone number or short code to send the message from. The phone
 number
        // or short code that you specify has to be associated with your Amazon
 Pinpoint
        // account. For best results, specify long codes in E.164 format.
        string originationNumber = configuration["OriginationNumber"]!;

        // The recipient's phone number.  For best results, you should specify
 the
        // phone number in E.164 format.
        string destinationNumber = configuration["DestinationNumber"]!;

        // The Pinpoint project/ application ID to use when you send this
 message.
        // Make sure that the SMS channel is enabled for the project or
 application
        // that you choose.
        string appId = configuration["AppId"]!;

        // The type of SMS message that you want to send. If you plan to send
        // time-sensitive content, specify TRANSACTIONAL. If you plan to send
        // marketing-related content, specify PROMOTIONAL.
        MessageType messageType = MessageType.TRANSACTIONAL;

        // The registered keyword associated with the originating short code.
        string? registeredKeyword = configuration["RegisteredKeyword"];

        // The sender ID to use when sending the message. Support for sender ID
        // varies by country or region. For more information, see
        // https://docs.aws.amazon.com/pinpoint/latest/userguide/channels-sms-
countries.html
        string? senderId = configuration["SenderId"];

        try
        {
            var response = await SendSmsMessage(region, appId, destinationNumber,
                originationNumber, registeredKeyword, senderId, messageType);
            Console.WriteLine($"Message sent to
 {response.MessageResponse.Result.Count} recipient(s).");
            foreach (var messageResultValue in
                     response.MessageResponse.Result.Select(r => r.Value))
```

```
            {
                Console.WriteLine($"{messageResultValue.MessageId} Status:
{messageResultValue.DeliveryStatus}");
            }
        }
        catch (Exception ex)
        {
            Console.WriteLine("The message wasn't sent. Error message: " +
ex.Message);
        }
    }

    public static async Task<SendMessagesResponse> SendSmsMessage(
        string region, string appId, string destinationNumber, string
originationNumber,
        string? keyword, string? senderId, MessageType messageType)
    {

        // The content of the SMS message.
        string message = "This message was sent through Amazon Pinpoint using" +
                         " the AWS SDK for .NET. Reply STOP to opt out.";


        var client = new
AmazonPinpointClient(RegionEndpoint.GetBySystemName(region));

        SendMessagesRequest sendRequest = new SendMessagesRequest
        {
            ApplicationId = appId,
            MessageRequest = new MessageRequest
            {
                Addresses =
                    new Dictionary<string, AddressConfiguration>
                    {
                        {
                            destinationNumber,
                            new AddressConfiguration { ChannelType =
ChannelType.SMS }
                        }
                    },
                MessageConfiguration = new DirectMessageConfiguration
                {
                    SMSMessage = new SMSMessage
                    {
```

```
                        Body = message,
                        MessageType = MessageType.TRANSACTIONAL,
                        OriginationNumber = originationNumber,
                        SenderId = senderId,
                        Keyword = keyword
                    }
                }
            }
        };
        SendMessagesResponse response = await
 client.SendMessagesAsync(sendRequest);
        return response;
    }
}
```

- API の詳細については、「 API リファレンスSendMessages」の「」を参照してください。 AWS SDK for .NET

CLI

AWS CLI

アプリケーションのエンドポイントを使用して SMS メッセージを送信するには

次の send-messages の例は、エンドポイントを使用してアプリケーションにダイレクトメッセージを送信します。

```
aws pinpoint send-messages \
    --application-id 611e3e3cdd47474c9c1399a505665b91 \
    --message-request file://myfile.json \
    --region us-west-2
```

myfile.json の内容:

```
{
    "MessageConfiguration": {
        "SMSMessage": {
            "Body": "hello, how are you?"
        }
    },
```

```
    "Endpoints": {
        "testendpoint": {}
    }
}
```

出力:

```
{
    "MessageResponse": {
        "ApplicationId": "611e3e3cdd47474c9c1399a505665b91",
        "EndpointResult": {
            "testendpoint": {
                "Address": "+12345678900",
                "DeliveryStatus": "SUCCESSFUL",
                "MessageId": "itnuqhai5alf1n6ahv3udc05n7hhddr6gb3lq6g0",
                "StatusCode": 200,
                "StatusMessage": "MessageId:
 itnuqhai5alf1n6ahv3udc05n7hhddr6gb3lq6g0"
            }
        },
        "RequestId": "c7e23264-04b2-4a46-b800-d24923f74753"
    }
}
```

詳細については、「Amazon Pinpoint ユーザーガイド」の「Amazon Pinpoint SMS チャンネル」を参照してください。

- API の詳細については、「 コマンドリファレンスSendMessages」の「」を参照してください。 AWS CLI

Java

SDK for Java 2.x

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

E メールメッセージを送信します。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.AddressConfiguration;
import software.amazon.awssdk.services.pinpoint.model.ChannelType;
import software.amazon.awssdk.services.pinpoint.model.SimpleEmailPart;
import software.amazon.awssdk.services.pinpoint.model.SimpleEmail;
import software.amazon.awssdk.services.pinpoint.model.EmailMessage;
import software.amazon.awssdk.services.pinpoint.model.DirectMessageConfiguration;
import software.amazon.awssdk.services.pinpoint.model.MessageRequest;
import software.amazon.awssdk.services.pinpoint.model.SendMessagesRequest;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import software.amazon.awssdk.services.pinpointemail.PinpointEmailClient;
import software.amazon.awssdk.services.pinpointemail.model.Body;
import software.amazon.awssdk.services.pinpointemail.model.Content;
import software.amazon.awssdk.services.pinpointemail.model.Destination;
import software.amazon.awssdk.services.pinpointemail.model.EmailContent;
import software.amazon.awssdk.services.pinpointemail.model.Message;
import software.amazon.awssdk.services.pinpointemail.model.SendEmailRequest;

import java.util.HashMap;
import java.util.Map;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class SendEmailMessage {

        // The character encoding the you want to use for the subject line and
        // message body of the email.
        public static String charset = "UTF-8";

    // The body of the email for recipients whose email clients support HTML
 content.
    static final String body = """
        Amazon Pinpoint test (AWS SDK for Java 2.x)
```

```
        This email was sent through the Amazon Pinpoint Email API using the AWS
SDK for Java 2.x

        """;

        public static void main(String[] args) {
                final String usage = """

                                Usage:    <subject> <appId> <senderAddress>
<toAddress>

            Where:
               subject - The email subject to use.
               senderAddress - The from address. This address has to be verified
in Amazon Pinpoint in the region you're using to send email\s
               toAddress - The to address. This address has to be verified in
Amazon Pinpoint in the region you're using to send email\s
            """;

        if (args.length != 3) {
            System.out.println(usage);
            System.exit(1);
        }

        String subject = args[0];
        String senderAddress = args[1];
        String toAddress = args[2];
        System.out.println("Sending a message");
        PinpointEmailClient pinpoint = PinpointEmailClient.builder()
            .region(Region.US_EAST_1)
            .build();

        sendEmail(pinpoint, subject, senderAddress, toAddress);
        System.out.println("Email was sent");
        pinpoint.close();
    }

    public static void sendEmail(PinpointEmailClient pinpointEmailClient, String
subject, String senderAddress, String toAddress) {
        try {
            Content content = Content.builder()
                .data(body)
                .build();
```

```
            Body messageBody = Body.builder()
                .text(content)
                .build();

            Message message = Message.builder()
                .body(messageBody)
                .subject(Content.builder().data(subject).build())
                .build();

            Destination destination = Destination.builder()
                .toAddresses(toAddress)
                .build();

            EmailContent emailContent = EmailContent.builder()
                .simple(message)
                .build();

            SendEmailRequest sendEmailRequest = SendEmailRequest.builder()
                .fromEmailAddress(senderAddress)
                .destination(destination)
                .content(emailContent)
                .build();

            pinpointEmailClient.sendEmail(sendEmailRequest);
            System.out.println("Message Sent");

        } catch (PinpointException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

CC の値を使用して E メールメッセージを送信します。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import  software.amazon.awssdk.services.pinpointemail.PinpointEmailClient;
import software.amazon.awssdk.services.pinpointemail.model.Body;
import software.amazon.awssdk.services.pinpointemail.model.Content;
import software.amazon.awssdk.services.pinpointemail.model.Destination;
import software.amazon.awssdk.services.pinpointemail.model.EmailContent;
```

```
import software.amazon.awssdk.services.pinpointemail.model.Message;
import software.amazon.awssdk.services.pinpointemail.model.SendEmailRequest;
import java.util.ArrayList;

/**
 * Before running this Java V2 code example, set up your development environment,
 including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class SendEmailMessageCC {

    // The body of the email.
    static final String body = """
        Amazon Pinpoint test (AWS SDK for Java 2.x)

        This email was sent through the Amazon Pinpoint Email API using the AWS
 SDK for Java 2.x

        """;
    public static void main(String[] args) {
        final String usage = """

            Usage:    <subject> <senderAddress> <toAddress> <ccAddress>

            Where:
               subject - The email subject to use.
               senderAddress - The from address. This address has to be verified
 in Amazon Pinpoint in the region you're using to send email\s
               toAddress - The to address. This address has to be verified in
 Amazon Pinpoint in the region you're using to send email\s
               ccAddress - The CC address.
            """;

        if (args.length != 4) {
            System.out.println(usage);
            System.exit(1);
        }

        String subject = args[0];
        String senderAddress = args[1];
```

```
        String toAddress = args[2];
        String ccAddress = args[3];

        System.out.println("Sending a message");
        PinpointEmailClient pinpoint = PinpointEmailClient.builder()
            .region(Region.US_EAST_1)
            .build();

        ArrayList<String> ccList = new ArrayList<>();
        ccList.add(ccAddress);
        sendEmail(pinpoint, subject, senderAddress, toAddress, ccList);
        pinpoint.close();
    }

    public static void sendEmail(PinpointEmailClient pinpointEmailClient, String
subject, String senderAddress, String toAddress, ArrayList<String> ccAddresses)
{
        try {
            Content content = Content.builder()
                .data(body)
                .build();

            Body messageBody = Body.builder()
                .text(content)
                .build();

            Message message = Message.builder()
                .body(messageBody)
                .subject(Content.builder().data(subject).build())
                .build();

            Destination destination = Destination.builder()
                .toAddresses(toAddress)
                .ccAddresses(ccAddresses)
                .build();

            EmailContent emailContent = EmailContent.builder()
                .simple(message)
                .build();

            SendEmailRequest sendEmailRequest = SendEmailRequest.builder()
                .fromEmailAddress(senderAddress)
                .destination(destination)
                .content(emailContent)
```

```
                .build();

            pinpointEmailClient.sendEmail(sendEmailRequest);
            System.out.println("Message Sent");

        } catch (PinpointException e) {
            // Handle exception
            e.printStackTrace();
        }
    }
}
```

SMS メッセージを送信します。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.DirectMessageConfiguration;
import software.amazon.awssdk.services.pinpoint.model.SMSMessage;
import software.amazon.awssdk.services.pinpoint.model.AddressConfiguration;
import software.amazon.awssdk.services.pinpoint.model.ChannelType;
import software.amazon.awssdk.services.pinpoint.model.MessageRequest;
import software.amazon.awssdk.services.pinpoint.model.SendMessagesRequest;
import software.amazon.awssdk.services.pinpoint.model.SendMessagesResponse;
import software.amazon.awssdk.services.pinpoint.model.MessageResponse;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import java.util.HashMap;
import java.util.Map;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class SendMessage {

        // The type of SMS message that you want to send. If you plan to send
        // time-sensitive content, specify TRANSACTIONAL. If you plan to send
        // marketing-related content, specify PROMOTIONAL.
```

```
        public static String messageType = "TRANSACTIONAL";

        // The registered keyword associated with the originating short code.
        public static String registeredKeyword = "myKeyword";

        // The sender ID to use when sending the message. Support for sender ID
        // varies by country or region. For more information, see
        // https://docs.aws.amazon.com/pinpoint/latest/userguide/channels-sms-
countries.html
        public static String senderId = "MySenderID";

        public static void main(String[] args) {
                final String usage = """

                                Usage:   <message> <appId> <originationNumber>
 <destinationNumber>\s

                                Where:
                                  message - The body of the message to send.
                                  appId - The Amazon Pinpoint project/application
 ID to use when you send this message.
                                  originationNumber - The phone number or
 short code that you specify has to be associated with your Amazon Pinpoint
 account. For best results, specify long codes in E.164 format (for example,
 +1-555-555-5654).
                                  destinationNumber - The recipient's phone
 number.  For best results, you should specify the phone number in E.164 format
 (for example, +1-555-555-5654).\s
                                  """;

                if (args.length != 4) {
                        System.out.println(usage);
                        System.exit(1);
                }

                String message = args[0];
                String appId = args[1];
                String originationNumber = args[2];
                String destinationNumber = args[3];
                System.out.println("Sending a message");
                PinpointClient pinpoint = PinpointClient.builder()
                                .region(Region.US_EAST_1)
                                .build();
```

```
                sendSMSMessage(pinpoint, message, appId, originationNumber,
destinationNumber);
                pinpoint.close();
        }

        public static void sendSMSMessage(PinpointClient pinpoint, String
message, String appId,
                        String originationNumber,
                        String destinationNumber) {
                try {
                        Map<String, AddressConfiguration> addressMap = new
HashMap<String, AddressConfiguration>();
                        AddressConfiguration addConfig =
AddressConfiguration.builder()
                                        .channelType(ChannelType.SMS)
                                        .build();

                        addressMap.put(destinationNumber, addConfig);
                        SMSMessage smsMessage = SMSMessage.builder()
                                        .body(message)
                                        .messageType(messageType)
                                        .originationNumber(originationNumber)
                                        .senderId(senderId)
                                        .keyword(registeredKeyword)
                                        .build();

                        // Create a DirectMessageConfiguration object.
                        DirectMessageConfiguration direct =
DirectMessageConfiguration.builder()
                                        .smsMessage(smsMessage)
                                        .build();

                        MessageRequest msgReq = MessageRequest.builder()
                                        .addresses(addressMap)
                                        .messageConfiguration(direct)
                                        .build();

                        // create a SendMessagesRequest object
                        SendMessagesRequest request =
SendMessagesRequest.builder()
                                        .applicationId(appId)
                                        .messageRequest(msgReq)
                                        .build();
```

```
                        SendMessagesResponse response =
 pinpoint.sendMessages(request);
                        MessageResponse msg1 = response.messageResponse();
                        Map map1 = msg1.result();

                        // Write out the result of sendMessage.
                        map1.forEach((k, v) -> System.out.println((k + ":" +
 v)));

                } catch (PinpointException e) {
                        System.err.println(e.awsErrorDetails().errorMessage());
                        System.exit(1);
                }
        }
}
```

SMS メッセージをバッチ送信します。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.DirectMessageConfiguration;
import software.amazon.awssdk.services.pinpoint.model.SMSMessage;
import software.amazon.awssdk.services.pinpoint.model.AddressConfiguration;
import software.amazon.awssdk.services.pinpoint.model.ChannelType;
import software.amazon.awssdk.services.pinpoint.model.MessageRequest;
import software.amazon.awssdk.services.pinpoint.model.SendMessagesRequest;
import software.amazon.awssdk.services.pinpoint.model.SendMessagesResponse;
import software.amazon.awssdk.services.pinpoint.model.MessageResponse;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import java.util.HashMap;
import java.util.Map;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class SendMessageBatch {
```

```
        // The type of SMS message that you want to send. If you plan to send
        // time-sensitive content, specify TRANSACTIONAL. If you plan to send
        // marketing-related content, specify PROMOTIONAL.
        public static String messageType = "TRANSACTIONAL";

        // The registered keyword associated with the originating short code.
        public static String registeredKeyword = "myKeyword";

        // The sender ID to use when sending the message. Support for sender ID
        // varies by country or region. For more information, see
        // https://docs.aws.amazon.com/pinpoint/latest/userguide/channels-sms-
countries.html
        public static String senderId = "MySenderID";

        public static void main(String[] args) {
                final String usage = """

                                Usage:   <message> <appId> <originationNumber>
 <destinationNumber> <destinationNumber1>\s

                                Where:
                                  message - The body of the message to send.
                                  appId - The Amazon Pinpoint project/application
 ID to use when you send this message.
                                  originationNumber - The phone number or
 short code that you specify has to be associated with your Amazon Pinpoint
 account. For best results, specify long codes in E.164 format (for example,
 +1-555-555-5654).
                                  destinationNumber - The recipient's phone
 number.  For best results, you should specify the phone number in E.164 format
 (for example, +1-555-555-5654).
                                  destinationNumber1 - The second recipient's
 phone number.  For best results, you should specify the phone number in E.164
 format (for example, +1-555-555-5654).\s
                                  """;

                if (args.length != 5) {
                        System.out.println(usage);
                        System.exit(1);
                }

                String message = args[0];
                String appId = args[1];
```

```
                String originationNumber = args[2];
                String destinationNumber = args[3];
                String destinationNumber1 = args[4];
                System.out.println("Sending a message");
                PinpointClient pinpoint = PinpointClient.builder()
                                .region(Region.US_EAST_1)
                                .build();

                sendSMSMessage(pinpoint, message, appId, originationNumber,
destinationNumber, destinationNumber1);
                pinpoint.close();
        }

        public static void sendSMSMessage(PinpointClient pinpoint, String
message, String appId,
                        String originationNumber,
                        String destinationNumber, String destinationNumber1) {
                try {
                        Map<String, AddressConfiguration> addressMap = new
HashMap<String, AddressConfiguration>();
                        AddressConfiguration addConfig =
AddressConfiguration.builder()
                                        .channelType(ChannelType.SMS)
                                        .build();

                        // Add an entry to the Map object for each number to whom
you want to send a
                        // message.
                        addressMap.put(destinationNumber, addConfig);
                        addressMap.put(destinationNumber1, addConfig);
                        SMSMessage smsMessage = SMSMessage.builder()
                                        .body(message)
                                        .messageType(messageType)
                                        .originationNumber(originationNumber)
                                        .senderId(senderId)
                                        .keyword(registeredKeyword)
                                        .build();

                        // Create a DirectMessageConfiguration object.
                        DirectMessageConfiguration direct =
DirectMessageConfiguration.builder()
                                        .smsMessage(smsMessage)
                                        .build();
```

```
                        MessageRequest msgReq = MessageRequest.builder()
                                        .addresses(addressMap)
                                        .messageConfiguration(direct)
                                        .build();

                        // Create a SendMessagesRequest object.
                        SendMessagesRequest request =
 SendMessagesRequest.builder()
                                        .applicationId(appId)
                                        .messageRequest(msgReq)
                                        .build();

                        SendMessagesResponse response =
 pinpoint.sendMessages(request);
                        MessageResponse msg1 = response.messageResponse();
                        Map map1 = msg1.result();

                        // Write out the result of sendMessage.
                        map1.forEach((k, v) -> System.out.println((k + ":" +
 v)));

                } catch (PinpointException e) {
                        System.err.println(e.awsErrorDetails().errorMessage());
                        System.exit(1);
                }
        }
}
```

- API の詳細については、「 API リファレンスSendMessages」の「」を参照してください。 AWS SDK for Java 2.x

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> については、「」を参照してください GitHub。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

別のモジュールでクライアントを作成し、エクスポートします。

```
import { PinpointClient } from "@aws-sdk/client-pinpoint";
// Set the AWS Region.
const REGION = "us-east-1";
export const pinClient = new PinpointClient({ region: REGION });
```

E メールメッセージを送信します。

```
// Import required AWS SDK clients and commands for Node.js
import { SendMessagesCommand } from "@aws-sdk/client-pinpoint";
import { pinClient } from "./libs/pinClient.js";

// The FromAddress must be verified in SES.
const fromAddress = "FROM_ADDRESS";
const toAddress = "TO_ADDRESS";
const projectId = "PINPOINT_PROJECT_ID";

// The subject line of the email.
var subject = "Amazon Pinpoint Test (AWS SDK for JavaScript in Node.js)";

// The email body for recipients with non-HTML email clients.
var body_text = `Amazon Pinpoint Test (SDK for JavaScript in Node.js)
-----------------------------------------------------
This email was sent with Amazon Pinpoint using the AWS SDK for JavaScript in
 Node.js.
For more information, see https://aws.amazon.com/sdk-for-node-js/`;

// The body of the email for recipients whose email clients support HTML content.
var body_html = `<html>
<head></head>
<body>
  <h1>Amazon Pinpoint Test (SDK for JavaScript in Node.js)</h1>
  <p>This email was sent with
    <a href='https://aws.amazon.com/pinpoint/'>the Amazon Pinpoint Email API</a>
 using the
    <a href='https://aws.amazon.com/sdk-for-node-js/'>
      AWS SDK for JavaScript in Node.js</a>.</p>
</body>
</html>`;

// The character encoding for the subject line and message body of the email.
```

```
var charset = "UTF-8";

const params = {
  ApplicationId: projectId,
  MessageRequest: {
    Addresses: {
      [toAddress]: {
        ChannelType: "EMAIL",
      },
    },
    MessageConfiguration: {
      EmailMessage: {
        FromAddress: fromAddress,
        SimpleEmail: {
          Subject: {
            Charset: charset,
            Data: subject,
          },
          HtmlPart: {
            Charset: charset,
            Data: body_html,
          },
          TextPart: {
            Charset: charset,
            Data: body_text,
          },
        },
      },
    },
  },
};

const run = async () => {
  try {
    const { MessageResponse } = await pinClient.send(
      new SendMessagesCommand(params),
    );

    if (!MessageResponse) {
      throw new Error("No message response.");
    }

    if (!MessageResponse.Result) {
      throw new Error("No message result.");
```

```
    }

    const recipientResult = MessageResponse.Result[toAddress];

    if (recipientResult.StatusCode !== 200) {
      throw new Error(recipientResult.StatusMessage);
    } else {
      console.log(recipientResult.MessageId);
    }
  } catch (err) {
    console.log(err.message);
  }
};

run();
```

SMS メッセージを送信します。

```
// Import required AWS SDK clients and commands for Node.js
import { SendMessagesCommand } from "@aws-sdk/client-pinpoint";
import { pinClient } from "./libs/pinClient.js";

/* The phone number or short code to send the message from. The phone number
 or short code that you specify has to be associated with your Amazon Pinpoint
account. For best results, specify long codes in E.164 format. */
const originationNumber = "SENDER_NUMBER"; //e.g., +1XXXXXXXXXX

// The recipient's phone number.  For best results, you should specify the phone
 number in E.164 format.
const destinationNumber = "RECEIVER_NUMBER"; //e.g., +1XXXXXXXXXX

// The content of the SMS message.
const message =
  "This message was sent through Amazon Pinpoint " +
  "using the AWS SDK for JavaScript in Node.js. Reply STOP to " +
  "opt out.";

/*The Amazon Pinpoint project/application ID to use when you send this message.
Make sure that the SMS channel is enabled for the project or application
that you choose.*/
const projectId = "PINPOINT_PROJECT_ID"; //e.g., XXXXXXXX66e4e9986478cXXXXXXXXX
```

```
/* The type of SMS message that you want to send. If you plan to send
time-sensitive content, specify TRANSACTIONAL. If you plan to send
marketing-related content, specify PROMOTIONAL.*/
var messageType = "TRANSACTIONAL";

// The registered keyword associated with the originating short code.
var registeredKeyword = "myKeyword";

/* The sender ID to use when sending the message. Support for sender ID
// varies by country or region. For more information, see
https://docs.aws.amazon.com/pinpoint/latest/userguide/channels-sms-
countries.html.*/

var senderId = "MySenderID";

// Specify the parameters to pass to the API.
var params = {
  ApplicationId: projectId,
  MessageRequest: {
    Addresses: {
      [destinationNumber]: {
        ChannelType: "SMS",
      },
    },
    MessageConfiguration: {
      SMSMessage: {
        Body: message,
        Keyword: registeredKeyword,
        MessageType: messageType,
        OriginationNumber: originationNumber,
        SenderId: senderId,
      },
    },
  },
};

const run = async () => {
  try {
    const data = await pinClient.send(new SendMessagesCommand(params));
    console.log(
      "Message sent! " +
        data["MessageResponse"]["Result"][destinationNumber]["StatusMessage"],
    );
```

```
  } catch (err) {
    console.log(err);
  }
};
run();
```

- API の詳細については、「 API リファレンスSendMessages」の「」を参照してください。 AWS SDK for JavaScript

SDK for JavaScript (v2)

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

E メールメッセージを送信します。

```
"use strict";

const AWS = require("aws-sdk");

// The AWS Region that you want to use to send the email. For a list of
// AWS Regions where the Amazon Pinpoint API is available, see
// https://docs.aws.amazon.com/pinpoint/latest/apireference/
const aws_region = "us-west-2";

// The "From" address. This address has to be verified in Amazon Pinpoint
// in the region that you use to send email.
const senderAddress = "sender@example.com";

// The address on the "To" line. If your Amazon Pinpoint account is in
// the sandbox, this address also has to be verified.
var toAddress = "recipient@example.com";

// The Amazon Pinpoint project/application ID to use when you send this message.
// Make sure that the SMS channel is enabled for the project or application
// that you choose.
const appId = "ce796be37f32f178af652b26eexample";
```

```
// The subject line of the email.
var subject = "Amazon Pinpoint (AWS SDK for JavaScript in Node.js)";

// The email body for recipients with non-HTML email clients.
var body_text = `Amazon Pinpoint Test (SDK for JavaScript in Node.js)
-----------------------------------------------------
This email was sent with Amazon Pinpoint using the AWS SDK for JavaScript in
 Node.js.
For more information, see https:\/\/aws.amazon.com/sdk-for-node-js/`;

// The body of the email for recipients whose email clients support HTML content.
var body_html = `<html>
<head></head>
<body>
  <h1>Amazon Pinpoint Test (SDK for JavaScript in Node.js)</h1>
  <p>This email was sent with
    <a href='https://aws.amazon.com/pinpoint/'>the Amazon Pinpoint API</a> using
 the
    <a href='https://aws.amazon.com/sdk-for-node-js/'>
      AWS SDK for JavaScript in Node.js</a>.</p>
</body>
</html>`;

// The character encoding the you want to use for the subject line and
// message body of the email.
var charset = "UTF-8";

// Specify that you're using a shared credentials file.
var credentials = new AWS.SharedIniFileCredentials({ profile: "default" });
AWS.config.credentials = credentials;

// Specify the region.
AWS.config.update({ region: aws_region });

//Create a new Pinpoint object.
var pinpoint = new AWS.Pinpoint();

// Specify the parameters to pass to the API.
var params = {
  ApplicationId: appId,
  MessageRequest: {
    Addresses: {
      [toAddress]: {
        ChannelType: "EMAIL",
```

```
      },
    },
    MessageConfiguration: {
      EmailMessage: {
        FromAddress: senderAddress,
        SimpleEmail: {
          Subject: {
            Charset: charset,
            Data: subject,
          },
          HtmlPart: {
            Charset: charset,
            Data: body_html,
          },
          TextPart: {
            Charset: charset,
            Data: body_text,
          },
        },
      },
    },
  },
};

//Try to send the email.
pinpoint.sendMessages(params, function (err, data) {
  // If something goes wrong, print an error message.
  if (err) {
    console.log(err.message);
  } else {
    console.log(
      "Email sent! Message ID: ",
      data["MessageResponse"]["Result"][toAddress]["MessageId"]
    );
  }
});
```

SMS メッセージを送信します。

```
"use strict";
```

```
var AWS = require("aws-sdk");

// The AWS Region that you want to use to send the message. For a list of
// AWS Regions where the Amazon Pinpoint API is available, see
// https://docs.aws.amazon.com/pinpoint/latest/apireference/.
var aws_region = "us-east-1";

// The phone number or short code to send the message from. The phone number
// or short code that you specify has to be associated with your Amazon Pinpoint
// account. For best results, specify long codes in E.164 format.
var originationNumber = "+12065550199";

// The recipient's phone number.  For best results, you should specify the
// phone number in E.164 format.
var destinationNumber = "+14255550142";

// The content of the SMS message.
var message =
  "This message was sent through Amazon Pinpoint " +
  "using the AWS SDK for JavaScript in Node.js. Reply STOP to " +
  "opt out.";

// The Amazon Pinpoint project/application ID to use when you send this message.
// Make sure that the SMS channel is enabled for the project or application
// that you choose.
var applicationId = "ce796be37f32f178af652b26eexample";

// The type of SMS message that you want to send. If you plan to send
// time-sensitive content, specify TRANSACTIONAL. If you plan to send
// marketing-related content, specify PROMOTIONAL.
var messageType = "TRANSACTIONAL";

// The registered keyword associated with the originating short code.
var registeredKeyword = "myKeyword";

// The sender ID to use when sending the message. Support for sender ID
// varies by country or region. For more information, see
// https://docs.aws.amazon.com/pinpoint/latest/userguide/channels-sms-
countries.html
var senderId = "MySenderID";

// Specify that you're using a shared credentials file, and optionally specify
// the profile that you want to use.
```

```
var credentials = new AWS.SharedIniFileCredentials({ profile: "default" });
AWS.config.credentials = credentials;

// Specify the region.
AWS.config.update({ region: aws_region });

//Create a new Pinpoint object.
var pinpoint = new AWS.Pinpoint();

// Specify the parameters to pass to the API.
var params = {
  ApplicationId: applicationId,
  MessageRequest: {
    Addresses: {
      [destinationNumber]: {
        ChannelType: "SMS",
      },
    },
    MessageConfiguration: {
      SMSMessage: {
        Body: message,
        Keyword: registeredKeyword,
        MessageType: messageType,
        OriginationNumber: originationNumber,
        SenderId: senderId,
      },
    },
  },
};

//Try to send the message.
pinpoint.sendMessages(params, function (err, data) {
  // If something goes wrong, print an error message.
  if (err) {
    console.log(err.message);
    // Otherwise, show the unique ID for the message.
  } else {
    console.log(
      "Message sent! " +
        data["MessageResponse"]["Result"][destinationNumber]["StatusMessage"]
    );
  }
});
```

- API の詳細については、「 API リファレンスSendMessages」の「」を参照してください。AWS SDK for JavaScript

Kotlin

SDK for Kotlin

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

```
/**
Before running this Kotlin code example, set up your development environment,
including your credentials.

For more information, see the following documentation topic:
https://docs.aws.amazon.com/sdk-for-kotlin/latest/developer-guide/setup.html
 */

val body: String = """
        Amazon Pinpoint test (AWS SDK for Kotlin)

        This email was sent through the Amazon Pinpoint Email API using the AWS
 SDK for Kotlin.

""".trimIndent()

suspend fun main(args: Array<String>) {
    val usage = """
    Usage:
        <subject> <appId> <senderAddress> <toAddress>

    Where:
        subject - The email subject to use.
        senderAddress - The from address. This address has to be verified in
 Amazon Pinpoint in the region you're using to send email
```

```
        toAddress - The to address. This address has to be verified in Amazon
 Pinpoint in the region you're using to send email
    """

    if (args.size != 3) {
        println(usage)
        exitProcess(0)
    }

    val subject = args[0]
    val senderAddress = args[1]
    val toAddress = args[2]
    sendEmail(subject, senderAddress, toAddress)
}

suspend fun sendEmail(subjectVal: String?, senderAddress: String, toAddressVal:
 String) {
    var content = Content {
        data = body
    }

    val messageBody = Body {
        text = content
    }

    val subContent = Content {
        data = subjectVal
    }

    val message = Message {
        body = messageBody
        subject = subContent
    }

    val destinationOb = Destination {
        toAddresses = listOf(toAddressVal)
    }

    val emailContent = EmailContent {
        simple = message
    }

    val sendEmailRequest = SendEmailRequest {
        fromEmailAddress = senderAddress
```

```
        destination = destination0b
        this.content = emailContent
    }

    PinpointEmailClient { region = "us-east-1" }.use { pinpointemail ->
        pinpointemail.sendEmail(sendEmailRequest)
        println("Message Sent")
    }
}
```

- API の詳細については、 AWS SDK for Kotlin API リファレンスSendMessagesの「」を参照してください。

Python

SDK for Python (Boto3)

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

E メールメッセージを送信します。

```
import logging
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)


def send_email_message(
    pinpoint_client,
    app_id,
    sender,
    to_addresses,
    char_set,
    subject,
    html_message,
```

```
    text_message,
):
    """
    Sends an email message with HTML and plain text versions.

    :param pinpoint_client: A Boto3 Pinpoint client.
    :param app_id: The Amazon Pinpoint project ID to use when you send this
 message.
    :param sender: The "From" address. This address must be verified in
                   Amazon Pinpoint in the AWS Region you're using to send email.
    :param to_addresses: The addresses on the "To" line. If your Amazon Pinpoint
 account
                         is in the sandbox, these addresses must be verified.
    :param char_set: The character encoding to use for the subject line and
 message
                      body of the email.
    :param subject: The subject line of the email.
    :param html_message: The body of the email for recipients whose email clients
 can
                         display HTML content.
    :param text_message: The body of the email for recipients whose email clients
                         don't support HTML content.
    :return: A dict of to_addresses and their message IDs.
    """
    try:
        response = pinpoint_client.send_messages(
            ApplicationId=app_id,
            MessageRequest={
                "Addresses": {
                    to_address: {"ChannelType": "EMAIL"} for to_address in
 to_addresses
                },
                "MessageConfiguration": {
                    "EmailMessage": {
                        "FromAddress": sender,
                        "SimpleEmail": {
                            "Subject": {"Charset": char_set, "Data": subject},
                            "HtmlPart": {"Charset": char_set, "Data":
 html_message},
                            "TextPart": {"Charset": char_set, "Data":
 text_message},
                        },
                    }
                },
```

```
            },
        )
    except ClientError:
        logger.exception("Couldn't send email.")
        raise
    else:
        return {
            to_address: message["MessageId"]
            for to_address, message in response["MessageResponse"]
["Result"].items()
        }


def main():
    app_id = "ce796be37f32f178af652b26eexample"
    sender = "sender@example.com"
    to_address = "recipient@example.com"
    char_set = "UTF-8"
    subject = "Amazon Pinpoint Test (SDK for Python (Boto3))"
    text_message = """Amazon Pinpoint Test (SDK for Python)
    -------------------------------------
    This email was sent with Amazon Pinpoint using the AWS SDK for Python
 (Boto3).
    For more information, see https://aws.amazon.com/sdk-for-python/
                """
    html_message = """<html>
    <head></head>
    <body>
      <h1>Amazon Pinpoint Test (SDK for Python (Boto3)</h1>
      <p>This email was sent with
        <a href='https://aws.amazon.com/pinpoint/'>Amazon Pinpoint</a> using the
        <a href='https://aws.amazon.com/sdk-for-python/'>
          AWS SDK for Python (Boto3)</a>.</p>
    </body>
    </html>
                """

    print("Sending email.")
    message_ids = send_email_message(
        boto3.client("pinpoint"),
        app_id,
        sender,
        [to_address],
        char_set,
```

```
        subject,
        html_message,
        text_message,
    )
    print(f"Message sent! Message IDs: {message_ids}")


if __name__ == "__main__":
    main()
```

SMS メッセージを送信します。

```
import logging
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)


def send_sms_message(
    pinpoint_client,
    app_id,
    origination_number,
    destination_number,
    message,
    message_type,
):
    """
    Sends an SMS message with Amazon Pinpoint.

    :param pinpoint_client: A Boto3 Pinpoint client.
    :param app_id: The Amazon Pinpoint project/application ID to use when you
 send
                   this message. The SMS channel must be enabled for the project
 or
                   application.
    :param destination_number: The recipient's phone number in E.164 format.
    :param origination_number: The phone number to send the message from. This
 phone
                               number must be associated with your Amazon
 Pinpoint
```

```
                                account and be in E.164 format.
    :param message: The content of the SMS message.
    :param message_type: The type of SMS message that you want to send. If you
 send
                         time-sensitive content, specify TRANSACTIONAL. If you
 send
                         marketing-related content, specify PROMOTIONAL.
    :return: The ID of the message.
    """
    try:
        response = pinpoint_client.send_messages(
            ApplicationId=app_id,
            MessageRequest={
                "Addresses": {destination_number: {"ChannelType": "SMS"}},
                "MessageConfiguration": {
                    "SMSMessage": {
                        "Body": message,
                        "MessageType": message_type,
                        "OriginationNumber": origination_number,
                    }
                },
            },
        )
    except ClientError:
        logger.exception("Couldn't send message.")
        raise
    else:
        return response["MessageResponse"]["Result"][destination_number]
["MessageId"]


def main():
    app_id = "ce796be37f32f178af652b26eexample"
    origination_number = "+12065550199"
    destination_number = "+14255550142"
    message = (
        "This is a sample message sent from Amazon Pinpoint by using the AWS SDK
 for "
        "Python (Boto 3)."
    )
    message_type = "TRANSACTIONAL"

    print("Sending SMS message.")
    message_id = send_sms_message(
```

```
        boto3.client("pinpoint"),
        app_id,
        origination_number,
        destination_number,
        message,
        message_type,
    )
    print(f"Message sent! Message ID: {message_id}.")


if __name__ == "__main__":
    main()
```

既存の E メールテンプレートを使用して E メールメッセージを送信します。

```
import logging
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)


def send_templated_email_message(
    pinpoint_client, project_id, sender, to_addresses, template_name,
 template_version
):
    """
    Sends an email message with HTML and plain text versions.

    :param pinpoint_client: A Boto3 Pinpoint client.
    :param project_id: The Amazon Pinpoint project ID to use when you send this
 message.
    :param sender: The "From" address. This address must be verified in
                   Amazon Pinpoint in the AWS Region you're using to send email.
    :param to_addresses: The addresses on the "To" line. If your Amazon Pinpoint
                         account is in the sandbox, these addresses must be
 verified.
    :param template_name: The name of the email template to use when sending the
 message.
    :param template_version: The version number of the message template.

    :return: A dict of to_addresses and their message IDs.
```

```
    """
    try:
        response = pinpoint_client.send_messages(
            ApplicationId=project_id,
            MessageRequest={
                "Addresses": {
                    to_address: {"ChannelType": "EMAIL"} for to_address in
 to_addresses
                },
                "MessageConfiguration": {"EmailMessage": {"FromAddress":
 sender}},
                "TemplateConfiguration": {
                    "EmailTemplate": {
                        "Name": template_name,
                        "Version": template_version,
                    }
                },
            },
        )
    except ClientError:
        logger.exception("Couldn't send email.")
        raise
    else:
        return {
            to_address: message["MessageId"]
            for to_address, message in response["MessageResponse"]
["Result"].items()
        }


def main():
    project_id = "296b04b342374fceb661bf494example"
    sender = "sender@example.com"
    to_addresses = ["recipient@example.com"]
    template_name = "My_Email_Template"
    template_version = "1"

    print("Sending email.")
    message_ids = send_templated_email_message(
        boto3.client("pinpoint"),
        project_id,
        sender,
        to_addresses,
        template_name,
```

```
        template_version,
    )
    print(f"Message sent! Message IDs: {message_ids}")


if __name__ == "__main__":
    main()
```

既存の SMS テンプレートを使用してテキストメッセージを送信します。

```
import logging
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)


def send_templated_sms_message(
    pinpoint_client,
    project_id,
    destination_number,
    message_type,
    origination_number,
    template_name,
    template_version,
):
    """
    Sends an SMS message to a specific phone number using a pre-defined template.

    :param pinpoint_client: A Boto3 Pinpoint client.
    :param project_id: An Amazon Pinpoint project (application) ID.
    :param destination_number: The phone number to send the message to.
    :param message_type: The type of SMS message (promotional or transactional).
    :param origination_number: The phone number that the message is sent from.
    :param template_name: The name of the SMS template to use when sending the
 message.
    :param template_version: The version number of the message template.

    :return The ID of the message.
    """
    try:
        response = pinpoint_client.send_messages(
```

```
            ApplicationId=project_id,
            MessageRequest={
                "Addresses": {destination_number: {"ChannelType": "SMS"}},
                "MessageConfiguration": {
                    "SMSMessage": {
                        "MessageType": message_type,
                        "OriginationNumber": origination_number,
                    }
                },
                "TemplateConfiguration": {
                    "SMSTemplate": {"Name": template_name, "Version":
 template_version}
                },
            },
        )

    except ClientError:
        logger.exception("Couldn't send message.")
        raise
    else:
        return response["MessageResponse"]["Result"][destination_number]
["MessageId"]


def main():
    region = "us-east-1"
    origination_number = "+18555550001"
    destination_number = "+14255550142"
    project_id = "7353f53e6885409fa32d07cedexample"
    message_type = "TRANSACTIONAL"
    template_name = "My_SMS_Template"
    template_version = "1"
    message_id = send_templated_sms_message(
        boto3.client("pinpoint", region_name=region),
        project_id,
        destination_number,
        message_type,
        origination_number,
        template_name,
        template_version,
    )
    print(f"Message sent! Message ID: {message_id}.")
```

```
if __name__ == "__main__":
    main()
```

- API の詳細については、 SendMessages AWS SDK for Python (Boto3) API リファレンス の「」を参照してください。

AWS SDK デベロッパーガイドとコード例の完全なリストについては、「」を参照してくださいAWS SDK での Amazon Pinpoint の使用。このトピックには、使用開始方法に関する情報と、以前の SDK バージョンの詳細も含まれています。

AWS SDK または CLI **UpdateEndpoint**で を使用する

次の例は、UpdateEndpoint を使用する方法を説明しています。

Java

SDK for Java 2.x

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.EndpointResponse;
import software.amazon.awssdk.services.pinpoint.model.EndpointRequest;
import software.amazon.awssdk.services.pinpoint.model.UpdateEndpointRequest;
import software.amazon.awssdk.services.pinpoint.model.UpdateEndpointResponse;
import software.amazon.awssdk.services.pinpoint.model.GetEndpointRequest;
import software.amazon.awssdk.services.pinpoint.model.GetEndpointResponse;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import software.amazon.awssdk.services.pinpoint.model.EndpointDemographic;
import software.amazon.awssdk.services.pinpoint.model.EndpointLocation;
import software.amazon.awssdk.services.pinpoint.model.EndpointUser;
import java.text.DateFormat;
import java.text.SimpleDateFormat;
import java.util.List;
import java.util.UUID;
```

```
import java.util.ArrayList;
import java.util.HashMap;
import java.util.Map;
import java.util.Date;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class UpdateEndpoint {
    public static void main(String[] args) {
        final String usage = """

                Usage:  <appId>

                Where:
                  appId - The ID of the application to create an endpoint for.

                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String appId = args[0];
        PinpointClient pinpoint = PinpointClient.builder()
                .region(Region.US_EAST_1)
                .build();

        EndpointResponse response = createEndpoint(pinpoint, appId);
        System.out.println("Got Endpoint: " + response.id());
        pinpoint.close();
    }

    public static EndpointResponse createEndpoint(PinpointClient client, String
 appId) {
        String endpointId = UUID.randomUUID().toString();
        System.out.println("Endpoint ID: " + endpointId);
```

```
        try {
            EndpointRequest endpointRequest = createEndpointRequestData();
            UpdateEndpointRequest updateEndpointRequest =
UpdateEndpointRequest.builder()
                    .applicationId(appId)
                    .endpointId(endpointId)
                    .endpointRequest(endpointRequest)
                    .build();

            UpdateEndpointResponse updateEndpointResponse =
client.updateEndpoint(updateEndpointRequest);
            System.out.println("Update Endpoint Response: " +
updateEndpointResponse.messageBody());

            GetEndpointRequest getEndpointRequest = GetEndpointRequest.builder()
                    .applicationId(appId)
                    .endpointId(endpointId)
                    .build();

            GetEndpointResponse getEndpointResponse =
client.getEndpoint(getEndpointRequest);
            System.out.println(getEndpointResponse.endpointResponse().address());

System.out.println(getEndpointResponse.endpointResponse().channelType());

System.out.println(getEndpointResponse.endpointResponse().applicationId());

System.out.println(getEndpointResponse.endpointResponse().endpointStatus());

System.out.println(getEndpointResponse.endpointResponse().requestId());
            System.out.println(getEndpointResponse.endpointResponse().user());

            return getEndpointResponse.endpointResponse();

        } catch (PinpointException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return null;
    }

    private static EndpointRequest createEndpointRequestData() {
        try {
```

```
            List<String> favoriteTeams = new ArrayList<>();
            favoriteTeams.add("Lakers");
            favoriteTeams.add("Warriors");
            HashMap<String, List<String>> customAttributes = new HashMap<>();
            customAttributes.put("team", favoriteTeams);

            EndpointDemographic demographic = EndpointDemographic.builder()
                    .appVersion("1.0")
                    .make("apple")
                    .model("iPhone")
                    .modelVersion("7")
                    .platform("ios")
                    .platformVersion("10.1.1")
                    .timezone("America/Los_Angeles")
                    .build();

            EndpointLocation location = EndpointLocation.builder()
                    .city("Los Angeles")
                    .country("US")
                    .latitude(34.0)
                    .longitude(-118.2)
                    .postalCode("90068")
                    .region("CA")
                    .build();

            Map<String, Double> metrics = new HashMap<>();
            metrics.put("health", 100.00);
            metrics.put("luck", 75.00);

            EndpointUser user = EndpointUser.builder()
                    .userId(UUID.randomUUID().toString())
                    .build();

            DateFormat df = new SimpleDateFormat("yyyy-MM-dd'T'HH:mm'Z'"); //
Quoted "Z" to indicate UTC, no timezone
                                                                           //
offset
            String nowAsISO = df.format(new Date());

            return EndpointRequest.builder()
                    .address(UUID.randomUUID().toString())
                    .attributes(customAttributes)
                    .channelType("APNS")
                    .demographic(demographic)
```

```
                    .effectiveDate(nowAsISO)
                    .location(location)
                    .metrics(metrics)
                    .optOut("NONE")
                    .requestId(UUID.randomUUID().toString())
                    .user(user)
                    .build();

            } catch (PinpointException e) {
                System.err.println(e.awsErrorDetails().errorMessage());
                System.exit(1);
            }
            return null;
        }
    }
```

- API の詳細については、「 API リファレンスUpdateEndpoint」の「」を参照してください。 AWS SDK for Java 2.x

AWS SDK デベロッパーガイドとコード例の完全なリストについては、「」を参照してくださいAWS SDK での Amazon Pinpoint の使用。このトピックには、使用開始方法に関する情報と、以前の SDK バージョンの詳細も含まれています。

SDK を使用した Amazon Pinpoint SMS および音声 API のコード例 AWS SDKs

次のコード例は、 AWS Software Development Kit (SDK) で Amazon Pinpoint SMS and Voice API を使用する方法を示しています。

アクションはより大きなプログラムからのコードの抜粋であり、コンテキスト内で実行する必要があります。アクションは個々のサービス機能を呼び出す方法を示していますが、関連するシナリオやサービス間の例ではアクションのコンテキストが確認できます。

AWS SDK デベロッパーガイドとコード例の完全なリストについては、「」を参照してくださいAWS SDK での Amazon Pinpoint の使用。このトピックには、使用開始方法に関する情報と、以前の SDK バージョンの詳細も含まれています。

コードの例

- SDK を使用した Amazon Pinpoint SMS および音声 API のアクション AWS SDKs

- AWS SDK または CLI SendVoiceMessageで を使用する

SDK を使用した Amazon Pinpoint SMS および音声 API のアクション AWS SDKs

次のコード例は、 AWS SDKs を使用して個々の Amazon Pinpoint SMS および音声 API アクションを実行する方法を示しています。これらの抜粋は Amazon Pinpoint SMS および音声 API を呼び出すもので、コンテキスト内で実行する必要がある大規模なプログラムからのコードの抜粋です。各例には へのリンクが含まれており GitHub、コードの設定と実行の手順を確認できます。

以下の例には、最も一般的に使用されるアクションのみ含まれています。詳細なリストについては、「Amazon Pinpoint SMS and Voice API Reference」を参照してください。

例

- AWS SDK または CLI SendVoiceMessageで を使用する

AWS SDK または CLI **SendVoiceMessage**で を使用する

以下のコード例は、SendVoiceMessage の使用方法を示しています。

Java

SDK for Java 2.x

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

```
import software.amazon.awssdk.core.client.config.ClientOverrideConfiguration;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpointsmsvoice.PinpointSmsVoiceClient;
import software.amazon.awssdk.services.pinpointsmsvoice.model.SSMLMessageType;
import
 software.amazon.awssdk.services.pinpointsmsvoice.model.VoiceMessageContent;
import
 software.amazon.awssdk.services.pinpointsmsvoice.model.SendVoiceMessageRequest;
```

```
import
 software.amazon.awssdk.services.pinpointsmsvoice.model.PinpointSmsVoiceException;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class SendVoiceMessage {

        // The Amazon Polly voice that you want to use to send the message. For a
 list
        // of voices, see https://docs.aws.amazon.com/polly/latest/dg/
voicelist.html
        static final String voiceName = "Matthew";

        // The language to use when sending the message. For a list of supported
        // languages, see
        // https://docs.aws.amazon.com/polly/latest/dg/SupportedLanguage.html
        static final String languageCode = "en-US";

        // The content of the message. This example uses SSML to customize and
 control
        // certain aspects of the message, such as by adding pauses and changing
        // phonation. The message can't contain any line breaks.
        static final String ssmlMessage = "<speak>This is a test message sent
 from "
                        + "<emphasis>Amazon Pinpoint</emphasis> "
                        + "using the <break strength='weak'/>AWS "
                        + "SDK for Java. "
                        + "<amazon:effect phonation='soft'>Thank "
                        + "you for listening.</amazon:effect></speak>";

        public static void main(String[] args) {

                final String usage = """
```

```
                                Usage:   <originationNumber> <destinationNumber>
\s

                                Where:
                                  originationNumber - The phone number or
 short code that you specify has to be associated with your Amazon Pinpoint
 account. For best results, specify long codes in E.164 format (for example,
 +1-555-555-5654).
                                  destinationNumber - The recipient's phone
 number.  For best results, you should specify the phone number in E.164 format
 (for example, +1-555-555-5654).\s
                                  """;

                if (args.length != 2) {
                        System.out.println(usage);
                        System.exit(1);
                }

                String originationNumber = args[0];
                String destinationNumber = args[1];
                System.out.println("Sending a voice message");

                // Set the content type to application/json.
                List<String> listVal = new ArrayList<>();
                listVal.add("application/json");
                Map<String, List<String>> values = new HashMap<>();
                values.put("Content-Type", listVal);

                ClientOverrideConfiguration config2 =
 ClientOverrideConfiguration.builder()
                                .headers(values)
                                .build();

                PinpointSmsVoiceClient client = PinpointSmsVoiceClient.builder()
                                .overrideConfiguration(config2)
                                .region(Region.US_EAST_1)
                                .build();

                sendVoiceMsg(client, originationNumber, destinationNumber);
                client.close();
        }
```

```
        public static void sendVoiceMsg(PinpointSmsVoiceClient client, String
 originationNumber,
                        String destinationNumber) {
                try {
                        SSMLMessageType ssmlMessageType =
 SSMLMessageType.builder()
                                        .languageCode(languageCode)
                                        .text(ssmlMessage)
                                        .voiceId(voiceName)
                                        .build();

                        VoiceMessageContent content =
 VoiceMessageContent.builder()
                                        .ssmlMessage(ssmlMessageType)
                                        .build();

                        SendVoiceMessageRequest voiceMessageRequest =
 SendVoiceMessageRequest.builder()

 .destinationPhoneNumber(destinationNumber)

 .originationPhoneNumber(originationNumber)
                                        .content(content)
                                        .build();

                        client.sendVoiceMessage(voiceMessageRequest);
                        System.out.println("The message was sent successfully.");

                } catch (PinpointSmsVoiceException e) {
                        System.err.println(e.awsErrorDetails().errorMessage());
                        System.exit(1);
                }
        }
}
```

- API の詳細については、「 API リファレンスSendVoiceMessage」の「」を参照してください。 AWS SDK for Java 2.x

JavaScript

SDK for JavaScript (v2)

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

```
"use strict";

var AWS = require("aws-sdk");

// The AWS Region that you want to use to send the voice message. For a list of
// AWS Regions where the Amazon Pinpoint SMS and Voice API is available, see
// https://docs.aws.amazon.com/pinpoint-sms-voice/latest/APIReference/
var aws_region = "us-east-1";

// The phone number that the message is sent from. The phone number that you
// specify has to be associated with your Amazon Pinpoint account. For best
 results, you
// should specify the phone number in E.164 format.
var originationNumber = "+12065550110";

// The recipient's phone number. For best results, you should specify the phone
// number in E.164 format.
var destinationNumber = "+12065550142";

// The language to use when sending the message. For a list of supported
// languages, see https://docs.aws.amazon.com/polly/latest/dg/
SupportedLanguage.html
var languageCode = "en-US";

// The Amazon Polly voice that you want to use to send the message. For a list
// of voices, see https://docs.aws.amazon.com/polly/latest/dg/voicelist.html
var voiceId = "Matthew";

// The content of the message. This example uses SSML to customize and control
// certain aspects of the message, such as the volume or the speech rate.
// The message can't contain any line breaks.
```

```
var ssmlMessage =
  "<speak>" +
  "This is a test message sent from <emphasis>Amazon Pinpoint</emphasis> " +
  "using the <break strength='weak'/>AWS SDK for JavaScript in Node.js. " +
  "<amazon:effect phonation='soft'>Thank you for listening." +
  "</amazon:effect>" +
  "</speak>";

// The phone number that you want to appear on the recipient's device. The phone
// number that you specify has to be associated with your Amazon Pinpoint
 account.
var callerId = "+12065550199";

// The configuration set that you want to use to send the message.
var configurationSet = "ConfigSet";

// Specify that you're using a shared credentials file, and optionally specify
// the profile that you want to use.
var credentials = new AWS.SharedIniFileCredentials({ profile: "default" });
AWS.config.credentials = credentials;

// Specify the region.
AWS.config.update({ region: aws_region });

//Create a new Pinpoint object.
var pinpointsmsvoice = new AWS.PinpointSMSVoice();

var params = {
  CallerId: callerId,
  ConfigurationSetName: configurationSet,
  Content: {
    SSMLMessage: {
      LanguageCode: languageCode,
      Text: ssmlMessage,
      VoiceId: voiceId,
    },
  },
  DestinationPhoneNumber: destinationNumber,
  OriginationPhoneNumber: originationNumber,
};

//Try to send the message.
pinpointsmsvoice.sendVoiceMessage(params, function (err, data) {
  // If something goes wrong, print an error message.
```

```
  if (err) {
    console.log(err.message);
    // Otherwise, show the unique ID for the message.
  } else {
    console.log("Message sent! Message ID: " + data["MessageId"]);
  }
});
```

- API の詳細については、「 API リファレンスSendVoiceMessage」の「」を参照してください。 AWS SDK for JavaScript

Python

SDK for Python (Boto3)

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

```
import logging
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)


def send_voice_message(
    sms_voice_client,
    origination_number,
    caller_id,
    destination_number,
    language_code,
    voice_id,
    ssml_message,
):
    """
```

```
    Sends a voice message using speech synthesis provided by Amazon Polly.

    :param sms_voice_client: A Boto3 PinpointSMSVoice client.
    :param origination_number: The phone number that the message is sent from.
                               The phone number must be associated with your
Amazon
                               Pinpoint account and be in E.164 format.
    :param caller_id: The phone number that you want to appear on the recipient's
                      device. The phone number must be associated with your
Amazon
                      Pinpoint account and be in E.164 format.
    :param destination_number: The recipient's phone number. Specify the phone
                               number in E.164 format.
    :param language_code: The language to use when sending the message.
    :param voice_id: The Amazon Polly voice that you want to use to send the
message.
    :param ssml_message: The content of the message. This example uses SSML to
control
                         certain aspects of the message, such as the volume and
the
                         speech rate. The message must not contain line breaks.
    :return: The ID of the message.
    """
    try:
        response = sms_voice_client.send_voice_message(
            DestinationPhoneNumber=destination_number,
            OriginationPhoneNumber=origination_number,
            CallerId=caller_id,
            Content={
                "SSMLMessage": {
                    "LanguageCode": language_code,
                    "VoiceId": voice_id,
                    "Text": ssml_message,
                }
            },
        )
    except ClientError:
        logger.exception(
            "Couldn't send message from %s to %s.",
            origination_number,
            destination_number,
        )
        raise
    else:
```

```
        return response["MessageId"]


def main():
    origination_number = "+12065550110"
    caller_id = "+12065550199"
    destination_number = "+12065550142"
    language_code = "en-US"
    voice_id = "Matthew"
    ssml_message = (
        "<speak>"
        "This is a test message sent from <emphasis>Amazon Pinpoint</emphasis> "
        "using the <break strength='weak'/>AWS SDK for Python (Boto3). "
        "<amazon:effect phonation='soft'>Thank you for listening."
        "</amazon:effect>"
        "</speak>"
    )
    print(f"Sending voice message from {origination_number} to
 {destination_number}.")
    message_id = send_voice_message(
        boto3.client("pinpoint-sms-voice"),
        origination_number,
        caller_id,
        destination_number,
        language_code,
        voice_id,
        ssml_message,
    )
    print(f"Message sent!\nMessage ID: {message_id}")


if __name__ == "__main__":
    main()
```

- API の詳細については、SendVoiceMessageAWS 「 SDK for Python (Boto3) API リファレンス」の「」を参照してください。

AWS SDK デベロッパーガイドとコード例の完全なリストについては、「」を参照してくださいAWS SDK での Amazon Pinpoint の使用。このトピックには、使用開始方法に関する情報と、以前の SDK バージョンの詳細も含まれています。

Amazon Pinpoint のセキュリティ

のクラウドセキュリティが最優先事項 AWS です。お客様は AWS 、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メリットを得られます。

セキュリティは、 AWS とユーザーの間で共有される責任です。責任共有モデルではこれを、クラウドのセキュリティ、およびクラウド内で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 — クラウドで AWS 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 AWS は AWS にあります。 AWS また、 では、安全に使用できるサービスも提供しています。コンプライアンスAWS プログラム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者は定期的にセキュリティの有効性をテストおよび検証。Amazon Pinpoint 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AWS による対象範囲内のサービスコンプライアンスプログラム」を参照してください。
- クラウドのセキュリティ — お客様の責任は、使用する AWS サービスによって決まります。また、お客様は、データの機密性、会社の要件、適用される法律や規制など、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、Amazon Pinpoint 使用時における責任共有モデルの適用法を理解するのに役立ちます。以下のトピックでは、セキュリティとコンプライアンスの目的を満たすように Amazon Pinpoint を設定する方法について説明します。また、Amazon Pinpoint リソースのモニタリングや保護に役立つ他の AWS のサービスの使用方法についても説明します。

詳細については、「Amazon Pinpoint Resilient Architecture Guide」のリファレンスアーキテクチャについての説明を参照してください。

トピック

- Amazon Pinpoint のセキュリティベストプラクティス

Amazon Pinpoint におけるデータ保護

責任 AWS 共有モデル、Amazon Pinpoint でのデータ保護に適用されます。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のよくある質問」を参照してください。欧州でのデータ保護の詳細については、AWS セキュリティブログに投稿された記事「AWS 責任共有モデルおよび GDPR」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 (IAM) 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。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。TLS 1.2 は必須であり TLS 1.3 がお勧めです。
- で API と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 API AWS を介して にアクセスするときに FIPS 140-2 検証済みの暗号化モジュールが必要な場合は、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) 140-2」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、API、または SDK を使用して Amazon Pinpoint AWS CLIまたは他の AWS のサービス を操作する場合も同様です。 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。外部サーバーへの URL を提供する場合は、そのサーバーへのリクエストを検証するための認証情報を URL に含めないように強くお勧めします。

サービスを設定および使用する方法に応じて、Amazon Pinpoint はお客様やお客様の顧客に関する次のタイプの個人データを保存します。

設定データ

これには、サポートされているチャンネルを介して Amazon Pinpoint からメッセージを送信する方法とタイミング、およびメッセージの送信先のユーザーセグメントを定義する設定や認証情報などのプロジェクト設定データが含まれます。メッセージを送信するために、このデータには、E メールメッセージ専用の IP アドレス、SMS テキストメッセージ用のショートコードと送信者 ID、Apple プッシュ通知サービス (APNs) や Firebase Cloud Messaging (FCM) などのプッシュ通知サービスと通信するための認証情報を含めることができます。

ユーザーおよびエンドポイントのデータ

これには、Amazon Pinpoint プロジェクトのユーザーとエンドポイントに関するデータを保存および管理するために使用する標準属性とカスタム属性が含まれます。属性には、特定のユーザー (ユーザー名など) またはユーザーの特定のエンドポイント (ユーザーの E メールアドレス、携帯電話番号、モバイルデバイストークンなど) に関する情報を格納できます。このデータには、Amazon Pinpoint プロジェクトのユーザーと外部システム (顧客関係管理システムなど) のユーザーを関連付ける外部ユーザー ID を含めることもできます。このデータに何が含まれるかの詳細については、「Amazon Pinpoint API リファレンス」の「ユーザー」および「エンドポイント」の「スキーマ」を参照してください。

分析データ

これには、重要業績評価指標 (KPI) とも呼ばれるメトリクスに関するデータが含まれます。メトリクスは、ユーザーエンゲージメントや購入活動などの分野で Amazon Pinpoint プロジェクトのパフォーマンスに関する分析情報を提供します。これには、プロジェクトのユーザー統計の分析情報を提供するメトリクスのデータも含まれます。データは、ユーザーの居住都市など、ユーザーおよびエンドポイントの標準属性とカスタム属性から派生できます。また、プロジェクトのために送信する E メールメッセージのオープンイベントやクリックイベントなどのイベントからも派生できます。

インポートされたデータ

これには、外部ソースから追加またはインポートして Amazon Pinpoint で使用するユーザー、セグメンテーション、および分析データが含まれます。例としては、静的セグメントを構築するために (コンソールから直接または Amazon S3 バケットから) Amazon Pinpoint 内にインポートする JSON ファイルがあります。その他の例としては、動的セグメントを構築するためにプログラムで追加するエンドポイントデータ、ダイレクトメッセージを送信する先のエンドポイントアド

レス、およびアプリケーションから Amazon Pinpoint にレポートするように設定するイベントがあります。

トピック

データ暗号化

Amazon Pinpoint のデータは転送中および保管時に暗号化されます。Amazon Pinpoint に送信したデータは、受信時に暗号化されて保存されます。Amazon Pinpoint から取得するデータは、現在のセキュリティプロトコルを使用して転送されます。

保管中の暗号化

Amazon Pinpoint は、保存するすべてのデータを自動的に暗号化します。これには、設定データ、ユーザーとエンドポイントのデータ、分析データ、および Amazon Pinpoint に追加またはインポートするすべてのデータが含まれます。データを暗号化するために、Amazon Pinpoint はサービスがユーザーに代わって所有および維持する内部 AWS Key Management Service（ AWS KMS) キーを使用します。これらのキーは定期的に更新されます。の詳細については AWS KMS、「 AWS Key Management Service デベロッパーガイド」を参照してください。

転送中の暗号化

Amazon Pinpoint は、HTTPS および Transport Layer Security (TLS) 1.2 以降を使用して、クライアントやアプリケーションと通信します。他の AWS サービスと通信するために、Amazon Pinpoint は HTTPS および TLS 1.2 を使用します。さらに、コンソール、 AWS SDK、または を使用して Amazon Pinpoint リソースを作成および管理する場合 AWS Command Line Interface、すべての通信は HTTPS および TLS 1.2 を使用して保護されます。

キー管理

Amazon Pinpoint データを暗号化するために、Amazon Pinpoint はサービスがユーザーに代わって所有および維持する内部 AWS KMS キーを使用します。これらのキーは定期的に更新されます。Amazon Pinpoint に保存しているデータを暗号化するために、独自のキー AWS KMS やその他のキーをプロビジョニングして使用することはできません。

インターネットトラフィックのプライバシー

インターネットトラフィックのプライバシーとは、Amazon Pinpoint とオンプレミスのクライアントとアプリケーション間、および同じ AWS リージョン内の Amazon Pinpoint と他の AWS リソース間の接続とトラフィックを保護することです。以下の機能とプラクティスは、Amazon Pinpoint のインターネットワークトラフィックのプライバシーを確保するのに役立ちます。

Amazon Pinpoint とオンプレミスのクライアントやアプリケーションとの間のトラフィック

Amazon Pinpoint とオンプレミスネットワークのクライアントやアプリケーションとの間でプライベート接続を確立するには、 AWS Direct Connectを使用できます。これにより、標準の光ファイバーイーサネットケーブルを使用して、ネットワークを AWS Direct Connect ロケーションにリンクできます。ケーブルの一端はユーザーのルーターに接続します。もう 1 つの端は AWS Direct Connect ルーターに接続されています。詳細については、『AWS Direct Connectユーザーガイド』の「What is AWS Direct Connect ?」を参照してください。

公開 API を介した Amazon Pinpoint へのアクセスを保護するために、API コールに関する Amazon Pinpoint の要件に準拠することをお勧めします。Amazon Pinpoint では、クライアントで Transport Layer Security (TLS) 1.2 以降を使用する必要があります。また、クライアントは、Ephemeral Diffie-Hellman (DHE) や Elliptic Curve Diffie-Hellman Ephemeral (ECDHE) などの Perfect Forward Secrecy (PFS) を使用した暗号スイートもサポートしている必要があります。モードは、Java 7 以降など、最近のほとんどのシステムでサポートされています。

さらに、リクエストは、 AWS アカウントの AWS Identity and Access Management (IAM) プリンシパルに関連付けられているアクセスキー 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) を使用して一時的な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。

Amazon Pinpoint と他の AWS リソース間のトラフィック

Amazon Pinpoint と同じ AWS リージョン内の他の AWS リソース間の通信を保護するために、Amazon Pinpoint はデフォルトで HTTPS と TLS 1.2 を使用します。

Amazon Pinpoint 用のインターフェイス VPC エンドポイントの作成

仮想プライベートクラウド (VPC) と Amazon Pinpoint のエンドポイント間のプライベート接続は、インターフェイス VPC エンドポイントを作成することで確立できます。

インターフェイスエンドポイントは、インターネットゲートウェイAWS PrivateLink、NAT デバイス、VPN 接続、または なしで Amazon Pinpoint APIs にプライベートにアクセスできるテクノロジーである を利用しています AWS Direct Connect。VPC のインスタンスは、パブリック IP アドレスがなくても、 AWS PrivateLinkと統合する Amazon Pinpoint API と通信できます。

詳細については、「AWS PrivateLink ガイド」を参照してください。

インターフェイス VPC エンドポイントの作成

Amazon VPC コンソールまたは AWS Command Line Interface () を使用してインターフェイスエンドポイントを作成できますAWS CLI。詳細については、「 AWS PrivateLink ガイド」の「インターフェイスエンドポイントの作成」を参照してください。

Amazon Pinpoint は、次のサービス名をサポートしています。

- `com.amazonaws.region.pinpoint`
- `com.amazonaws.region.pinpoint-sms-voice-v2`

インターフェイスエンドポイントのプライベート DNS を有効にすると、 のデフォルトの DNS 名 AWS リージョン、例えば を使用して Amazon Pinpoint に API リクエストを行うことができますcom.amazonaws.*us-east-1*.pinpoint。詳細については、「AWS PrivateLink ガイド」の「DNS ホスト名」を参照してください。

Amazon Pinpoint で現在利用可能なすべてのリージョンとエンドポイントの一覧については、「Amazon Web Services 全般のリファレンス」の「AWS サービスエンドポイント」を参照してください。

VPCエンドポイントポリシーの作成

VPC エンドポイントには、アクセスを制御するエンドポイントポリシーをアタッチできます。このポリシーでは、以下の情報を指定します。

- アクションを実行できるプリンシパル。
- 実行可能なアクション。
- このアクションを実行できるリソース。

詳細については、「AWS PrivateLink ガイド」の「Control access to services using endpoint policies (エンドポイントポリシーを使用してサービスへのアクセスをコントロールする)」を参照してください。

例: VPC エンドポイントポリシー

次の VPC エンドポイントポリシーは、リストされた Amazon Pinpoint アクションへのアクセスを、すべてのプリンシパルに、すべてのリソースについて許可します。

```
{
"Statement": [
    {
      "Principal": "*",
      "Action": [
        "mobiletargeting:CreateCampaign",
        "mobiletargeting:CreateApp",
        "mobiletargeting:DeleteApp",
      ],
      "Effect": "Allow",
      "Resource": "*"
    }
  ]
}
```

Amazon Pinpoint のための Identity and Access Management

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 Amazon Pinpoint 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
- Amazon Pinpoint で IAM が機能する仕組み
- IAM ポリシーの Amazon Pinpointアクション
- Amazon Pinpoint のアイデンティティベースポリシーの例
- Amazon Pinpoint の一般的なタスクの IAM ロール
- Amazon Pinpoint Identity and Access Management 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、Amazon Pinpoint で行う作業によって異なります。

サービスユーザー – ジョブを実行するために Amazon Pinpoint サービスを使用する場合は、管理者から必要なアクセス許可と認証情報が与えられます。さらに多くの Amazon Pinpoint 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。Amazon Pinpoint の機能にアクセスできない場合は、「Amazon Pinpoint Identity and Access Management のトラブルシューティング」を参照してください。

サービス管理者 – 社内の Amazon Pinpoint リソースを担当している場合は、通常、Amazon Pinpoint へのフルアクセスがあります。サービスのユーザーがどの Amazon Pinpoint の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。その後、IAM 管理者にリクエストを送信して、サービスユーザーの権限を変更する必要があります。このページの情報を点検して、IAM の基本概念を理解してください。会社で Amazon Pinpoint と IAM を併用する方法の詳細については、「Amazon Pinpoint で IAM が機能する仕組み」を参照してください。

IAM 管理者 – 管理者は、Amazon Pinpoint 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。IAM で使用できる Amazon Pinpoint 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、「Amazon Pinpoint のアイデンティティベースポリシーの例」を参照してください。

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アイデンティティとしてサインインする場合、IAM ロールを使用して、前もって管理者により ID フェデレーションが設定されています。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「 にサインインする方法 AWS アカウント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 Software Development Kit (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 API AWS 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、多要素認証 (MFA) を使用してアカウントのセキュリティを向上させることをお勧めします。詳細については、『AWS IAM Identity Center ユーザーガイド』の「Multi-factor authentication」(多要素認証) および『IAM ユーザーガイド』の「AWSにおける多要素認証 (MFA) の使用」を参照してください。

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 およびリソースへの AWS のサービス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「ルートユーザー認証情報が必要なタスク」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保有する IAM ユーザーを作成する代わりに、一時認証情報を使用することをお勧めします。ただし、IAM ユーザーでの長期的な認証情報が必要な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、IAM 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のためにアクセスキーを定期的にローテーションする」を参照してください。

IAM グループは、IAM ユーザーの集団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、IAMAdmins という名前のグループを設定して、そのグループに IAM リソースを管理する権限を与えることができます。

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、ロールでは一時的な認証情報が提供されます。詳細については、『IAM ユーザーガイド』の「IAM ユーザー (ロールではなく) の作成が適している場合」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 ユーザーに似ていますが、特定のユーザーには関連付けられていません。ロール を切り替える AWS Management Console ことで、 で IAM 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 AWS CLI オペレーションを呼び出すか、カスタム URL を使用します。ロールを使用する方法の詳細については、「IAM ユーザーガイド」の「IAM ロールの使用」を参照してください。

IAM ロールと一時的な認証情報は、次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アイデンティティに権限を割り当てるには、ロールを作成してそのロールの権限を定義します。フェデレーティッドアイデンティティが認証されると、そのアイデンティティはロールに関連付けられ、ロールで定義されている権限が付与されます。フェデレーションの詳細については、『IAM ユーザーガイド』の「サードパーティーアイデンティティプロバイダー向けロールの作成」 を参照してください。IAM アイデンティティセンターを使用する場合、権限セットを設定します。アイデンティティが認証後にアクセスできるものを制御するため、IAM Identity Center は、権限セットを IAM のロールに関連付けます。権限セットの詳細については、『AWS IAM Identity Center ユーザーガイド』の「権限セット」を参照してください。
- 一時的な IAM ユーザー権限 - IAM ユーザーまたはロールは、特定のタスクに対して複数の異なる権限を一時的に IAM ロールで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、自分のアカウントのリソースにアクセスすることを、別のアカウントの人物 (信頼済みプリンシパル) に許可できます。クロスアカウントアクセス権を付与する主な方法は、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) リソースに直接ポリシーをアタッチできます。クロスアカウントアクセスにおけるロールとリソースベースのポリシーの違いについては、『IAM ユーザーガイド』の「IAM ロールとリソースベースのポリシーとの相違点」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、あるサービスで呼び出しを行うと、通常そのサービスによって Amazon EC2 でア

プリケーションが実行されたり、Amazon S3 にオブジェクトが保存されたりします。サービスでは、呼び出し元プリンシパルの権限、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場合 AWS、ユーザーは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 へのリクエストリクエスト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アクセス許可が必要です。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 - サービスがユーザーに代わってアクションを実行するために引き受ける IAM ロールです。IAM 管理者は、IAM 内からサービスロールを作成、変更、削除できます。詳細については、IAM ユーザーガイドの「 AWS のサービスにアクセス許可を委任するロールの作成」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権限を表示できますが、編集することはできません。

- Amazon EC2 で実行されているアプリケーション – 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 インスタンス内でのアクセスキーの保存に推奨されます。 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、EC2 インスタンスで実行されるプログラムは一時的な認証情報を取得できます。詳細については、『IAM ユーザーガイド』の「Amazon EC2 インスタンスで実行されるアプリケーションに IAM ロールを使用して権限を付与する」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、『IAM ユーザーガイド』の「(IAM ユーザーではなく) IAM ロールをいつ作成したら良いのか?」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、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) AWS 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、「IAM ユーザーガイド」の「JSON ポリシー概要」を参照してください。

管理者は AWS JSON ポリシーを使用して、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んなリソースにどんな条件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。IAM 管理者は、リソースで必要なアクションを実行するための権限をユーザーに付与する IAM ポリシーを作成できます。その後、管理者はロールに IAM ポリシーを追加し、ユーザーはロールを引き継ぐことができます。

IAM ポリシーは、オペレーションの実行方法を問わず、アクションの権限を定義します。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 API からロール情報を取得できます。

アイデンティティベースのポリシー

アイデンティティベースポリシーは、IAM ユーザー、ユーザーのグループ、ロールなど、アイデンティティにアタッチできる JSON 権限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、IAM ユーザーガイドの「IAM ポリシーの作成」を参照してください。

アイデンティティベースポリシーは、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。管理ポリシーまたはインラインポリシーのいずれ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比較」を参照してください。

Amazon Pinpoint は、ID ベースのポリシーを使用した Amazon Pinpoint リソースへのアクセスの制御をサポートしています。

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、リソースに添付する 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には例として、IAM ロールの信頼ポリシーや Amazon S3 バケットポリシーがあげられます。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制御できます。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では、IAM の AWS マネージドポリシーを使用できません。

Amazon Pinpoint は、リソースベースのポリシーを使用した Amazon Pinpoint リソースへのアクセスの制御をサポートしています。

アクセスコントロールリスト (ACL)

アクセスコントロールリスト (ACL) は、どのプリンシパル (アカウントメンバー、ユーザー、またはロール) がリソースにアクセスするための権限を持つかをコントロールします。ACL 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は使用しません。

Amazon S3、 AWS WAF、および Amazon VPC は、ACLs。ACL の詳細については、『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。

Amazon Pinpoint は、ACL を使用した Amazon Pinpoint リソースへのアクセスの制御をサポートしていません。

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によって IAM エンティティ (IAM ユーザーまたはロール) に付与できる権限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。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 (SCP) を一部またはすべてのアカウントに適用できます。SCP 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を制限します AWS アカウントのルートユーザー。Organizations と SCP の詳細については、『AWS Organizations ユーザーガイド』の「SCP の仕組み」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、『IAM ユーザーガイド』の「セッションポリシー」を参照してください。

Amazon Pinpoint は、これらのタイプのポリシーを使用した Amazon Pinpoint リソースへのアクセスの制御をサポートしています。

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与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、IAM ユーザーガイドの「ポリシー評価ロジック」を参照してください。

Amazon Pinpoint で IAM が機能する仕組み

Amazon Pinpoint を使用するには、 AWS アカウントのユーザーには、分析データの表示、プロジェクトの作成、ユーザーセグメントの定義、キャンペーンのデプロイなどを許可するアクセス許可が必要です。モバイルアプリケーションまたはウェブアプリケーションを Amazon Pinpoint と統合する場合、アプリケーションのユーザーは Amazon Pinpoint へのアクセスも必要とします。このアクセスにより、アプリケーションはエンドポイントを登録し、使用状況データを Amazon Pinpoint にレポートできます。Amazon Pinpoint 機能へのアクセスを許可するには、IAM ID または Amazon Pinpoint リソースの Amazon Pinpoint アクションを許可する AWS Identity and Access Management (IAM) ポリシーを作成します。

IAM は、管理者が AWS リソースへのアクセスを安全に制御できるようにするサービスです。IAM ポリシーには、特定のリソースでユーザーが実行できる特定のアクションを許可または拒否するステートメントが含まれています。Amazon Pinpoint は、 ポリシーの一連のアクションを提供し、Amazon Pinpoint ユーザーの詳細なアクセス権限を指定できます。これにより、重要なデータの漏洩やリソースの侵害を起こすような過度に寛容なポリシーを作成することなく、Amazon Pinpoint への適切なレベルのアクセスを付与できます。例えば、Amazon Pinpoint の管理者には無制限のアクセスを許可する一方で、特定のプロジェクトにのみアクセスする必要のある個人には読み取り専用アクセスを付与できます。

IAM を使用して Amazon Pinpoint へのアクセスを管理する前に、Amazon Pinpoint で使用できる IAM 機能について理解しておく必要があります。Amazon Pinpoint およびその他の AWS のサービスが IAM と連携する方法の概要を把握するには、「IAM ユーザーガイド」の「IAM AWS と連携するのサービス」を参照してください。

トピック

- Amazon Pinpoint アイデンティティベースのポリシー
- Amazon Pinpoint リソースベースのアクセス許可ポリシー
- Amazon Pinpoint タグに基づいた認可
- Amazon Pinpoint IAM ロール

Amazon Pinpoint アイデンティティベースのポリシー

IAM アイデンティティベースのポリシーでは、許可または拒否するアクションとリソース、アクションを許可または拒否する条件を指定できます。Amazon Pinpoint 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ています。JSON ポリシーで使用できるすべての要素について学ぶには、『IAM ユーザーガイド』の「IAM JSON ポリシーの要素のリファレンス」を参照してください。

アクション

管理者は AWS JSON ポリシーを使用して、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んなリソースにどんな条件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`要素には、ポリシー内の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が記述されます。ポリシーアクションの名前は通常、関連付けられた AWS API オペレーションと同じです。一致する API オペレーションのない権限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。

つまり、ポリシーアクションによって、ユーザーが Amazon Pinpoint コンソールで実行できる操作が制御されます。また、 AWS SDKs、または Amazon Pinpoint APIsを直接使用して、ユーザーがプログラムで実行できる操作も制御します。 AWS Command Line Interface AWS CLI

Amazon Pinpoint のポリシーアクションでは、次のプレフィックスを使用します。

- **`mobiletargeting`** – Amazon Pinpoint API (Amazon Pinpoint のプライマリ API) から派生するアクション用。
- **`sms-voice`** – Amazon Pinpoint SMS および音声 API から派生するアクション用。これは、Amazon Pinpoint の SMS チャンネルおよび音声チャンネルを使用および管理するための高度なオプションを提供する補足 API です。

例えば、プロジェクトのすべてのセグメントに関する情報を表示するためのアクセス許可をユーザーに付与するには、ポリシーに `mobiletargeting:GetSegments` アクションを含めます (これは、Amazon Pinpoint API の `GetSegments` オペレーションに対応するアクションです)。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。Amazon Pinpoint は、ユーザーが実行できるタスクを記述した独自のアクションのセットを定義します。

単一のステートメントで複数のアクションを指定するには、アクション間をコンマで区切ります。

```
"Action": [
      "mobiletargeting:action1",
      "mobiletargeting:action2"
```

ワイルドカード (*) を使用して複数のアクションを指定することもできます。例えば、`Get` 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": "mobiletargeting:Get*"
```

ただしベストプラクティスとして、最小特権 の原則に準拠したポリシーを作成してください。つまり、特定のアクションを実行するために必要なアクセス許可のみが含まれたポリシーを作成します。

ポリシーで使用できる Amazon Pinpoint アクションのリストについては、IAM ポリシーの Amazon Pinpointアクション を参照してください。

リソース

管理者は AWS JSON ポリシーを使用して、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どのような条件でアクションを実行できるかということです。

Resource JSON ポリシー要素は、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、リソースレベルの権限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": "*"
```

例えば、mobiletargeting:GetSegments アクションは、特定の Amazon Pinpoint プロジェクトに関連付けられているすべてのセグメントに関する情報を取得します。プロジェクトを識別するには、次の形式の ARN を使用します。

```
arn:aws:mobiletargeting:${Region}:${Account}:apps/${projectId}
```

ARN の形式の詳細については、「AWS 全般のリファレンス」の「Amazon リソースネーム (ARN)」を参照してください。

IAM ポリシーでは、次のタイプの Amazon Pinpoint リソースに対して ARN を指定できます。

- キャンペーン
- ジャーニー
- メッセージテンプレート (一部のコンテキストではテンプレートと呼ばれます)
- プロジェクト (一部のコンテキストではアプリまたはアプリケーションと呼ばれます)
- 推奨モデル (一部のコンテキストでは 推奨と呼ばれます)
- セグメント

例えば、プロジェクト ID 810c7aab86d42fb2b56c8c966example を持つプロジェクトのポリシーステートメントを作成するには、次の ARN を使用します。

```
"Resource": "arn:aws:mobiletargeting:us-
east-1:123456789012:apps/810c7aab86d42fb2b56c8c966example"
```

特定のアカウントに属するすべてのプロジェクトを指定するには、ワイルドカード (*) を使用します。

```
"Resource": "arn:aws:mobiletargeting:us-east-1:123456789012:apps/*"
```

リソースを作成するための特定のアクションなど、一部の Amazon Pinpoint アクションは、特定のリソースに対して実行できません。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。

```
"Resource": "*"
```

IAM ポリシーでは、次のタイプの Amazon Pinpoint SMS および音声リソースに対しても ARN を指定できます。

- 設定セット
- オプトアウトリスト
- 電話番号
- プール
- 送信者 ID

例えば、電話番号 ID が phone-1234567890123456789012345678901 2 の電話番号についてポリシーステートメントを作成するには、次の ARN を使用します。

```
"Resource": "arn:aws:sms-voice:us-east-1:123456789012:phone-number/
phone-1234567890123456789012345678901 2"
```

特定のアカウントに属するすべての電話番号を指定するには、電話番号 ID の代わりにワイルドカード (*) を使用します。

```
"Resource": "arn:aws:sms-voice:us-east-1:123456789012:phone-number/*"
```

Amazon Pinpoint SMS および音声アクションの中には、使用限度のようなアカウントレベルの設定管理に関するアクションなど、特定のリソースに対して実行されないものもあります。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。

```
"Resource": "*"
```

一部の Amazon Pinpoint API アクションには、複数のリソースが関連します。例えば、TagResource アクションは複数のプロジェクトにタグを追加できます。単一のステートメントで複数のリソースを指定するには、ARN 間をカンマで区切ります。

```
"Resource": [
      "resource1",
      "resource2"
```

Amazon Pinpoint のリソースタイプとそれらの ARN のリストを確認するには、『IAM ユーザーガイド』の「Amazon Pinpoint で定義されるリソースタイプ」を参照してください。どのアクションで各リソースタイプの ARN を指定できるかについては、『IAM ユーザーガイド』の「Amazon Pinpoint で定義されるアクション」を参照してください。

条件キー

管理者は AWS JSON ポリシーを使用して、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んなリソースにどんな条件でアクションを実行できるかということです。

Condition 要素 (または Condition ブロック) を使用すると、ステートメントが有効な条件を指定できます。Condition 要素はオプションです。イコールや未満などの 条件演算子 を使用して条件式を作成することで、ポリシーの条件とリクエスト内の値を一致させることができます。

1 つのステートメントに複数の Condition 要素を指定するか、1 つの Condition 要素に複数のキーを指定すると、 AWS は AND 論理演算子を使用してそれらを評価します。1 つの条件キーに複数の値を指定すると、 は論理ORオペレーションを使用して条件 AWS を評価します。ステートメントの権限が付与される前にすべての条件が満たされる必要があります。

条件を指定する際にプレースホルダー変数も使用できます。例えば IAM ユーザーに、IAM ユーザー名がタグ付けされている場合のみリソースにアクセスできる権限を付与することができます。詳細については、『IAM ユーザーガイド』の「IAM ポリシーの要素: 変数およびタグ」を参照してください。

AWS は、グローバル条件キーとサービス固有の条件キーをサポートします。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」のAWS 「 グローバル条件コンテキストキー」を参照してください。

Amazon Pinpoint は一連の独自の条件キーを定義し、一部のグローバル条件キーもサポートします。すべての AWS グローバル条件キーのリストを確認するには、「IAM ユーザーガイド」のAWS 「グローバル条件コンテキストキー」を参照してください。 Amazon Pinpoint 条件キーのリストについては、『IAM ユーザーガイド』の「Amazon Pinpoint の条件キー」を参照してください。条件キーを使用できるアクションとリソースについては、『IAM ユーザーガイド』の「Amazon Pinpoint で定義されるアクション」を参照してください。

例

Amazon Pinpoint でのアイデンティティベースのポリシーの例は、「Amazon Pinpoint のアイデンティティベースポリシーの例」でご確認ください。

Amazon Pinpoint リソースベースのアクセス許可ポリシー

リソースベースのアクセス許可ポリシーとは、指定されたプリンシパルが Amazon Pinpoint リソースに対して、どのアクションをどの条件で実行できるかを指定する JSON ポリシードキュメントです。Amazon Pinpoint は、キャンペーン、ジャーニー、メッセージテンプレート (テンプレート)、レコメンダーモデル (レコメンダー)、プロジェクト (アプリケーション)、セグメントに対するリソースベースのパーミッションポリシーをサポートしています。

例

Amazon Pinpoint リソースベースのポリシーの例を表示するには、「the section called “アイデンティティベースポリシーの例”」を参照してください。

Amazon Pinpoint タグに基づいた認可

タグは、特定のタイプの Amazon Pinpoint リソースに関連付けたり、Amazon Pinpoint へのリクエストに渡したりすることができます。タグに基づいてアクセスを管理するには、`aws:ResourceTag/${TagKey}`、`aws:RequestTag/${TagKey}`、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。

Amazon Pinpoint リソースのタグ付けの詳細 (IAM ポリシーの例を含む) については、Amazon Pinpoint のリソースにタグを付ける を参照してください。

Amazon Pinpoint IAM ロール

IAM ロールは AWS アカウント内のエンティティで、特定の許可を持っています。

Amazon Pinpoint での一時的な認証情報の使用

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する、IAM ロールを引き受ける、またはクロスアカウントロールを引き受けることができます。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 (AWS STS) API オペレーションを呼び出し AWS Security Token Service ますGetFederationToken。

Amazon Pinpoint は、一時的な認証情報の使用をサポートします。

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。サービスリンクロールは IAM アカウント内に表示され、サービスによって所有されます。IAM 管理者は、サービスリンクロールの許可を表示できますが、編集することはできません。

Amazon Pinpoint は、サービスにリンクされたロールを使用しません。

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、IAM 管理者は、このロールの権限を変更できます。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。

Amazon Pinpoint は、サービスロールの使用をサポートしています。

IAM ポリシーの Amazon Pinpointアクション

AWS アカウントの Amazon Pinpoint リソースへのアクセスを管理するには、Amazon Pinpoint アクションを AWS Identity and Access Management (IAM) ポリシーに追加します。ポリシーでアクションを使用することで、ユーザーが実行できる操作を Amazon Pinpoint コンソールで制御できます。AWS SDKs 、 AWS Command Line Interface （ AWS CLI ）、または Amazon Pinpoint APIs を直接使用して、ユーザーがプログラムで実行できる操作を制御することもできます。

ポリシーでは、適切な Amazon Pinpoint 名前空間に続けてコロンとアクション名 (`GetSegments` など) を使用して、各アクションを指定します。ほとんどのアクションは、特定の URI および HTTP メソッドを使用した Amazon Pinpoint API へのリクエストに対応しています。例えば、ユーザーのポリシーで `mobiletargeting:GetSegments` アクションを許可すると、ユーザーは /

apps/*projectId*/segments URI に HTTP GET リクエストを送信することで、プロジェクトのすべてのセグメントに関する情報を取得することが許可されます。このポリシーでは、ユーザーはその情報をコンソールで表示し、 AWS SDK または を使用してその情報を取得することもできます AWS CLI。

各アクションは、Amazon リソースネーム (ARN) により、ポリシーステートメントで識別する、特定の Amazon Pinpoint リソースで実行されます。例えば、`mobiletargeting:GetSegments` アクションが、ARN `arn:aws:mobiletargeting:`*region*`:`*accountId*`:apps/`*projectId* を使用して識別する特定のアプリケーションで実行されます。

このトピックでは、 AWS アカウントの IAM ポリシーに追加できる Amazon Pinpoint アクションを示します。ポリシーでアクションを使用して Amazon Pinpoint リソースへのアクセスを管理する方法を示す例については、Amazon Pinpoint のアイデンティティベースポリシーの例 を参照してください。

トピック

- Amazon Pinpoint API のアクション
- Amazon Pinpoint SMS および音声バージョン 1 API アクション

Amazon Pinpoint API のアクション

このセクションでは、Amazon Pinpoint のプライマリ API である Amazon Pinpoint API から使用できる機能のアクションを示します。この API の詳細については、「Amazon Pinpoint API リファレンス」を参照してください。

カテゴリ:

- 分析とメトリクス
- キャンペーン
- チャネル
- エンドポイント
- イベントストリーム
- イベント
- エクスポートジョブ
- インポートジョブ
- ジャーニー

- メッセージテンプレート
- メッセージ
- ワンタイムパスワード
- 電話番号検証
- プロジェクト
- 推奨モデル
- セグメント
- タグ
- [ユーザー]

分析とメトリクス

次のアクセス許可は、Amazon Pinpoint コンソールでの分析データの表示に関連しています。また、プロジェクト、キャンペーン、およびジャーニーに適用される、重要業績評価指標 (KPI) とも呼ばれる標準メトリクスの集計データの取得 (クエリ) にも関連しています。

`mobiletargeting:GetReports`

Amazon Pinpoint コンソールでデータ分析を表示します。このアクセス許可は、Amazon Pinpoint コンソールを使用してカスタム属性を含むセグメントを作成するためにも必要です。また、Amazon Pinpoint コンソールでセグメントのサイズの推定値を取得するためにも必要です。

- URI - 該当なし
- メソッド - 該当なし
- リソース ARN – `arn:aws:mobiletargeting:`*`region`*`:`*`accountId`*`:*`

`mobiletargeting:GetApplicationDateRangeKpi`

標準アプリケーションメトリクスの集計データを取得 (クエリ) します。これは、プロジェクトに関連付けられているすべてのキャンペーンまたはトランザクションメッセージに適用されるメトリクスです。

- URI - `/apps/`*`projectId`*`/kpis/daterange/`*`kpi-name`*
- メソッド - GET
- リソース ARN – `arn:aws:mobiletargeting:`*`region`*`:`*`accountId`*`:apps/`*`projectId`*`/kpis/daterange/`*`kpi-name`*

mobiletargeting:GetCampaignDateRangeKpi

標準キャンペーンメトリクスの集計データを取得 (クエリ) します。これは、個々のキャンペーンに適用されるメトリクスです。

- URI - /apps/*projectId*/campaigns/*campaignId*/kpis/daterange/*kpi-name*
- メソッド - GET
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/campaigns/*campaignId*/kpis/daterange/*kpi-name*

mobiletargeting:GetJourneyDateRangeKpi

標準的なジャーニーエンゲージメントメトリクスの集計データを取得 (クエリ) します。これは、個々のジャーニーに適用されるエンゲージメントメトリクスです。例えば、ジャーニー内のすべてのアクティビティにおいて、参加者が開封したメッセージの数などです。

- URI - /apps/*projectId*/journeys/*journeyId*/kpis/daterange/*kpi-name*
- メソッド - GET
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/journeys/*journeyId*/kpis/daterange/*kpi-name*

mobiletargeting:GetJourneyExecutionMetrics

個々のジャーニーに適用される標準的な実行メトリクス (ジャーニーのすべてのアクティビティをアクティブに進めている参加者の数など) の集計データを取得 (クエリ) します。

- URI - /apps/*projectId*/journeys/*journeyId*/execution-metrics
- メソッド - GET
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/journeys/*journeyId*/execution-metrics

mobiletargeting:GetJourneyExecutionActivityMetrics

ジャーニー内の個々のアクティビティ (アクティビティを開始または完了した参加者の数など) に適用される標準実行メトリクスの集計データを取得 (クエリー) します。

- URI - /apps/*projectId*/journeys/*journeyId*/activities/*journey-activity-id*/execution-metrics
- メソッド - GET
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/journeys/*journeyId*/activities/*journey-activity-id*/execution-metrics

キャンペーン

以下のアクセス権限は、Amazon Pinpoint アカウント内のキャンペーンの管理に関連しています。

mobiletargeting:CreateCampaign

プロジェクトのキャンペーンを作成します。

- URI - /apps/*projectId*/campaigns
- メソッド - POST
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/campaigns

mobiletargeting:DeleteCampaign

特定のキャンペーンを削除します。

- URI - /apps/*projectId*/campaigns/*campaignId*
- メソッド - DELETE
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/campaigns/*campaignId*

mobiletargeting:GetCampaign

特定のキャンペーンに関する情報を取得します。

- URI - /apps/*projectId*/campaigns/*campaignId*
- メソッド - GET
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/campaigns/*campaignId*

mobiletargeting:GetCampaignActivities

キャンペーンによって実行されたアクティビティに関する情報を取得します。

- URI - /apps/*projectId*/campaigns/*campaignId*/activities
- メソッド - GET
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/campaigns/*campaignId*

mobiletargeting:GetCampaigns

プロジェクトのすべてのキャンペーンに関する情報を取得します。

- URI - /apps/*projectId*/campaigns

- メソッド - GET
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*

mobiletargeting:GetCampaignVersion

特定のキャンペーンのバージョンに関する情報を取得します。

- URI - /apps/*projectId*/campaigns/*campaignId*/versions/*versionId*
- メソッド - GET
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/campaigns/*campaignId*

mobiletargeting:GetCampaignVersions

現在および以前のキャンペーンのバージョンに関する情報を取得します。

- URI - /apps/*projectId*/campaigns/*campaignId*/versions
- メソッド - GET
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/campaigns/*campaignId*

mobiletargeting:UpdateCampaign

特定のキャンペーンを更新します。

- URI - /apps/*projectId*/campaigns/*campaignId*
- メソッド – PUT
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/campaigns/*campaignId*

チャネル

以下のアクセス権限は、Amazon Pinpoint アカウント内のチャンネルの管理に関連しています。Amazon Pinpoint で、チャンネルは、顧客に連絡するための方法を指します。例えば、E メール、SMS メッセージ、プッシュ通知の送信などです。

mobiletargeting:DeleteAdmChannel

プロジェクトの Amazon Device Messaging (ADM) チャンネルを無効にします。

- URI - /apps/*projectId*/channels/adm
- メソッド - DELETE

- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/channels/adm

mobiletargeting:GetAdmChannel

プロジェクトの ADM チャンネルに関する情報を取得します。

- URI - /apps/*projectId*/channels/adm
- メソッド - GET
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/channels/adm

mobiletargeting:UpdateAdmChannel

プロジェクトの ADM チャンネルを有効化または更新します。

- URI - /apps/*projectId*/channels/adm
- メソッド – PUT
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/channels/adm

mobiletargeting:DeleteApnsChannel

プロジェクトの Apple Push Notification Service (APNS) チャンネルを無効にします。

- URI - /apps/*projectId*/channels/apns
- メソッド - DELETE
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/channels/apns

mobiletargeting:GetApnsChannel

プロジェクトの APN チャンネルに関する情報を取得します。

- URI - /apps/*projectId*/channels/apns
- メソッド - GET
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/channels/apns

mobiletargeting:UpdateApnsChannel

プロジェクトの APNs チャンネルを有効化または更新します。

- URI - /apps/*projectId*/channels/apns

- メソッド – PUT
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/channels/apns

mobiletargeting:DeleteApnsSandboxChannel

プロジェクトの APN サンドボックスチャンネルを無効にします。

- URI - /apps/*projectId*/channels/apns_sandbox
- メソッド - DELETE
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/channels/apns_sandbox

mobiletargeting:GetApnsSandboxChannel

プロジェクトの APN サンドボックスチャンネルに関する情報を取得します。

- URI - /apps/*projectId*/channels/apns_sandbox
- メソッド - GET
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/channels/apns_sandbox

mobiletargeting:UpdateApnsSandboxChannel

プロジェクトの APNs サンドボックスチャンネルを有効化または更新します。

- URI - /apps/*projectId*/channels/apns_sandbox
- メソッド – PUT
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/channels/apns_sandbox

mobiletargeting:DeleteApnsVoipChannel

プロジェクトの APN VoIP チャンネルを無効にします。

- URI - /apps/*projectId*/channels/apns_voip
- メソッド - DELETE
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/channels/apns_voip

mobiletargeting:GetApnsVoipChannel

プロジェクトの APN VoIP チャンネルに関する情報を取得します。

- URI - /apps/*projectId*/channels/apns_voip
- メソッド - GET
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/channels/apns_voip

mobiletargeting:UpdateApnsVoipChannel

プロジェクトの APNs VoIP チャンネルを有効化または更新します。

- URI - /apps/*projectId*/channels/apns_voip
- メソッド – PUT
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/channels/apns_voip

mobiletargeting:DeleteApnsVoipSandboxChannel

プロジェクトの APN VoIP サンドボックスチャンネルを無効にします。

- URI - /apps/*projectId*/channels/apns_voip_sandbox
- メソッド - DELETE
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/channels/apns_voip_sandbox

mobiletargeting:GetApnsVoipSandboxChannel

プロジェクトの APN VoIP サンドボックスチャンネルに関する情報を取得します。

- URI - /apps/*projectId*/channels/apns_voip_sandbox
- メソッド - GET
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/channels/apns_voip_sandbox

mobiletargeting:UpdateApnsVoipSandboxChannel

プロジェクトの APNs VoIP サンドボックスチャンネルを有効化または更新します。

- URI - /apps/*projectId*/channels/apns_voip_sandbox
- メソッド – PUT
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/channels/apns_voip_sandbox

mobiletargeting:DeleteBaiduChannel

プロジェクトの Baidu Cloud Push チャンネルを無効にします。

- URI - /apps/*projectId*/channels/baidu
- メソッド - DELETE
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/channels/baidu

mobiletargeting:GetBaiduChannel

プロジェクトの Baidu Cloud Push チャンネルに関する情報を取得します。

- URI - /apps/*projectId*/channels/baidu
- メソッド - GET
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/channels/baidu

mobiletargeting:UpdateBaiduChannel

プロジェクトの Baidu Cloud Push チャンネルを有効化または更新します。

- URI - /apps/*projectId*/channels/baidu
- メソッド – PUT
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/channels/baidu

mobiletargeting:DeleteEmailChannel

プロジェクトの E メールチャンネルを無効にします。

- URI - /apps/*projectId*/channels/email
- メソッド - DELETE
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/channels/email

mobiletargeting:GetEmailChannel

プロジェクトの E メールチャンネルに関する情報を取得します。

- URI - /apps/*projectId*/channels/email
- メソッド - GET
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/channels/email

mobiletargeting:UpdateEmailChannel

プロジェクトの E メールチャンネルを有効化または更新します。

- URI - /apps/*projectId*/channels/email
- メソッド – PUT
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/channels/email

mobiletargeting:DeleteGcmChannel

プロジェクトの Firebase Cloud Messaging (FCM) チャンネルを無効にします。このチャンネルでは、Amazon Pinpoint は Google クラウドメッセージング (GCM) サービスに代わる FCM サービスを通じて Android アプリケーションにプッシュ通知を送信することができます。

- URI - /apps/*projectId*/channels/gcm
- メソッド - DELETE
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/channels/gcm

mobiletargeting:GetGcmChannel

プロジェクトの FCM チャンネルに関する情報を取得します。このチャンネルでは、Amazon Pinpoint は Google クラウドメッセージング (GCM) サービスに代わる FCM サービスを通じて Android アプリケーションにプッシュ通知を送信することができます。

- URI - /apps/*projectId*/channels/gcm
- メソッド - GET
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/channels/gcm

mobiletargeting:UpdateGcmChannel

プロジェクトの FCM チャンネルを有効化または更新します。このチャンネルでは、Amazon Pinpoint は Google クラウドメッセージング (GCM) サービスに代わる FCM サービスを通じて Android アプリケーションにプッシュ通知を送信することができます。

- URI - /apps/*projectId*/channels/gcm
- メソッド – PUT
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/channels/gcm

mobiletargeting:DeleteSmsChannel

プロジェクトの SMS チャンネルを無効にします。

- URI - /apps/*projectId*/channels/sms
- メソッド - DELETE
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/channels/sms

mobiletargeting:GetSmsChannel

プロジェクトの SMS チャンネルに関する情報を取得します。

- URI - /apps/*projectId*/channels/sms
- メソッド - GET
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/channels/sms

mobiletargeting:UpdateSmsChannel

プロジェクトの SMS チャンネルを有効化または更新します。

- URI - /apps/*projectId*/channels/sms
- メソッド – PUT
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/channels/sms

mobiletargeting:GetChannels

アプリケーションの各チャンネルの履歴とステータスに関する情報を取得します。

- URI - /apps/*application-id*/channels
- メソッド - GET
- リソース ARN – arn:aws:mobiletargeting:region:*accountId*:apps/*projectId*/channels

mobiletargeting:DeleteVoiceChannel

アプリケーションの音声チャンネルを無効にし、チャンネルの既存の設定を削除します。

- URI - /apps/*application-id*/channels/voice
- メソッド - DELETE
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectid*/channels/voice

mobiletargeting:GetVoiceChannel

アプリケーションの音声チャンネルのステータスおよび設定に関する情報を取得します。

- URI - /apps/*application-id*/channels/voice
- メソッド - GET
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectid*/channels/voice

mobiletargeting:UpdateVoiceChannel

アプリケーションの音声チャンネルを有効にするか、アプリケーションの音声チャンネルのステータスおよび設定を更新します。

- URI - /apps/*application-id*/channels/voice
- メソッド – PUT
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectid*/channels/voice

エンドポイント

以下のアクセス権限は、Amazon Pinpoint アカウント内のエンドポイントの管理に関連しています。Amazon Pinpoint で、エンドポイントはメッセージの単一の送信先を指します。例えば、エンドポイントはお客様の E メールアドレス、電話番号、モバイルデバイストークンなどになります。

mobiletargeting:DeleteEndpoint

エンドポイントを削除します。

- URI - /apps/*projectId*/endpoints/*endpointId*
- メソッド - DELETE
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/endpoints/*endpointId*

mobiletargeting:GetEndpoint

特定のエンドポイントに関する情報を取得します。

- URI - /apps/*projectId*/endpoints/*endpointId*
- メソッド - GET
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/endpoints/*endpointId*

mobiletargeting:RemoveAttributes

アプリケーションに関連付けられているすべてのエンドポイントから、同じ属性タイプの 1 つまたは複数の属性を削除します。

- URI - apps/*application-id*/attributes/*attribute-type*
- メソッド – PUT
- リソース ARN – arn:aws:mobiletargeting:region:*accountId*:apps/*projectId*/attributes/*attribute-type*

mobiletargeting:UpdateEndpoint

エンドポイントを作成、または、エンドポイントの情報を更新します。

- URI - /apps/*projectId*/endpoints/*endpointId*
- メソッド – PUT
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/endpoints/*endpointId*

mobiletargeting:UpdateEndpointsBatch

バッチオペレーションとしてエンドポイントを作成または更新します。

- URI - /apps/*projectId*/endpoints
- メソッド – PUT
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*

イベントストリーム

以下のアクセス許可は、Amazon Pinpoint アカウント内のイベントストリームの管理に関連しています。

mobiletargeting:DeleteEventStream

プロジェクトのイベントストリームを削除します。

- URI - /apps/*projectId*/eventstream/
- メソッド - DELETE
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/eventstream

mobiletargeting:GetEventStream

プロジェクトのイベントストリームに関する情報を取得します。

- URI - /apps/*projectId*/eventstream/
- メソッド - GET
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/eventstream

mobiletargeting:PutEventStream

プロジェクトのイベントストリームを作成または更新します。

- URI - /apps/*projectId*/eventstream/
- メソッド - POST
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/eventstream

イベント

以下のアクセス権限は、Amazon Pinpoint アカウント内のイベントジョブの管理に関連しています。Amazon Pinpoint では、インポートジョブを作成し、Amazon S3 バケットに保存されたエンドポイントの定義に基づいてセグメントを作成します。

mobiletargeting:PutEvents

エンドポイントに記録する新しいイベントを作成したり、既存のイベントが関連付けられているエンドポイントデータを作成または更新します。

- URI - /apps/*application-id*/events
- メソッド - POST
- リソース ARN – arn:aws:mobiletargeting:region:*accountId*:apps/*projectId*/events

エクスポートジョブ

以下のアクセス権限は、Amazon Pinpoint アカウント内のエクスポートジョブの管理に関連しています。Amazon Pinpoint では、エクスポートジョブを作成して、エンドポイントに関する情報を保存または分析のため Amazon S3 バケットに送信します。

mobiletargeting:CreateExportJob

エンドポイントの定義を Amazon S3 にエクスポートするためのエクスポートジョブを作成します。

- URI - /apps/*projectId*/jobs/export
- メソッド - POST
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/jobs/export

mobiletargeting:GetExportJob

プロジェクトの特定のエクスポートジョブに関する情報を取得します。

- URI - /apps/*projectId*/jobs/export/*jobId*
- メソッド - GET
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/jobs/export/*jobId*

mobiletargeting:GetExportJobs

プロジェクトのすべてのエクスポートジョブのリストを取得します。

- URI - /apps/*projectId*/jobs/export
- メソッド - GET
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/jobs/export

インポートジョブ

以下のアクセス権限は、Amazon Pinpoint アカウント内のインポートジョブの管理に関連しています。Amazon Pinpoint では、インポートジョブを作成し、Amazon S3 バケットに保存されたエンドポイントの定義に基づいてセグメントを作成します。

mobiletargeting:CreateImportJob

Amazon S3 からエンドポイント定義をインポートしてセグメントを作成します。

- URI - /apps/*projectId*/jobs/import
- メソッド - POST
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*

mobiletargeting:GetImportJob

プロジェクトの特定のインポートジョブに関する情報を取得します。

- URI - /apps/*projectId*/jobs/import/*jobId*
- メソッド - GET
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/jobs/import/*jobId*

mobiletargeting:GetImportJobs

プロジェクトのすべてのインポートジョブに関する情報を取得します。

- URI - /apps/*projectId*/jobs/import
- メソッド - GET
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*

ジャーニー

以下のアクセス許可は、Amazon Pinpoint アカウント内のセグメントの管理に関連しています。

mobiletargeting:CreateJourney

プロジェクトのジャーニーを作成します。

- URI - /apps/*projectId*/journeys
- メソッド - POST
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/journeys

mobiletargeting:GetJourney

特定のジャーニーに関する情報を取得します。

- URI - /apps/*projectId*/journeys/*journeyId*
- メソッド - GET
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/journeys/*journeyId*

mobiletargeting:ListJourneys

プロジェクトのすべてのジャーニーに関する情報を取得します。

- URI - /apps/*projectId*/journeys
- メソッド - GET
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/journeys

mobiletargeting:UpdateJourney

特定のジャーニーに合わせて設定やその他の設定を更新します。

- URI - /apps/*projectId*/journeys/*journeyId*
- メソッド – PUT
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/journeys/*journeyId*

mobiletargeting:UpdateJourneyState

アクティブなジャーニーをキャンセルします。

- URI - /apps/*projectId*/journeys/*journeyId*/state
- メソッド – PUT
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/journeys/*journeyId*/state

mobiletargeting:DeleteJourney

特定のジャーニーを削除します。

- URI - /apps/*projectId*/journeys/*journeyId*
- メソッド - DELETE
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/journeys/*journeyId*

メッセージテンプレート

以下のアクセス許可は、Amazon Pinpoint アカウントのメッセージテンプレートの作成と管理に関連しています。メッセージテンプレートは、Amazon Pinpoint プロジェクトのいずれかに送信するメッセージで定義、保存、再利用できるコンテンツと設定のセットです。

mobiletargeting:ListTemplates

Amazon Pinpoint アカウントに関連付けられているすべてのメッセージテンプレートに関する情報を取得します。

- URI - /templates
- メソッド - GET
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:templates

mobiletargeting:ListTemplateVersions

特定のメッセージテンプレートのすべてのバージョンに関する情報を取得します。

- URI - /templates/*template-name*/*template-type*/versions
- メソッド - GET
- リソース ARN - 該当なし

mobiletargeting:UpdateTemplateActiveVersion

メッセージテンプレートの特定のバージョンをテンプレートのアクティブバージョンとして指定します。

- URI - /templates/*template-name*/*template-type*/active-version
- メソッド - GET
- リソース ARN - 該当なし

mobiletargeting:GetEmailTemplate

E メールチャンネルを介して送信されるメッセージのメッセージテンプレートに関する情報を取得します。

- URI - /templates/*template-name*/email
- メソッド - GET
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:templates/*template-name*/EMAIL

mobiletargeting:CreateEmailTemplate

E メールチャンネルを介して送信されるメッセージのメッセージテンプレートを作成します。

- URI - /templates/*template-name*/email
- メソッド - POST

- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:templates/*template-name*/EMAIL

mobiletargeting:UpdateEmailTemplate

E メールチャンネルを介して送信されるメッセージの既存のメッセージテンプレートを更新します。

- URI - /templates/*template-name*/email
- メソッド – PUT
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:templates/*template-name*/EMAIL

mobiletargeting:DeleteEmailTemplate

E メールチャンネルを介して送信されたメッセージのメッセージテンプレートを削除します。

- URI - /templates/*template-name*/email
- メソッド - DELETE
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:templates/*template-name*/EMAIL

mobiletargeting:GetPushTemplate

プッシュ通知チャンネルを介して送信されるメッセージのメッセージテンプレートに関する情報を取得します。

- URI - /templates/*template-name*/push
- メソッド - GET
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:templates/*template-name*/PUSH

mobiletargeting:CreatePushTemplate

プッシュ通知チャンネルを介して送信されるメッセージのメッセージテンプレートを作成します。

- URI - /templates/*template-name*/push

- メソッド - POST
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:templates/*template-name*/PUSH

mobiletargeting:UpdatePushTemplate

プッシュ通知チャンネルを介して送信されるメッセージの既存のメッセージテンプレートを更新します。

- URI - /templates/*template-name*/push
- メソッド – PUT
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:templates/*template-name*/PUSH

mobiletargeting:DeletePushTemplate

プッシュ通知チャンネルを介して送信されたメッセージのメッセージテンプレートを削除します。

- URI - /templates/*template-name*/push
- メソッド - DELETE
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:templates/*template-name*/PUSH

mobiletargeting:GetSmsTemplate

SMS チャンネルを介して送信されるメッセージのメッセージテンプレートに関する情報を取得します。

- URI - /templates/*template-name*/sms
- メソッド - GET
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:templates/*template-name*/SMS

mobiletargeting:CreateSmsTemplate

SMS チャンネルを介して送信されるメッセージのメッセージテンプレートを作成します。

- URI - /templates/*template-name*/sms
- メソッド - POST
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:templates/*template-name*/SMS

mobiletargeting:UpdateSmsTemplate

SMS チャンネルを介して送信されるメッセージの既存のメッセージテンプレートを更新します。

- URI - /templates/*template-name*/sms
- メソッド – PUT
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:templates/*template-name*/SMS

mobiletargeting:DeleteSmsTemplate

SMS チャンネルを介して送信されたメッセージのメッセージテンプレートを削除します。

- URI - /templates/*template-name*/sms
- メソッド - DELETE
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:templates/*template-name*/SMS

mobiletargeting:GetVoiceTemplate

音声チャンネルを介して送信されるメッセージのメッセージテンプレートに関する情報を取得します。

- URI - /templates/*template-name*/voice
- メソッド - GET
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:templates/*template-name*/VOICE

mobiletargeting:CreateVoiceTemplate

音声チャンネルを介して送信されるメッセージのメッセージテンプレートを作成します。

- URI - /templates/*template-name*/voice
- メソッド - POST

- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:templates/*template-name*/VOICE

mobiletargeting:UpdateVoiceTemplate

音声チャンネルを介して送信されるメッセージの既存のメッセージテンプレートを更新します。

- URI - /templates/*template-name*/voice
- メソッド – PUT
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:templates/*template-name*/VOICE

mobiletargeting:DeleteVoiceTemplate

音声チャンネルを介して送信されたメッセージのメッセージテンプレートを削除します。

- URI - /templates/*template-name*/voice
- メソッド - DELETE
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:templates/*template-name*/VOICE

メッセージ

次のアクセス許可は、Amazon Pinpoint アカウントからのメッセージとプッシュ通知の送信に関連しています。SendMessages および SendUsersMessages オペレーションを使用して、最初にセグメントやキャンペーンを作成することなく、特定のエンドポイントにメッセージを送信できます。

mobiletargeting:SendMessages

特定のエンドポイントにメッセージまたはプッシュ通知を送信します。

- URI - /apps/*projectId*/messages
- メソッド - POST
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/messages

mobiletargeting:SendUsersMessages

特定のユーザー ID に関連付けられているすべてのエンドポイントに、メッセージまたはプッシュ通知を送信します。

- URI - /apps/*projectId*/users-messages

- メソッド - POST
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/messages

ワンタイムパスワード

以下のアクセス許可は、Amazon Pinpoint でのワンタイムパスワード (OTP) の送信と検証に関連しています。

mobiletargeting:SendOTPMessage

ワンタイムパスワードを含むテキストメッセージを送信します。

- URI - /apps/*projectId*/otp
- メソッド - POST
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/otp

mobiletargeting:VerifyOTPMessage

SendOTPMessage オペレーションを使用して、生成されたワンタイムパスワード (OTP) の有効性を確認します。

- URI - /apps/*projectId*/verify-otp
- メソッド - POST
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/verify-otp

電話番号検証

以下のアクセス権限は、Amazon Pinpoint での電話番号検証サービスの使用に関連しています。

mobiletargeting:PhoneNumberValidate

電話番号に関する情報を取得します。

- URI - /phone/number/validate
- メソッド - POST
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:phone/number/validate

プロジェクト

以下のアクセス権限は、Amazon Pinpoint アカウント内のプロジェクトの管理に関連しています。当初、プロジェクトはアプリケーションと呼ばれていました。これらのオペレーションでは、Amazon Pinpoint アプリケーションは Amazon Pinpoint プロジェクトと同じです。

mobiletargeting:CreateApp

Amazon Pinpoint プロジェクトを作成します。

- URI - /apps
- メソッド - POST
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps

mobiletargeting:DeleteApp

Amazon Pinpoint プロジェクトを削除します。

- URI - /apps/*projectId*
- メソッド - DELETE
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*

mobiletargeting:GetApp

Amazon Pinpoint プロジェクトに関する情報を取得します。

- URI - /apps/*projectId*
- メソッド - GET
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*

mobiletargeting:GetApps

Amazon Pinpoint 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に関する情報を取得します。

- URI - /apps
- メソッド - GET
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps

mobiletargeting:GetApplicationSettings

Amazon Pinpoint プロジェクトのデフォルト設定を取得します。

- URI - /apps/*projectId*/settings

- メソッド - GET
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*

mobiletargeting:UpdateApplicationSettings

Amazon Pinpoint プロジェクトのデフォルト設定を更新します。

- URI - /apps/*projectId*/settings
- メソッド – PUT
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*

推奨モデル

以下のアクセス許可は、推奨モデルから推奨データを取得および処理するための Amazon Pinpoint 設定の管理に関連しています。推奨モデルは、データ内のパターンを見つけることによって、パーソナライズされた推奨事項を予測し、生成する機械学習モデルの一種です。

mobiletargeting:CreateRecommenderConfiguration

推奨モデル用の Amazon Pinpoint 設定を作成します。

- URI - /recommenders
- メソッド - POST
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:recommenders

mobiletargeting:GetRecommenderConfigurations

Amazon Pinpoint アカウントに関連付けられているすべての推奨モデルの設定に関する情報を取得します。

- URI - /recommenders
- メソッド - GET
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:recommenders

mobiletargeting:GetRecommenderConfiguration

推奨モデルの個別の Amazon Pinpoint 設定に関する情報を取得します。

- URI - /recommenders/*recommenderId*
- メソッド - GET
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:recommenders/*recommenderId*

mobiletargeting:UpdateRecommenderConfiguration

推奨モデルの Amazon Pinpoint 設定を更新します。

- URI - /recommenders/*recommenderId*
- メソッド – PUT
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:recommenders/*recommenderId*

mobiletargeting:DeleteRecommenderConfiguration

推奨モデルの Amazon Pinpoint 設定を削除します。

- URI - /recommenders/*recommenderId*
- メソッド - DELETE
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:recommenders/*recommenderId*

セグメント

以下のアクセス権限は、Amazon Pinpoint アカウント内のセグメントの管理に関連しています。Amazon Pinpoint で、セグメントはお客様が定義した特定の属性を共有する、キャンペーンの受信者のグループを指します。

mobiletargeting:CreateSegment

セグメントを作成します。ユーザーが Amazon Pinpoint の外部からエンドポイントデータをインポートしてセグメントを作成することを許可するには、mobiletargeting:CreateImportJob アクションを許可します。

- URI - /apps/*projectId*/segments
- メソッド - POST
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*

mobiletargeting:DeleteSegment

セグメントを削除します。

- URI - /apps/*projectId*/segments/*segmentId*
- メソッド - DELETE
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/segments/*segmentId*

mobiletargeting:GetSegment

特定のセグメントに関する情報を取得します。

- URI - /apps/*projectId*/segments/*segmentId*
- メソッド - GET
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/segments/*segmentId*

mobiletargeting:GetSegmentExportJobs

セグメントのエンドポイント定義をエクスポートするジョブに関する情報を取得します。

- URI - /apps/*projectId*/segments/*segmentId*/jobs/export
- メソッド - GET
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/segments/*segmentId*/jobs/export

mobiletargeting:GetSegments

プロジェクトのすべてのセグメントに関する情報を取得します。

- URI - /apps/*projectId*/segments
- メソッド - GET
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*

mobiletargeting:GetSegmentImportJobs

Amazon S3 からエンドポイント定義をインポートしてセグメントを作成するジョブに関する情報を取得します。

- URI - /apps/*projectId*/segments/*segmentId*/jobs/import
- メソッド - GET
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/segments/*segmentId*

mobiletargeting:GetSegmentVersion

特定のセグメントバージョンに関する情報を取得します。

- URI - /apps/*projectId*/segments/*segmentId*/versions/*versionId*
- メソッド - GET
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/segments/*segmentId*

mobiletargeting:GetSegmentVersions

現在および以前のセグメントのバージョンに関する情報を取得します。

- URI - /apps/*projectId*/segments/*segmentId*/versions
- メソッド - GET
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/segments/*segmentId*

mobiletargeting:UpdateSegment

特定のセグメントを更新します。

- URI - /apps/*projectId*/segments/*segmentId*
- メソッド – PUT
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/segments/*segmentId*

タグ

以下のアクセス許可は、Amazon Pinpoint リソースのタグの表示と管理に関連しています。

mobiletargeting:ListTagsForResource

プロジェクト、キャンペーン、メッセージテンプレート、またはセグメントに関連付けられているタグに関する情報を取得します。

- URI - /tags/*resource-arn*
- メソッド - GET
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:*

mobiletargeting:TagResource

プロジェクト、キャンペーン、メッセージテンプレート、またはセグメントに 1 つ以上のタグを追加します。

- URI - /tags/*resource-arn*
- メソッド - POST
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:*

mobiletargeting:UntagResource

プロジェクト、キャンペーン、メッセージテンプレート、セグメントから 1 つ以上のタグを削除します。

- URI - /tags/*resource-arn*
- メソッド - DELETE
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:*

[ユーザー]

以下のアクセス許可は、ユーザーの管理に関連しています。Amazon Pinpoint で、ユーザーはお客様からメッセージを受け取る人物を指します。1 人のユーザーが複数のエンドポイントに関連付けられる場合があります。

mobiletargeting:DeleteUserEndpoints

ユーザー ID に関連付けられているすべてのエンドポイントを削除します。

- URI - /apps/*projectId*/users/*userId*
- メソッド - DELETE
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/users/*userId*

mobiletargeting:GetUserEndpoints

ユーザー ID に関連付けられているすべてのエンドポイントに関する情報を取得します。

- URI - /apps/*projectId*/users/*userId*
- メソッド - GET
- リソース ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/users/*userId*

Amazon Pinpoint SMS および音声バージョン 1 API アクション

このセクションでは、Amazon Pinpoint SMS および音声 API から使用できる機能のアクションを示します。これは、Amazon Pinpoint で SMS チャンネルと音声チャンネルを使用および管理するための高度なオプションを提供する補足 API です。この API の詳細については、「Amazon Pinpoint SMS および音声 API リファレンス」を参照してください。

sms-voice:CreateConfigurationSet

音声メッセージを送信するための設定セットを作成します。

- URI - /sms-voice/configuration-sets
- メソッド - POST
- リソース ARN - 使用できません。* を使用します。

sms-voice:DeleteConfigurationSet

音声メッセージを送信するための設定セットを削除します。

- URI – /sms-voice/configuration-sets/*ConfigurationSetName*
- メソッド - DELETE
- リソース ARN - 使用できません。* を使用します。

sms-voice:GetConfigurationSetEventDestinations

設定セットとそれに含まれるイベント送信先に関する情報を取得します。

- URI – /sms-voice/configuration-sets/*ConfigurationSetName*/event-destinations
- メソッド - GET
- リソース ARN - 使用できません。* を使用します。

sms-voice:CreateConfigurationSetEventDestination

音声イベントのイベント送信先を作成します。

- URI – /sms-voice/configuration-sets/*ConfigurationSetName*/event-destinations
- メソッド - POST
- リソース ARN - 使用できません。* を使用します。

sms-voice:UpdateConfigurationSetEventDestination

音声イベントのイベント送信先を更新します。

- URI – /sms-voice/configuration-sets/*ConfigurationSetName*/event-destinations/*EventDestinationName*
- メソッド – PUT

- リソース ARN - 使用できません。* を使用します。

sms-voice:DeleteConfigurationSetEventDestination

音声イベントのイベント送信先を削除します。

- URI – /sms-voice/configuration-sets/*ConfigurationSetName*/event-destinations/*EventDestinationName*
- メソッド - DELETE
- リソース ARN - 使用できません。* を使用します。

sms-voice:SendVoiceMessage

音声メッセージを作成して送信します。

- URI - /sms-voice/voice/message
- メソッド - POST
- リソース ARN - 使用できません。* を使用します。

Amazon Pinpoint のアイデンティティベースポリシーの例

デフォルトでは、ユーザーとロールには Amazon Pinpoint リソースを作成または変更するためのアクセス許可がありません。また、、 AWS Management Console AWS CLI、または AWS API を使用してタスクを実行することはできません。IAM 管理者は、リソースで特定の API オペレーションを実行するために必要な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。続いて、管理者はそれらのアクセス許可が必要なユーザーまたはグループにそのポリシーをアタッチします。

これらの JSON ポリシードキュメント例を使用して IAM の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、『IAM ユーザーガイド』の「JSON タブでのポリシーの作成」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
- Amazon Pinpoint コンソールを使用する
- 例:単一の Amazon Pinpoint プロジェクトにアクセスする
- 例: タグに基づいて Amazon Pinpoint リソースを表示する

- 例: ユーザー自身のアクセス許可を表示することをユーザーに許可する
- 例: Amazon Pinpoint API アクションへのアクセスを許可する
- 例: Amazon Pinpoint SMS および音声 API アクションへのアクセスを許可する
- 例: 特定の IP アドレスに対して Amazon Pinpoint プロジェクトへのアクセスを制限する
- 例: タグに基づいて Amazon Pinpoint へのアクセスを制限する
- 例: Amazon Pinpoint から、Amazon SES で検証されたアイデンティティを使用して E メールを送信することを許可する

ポリシーのベストプラクティス

アイデンティティベースのポリシーは、ユーザーのアカウント内で誰かが Amazon Pinpoint 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、IAM ユーザーガイドの「AWS マネージドポリシー」または「AWS ジョブ機能の管理ポリシー」を参照してください。
- 最小特権を適用する – IAM ポリシーで権限を設定するときは、タスクの実行に必要な権限のみを付与します。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、最小特権権限とも呼ばれています。IAM を使用して権限を適用する方法の詳細については、『IAM ユーザーガイド』の「IAM でのポリシーと権限」を参照してください。
- IAM ポリシーで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、アクションや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 SSL を使用して送信するように指定できます。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、IAM ユーザーガイドの [IAM JSON policy elements: Condition] (IAM JSON ポリシー要素：条件) 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、安全で機能的な権限を確保する - IAM Access Analyzer は、新規および既存のポリシーを検証して、ポリシーが IAM ポリシー言語 (JSON) および IAM のベストプラクティスに準拠するようにします。IAM アクセスアナライザーは 100 を超えるポリシーチェックと実用的な推奨事項を提供し、安全で機能的なポリシーの作成をサ

ポートします。詳細については、「IAM ユーザーガイド」の「IAM Access Analyzer ポリシーの検証」を参照してください。

- 多要素認証 (MFA) を要求する – 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するために MFA を有効にします。API オペレーションが呼び出されるときに MFA を必須にするには、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 保護 API アクセスの設定」を参照してください。

IAM で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「IAM での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。

Amazon Pinpoint コンソールを使用する

Amazon Pinpoint コンソールにアクセスするには、許可の最小限のセットが必要です。これらのアクセス許可により、 AWS アカウントの Amazon Pinpoint リソースの詳細を一覧表示および表示できます。最小限必要なアクセス許可よりも厳しい制限を適用するアイデンティティベースのポリシーを作成すると、そのポリシーにより、コンソールはエンティティ (ユーザーまたはロール) について意図されたとおりには機能しなくなります。これらのエンティティで Amazon Pinpoint コンソールを使用可能にするには、エンティティにポリシーをアタッチします。詳細については、『IAM ユーザーガイド』の「ユーザーへの許可の追加」を参照してください。

次のポリシー例では、特定のリージョンの Amazon Pinpoint コンソールへの読み取り専用アクセス AWS を提供します。Amazon Pinpoint コンソールが依存するその他のサービス (Amazon Simple Email Service (Amazon SES)、IAM、Amazon Kinesis など) への読み取り専用アクセスも含まれます。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "UseConsole",
            "Effect": "Allow",
            "Action": [
                "mobiletargeting:Get*",
                "mobiletargeting:List*"
             ],
            "Resource": "arn:aws:mobiletargeting:region:accountId:*"
        },
        {
            "Effect": "Allow",
            "Action": [
```

```
                "firehose:ListDeliveryStreams",
                "iam:ListRoles",
                "kinesis:ListStreams",
                "s3:List*",
                "ses:Describe*",
                "ses:Get*",
                "ses:List*",
                "sns:ListTopics"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "accountId"
                }
            }
        }
    ]
}
```

前述のポリシーの例では、*region* を AWS リージョンの名前に置き換え、*accountId*を AWS アカウント ID に置き換えます。

AWS CLI または AWS API 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。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が許可されます。

例:単一の Amazon Pinpoint プロジェクトにアクセスする

特定のプロジェクトのみへのアクセスを付与する読み取り専用ポリシーを作成することもできます。次のポリシー例を使用すると、ユーザーはコンソールにサインインしてプロジェクトを一覧表示できます。また、ユーザーは Amazon SES、IAM、Amazon Kinesisなど、Amazon Pinpoint コンソールが依存する AWS の他のサービスに関連するリソースの情報も表示できます。ただし、ユーザーが表示できるのはポリシーに指定されているプロジェクトに関する追加情報のみです。このポリシーを変更して、追加のプロジェクトまたは AWS リージョンへのアクセスを許可できます。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewProject",
            "Effect": "Allow",
            "Action": "mobiletargeting:GetApps",
```

```
            "Resource": "arn:aws:mobiletargeting:region:accountId:*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "mobiletargeting:Get*",
                "mobiletargeting:List*"
            ],
            "Resource": [
                "arn:aws:mobiletargeting:region:accountId:apps/projectId",
                "arn:aws:mobiletargeting:region:accountId:apps/projectId/*",
                "arn:aws:mobiletargeting:region:accountId:reports"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "ses:Get*",
                "kinesis:ListStreams",
                "firehose:ListDeliveryStreams",
                "iam:ListRoles",
                "ses:List*",
                "sns:ListTopics",
                "ses:Describe*",
                "s3:List*"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "accountId"
                }
            }
        }
    ]
}
```

前の例では、*region* を AWS リージョンの名前に置き換え、*accountId* を AWS アカウント ID に置き換え、*projectId* をアクセスを提供する Amazon Pinpoint プロジェクトの ID に置き換えます。

同様に、プロジェクト 810c7aab86d42fb2b56c8c966example ID を持つプロジェクトなど、Amazon Pinpoint プロジェクトの 1 つへの書き込みアクセスが制限された AWS アカウント内のユーザーに許可するポリシーを作成できます。この場合、セグメントやキャンペーンなどのプロジェ

クトコンポーネントの表示、追加、更新はユーザーに許可しますが、コンポーネントの削除は許可しません。

mobiletargeting:Get アクションや mobiletargeting:List アクションに対するアクセス許可を付与するほかに、mobiletargeting:Create、mobiletargeting:Update、および mobiletargeting:Put の各アクションに対するアクセス許可を付与するポリシーを作成します。これらは、ほとんどのプロジェクトコンポーネントを作成および管理するために必要な追加のアクセス許可です。例:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "LimitedWriteProject",
            "Effect": "Allow",
            "Action": "mobiletargeting:GetApps",
            "Resource": "arn:aws:mobiletargeting:region:accountId:*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "mobiletargeting:Get*",
                "mobiletargeting:List*",
                "mobiletargeting:Create*",
                "mobiletargeting:Update*",
                "mobiletargeting:Put*"
            ],
            "Resource": [

 "arn:aws:mobiletargeting:region:accountId:apps/810c7aab86d42fb2b56c8c966example",

 "arn:aws:mobiletargeting:region:accountId:apps/810c7aab86d42fb2b56c8c966example/*",
                "arn:aws:mobiletargeting:region:accountId:reports"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "ses:Get*",
                "kinesis:ListStreams",
                "firehose:ListDeliveryStreams",
                "iam:ListRoles",
                "ses:List*",
```

```
                "sns:ListTopics",
                "ses:Describe*",
                "s3:List*"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "accountId"
                }
            }
        }
    ]
}
```

例: タグに基づいて Amazon Pinpoint リソースを表示する

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 Amazon Pinpoint リソースへのアクセスを制御できます。このポリシー例は、このようなポリシーを作成して Amazon Pinpoint リソースの表示を許可する方法を示しています。ただし、ユーザーにアクセス許可が付与されるのは、Owner リソースタグの値がユーザー名になっている場合のみです。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": "ListResources",
            "Effect": "Allow",
            "Action": [
                "mobiletargeting:Get*",
                "mobiletargeting:List*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "ViewResourceIfOwner",
            "Effect": "Allow",
            "Action": [
                "mobiletargeting:Get*",
                "mobiletargeting:List*"
            ],
            "Resource": "arn:aws:mobiletargeting:*:*:*",
            "Condition": {
```

```
                "StringEquals": {
                    "aws:ResourceTag/Owner": "userName"
                },
                "StringEquals": {
                    "aws:SourceAccount": "accountId"
                },
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:mobiletargeting:region:accountId:*"
                }
            }
        }
    ]
}
```

このタイプのポリシーをアカウントの ユーザーにアタッチできます。richard-roe という名前のユーザーが Amazon Pinpoint リソースを表示する場合、リソースには Owner=richard-roe または owner=richard-roe というタグが付いている必要があります。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、『IAM ユーザーガイド』の「IAM JSON ポリシー要素: 条件」を参照してください。

例: ユーザー自身のアクセス許可を表示することをユーザーに許可する

この例では、ユーザーアイデンティティにアタッチされたインラインおよびマネージドポリシーの表示を IAM ユーザーに許可するポリシーの作成方法を示します。このポリシーには、コンソールで、または AWS CLI または AWS API 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。

```
{
    "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": "*"
        }
    ]
}
```

例: Amazon Pinpoint API アクションへのアクセスを許可する

このセクションでは、Amazon Pinpoint のプライマリ API である Amazon Pinpoint API から利用できる機能へのアクセスを許可するポリシーの例を示します。この API の詳細については、「Amazon Pinpoint API リファレンス」を参照してください。

読み取り専用アクセス

次のポリシー例では、特定のリージョンの Amazon Pinpoint アカウント内のすべてのリソースへの読み取り専用アクセスを許可します AWS 。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewAllResources",
            "Effect": "Allow",
            "Action": [
                "mobiletargeting:Get*",
                "mobiletargeting:List*"
            ],
            "Resource": "arn:aws:mobiletargeting:region:accountId:*"
        }
    ]
```

```
}
```

前の例では、*region* を AWS リージョンの名前に置き換え、*accountId*を自分のアカウント ID に置き換えます AWS 。

管理者アクセス権

次のポリシー例では、Amazon Pinpoint アカウント内のすべての Amazon Pinpoint アクションとリソースへのフルアクセスを許可します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "FullAccess",
            "Effect": "Allow",
            "Action": [
                "mobiletargeting:*"
            ],
            "Resource": "arn:aws:mobiletargeting:region:accountId:*"
        }
    ]
}
```

前述の例で、*accountId* は自分の AWS アカウント ID に置き換えます。

例: Amazon Pinpoint SMS および音声 API アクションへのアクセスを許可する

このセクションでは、Amazon Pinpoint SMS および音声 API から利用できる機能へのアクセスを許可するポリシーの例を示します。これは、Amazon Pinpoint で SMS チャンネルと音声チャンネルを使用および管理するための高度なオプションを提供する補足 API です。この API の詳細については、「Amazon Pinpoint SMS および音声リファレンス」を参照してください。

読み取り専用アクセス

次のポリシー例では、 AWS アカウント内のすべての Amazon Pinpoint SMS および音声 API アクションとリソースへの読み取り専用アクセスを許可します。

```
{
    "Version": "2012-10-17",
    "Statement": [
```

```
        {
            "Sid": "SMSVoiceReadOnly",
            "Effect": "Allow",
            "Action": [
                "sms-voice:Get*",
                "sms-voice:List*"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "accountId"
                },
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:sms-voice:region:accountId:*"
                }
            }
        }
    ]
}
```

管理者アクセス権

次のポリシー例では、アカウント内のすべての Amazon Pinpoint SMS および音声 API アクションとリソースへのフルアクセスを許可します AWS 。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "SMSVoiceFullAccess",
            "Effect": "Allow",
            "Action": [
                "sms-voice:*",
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "accountId"
                },
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:sms-voice:region:accountId:*"
                }
            }
```

```
        }
    ]
}
```

例: 特定の IP アドレスに対して Amazon Pinpoint プロジェクトへのアクセスを制限する

次のポリシーの例では、指定したプロジェクト (*ProjectId*) で Amazon Pinpoint アクションを実行するアクセス許可をすべてのユーザーに付与します。ただし、リクエストは条件に指定されている IP アドレス範囲から行う必要があります。

このステートメントの条件では、許可されたインターネットプロトコルバージョン 4 (IPv4) のアドレスの 54.240.143.* 範囲を識別します。ただし、54.240.143.188 を除きます。Condition ブロックは、 IpAddress および NotIpAddress条件と、 AWS全体のaws:SourceIp条件キーである 条件キーを使用します。これらの条件キーの詳細については、『IAM ユーザーガイド』の「ポリシーでの条件の指定」を参照してください。aws:SourceIp IPv4 値は標準の CIDR 表記を使用します。詳細については、『IAM ユーザーガイド』の「IP アドレス条件演算子」を参照してください。

```
{
    "Version":"2012-10-17",
    "Id":"AMZPinpointPolicyId1",
    "Statement":[
        {
            "Sid":"IPAllow",
            "Effect":"Allow",
            "Principal":"*",
            "Action":"mobiletargeting:*",
            "Resource":[
                "arn:aws:mobiletargeting:region:accountId:apps/projectId",
                "arn:aws:mobiletargeting:region:accountId:apps/projectId/*"
            ],
            "Condition":{
                "IpAddress":{
                    "aws:SourceIp":"54.240.143.0/24"
                },
                "NotIpAddress":{
                    "aws:SourceIp":"54.240.143.188/32"
                }
            }
        }
```

```
    ]
}
```

例: タグに基づいて Amazon Pinpoint へのアクセスを制限する

次のポリシーの例では、指定したプロジェクト (*projectId*) で Amazon Pinpoint アクションを実行するアクセス許可を付与します。ただし、アクセス許可が付与されるのは、リクエスト元のユーザーの名前が Owner リソースタグの値として、条件に指定されている場合のみです。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ModifyResourceIfOwner",
            "Effect": "Allow",
            "Action": "mobiletargeting:*",
            "Resource": [
                "arn:aws:mobiletargeting:region:accountId:apps/projectId",
                "arn:aws:mobiletargeting:region:accountId:apps/projectId/*"
                ],
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/Owner": "userName"
                }
            }
        }
    ]
}
```

例: Amazon Pinpoint から、Amazon SES で検証されたアイデンティティを使用して E メールを送信することを許可する

Amazon Pinpoint コンソールで E メール ID (E メールアドレスやドメインなど) を検証すると、その ID は Amazon Pinpoint と Amazon SES の両方で使用できるように自動的に設定されます。ただし、Amazon SES で E メール ID を検証し、その ID を Amazon Pinpoint で使用する場合は、その ID にポリシーを適用する必要があります。

次のポリシー例では、Amazon SES で検証された E メール ID を使用して E メールを送信するためのアクセス許可を Amazon Pinpoint に付与します。

```
{
```

```
    "Version":"2008-10-17",
    "Statement":[
        {
            "Sid":"PinpointEmail",
            "Effect":"Allow",
            "Principal":{
                "Service":"pinpoint.amazonaws.com"
            },
            "Action":"ses:*",
            "Resource":"arn:aws:ses:region:accountId:identity/emailId",
            "Condition":{
                "StringEquals":{
                    "aws:SourceAccount":"accountId"
                },
                "StringLike":{
                    "aws:SourceArn":"arn:aws:mobiletargeting:region:accountId:apps/*"
                }
            }
        }
    ]
}
```

AWS GovCloud（米国西部）リージョンで Amazon Pinpoint を使用する場合は、代わりに次のポリシー例を使用します。

```
{
    "Version":"2008-10-17",
    "Statement":[
        {
            "Sid":"PinpointEmail",
            "Effect":"Allow",
            "Principal":{
                "Service":"pinpoint.amazonaws.com"
            },
            "Action":"ses:*",
            "Resource":"arn:aws-us-gov:ses:us-gov-west-1:accountId:identity/emailId",
            "Condition":{
                "StringEquals":{
                    "aws:SourceAccount":"accountId"
                },
                "StringLike":{
                    "aws:SourceArn":"arn:aws-us-gov:mobiletargeting:us-gov-
west-1:accountId:apps/*"
```

```
                }
            }
        }
    ]
}
```

Amazon Pinpoint の一般的なタスクの IAM ロール

IAM ロールは、 AWS アカウントで作成し、特定のアクセス許可を付与できる AWS Identity and Access Management (IAM) ID です。IAM ロールは、 AWS で ID が実行できることとできないことを決定するアクセス許可ポリシーを持つ ID です AWS。ただし、ロールは、1 人のユーザーに一意に関連付ける代わりに、それを必要とする任意のユーザーに割り当てることができます。

また、ロールには標準の長期的な認証情報も関連付けられません。代わりに、セッション用の一時的なセキュリティ認証情報を提供します。IAM ロールを使用して、通常は AWS リソースにアクセスできないユーザー、アプリケーション、アプリケーション、またはサービスへのアクセスを委任できます。

このような理由から、IAM ロールを使用して、Amazon Pinpoint をアカウントの特定の AWS のサービスやリソースと統合できます。例えば、Amazon Simple Storage Service (Amazon S3) バケットに保存したセグメント用のエンドポイント定義へのアクセスを Amazon Pinpoint に許可できます。または、イベントデータをアカウントの Amazon Kinesis ストリームにストリーミングすることを Amazon Pinpoint に許可できます。同様に、IAM ロールを使用して、キーをアプリケーションに埋め込むことなく (AWS ローテーションが難しく、ユーザーがエンドポイントを抽出できる可能性がある)、ウェブアプリケーションやモバイルアプリケーションが Amazon Pinpoint プロジェクトのエンドポイントを登録したり、使用状況データをレポートしたりできるようにすることもできます。

これらのシナリオでは、IAM ロールを使用して Amazon Pinpoint にアクセスを委任できます。このセクションでは、IAM ロールを使用して AWS の他のサービスを使用する一般的な Amazon Pinpoint タスクについて説明し、例を示します。IAM ロールを ウェブアプリケーションおよびモバイルアプリケーションで使用する具体的な方法については、『IAM ユーザーガイド』の「外部で認証されたユーザー (ID フェデレーション) へのアクセスの許可」を参照してください。

トピック

- エンドポイントまたはセグメントをインポートするための IAM ロール
- エンドポイントまたはセグメントをエクスポートするための IAM ロール
- Amazon Personalize から推奨事項を取得するための IAM ロール
- Kinesis にイベントをストリーミングするための IAM ロール

- Amazon SES で E メールを送信するための IAM ロール

エンドポイントまたはセグメントをインポートするための IAM ロール

Amazon Pinpoint では、 AWS アカウントの Amazon Simple Storage Service (Amazon S3) バケットからエンドポイント定義をインポートすることで、ユーザーセグメントを定義できます。インポートする前に、必要なアクセス権限を Amazon Pinpoint に与える必要があります。これを行うには、AWS Identity and Access Management (IAM) ロールを作成し、次のポリシーをロールにアタッチします。

- `AmazonS3ReadOnlyAccess` AWS マネージドポリシー。このポリシーは によって作成および管理され AWS、Amazon S3 バケットへの読み取り専用アクセスを許可します。
- Amazon Pinpoint がロールを引き受けるのを許可する信頼ポリシー。

ロールを作成すると、Amazon Pinpoint を使用して Amazon S3 バケットからセグメントをインポートできます。バケットの作成、エンドポイントファイルの作成、およびコンソールを使用したセグメントのインポートの詳細については、『Amazon Pinpoint ユーザーガイド』の「セグメントのインポート」を参照してください。を使用してプログラムでセグメントをインポートする方法の例については AWS SDK for Java、セグメントのインポートこのガイドの「」を参照してください。

IAM ロールの作成 (AWS CLI)

AWS Command Line Interface () を使用して IAM ロールを作成するには、次のステップを実行しますAWS CLI。をインストールしていない場合は AWS CLI、「 AWS Command Line Interface ユーザーガイド」の「 AWS CLIのインストール」を参照してください。

を使用して IAM ロールを作成するには AWS CLI

1. ロールの信頼ポリシーを含む JSON ファイルを作成し、ローカルにファイルを保存します。次の信頼ポリシーを使用できます。

```
{
     "Version": "2012-10-17",
     "Statement": [
          {
               "Action": "sts:AssumeRole",
               "Effect": "Allow",
               "Principal": {
```

```
                    "Service": "pinpoint.amazonaws.com"
                },
                "Condition": {
                    "StringEquals": {
                        "aws:SourceAccount": "accountId"
                    },
                "ArnLike": {
                    "arn:aws:mobiletargeting:region:accountId:apps/application-id"
                    }
                }
            }
        ]
}
```

前述の例で、以下を実行します。

- *region* を、Amazon Pinpoint を使用する AWS リージョンに置き換えます。
- *accountId* を AWS アカウントの一意の ID に置き換えます。
- *application-id* をプロジェクトの一意の ID に置き換えます。

2. コマンドラインで、create-role コマンドを使用して、ロールを作成し、信頼ポリシーをアタッチします。

```
aws iam create-role --role-name PinpointSegmentImport --assume-role-policy-document
 file://PinpointImportTrustPolicy.json
```

`file://` に続けて、信頼ポリシーを含む JSON ファイル へのパスを指定します。

このコマンドを実行すると、ターミナルに次のような出力が表示されます。

```
{
    "Role": {
        "AssumeRolePolicyDocument": {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Action": "sts:AssumeRole",
                    "Effect": "Allow",
                    "Principal": {
                        "Service": "pinpoint.amazonaws.com"
                    },
                    "Condition": {
```

```
                        "StringEquals": {
                            "aws:SourceAccount": "accountId"
                        },
                        "ArnLike": {
                            "aws:SourceArn":
 "arn:aws:mobiletargeting:region:accountId:apps/application-id"
                        }
                    }
                }
            ]
        },
        "RoleId": "AIDACKCEVSQ6C2EXAMPLE",
        "CreateDate": "2016-12-20T00:44:37.406Z",
        "RoleName": "PinpointSegmentImport",
        "Path": "/",
        "Arn": "arn:aws:iam::accountId:role/PinpointSegmentImport"
    }
}
```

3. attach-role-policy コマンドを使用して、AmazonS3ReadOnlyAccess AWS 管理ポリシーをロールにアタッチします。

```
aws iam attach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonS3ReadOnlyAccess --role-name PinpointSegmentImport
```

エンドポイントまたはセグメントをエクスポートするための IAM ロール

エンドポイントのリストを取得するには、エクスポートジョブを作成します。エクスポートジョブを作成したら、プロジェクト ID を指定します。また、セグメント ID をオプションで指定できます。Amazon Pinpoint は、プロジェクトまたはセグメントに関連付けられたエンドポイントのリストを Amazon Simple Storage Service (Amazon S3) バケットにエクスポートします。結果のファイルには、エンドポイントとその属性 (チャンネル、アドレス、オプトイン/オプトアウト状態、作成日、エンドポイント ID など) のリストが JSON 形式で含まれています。

エクスポートジョブを作成するには、Amazon Pinpoint による Amazon S3 バケットへの書き込みを許可する IAM ロールを設定する必要があります。ロールを設定するプロセスは、2 つのステップで構成されます。

1. エンティティ (この場合は Amazon Pinpoint) による特定の Amazon S3 バケットへの書き込みを許可する IAM ポリシーを作成します。

2. IAM ロールを作成して、それにポリシーをアタッチします。

このトピックには、これらのステップの両方を完了する手順が含まれています。これらの手順は、エクスポートされたセグメントの保存を目的として、 Amazon S3 バケットだけでなく、そのバケット内にすでにフォルダを作成していることを前提としています。バケットの作成の詳細については、『Amazon Simple Storage Service ユーザーガイド』の「バケットの作成」を参照してください。

また、次の手順でも AWS Command Line Interface (AWS CLI) がインストール済みおよび設定済みであることを前提とします。のセットアップの詳細については AWS CLI、「 AWS Command Line Interface ユーザーガイド」の「 AWS CLIのインストール」を参照してください。

ステップ 1: IAM ポリシーの作成

IAM ポリシーは、アイデンティティやリソースなどのエンティティのアクセス許可を定義します。Amazon Pinpoint エンドポイントをエクスポートするためのロールを作成するには、特定の Amazon S3 バケットの特定のフォルダに書き込むアクセス許可を付与するポリシーを作成する必要があります。次のポリシー例では、最小特権、つまり、1 つのタスクを実行するのに必要なアクセス許可のみを付与するセキュリティプラクティスを示します。

IAM ポリシーを作成するには

1. テキストエディタで新規ファイルを作成します。ファイルに次のコードを貼り付けます。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowUserToSeeBucketListInTheConsole",
            "Action": [
                "s3:ListAllMyBuckets",
                "s3:GetBucketLocation"
            ],
            "Effect": "Allow",
            "Resource": [ "arn:aws:s3:::*" ]
        },
        {
            "Sid": "AllowRootAndHomeListingOfBucket",
            "Action": [
                "s3:ListBucket"
            ],
            "Effect": "Allow",
```

```
            "Resource": [ "arn:aws:s3:::example-bucket" ],
            "Condition": {
                "StringEquals": {
                    "s3:delimiter": [ "/" ],
                    "s3:prefix": [
                        "",
                        "Exports/"
                    ]
                }
            }
        },
        {
            "Sid": "AllowListingOfUserFolder",
            "Action": [
                "s3:ListBucket"
            ],
            "Effect": "Allow",
            "Resource": [ "arn:aws:s3:::example-bucket" ],
            "Condition": {
                "StringLike": {
                    "s3:prefix": [
                        "Exports/*"
                    ]
                }
            }
        },
        {
            "Sid": "AllowAllS3ActionsInUserFolder",
            "Action": [ "s3:*" ],
            "Effect": "Allow",
            "Resource": [ "arn:aws:s3:::example-bucket/Exports/*" ]
        }
    ]
}
```

前述のコードで、*example-bucket* のすべてのインスタンスを、セグメント情報のエクスポート先フォルダを含む Amazon S3 バケットの名前に置き換えます。また、*Exports* のすべてのインスタンスをフォルダの名前に置き換えます。

終了したら、s3policy.json としてファイルを保存します。

2. を使用して AWS CLI、s3policy.jsonファイルがあるディレクトリに移動します。次のコマンドを入力してポリシーを作成します。

```
aws iam create-policy --policy-name s3ExportPolicy --policy-document
 file://s3policy.json
```

ポリシーが正常に作成されたら、次のような出力が表示されます。

```
{
    "Policy": {
        "CreateDate": "2018-04-11T18:44:34.805Z",
        "IsAttachable": true,
        "DefaultVersionId": "v1",
        "AttachmentCount": 0,
        "PolicyId": "ANPAJ2YJQRJCG3EXAMPLE",
        "UpdateDate": "2018-04-11T18:44:34.805Z",
        "Arn": "arn:aws:iam::123456789012:policy/s3ExportPolicy",
        "PolicyName": "s3ExportPolicy",
        "Path": "/"
    }
}
```

ポリシーの Amazon リソースネーム (ARN) (前述の例の `arn:aws:iam::123456789012:policy/s3ExportPolicy`) をコピーします。次のセクションでは、ロールを作成するときに、この ARN を指定する必要があります。

> **Note**
>
> `CreatePolicy` オペレーションを実行する権限がアカウントに付与されていないというメッセージが表示された場合は、新しい IAM ポリシーやロールの作成を可能にするポリシーをユーザー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「IAM ID アクセス許可の追加と削除」を参照してください。

ステップ 2: IAM ロールを作成する

IAM ポリシーが作成され、ロールを作成してそのポリシーをアタッチできるようになりました。IAM ロールにはそれぞれ、信頼ポリシーが含まれます。信頼ポリシーは、ロールを引き受けることができるエンティティを指定するルールのセットです。このセクションでは、Amazon Pinpoint がロールを引き受けることを許可する信頼ポリシーを作成します。次に、ロール自体を作成し、前のセクションで作成したポリシーをアタッチします。

IAM ロールを作成するには

1. テキストエディタで新規ファイルを作成します。ファイルに次のコードを貼り付けます。

```
{
    "Version":"2012-10-17",
    "Statement":[
        {
            "Effect":"Allow",
            "Principal":{
                "Service":"pinpoint.amazonaws.com"
            },
            "Action":"sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "accountId"
                },
                "ArnLike": {
                    "aws:SourceArn":
 "arn:aws:mobiletargeting:region:accountId:apps/applicationId"
                }
            }
        }
    ]
}
```

   trustpolicy.json という名前でファイルを保存します。

2. を使用して AWS CLI、trustpolicy.jsonファイルがあるディレクトリに移動します。次のコマンドを入力して、新しいロールを作成します。

```
aws iam create-role --role-name s3ExportRole --assume-role-policy-document
 file://trustpolicy.json
```

3. コマンドラインに次のコマンドを入力し、前のセクションで作成したポリシーを、先ほど作成したロールにアタッチします。

```
aws iam attach-role-policy --policy-arn arn:aws:iam::123456789012:policy/
s3ExportPolicy --role-name s3ExportRole
```

   前述のコマンドで、*arn:aws:iam::123456789012:policy/s3ExportPolicy* を、前のセクションで作成したポリシーの ARN に置き換えます。

Amazon Personalize から推奨事項を取得するための IAM ロール

Amazon Personalize キャンペーンとしてデプロイされた Amazon Personalize ソリューションから推奨データを取得するように Amazon Pinpoint を設定できます。このデータを使用して、各受信者の属性と動作に基づいてパーソナライズされた推奨事項をメッセージ受信者に送信できます。詳細については、『Amazon Pinpoint ユーザーガイド』の「機械学習モデル」を参照してください。

Amazon Personalize キャンペーンから推奨事項データを取得する前に、Amazon Pinpoint がキャンペーンからデータを取得できる AWS Identity and Access Management (IAM) ロールを作成する必要があります。Amazon Pinpoint は、コンソールを使用して Amazon Pinpoint で推奨モデルを設定した場合は、自動的にこのロールを作成できます。または、このロールを手動で作成することもできます。

ロールを手動で作成するには、IAM API を使用して次の手順を実行します。

1. エンティティ (この場合は Amazon Pinpoint) が Amazon Personalize キャンペーンから推奨データを取得できるようにする IAM ポリシーを作成します。
2. IAM ロールを作成して、それに IAM ポリシーをアタッチします。

このトピックでは、 AWS Command Line Interface () を使用してこれらのステップを完了する方法について説明しますAWS CLI。この手順では、Amazon Personalize ソリューションを作成済みであり、それを Amazon Personalize キャンペーンとしてデプロイ済みであるものとします。キャンペーンの作成およびデプロイの詳細については、『Amazon Personalize デベロッパーガイド』の「キャンペーンの作成」を参照してください。

このトピックでは、 AWS CLIがすでにインストールされ、設定されていることを前提としています。のセットアップの詳細については AWS CLI、「 AWS Command Line Interface ユーザーガイド」の「 AWS CLIのインストール」を参照してください。

ステップ 1: IAM ポリシーの作成

IAM ポリシーは、アイデンティティやリソースなどのエンティティのアクセス許可を定義します。Amazon Pinpoint が Amazon Personalize キャンペーンから推奨事項データを取得することを許可するロールを作成するには、まずロールに IAM ポリシーを作成する必要があります。このポリシーでは、Amazon Pinpoint に以下を許可する必要があります。

- キャンペーンによってデプロイされたソリューションの設定情報を取得します (`DescribeSolution`)。

- キャンペーンのステータスを確認します (DescribeCampaign)。
- キャンペーンから推奨データを取得します (GetRecommendations)。

次の手順では、ポリシーの例により、特定の Amazon Personalize キャンペーンによってデプロイされた特定の Amazon Personalize ソリューションに対してこのアクセスが許可されます。

IAM ポリシーを作成するには

1. テキストエディタで新規ファイルを作成します。ファイルに次のコードを貼り付けます。

```
{
    "Version":"2012-10-17",
    "Statement":[
        {
            "Sid":"RetrieveRecommendationsOneCampaign",
            "Effect":"Allow",
            "Action":[
                "personalize:DescribeSolution",
                "personalize:DescribeCampaign",
                "personalize:GetRecommendations"
            ],
            "Resource":[
                "arn:aws:personalize:region:accountId:solution/solutionId",
                "arn:aws:personalize:region:accountId:campaign/campaignId"
            ]
        }
    ]
}
```

   前述の例では、*######*のテキストを自分の情報に置き換えます。

   - *region* – Amazon Personalize ソリューションとキャンペーンをホストする AWS リージョンの名前。
   - *accountId* — お客様の AWS アカウント ID。
   - *solutionId* – キャンペーンによってデプロイされる Amazon Personalize ソリューションの一意のリソース ID。
   - *campaignId* – 推奨事項データを取得する Amazon Personalize キャンペーンの一意のリソース ID。
2. 終了したら、RetrieveRecommendationsPolicy.json としてファイルを保存します。

3. コマンドラインインターフェイスを使用して、RetrieveRecommendationsPolicy.json ファイルを保存したディレクトリに移動します。
4. 次のコマンドを入力して、ポリシーを作成し、RetrieveRecommendationsPolicy と名前を付けます。別の名前を使用するには、 を目的の名前*RetrieveRecommendationsPolicy*に変更します。

```
aws iam create-policy --policy-name RetrieveRecommendationsPolicy --policy-document
 file://RetrieveRecommendationsPolicy.json
```

> **Note**
>
> CreatePolicy オペレーションを実行する権限がアカウントに付与されていないというメッセージを受信した場合は、アカウントの新しい IAM ポリシーやロールの作成を可能にするポリシーをユーザー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「IAM ID アクセス許可の追加と削除」を参照してください。

5. ポリシーの Amazon リソースネーム (ARN) (前述の例の arn:aws:iam::123456789012:policy/RetrieveRecommendationsPolicy) をコピーします。次のセクションで、IAM ロールを作成するために、この ARN が必要になります。

ステップ 2: IAM ロールを作成する

IAM ポリシーを作成後、IAM ロールを作成してそのポリシーをアタッチできます。

IAM ロールにはそれぞれ、信頼ポリシーが含まれます。信頼ポリシーは、ロールを引き受けることができるエンティティを指定するルールのセットです。このセクションでは、Amazon Pinpoint がロールを引き受けることを許可する信頼ポリシーを作成します。次に、ロール自体を作成します。そのロールをグループにアタッチします。

IAM ロールを作成するには

1. テキストエディタで新規ファイルを作成します。ファイルに次のコードを貼り付けます。

```
{
    "Version":"2012-10-17",
    "Statement":[
        {
            "Effect": "Allow",
            "Principal": {
```

```
                "Service": "pinpoint.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "AWS:SourceAccount": "accountId"
                },
                "ArnLike": {
                    "AWS:SourceArn":
 "arn:aws:mobiletargeting:region:accountId:apps/*"
                }
            }
        }
    ]
}
```

2. RecommendationsTrustPolicy.json という名前でファイルを保存します。
3. コマンドラインインターフェイスを使用して、RecommendationsTrustPolicy.json ファイルを保存したディレクトリに移動します。
4. 次のコマンドを入力して、新しいロールを作成し、PinpointRoleforPersonalize と名前を付けます。別の名前を使用するには、 を目的の名前*PinpointRoleforPersonalize*に変更します。

```
aws iam create-role --role-name PinpointRoleforPersonalize --assume-role-policy-
document file://RecommendationsTrustPolicy.json
```

5. 次のコマンドを入力し、前のセクションで作成したポリシーを、先ほど作成したロールにアタッチします。

```
aws iam attach-role-policy --policy-arn arn:aws:iam::123456789012:policy/
RetrieveRecommendationsPolicy --role-name PinpointRoleforPersonalize
```

前述のコマンドで、*arn:aws:iam::123456789012:policy/RetrieveRecommendationsPolicy* を、前のセクションで作成したポリシーの ARN に置き換えます。また、ロールに別の名前を指定した場合は、 をステップ 4 で指定したロールの名前*PinpointRoleforPersonalize*に置き換えます。

Kinesis にイベントをストリーミングするための IAM ロール

Amazon Pinpoint は、アプリケーションから AWS アカウント内の Amazon Kinesis データストリームまたは Amazon Data Firehose 配信ストリームに、アプリケーション使用状況データまたはイベントデータを自動的に送信できます。Amazon Pinpoint でイベントデータのストリーミングを開始するには、事前に Amazon Pinpoint に必要なアクセス権限を割り当てなければなりません。

コンソールを使用してイベントストリーミングをセットアップする場合、Amazon Pinpoint は自動的に必要なアクセス権限を持つ AWS Identity and Access Management (IAM) ロールを作成します。詳細については、『Amazon Pinpoint ユーザーガイド』の「Amazon Pinpoint イベントを Amazon Kinesis へストリーミングする」を参照してください。

手動でロールを作成する場合は、次のポリシーをロールにアタッチします。

- イベントデータをストリームに送信することを Amazon Pinpoint に許可するアクセス許可ポリシー。
- Amazon Pinpoint がロールを引き受けるのを許可する信頼ポリシー。

ロールの作成が完了したら、Amazon Pinpoint を設定して自動的にストリームにイベントを送信できます。詳細については、このガイドの「Amazon Pinpoint のイベントを Kinesis にストリーミングする」を参照してください。

IAM ロールの作成 (AWS CLI)

AWS Command Line Interface (AWS CLI) を使って IAM ロールを手動で作成するには、次の手順を実行します。Amazon Pinpoint コンソールを使用してロールを作成する方法については、『Amazon Pinpoint ユーザーガイド』の「Amazon Pinpoint のイベントを Kinesis にストリーミングする」を参照してください。

をインストールしていない場合は AWS CLI、「 AWS Command Line Interface ユーザーガイド」の「 AWS CLIのインストール」を参照してください。また、Kinesis ストリームまたは Firehose ストリームを作成している必要があります。これらのリソースの作成の詳細については、「Amazon Kinesis Data Streams デベロッパーガイド」の「ストリームの作成と管理」または「Amazon Data Firehose デベロッパーガイド」の Amazon Kinesis「Amazon Data Firehose 配信ストリームの作成」を参照してください。

を使用して IAM ロールを作成するには AWS CLI

1. 新しいファイルを作成します。次のポリシーをドキュメントに貼り付け、次の変更を加えます。

- *region* は、Amazon Pinpoint を使用する AWS リージョンに置き換えます。
- *accountId* を AWS アカウントの一意の ID に置き換えます。
- *applicationId* をプロジェクトの一意の ID に置き換えます。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "pinpoint.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "accountId"
                },
                "ArnLike": {
                    "aws:SourceArn":
 "arn:aws:mobiletargeting:region:accountId:apps/applicationId"
                }
            }
        }
    ]
}
```

終了したら、PinpointEventStreamTrustPolicy.json としてファイルを保存します。

2. create-role コマンドを使用して、ロールを作成し、信頼ポリシーをアタッチします。

```
aws iam create-role --role-name PinpointEventStreamRole --assume-role-policy-
document file://PinpointEventStreamTrustPolicy.json
```

3. ロールのアクセス許可ポリシーを含む新しいファイルを作成します。

Kinesis ストリームにデータを送信するように Amazon Pinpoint を設定する場合は、次のポリシーをファイルに貼り付け、以下を置き換えます。

- *region* は、Amazon Pinpoint を使用する AWS リージョンに置き換えます。
- *accountId* を AWS アカウントの一意の ID に置き換えます。

- *streamName* を Kinesis ストリームの名前に置き換えます。

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Action": [
            "kinesis:PutRecords",
            "kinesis:DescribeStream"
        ],
        "Effect": "Allow",
        "Resource": [
            "arn:aws:kinesis:region:accountId:stream/streamName"
        ]
    }
}
```

または、Firehose ストリームにデータを送信するように Amazon Pinpoint を設定する場合は、次のポリシーを ファイルに貼り付け、以下を置き換えます。

- *region* は、Amazon Pinpoint を使用する AWS リージョンに置き換えます。
- *accountId* を AWS アカウントの一意の ID に置き換えます。
- を Firehose ストリームの名前*delivery-stream-name*に置き換えます。

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
         "firehose:PutRecordBatch",
         "firehose:DescribeDeliveryStream"
        ],
        "Resource": [
         "arn:aws:firehose:region:accountId:deliverystream/delivery-stream-name"
     ]
    }
}
```

終了したら、PinpointEventStreamPermissionsPolicy.json としてファイルを保存します。

4. put-role-policy コマンドを使用して、ロールにアクセス権限ポリシーをアタッチします。

```
aws iam put-role-policy --role-name PinpointEventStreamRole --policy-
name PinpointEventStreamPermissionsPolicy --policy-document file://
PinpointEventStreamPermissionsPolicy.json
```

Amazon SES で E メールを送信するための IAM ロール

Amazon Pinpoint は Amazon SES リソースを使用して、キャンペーンまたはジャーニーの E メールを送信します。Amazon Pinpoint が Amazon SES リソースを使用して E メールを送信する前に、Amazon Pinpoint に必要なアクセス許可を付与する必要があります。アカウントには、IAM ロールを更新または作成するための `iam:PutRolePolicy`および アクセス`iam:UpdateAssumeRolePolicy`許可が必要です。

Amazon Pinpoint コンソールは、必要なアクセス許可を持つ AWS Identity and Access Management (IAM) ロールを自動的に作成できます。詳細については、Amazon Pinpoint ユーザーガイド」の「E メールオーケストレーション送信ロールの作成」を参照してください。

手動でロールを作成する場合は、次のポリシーをロールにアタッチします。

- Amazon SES リソースへの Amazon Pinpoint アクセスを許可するアクセス許可ポリシー。
- Amazon Pinpoint がロールを引き受けるのを許可する信頼ポリシー。

ロールを作成したら、Amazon SES リソースを使用するように Amazon Pinpoint Amazon SES を設定できます。

IAM ポリシーシミュレーターを使用して IAM ポリシーをテストできます。詳細については、「IAM ユーザーガイド」の「IAM Policy Simulator を使用した IAM ポリシーのテストhttps://docs.aws.amazon.com/IAM/latest/UserGuide/」を参照してください。

IAM ロールの作成 (AWS Management Console)

キャンペーンまたは E メールを送信するジャーニーの IAM ロールを手動で作成するには、次のステップを実行します。

1. IAM ユーザーガイドの「JSON エディタを使用したポリシーの作成」の指示に従って、新しいアクセス許可ポリシーを作成します。

    - ステップ 5 では、IAM ロールに次のアクセス許可ポリシーを使用します。

- partition は、リソースが存在するパーティションに置き換えます。標準 の場合 AWS リージョン、パーティションは ですaws。他のパーティションにリソースがある場合、パーティションは aws-partitionname です。例えば、 AWS GovCloud （ 米国西部) のリソースのパーティションは ですaws-us-gov。
- *region* を、Amazon Pinpoint プロジェクトをホスト AWS リージョン する の名前に置き換えます。
- *accountId* を の一意の ID に置き換えます AWS アカウント。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PinpointUsesSESForEmailSends",
            "Effect": "Allow",
            "Action": [
                "ses:SendEmail",
                "ses:SendRawEmail"
            ],
            "Resource": [
                "arn:partition:ses:region:accountId:identity/*",
                "arn:partition:ses:region:accountId:configuration-set/*"
            ]
        }
    ]
}
```

2. IAM ユーザーガイドの「カスタム信頼ポリシーを使用してロールを作成する」の指示に従って、新しい信頼ポリシーを作成します。 https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles_create_for-custom.html https://docs.aws.amazon.com/IAM/latest/UserGuide/

   a. ステップ 4 では、次の信頼ポリシー を使用します。

   - *accountId* を の一意の ID に置き換えます AWS アカウント。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
```

```
            "Sid": "AllowPinpoint",
            "Effect": "Allow",
            "Principal": {
                "Service": "pinpoint.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "accountId"
                }
            }
        }
    ]
}
```

b. ステップ 11 で、前のステップで作成したアクセス許可ポリシーを追加します。

Amazon Pinpoint Identity and Access Management のトラブルシューティング

次の情報は、Amazon Pinpoint と IAM の利用に伴って発生する可能性がある一般的な問題の診断や修正をします。

トピック

- Amazon Pinpoint でアクションを実行する権限がありません。
- iam を実行する権限がありません。PassRole
- AWS アカウント外のユーザーに Amazon Pinpoint リソースへのアクセスを許可したい

Amazon Pinpoint でアクションを実行する権限がありません。

がアクションを実行する権限がないと AWS Management Console 通知した場合は、管理者に連絡してサポートを依頼する必要があります。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、mateojackson ユーザーがコンソールを使用してプロジェクトの詳細を表示する際に mobiletargeting:*GetApp* アクセス許可を持っていない場合に発生します。

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 mobiletargeting:GetApp on resource: my-example-project
```

この場合、Mateo は、mobiletargeting:*GetApp* アクションを使用して *my-example-project* 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 Amazon Pinpoint にロールを渡せるようにする必要があります。

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。

以下の例のエラーは、marymajor という IAM ユーザーがコンソールを使用して Amazon Pinpoint でアクションを実行しようする場合に発生します。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。Mary には、ロールをサービスに渡す権限がありません。

```
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。サインイン資格情報を提供した担当者が管理者です。

AWS アカウント外のユーザーに Amazon Pinpoint 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 (ACL) をサポートするサービスの場合、それらのポリシーを使用して、リソースへのアクセスを付与できます。

詳細については、以下を参照してください:

- Amazon Pinpoint がこれらの機能をサポートしているかどうかについては、「Amazon Pinpoint で IAM が機能する仕組み」を参照してください。

- 所有 AWS アカウント 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、IAM 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」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介し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「外部で認証されたユーザー (ID フェデレーション) へのアクセス権限」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の詳細については、『IAM ユーザーガイド』の「IAM ロールとリソースベースのポリシーとの相違点」を参照してください。

Amazon Pinpoint でのログ記録とモニタリング

ログ記録とモニタリングは、Amazon Pinpoint プロジェクトやその他の種類の Amazon Pinpoint リソースの信頼性、可用性、およびパフォーマンスを維持する上で重要な要素です。マルチポイント障害が発生した場合は、Amazon Pinpoint プロジェクトとリソースのすべての部分からモニタリングデータをログに記録して収集し、より簡単にデバッグする必要があります。 は、このデータのログ記録と収集、および潜在的なインシデントへの対応に役立つ複数のツール AWS を提供します。

AWS CloudTrail

Amazon Pinpoint は、ユーザー AWS CloudTrail、ロール、または別のサービスによって Amazon Pinpoint で実行されたアクションを記録する AWS サービスである と統合されます。これには、Amazon Pinpoint コンソールからのアクションと Amazon Pinpoint API オペレーションへのプログラムによる呼び出しが含まれます。によって収集された情報を使用して CloudTrail、Amazon Pinpoint に対して行われたリクエストを判断できます。リクエストごとに、リクエスト日時、リクエスト元の IP アドレス、作成者、その他の詳細を確認できます。詳細については、このガイドの「を使用した Amazon Pinpoint API コールのログ記録 AWS CloudTrail」を参照してください。

Amazon CloudWatch

Amazon を使用して、Amazon Pinpoint アカウントとプロジェクトに関連するいくつかの重要なメトリクスを CloudWatch 収集、表示、分析できます。を使用して CloudWatch 、メトリクスの値が特定の条件を満たし、定義したしきい値の範囲内または超えた場合に通知するアラームを作成することもできます。アラームを作成すると、 CloudWatch は指定した Amazon Simple Notification Service (Amazon SNS) トピックに通知を送信します。詳細については、Amazon

Pinpoint ユーザーガイド」の「Amazon Pinpoint による Amazon Pinpoint のモニタリング CloudWatchAmazon Pinpoint」を参照してください。

AWS Health ダッシュボード

AWS Health ダッシュボードを使用すると、Amazon Pinpoint 環境のステータスを確認およびモニタリングできます。Amazon Pinpoint サービス全体のステータスを確認するには、 AWS Service Health Dashboard を使用します。 AWS より具体的に環境に影響を与える可能性のあるイベントや問題に関する履歴データを確認、モニタリング、表示するには、 AWS Personal Health Dashboard を使用します。これらのダッシュボードの詳細については、『AWS Health ユーザーガイド』を参照してください。

AWS Trusted Advisor

AWS Trusted Advisor は AWS 環境を検査し、セキュリティギャップに対処し、システムの可用性とパフォーマンスを向上させ、コストを削減する機会に関する推奨事項を提供します。すべての AWS お客様は、コア Trusted Advisor チェックセットにアクセスできます。ビジネスまたはエンタープライズサポートプランをご利用のお客様は、追加の Trusted Advisor チェックにアクセスできます。

これらのチェックの多くは、 AWS アカウント全体の一部として Amazon Pinpoint リソースのセキュリティ体制を評価するのに役立ちます。例えば、 Trusted Advisor チェックのコアセットには、次のものが含まれます。

- サポートされている各 AWS リージョンの AWS アカウントのログ記録設定。
- Amazon Simple Storage Service (Amazon S3) バケットへのアクセス許可。このバケットには、Amazon Pinpoint にインポートしてセグメントを構築するためのファイルが含まれている場合があります。
- Amazon Pinpoint リソースへのアクセスを制御するための AWS Identity and Access Management ユーザー、グループ、ロールの使用。
- AWS 環境と Amazon Pinpoint リソースのセキュリティを侵害する可能性のある IAM 設定とポリシー設定。

詳細については、『AWS Support ユーザーガイド』のAWS Trusted Advisor を参照してください。

Amazon Pinpoint のコンプライアンス検証

Amazon Pinpoint のセキュリティとコンプライアンスは、複数の AWS コンプライアンスプログラムの一環として、サードパーティー監査機関によって評価されます。これには、 AWS System and

Organization Controls (SOC)、FedRAMP、HIPAA、セキュリティ管理コントロール用の ISO/IEC 27001:2013、クラウド固有のコントロール用の ISO/IEC 27017:2015、個人データ保護用の ISO/IEC 27018:2014、品質管理システム用の ISO/IEC 9001:2015 などが含まれます。

特定のコンプライアンスプログラムの対象となる AWS サービスのリストについては、「コンプライアンスAWS プログラムによる対象範囲内のサービス」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。

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 AWS Artifact。詳細については、AWS 「アーティファクト でのレポートのダウンロード」を参照してください。

Amazon Pinpoint を使用する際のお客様のコンプライアンス責任は、お客様のデータの機密性、貴社のコンプライアンス目的、適用される法律および規制によって決まります。 では、コンプライアンスに役立つ以下のリソース AWS を提供しています。

- セキュリティとコンプライアンスのクイックスタートガイド – これらのデプロイガイドでは、アーキテクチャ上の考慮事項について説明し、セキュリティとコンプライアンスに重点を置いたベースライン環境を にデプロイする手順について説明します AWS。
- HIPAA セキュリティとコンプライアンスのアーキテクチャに関するホワイトペーパー — このホワイトペーパーでは、企業が AWS を使用して HIPAA 準拠のアプリケーションを作成する方法について説明します。
- AWS コンプライアンスリソース – このワークブックとガイドのコレクションは、お客様の業界や地域に適用される場合があります。
- 「 デベロッパーガイド」のルールによるリソースの評価 — この AWS Config サービスは、リソース設定が社内プラクティス、業界ガイドライン、および規制にどの程度準拠しているかを評価します。 AWS Config
- AWS Security Hub – この AWS サービスは、 内のセキュリティ状態を包括的に把握 AWS し、セキュリティ業界標準とベストプラクティスへの準拠を確認するのに役立ちます。

Amazon Pinpoint は、お客様が適切な通信チャネルを使用する場合の AWS HIPAA 対応サービスです。Amazon Pinpoint を使用して HIPAA および関連法規が定義する保護された健康情報 (PHI) が含まれているワークロードを実行する場合は、E メールチャンネル、プッシュ通知チャンネル、または SMS チャンネルを使用して、PHI が含まれているメッセージを送信する必要があります。SMS チャネルを使用して PHI を含むメッセージを送信する場合は、PHI を含む、または含む可能性のあるメッセージを送信する明示的な目的で、 AWS アカウントに対してリクエストした専用のショートコードからメッセージを送信する必要があります。音声チャネルは AWS HIPAA に対応していません。PHI を含むメッセージを送信するために音声チャネルを使用しないでください。

Amazon Pinpoint の耐障害性

AWS 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ています。 AWS リージョンは、低レイテンシー、高スループット、および高度に冗長なネットワークで接続された、物理的に分離および分離された複数のアベイラビリティーゾーンを提供します。アベイラビリティーゾーンでは、ゾーン間で中断することなく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が高く、フォールトトレラントで、スケーラブルです。

詳細については、「Amazon Pinpoint Resilient Architecture Guide」のリファレンスアーキテクチャについての説明を参照してください。

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。

Amazon Pinpoint でのインフラストラクチャセキュリティ

マネージドサービスである Amazon Pinpoint 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、「セキュリティの柱 AWS Well-Architected Framework」の「Infrastructure Protection」を参照してください。

が AWS 公開した API コールを使用して、ネットワーク経由で Amazon Pinpoint 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)。TLS 1.2 は必須で TLS 1.3 がお勧めです。
- DHE (楕円ディフィー・ヘルマン鍵共有) や ECDHE (楕円曲線ディフィー・ヘルマン鍵共有) などの完全前方秘匿性 (PFS) による暗号スイート。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。

また、リクエストには、アクセスキー ID と、IAM プリンシパルに関連付けられている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、一時的な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。

これらの API コールは任意のネットワークロケーションから実行できますが、Amazon Pinpoint はリソースベースのアクセスポリシーをサポートしています。これらのポリシーには、発信元 IP アドレスに基づく制限を含めることができます。このタイプのポリシーの詳細については、「ポリシーを使用したアクセスの管理」を参照してください。

さらに、さまざまな AWS セキュリティ機能を設定および使用して、Amazon Pinpoint と統合するモバイルアプリケーションまたはウェブアプリケーションから Amazon Pinpoint リソースへのアクセスを制御できます。これには、エンドポイントの追加、エンドポイントデータの更新、イベントデータの送信、使用状況データのレポートなどのタスクの API コールに対する制限が含まれます。

これらの機能を使用するには、 AWS Mobile SDKs または AWS Amplify JavaScript ライブラリを使用して、モバイルアプリケーションとウェブアプリケーションを Amazon Pinpoint と統合することをお勧めします。Android または iOS アプリの場合は、 AWS Mobile SDK for iOSそれぞれ AWS Mobile SDK for Android または を使用することをお勧めします。 JavaScriptベースのモバイルアプリまたはウェブアプリの場合は、ウェブ用ライブラリまたは React Native 用 AWS Amplify JavaScript ライブラリを使用する AWS Amplify JavaScriptことをお勧めします。これらのリソースの詳細については、AWS 「モバイル SDKs」、「ウェブ 用 AWS Amplify ライブラリの開始方法」、「React native 用 AWS Amplify ライブラリの開始方法」を参照してください。

Amazon Pinpoint での設定と脆弱性の分析

マネージドサービスである Amazon Pinpoint は、ホワイトペーパー「Amazon Web Services: セキュリティプロセスの概要」に記載されている AWS グローバルネットワークセキュリティの手順で保護されています。 https://d0.awsstatic.com/whitepapers/Security/AWS_Security_Whitepaper.pdfつまり、 は基本的なセキュリティタスクと手順を AWS 管理および実行して、Amazon Pinpoint アカウントとリソースの基盤となるインフラストラクチャを強化、パッチ適用、更新、その他の方法で維持します。これらの手順は適切なサードパーティーによって確認され、認証されています。

詳細については、以下のリソースを参照してください。

- Amazon Pinpoint のコンプライアンス検証
- 責任共有モデル
- アマゾン ウェブ サービス: セキュリティプロセスの概要 (ホワイトペーパー)

Amazon Pinpoint のセキュリティベストプラクティス

AWS Identity and Access Management (IAM) アカウントを使用して、API オペレーション、特に Amazon Pinpoint リソースを作成、変更、削除するオペレーションへのアクセス Amazon Pinpoint を

制御します。 Amazon Pinpoint API の場合、このようなリソースにはプロジェクト、キャンペーン、ジャーニーが含まれます。 Amazon Pinpoint SMS および音声 API の場合、このようなリソースには電話番号、プール、設定セットが含まれます。

- 自分自身を含め、 Amazon Pinpoint リソースを管理するユーザーごとに個別のユーザーを作成します。Amazon Pinpoint リソースの管理に AWS ルート認証情報を使用しないでください。
- それぞれの職務の実行に最低限必要になる一連のアクセス許可を各ユーザーに付与します。
- IAM グループを使用して、複数のユーザーのアクセス許可を効果的に管理します。
- IAM 認証情報のローテーションを定期的に行います。

Amazon Pinpoint セキュリティの詳細については、Amazon Pinpoint のセキュリティ」を参照してください。IAM の詳細については、「AWS Identity and Access Management」を参照してください。IAM のベストプラクティスについては、「IAM のベストプラクティス」を参照してください。

Amazon Pinpoint のクォータ

以下のセクションでは、Amazon Pinpointのリソースと操作に適用されるクォータ (以前は制限と呼ばれていました) のリストと説明をします。クォータによっては、引き上げることができないものもあります。クォータの引き上げをリクエストできるかどうかを判断するには、各セクションの「Eligible for Increase」列または説明を参照してください。

トピック

プロジェクトクォータ

次の表に、Amazon Pinpoint に関連するクォータの一覧を示します。

リソース	デフォルトのクォータ	引き上げの対象かどうかの確認
プロジェクト	各 で AWS リージョン、最大 100 のプロジェクトを作成できます。	いいえ

API リクエストのクォータ

Amazon Pinpoint は、 AWS アカウントから Amazon Pinpoint API に対して実行できるリクエストのサイズと数を制限するクォータを実装します。

呼び出し (リクエストとレスポンス) ペイロードの最大サイズは 7 MB です。ただし、特定のタイプのリソースに対して特に指定されている場合を除きます。リソースのクォータが異なるかどうかを判断するには、このトピックの該当セクションでそのリソースのタイプを参照してください。

最大リクエスト数は、クォータタイプと API 操作によって異なります。Amazon Pinpoint は、API リクエストに対して 2 種類のクォータを実装します。

- Rate quotas – レート制限とも呼ばれるこのタイプのクォータは、特定のオペレーションに対して 1 秒あたりに実行できるリクエストの最大数を定義します。これは、アカウントごとの送信または受信されるリクエストのレートを制御します。
- Burst quotas – バースト制限またはバーストキャパシティとも呼ばれるこのタイプのクォータは、アカウントで同時に送信されるリクエストの最大数を定義します。

次の表に、Amazon Pinpoint API のレートとバーストクォータを示します。

操作	デフォルトのバースト/レートクォータ (1 秒あたりのリクエスト数)
CreateCampaign	25
CreateEmailTemplate	10
CreateInAppTemplate	10
CreateImportJob	300

操作	デフォルトのバースト/レートクォータ (1 秒あたりのリクエスト数)
CreatePushTemplate	10
CreateSegment	25
CreateSmsTemplate	10
CreateVoiceTemplate	10
DeleteCampaign	25
DeleteEndpoint	5
DeleteSegment	25
GetEndpoint	10
PhoneNumberValidate	20
PutEvents	15
SendMessages	4,000
SendUsersMessages	6,000
UpdateCampaign	25
UpdateEmailTemplate	10
UpdateEndpoint	10
UpdateEndpointsBatch	2
UpdateInAppTemplate	10
UpdatePushTemplate	10
UpdateSegment	25
UpdateSmsTemplate	10

操作	デフォルトのバースト/レートクォータ (1 秒あたりのリクエスト数)
UpdateVoiceTemplate	10
その他すべてのオペレーション	300

次の表に、CreateImportJob のファイルインポートクォータを示します。

操作	デフォルトのクォータ	引き上げの対象かどうかの確認
インポートファイルの最大数	インポートジョブごとに 10,000 ファイル	いいえ

これらのクォータのいずれかを超えると、Amazon Pinpoint によってリクエストがスロットリングされます。つまり、それ以外の有効なリクエストが拒否され、TooManyRequests エラーが返されます。スロットリングは、特定の AWS リージョンにおける特定のオペレーションに対してアカウントから行ったリクエストの総数に基づきます。さらに、スロットリングの決定は、オペレーションごとに個別に計算されます。例えば、Amazon Pinpoint オペレーションのリクエストを SendMessages がスロットルする場合、同時に行われた UpdateEndpoint オペレーションのリクエストは正常に完了できます。

キャンペーンクォータ

Amazon Pinpoint API のキャンペーンリソースには、以下のクォータが適用されます。

ごとに次のクォータが適用され AWS リージョン 、一部引き上げることができます。詳細については、「Service Quotas ユーザーガイド」の「クォータの増加をリクエスト」を参照してください。

リソース	デフォルトのクォータ	引き上げの対象かどうかの確認
アクティブなキャンペーン	アカウントあたり 200	いいえ

リソース	デフォルトのクォータ	引き上げの対象かどうかの確認
	Note アクティブなキャンペーンとは、完了も失敗もしていないキャンペーンのことです。アクティブなキャンペーンのステータスは SCHEDULED 、EXECUTING 、または PENDING_NEXT_RUN です。	
最大セグメントサイズ	インポートされたセグメントの場合: 1 キャンペーンにつき 100,000,000 ダイナミックセグメントの場合: 無制限	いいえ

リソース	デフォルトのクォータ	引き上げの対象かどうかの確認
イベントベースのキャンペーン	各プロジェクトには、イベントが発生したときに送信される最大 25 個のキャンペーンを含めることができます。 イベントベースのトリガーを使用するキャンペーンでは、動的セグメントを使用する必要があります。インポートされたセグメントは使用できません。 AWS Mobile SDK を使用してアプリケーションを Amazon Pinpoint と統合する場合、イベントベースのキャンペーンからのメッセージは、アプリケーションが AWS Mobile SDK for Android バージョン 2.7.2 以降、または AWS Mobile SDK for iOS バージョン 2.6.30 以降を実行している顧客のみに送信されます。 Amazon Pinpoint が 5 分以内にイベントベースのキャンペーンからメッセージを配信できない場合、そのメッセージはドロップされ、配信を再試行しません。	いいえ

E メールクォータ

次のセクションのクォータは E メールチャネルに適用されます。

E メールメッセージのクォータ

リソース	デフォルトのクォータ	引き上げの対象かどうかの確認
最大メッセージサイズ (添付ファイルを含む)	メッセージごとに 10 MB。	いいえ
検証済み ID の数	10,000 ID **Note** ID とは、E メールアドレスまたはドメイン、またはその 2 つの組み合わせを指します。Amazon Pinpoint を使用して送信する E メールはすべて、検証済み ID から送信される必要があります。	いいえ

E メール送信者と受取人のクォータ

リソース	デフォルトのクォータ	引き上げの対象かどうかの確認
送信者アドレス	すべての送信元のメールアドレスまたはドメインを検証する必要があります。	いいえ
受取人アドレス	アカウントがサンドボックスにある場合は、受取人 E メールアドレスまたはドメインを検証する必要があります。	はい

リソース	デフォルトのクォータ	引き上げの対象かどうかの確認
	アカウントがサンドボックスの外にある場合、任意の有効なアドレスに送信することができます。	
メッセージあたりの受信者の最大数	メッセージあたり 50 人の受取人。	いいえ
検証できるアイデンティティの数	AWS リージョンあたり 10,000 ID **Note** ID とは、E メールアドレスまたはドメイン、またはその 2 つの組み合わせを指します。Amazon Pinpoint を使用して送信する E メールはすべて、検証済み ID から送信される必要があります。	いいえ

E メール送信クォータ

送信クォータ、送信レート、サンドボックスの上限は、同じリージョンにある 2 つのサービスで共有されます。us-east-1 で Amazon SES を使用していて、サンドボックスから削除され、送信クォータ / レートが増加した場合、これらの変更はすべて us-east-1 の Pinpoint アカウントに適用されます。

リソース	デフォルトのクォータ	引き上げの対象かどうかの確認
24 時間ごとに送信可能な E メールの数 (送信クォータ)	アカウントがサンドボックスにある場合、24 時間あたり 200 通の E メール。 アカウントがサンドボックスの外にある場合、クォータは具体的なユースケースによって異なります。 **Note** このクォータは、一意の送信メッセージの数ではなく、受取人の数に基づいています。受取人は To: 行にあるすべての E メールアドレスです。	はい
1 秒ごとに送信できる E メールの数 (送信レート)	アカウントがサンドボックスにある場合、1 秒あたり 1 通の E メール。 アカウントがサンドボックスの外にある場合、レートは具体的なユースケースによって異なります。 **Note** このレートは、一意の送信メッセージの数ではなく、受取人の数に基づいています。受取	はい

リソース	デフォルトのクォータ	引き上げの対象かどうかの確認
	人は To: 行にあるすべての E メールアドレスです。	

エンドポイントクォータ

Amazon Pinpoint API のエンドポイントリソースには、以下のクォータが適用されます。

エンドポイントごとにサポートされる属性の最大数は 250 で、エンドポイントの最大サイズは 15 KB です。ただし、この属性の数は、すべての属性を含むエンドポイントの合計サイズによって制限される場合があります。テンプレートに属性を追加する際にエラーが発生した場合は、各属性のデータ量を減らすか、属性の数を減らすことを検討してください。

リソース	デフォルトのクォータ	引き上げの対象かどうかの確認
エンドポイントのサイズ	最大サイズ: 15 KB	いいえ
`Attributes` 、`Metrics`、`UserAttributes` パラメータにまとめて割り当てられた属性	アプリケーションあたりのすべての属性パラメータに対して 250	いいえ
`Attributes` パラメータにまとめて割り当てられた属性	アプリケーションあたりのすべての属性パラメータに対して 250	いいえ
`Metrics` パラメータにまとめて割り当てられた属性	アプリケーションあたりのすべての属性パラメータに対して 250	いいえ
`UserAttributes` パラメータにまとめて割り当てられた属性	アプリケーションあたりのすべての属性パラメータに対して 250	いいえ

リソース	デフォルトのクォータ	引き上げの対象かどうかの確認
属性名の長さ	50 文字	いいえ
属性値の長さ	100 文字	いいえ
`EndpointBatchItem` ペイロードの `EndpointBatchRequest` オブジェクト	ペイロードごとに 100 個。ペイロードサイズが 7 MB を超えることはできません。	いいえ
同じユーザー ID を持つエンドポイント	ユーザー ID ごとに 15 個の一意のエンドポイント	いいえ
`Attributes` パラメータ属性に割り当てられた値	属性ごとに 50 個	いいえ
`UserAttributes` パラメータ属性に割り当てられた値	属性ごとに 50 個	いいえ

エンドポイントインポートのクォータ

エンドポイントの Amazon Pinpoint へのインポートには、次のクォータが適用されます。

リソース	デフォルトのクォータ	引き上げの対象かどうかの確認
アクティブなインポートジョブ	アカウントあたり 10 個 インポートジョブは、実行中の場合にのみ、このクォータにカウントされます。インポートジョブが完了すると、このクォータにカウントされなくなります。	いいえ
インポートサイズ	インポートジョブごとに 1 GB	いいえ

リソース	デフォルトのクォータ	引き上げの対象かどうかの確認
	たとえば、各エンドポイントが 4 KB 以下の場合、250,000 エンドポイントをインポートできます。	

イベント取り込みのクォータ

Mobile SDKs と Amazon Pinpoint API の Events AWS リソースを使用したイベントの取り込みには、次のクォータが適用されます。

リソース	デフォルトのクォータ	引き上げの対象かどうかの確認
カスタムのイベントタイプの最大数	アプリごとに 1,500	いいえ
カスタム属性キーの最大数	アプリごとに 500	いいえ
属性キーごとのカスタム属性値の最大数	100,000。100,000を超える数値も登録できますが、Amazon Pinpointの分析コンソールでは利用できなくなります。	いいえ
属性キーごとの最大文字数	50	いいえ
属性値ごとの最大文字数	200。文字数が 200 を超えると、イベントは中止されます。	いいえ
カスタムメトリクスキーの最大数	アプリごとに 500	いいえ
1 件のリクエストにおけるイベントの最大数。	リクエストあたり 100	いいえ

リソース	デフォルトのクォータ	引き上げの対象かどうかの確認
リクエストの最大サイズ	4 MB	いいえ
個々のイベントの最大サイズ	1,000 KB	いいえ
イベントごとの属性キーおよびメトリクスキーの最大数	リクエストあたり 40 個	いいえ

ジャーニークォータ

ジャーニーには、以下のクォータが適用されます。

ごとに次のクォータが適用され AWS リージョン 、一部引き上げることができます。詳細については、「Service Quotas ユーザーガイド」の「クォータの増加をリクエスト」を参照してください。

リソース	デフォルトのクォータ	引き上げの対象かどうかの確認
アクティブなジャーニーの最大数	アカウントあたり 50 個	いいえ
アクティブな の最大数 EventTriggeredJourneys	アカウントあたり 20 個	いいえ
ジャーニーアクティビティの最大数	ジャーニーあたり 40 個	いいえ
最大セグメントサイズ	インポートされたセグメントの場合: ジャーニーあたり 100,000,000 ダイナミックセグメントの場合: 無制限	いいえ
コンタクトセンターのアクティビティの最大数	ジャーニーあたり 3	いいえ

リソース	デフォルトのクォータ	引き上げの対象かどうかの確認
[最大休業日数] ルール	チャネルあたり 20	いいえ
[休業日の最大長] ルール名	150 文字	いいえ
[休業日の開始時刻と終了時刻の間の最大日数] ルール	7 日間	いいえ
[最大営業時間数] ルール	1 日あたり 4	いいえ

Lambda クォータ

以下のクォータは、Lambda からデータを取得および処理するための Amazon Pinpoint 設定に適用されます。

リソース	デフォルトのクォータ	引き上げの対象かどうかの確認
Lambda 関数の呼び出しペイロード (リクエストと応答) の最大サイズ	6 MB	いいえ
Lambda 関数がデータを処理するのを待機する最大時間	15 秒	いいえ
エンドポイントあたりのイベント属性の最大数	5	いいえ
イベント属性名の最大文字数	128 文字	いいえ
イベント属性値の最大文字数	128 文字	いいえ
ジャーニーを実行できる最大日数	540 日	いいえ

機械学習のクォータ

以下のクォータは、機械学習 (ML) モデルからデータを取得および処理するための Amazon Pinpoint 設定に適用されます。

リソース	デフォルトのクォータ	引き上げの対象かどうかの確認
モデル設定の最大数	メッセージテンプレートごとに 1 つ アカウントあたり 100 個	いいえ
推奨事項の最大数	エンドポイントまたはユーザーあたり 5 個。	いいえ
エンドポイントまたはユーザーごとの推奨属性の最大数	属性値が AWS Lambda 関数によって処理されない場合、1 属性値が AWS Lambda 関数によって処理される場合、10	いいえ
推奨属性名の最大長	属性名は 50 文字 25 文字の属性表示名 (コンソールの [Attribute finder (属性ファインダー)] に表示される名前)	いいえ
Amazon Personalize から取得される推奨属性値の最大文字数	100 文字	いいえ
Lambda 関数の呼び出しペイロード (リクエストと応答) の最大サイズ	6 MB	いいえ
Lambda 関数がデータを処理するのを待機する最大時間	15 秒	いいえ

リソース	デフォルトのクォータ	引き上げの対象かどうかの確認
Lambda 関数の呼び出しの最大試行回数	試行 3 回	いいえ

ML モデルを使用するように Amazon Pinpoint を設定する方法によっては、追加のクォータが適用される場合があります。Amazon Personalize のクォータの詳細については、Amazon Personalize デベロッパーガイドを参照してください。Amazon Personalize の AWS Lambda クォータの詳細については、AWS Lambda Amazon Personalize デベロッパーガイドのクォータを参照してください。

メッセージテンプレートのクォータ

次のクォータは、Amazon Pinpoint アカウントのメッセージテンプレートに適用されます。

リソース	デフォルトのクォータ	引き上げの対象かどうかの確認
メッセージテンプレートの最大数	アカウントあたり 20,000	いいえ
バージョンの最大数	テンプレートあたり 5,000 個	いいえ
E メールテンプレートの最大文字数	600,000 文字	いいえ
アプリ内テンプレートの最大文字数	200,000 文字	いいえ
プッシュ通知テンプレートのデフォルトテンプレート部分の最大文字数	4,000 文字	いいえ
プッシュ通知テンプレートの ADM 固有のテンプレート部分の最大文字数	6,000 文字	いいえ

リソース	デフォルトのクォータ	引き上げの対象かどうかの確認
プッシュ通知テンプレートの APN 固有のテンプレート部分の最大文字数	4,000 文字	いいえ
プッシュ通知テンプレートの Baidu 固有のテンプレート部分の最大文字数	4,000 文字	いいえ
プッシュ通知テンプレートの FCM 固有のテンプレート部分の最大文字数	4,000 文字	いいえ
SMS テンプレートの最大文字数	1,600 文字	いいえ
音声テンプレートの最大文字数	10,000 文字	いいえ

プッシュ通知のクォータ

プッシュ通知チャネルを通じて Amazon Pinpoint が送信するメッセージには、以下のクォータが適用されます。

リソース	デフォルトのクォータ	引き上げの対象かどうかの確認
キャンペーンで 1 秒あたりに送信できるプッシュ通知の最大数	25000 通知 / 秒	はい
Amazon Device Messaging (ADM) のメッセージペイロードサイズ	メッセージごとに 6 KB	いいえ

リソース	デフォルトのクォータ	引き上げの対象かどうかの確認
Apple Push Notification サービス (APN) メッセージペイロードサイズ	メッセージごとに 4 KB	いいえ
APNS サンドボックスメッセージのペイロードサイズ	メッセージごとに 4 KB	いいえ
Baidu Cloud Push メッセージペイロードサイズ	メッセージごとに 4 KB	いいえ
Firebase Cloud Messaging (FCM) メッセージペイロードサイズ	メッセージごとに 4 KB	いいえ

アプリケーション内のメッセージのクォータ

Amazon Pinpoint で管理するアプリケーション内のメッセージには、以下のクォータが適用されます。

ごとに次のクォータが適用され AWS リージョン 、一部引き上げることができます。詳細については、「Service Quotas ユーザーガイド」の「クォータの増加をリクエスト」を参照してください。

リソース	デフォルトのクォータ	引き上げの対象かどうかの確認
1 秒間に `GetInAppMessages` APIを呼び出すことができる最大回数。	1 秒あたり 5,000 個のリクエスト	はい
アプリ内メッセージングキャンペーン	各プロジェクトには、アプリ内メッセージングチャネルを使用する最大 25 のキャンペーンを含めることができます。	はい。「Service Quotas ユーザーガイド」の「クォータの増加をリクエスト」を参照してください。

セグメントクォータ

Amazon Pinpoint API のセグメントリソースには、以下のクォータが適用されます。

リソース	デフォルトのクォータ	引き上げの対象かどうかの確認
セグメントの作成に使用できるディメンションの最大数	セグメントごとに 100 個	いいえ
セグメントあたりのセグメントグループの最大数	5	いいえ
セグメントあたりのソースセグメントの最大数	5	いいえ
ソースセグメントの最大深度。 例えば、あるセグメントにソースセグメントがあり、そのソースセグメントにもソースセグメントがある場合、深度チェーンはこの制限を超えません。	5	いいえ

ショートメッセージクォータ

SMS チャネルには、以下のクォータが適用されます。

SMS コストの詳細については、「Amazon Pinpoint ユーザーガイド」の「Amazon Pinpoint の SMS 請求および使用状況レポートを理解する」を参照してください。

リソース	デフォルトのクォータ	引き上げの対象かどうかの確認
使用量しきい値	アカウントごとに 1.00 米ドル (USD)。	はい。ただし、リージョンによって上限が異なります。上

リソース	デフォルトのクォータ	引き上げの対象かどうかの確認
		限の引き上げが必要なリージョンを指定する必要があります。
1 秒ごとに送信できる SMS メッセージの数 (送信レート)	送信先の国および発信元の電話番号によって異なります。詳細については、『Amazon Pinpoint ユーザーガイド』の「Message parts per second limits」を参照してください。	はい。ただし、より高いスループットに対応した電話番号の取得が必要な場合があります。使用する番号タイプが不明な場合は、 AWS Support または AWS アカウントマネージャーにお問い合わせください。 英数字の送信者 ID を使用してメッセージを送信すると、スループットレート向上する場合があります。お使いの送信者 ID でスループットの増加が可能かどうかを確認するには、サポートセンターコンソールの[送信者 ID リクエスト] を開きます。リクエストには、既存の送信者 ID、その送信者 ID を使用している国、希望するスループットレートを含めてください。
1 秒ごとに受信者に送信できる SMS メッセージの数	1 秒あたり 1 メッセージ	いいえ
双方向 SMS の Amazon SNS トピック数	アカウントあたり 100,000 個	はい
双方向 SMS のキーワードの数	番号あたり 30 個のキーワード	はい

リソース	デフォルトのクォータ	引き上げの対象かどうかの確認
SMS および音声番号の数	アカウントおよびリージョンあたり 25	はい
専用電話番号の数	アカウントあたり 25	はい
オプトアウトリストの数 注: 必須のデフォルトオプトアウトリストは、このクォータにカウントされます。	アカウントあたり 25	はい
設定セットの数	アカウントあたり 25	はい
イベント送信先の数	設定セットあたり 5	いいえ
SMS サンドボックス内で検証済みの宛先電話番号の数	アカウントあたり 10 個	はい
電話番号プールの数	アカウントあたり 25	はい
電話番号プールに関連付けることができる発信元 ID の数	電話番号プールあたり 100	はい

10DLC クォータ

10DLC の電話番号を使用して送信される SMS メッセージには、以下のクォータが適用されます。10DLC 番号は、米国内の受信者へのメッセージ送信にのみ使用できます。

リソース	デフォルトのクォータ	引き上げの対象かどうかの確認
あたり最大 10DLC 企業 AWS アカウント	25	はい

リソース	デフォルトのクォータ	引き上げの対象かどうかの確認
10DLC 企業あたりの最大 10DLC キャンペーン	10	はい
10DLC キャンペーンあたりの最大 10DLC 番号	49	いいえ

音声クォータ

音声チャネルには、以下のクォータが適用されます。

> **Note**
>
> アカウントがサンドボックスから削除されると、以下の表に示す最大クォータが自動的に適用されます。

リソース	デフォルトのクォータ	引き上げの対象かどうかの確認
24 時間に送信できる音声メッセージの最大数。	アカウントがサンドボックスにある場合: 20 メッセージ。	いいえ
24 時間に 1 人の受信者に送信できる音声メッセージの数	5 メッセージ	いいえ
1 分あたりに送信できる音声メッセージの最大数。	アカウントがサンドボックスにある場合: 5 分あたり 5 個の呼び出し。 アカウントがサンドボックスの外にある場合: 1 分あたり 20 個の呼び出し。	いいえ

リソース	デフォルトのクォータ	引き上げの対象かどうかの確認
1 秒間に発信元の単一の電話番号から送信できる音声メッセージの数	1 秒あたり 1 メッセージ	いいえ
音声メッセージの長さ。	アカウントがサンドボックスにある場合: 30 秒。 アカウントがサンドボックスの外にある場合: 5 分。	いいえ

リソース	デフォルトのクォータ	引き上げの対象かどうかの確認
国際電話番号に音声メッセージを送信する機能	アカウントがサンドボックスにある場合は、次の国の受信者にのみメッセージを送信できます。 • オーストラリア • カナダ • ドイツ • 香港 • イスラエル • 日本 • メキシコ • シンガポール • スウェーデン • アメリカ • 英国 アカウントがサンドボックスの外にある場合は、どの国の受信者にもメッセージを送信できます。 Note 国際電話には追加料金がかかりますが、これは送信先の国やリージョンによって異なります。	いいえ

リソース	デフォルトのクォータ	引き上げの対象かどうかの確認
音声メッセージの文字数	3,000 の請求可能な文字（話されている単語の中の文字） 合計 6,000 文字（請求可能な文字と SSML タグを含む）	いいえ
設定セットの数	AWS リージョンあたり 10,000 個の音声設定セット	いいえ

クォータ引き上げのリクエスト

上記のいずれかの表の [Eligible for Increase] 列の値が [Yes] の場合、そのクォータの引き上げをリクエストすることができます。

クォータの引き上げをリクエストするには

1. https://console.aws.amazon.com/ AWS Management Console で にサインインします。
2. https://console.aws.amazon.com/support/home#/case/create で新しい AWS サポートケースを作成します。
3. [お客様のサポートケース] ペインで、[ケースを作成] を選択します。
4. [サービス制限の引き上げをご希望ですか?] リンクを選択します。
5. 「サービスクォータの引き上げ」で、「サービス」で、次のいずれかのオプションを選択します。

   - メールチャネルに関連するクォータの引き上げをリクエストするには、[Pinpoint Email] を選択します。
   - SMS 使用限度または SMS 送信レートのクォータの増加をリクエストするには、[Pinpoint SMS] を選択します。その他すべての SMS クォータの増加については、[Pinpoint] を選択します。
   - 音声チャネルに関連するクォータの引き上げをリクエストするには、[Pinpoint Voice] を選択します。
   - 他の Amazon Pinpoint 機能に関連するクォータの引き上げをリクエストするには、[Pinpoint] を選択します。

6. 選択したサービスによっては、次のように入力するように求められる場合があります。

   - (オプション) [Provide a link to the site or app which will be sending SMS messages (SMS メッセージを送信するサイトまたはアプリケーションへのリンクを指定する)] で、SMS メッセージを送信する Web サイト、アプリケーション、またはサービスに関する情報を入力します。
   - (オプション) [What type of messages do you plan to send (送信するメッセージのタイプ)] で、ロングコードを使用して送信する予定のメッセージのタイプを選択します。
     - [ワンタイムパスワード] - ウェブサイトまたはアプリケーションを認証するために顧客が使用するパスワードを提供するメッセージ。
     - [プロモーション]- 特価販売やお知らせなど、ビジネスやサービスを宣伝する非クリティカルなメッセージ。
     - [トランザクション]- 注文確認やアカウントアラートなど、顧客のトランザクションをサポートする重要な情報メッセージ。トランザクションメッセージにプロモーションコンテンツまたはマーケティングコンテンツを含めることはできません。
   - (オプション) からメッセージを送信する AWS リージョンで、メッセージを送信するリージョンを選択します。
   - (オプション) [Which countries do you plan to send messages to] で、ショートコードを購入する国または地域を入力します。
   - (オプション) [How do your customers opt to receive messages from you] で、オプトインプロセスの詳細を入力します。
   - (オプション) [How do your customers opt to receive messages from you] で、オプトインプロセスの詳細を入力します。[Please provide the message template that you plan to use to send messages to your customers] の項目に使用するテンプレートを入力します。

7. [リクエスト]で、以下の操作を行います。

   - リージョン で、 を選択します AWS リージョン。
   - [リソースタイプ] で、[一般的な制限] を選択します。Resource Type フィールドは、一部のサービスにのみ存在します。
   - Quota で、変更するクォータを選択します。
   - 新しいクォータ値には、クォータの新しい値を入力します。
   - 追加の で同じクォータの引き上げをリクエストするには AWS リージョン、別のリクエストを追加 を選択し、追加を選択して新しいリクエスト AWS リージョン を入力します。

8. 引き上げるクォータを選択し、クォータに使用する新しい値を入力します。

9. 「ケースの説明」で、クォータの引き上げをリクエストする理由を説明してください。

10. 連絡先オプション で、優先連絡先言語 で、 AWS サポートチームと通信するときに使用する言語を選択します。
11. 問い合わせ方法 で、 AWS サポートチームとの連絡方法を選択します。
12. [送信] を選択します。

AWS サポートチームは、リクエストに 24 時間以内に初期応答を提供します。

迷惑なコンテンツや悪意のあるコンテンツを送信するためにシステムが悪用されないように、各リクエストを慎重に検討する必要があります。可能であれば、24 時間以内にリクエストを承認します。ただし、お客様から追加情報を取得する必要がある場合は、お客様のリクエストの解決に時間がかかる場合があります。

お客様のユースケースが当社の方針と一致しない場合は、リクエストを承認できない場合があります。

Amazon Pinpoint のドキュメント履歴

次の表は、2018年12月以降の 『Amazon Pinpoint デベロッパーガイド』の各リリースにおける重要な変更点を説明しています。このドキュメントの更新に関する通知を受け取るには、RSS フィードにサブスクライブできます。

- ドキュメントの最新更新日: 2023 年 11 月 16 日

変更	説明	日付
E メールヘッダー	E メールメッセージに E メールヘッダーを追加できます。詳細については、「サブスクリプション解除ヘッダーを含む E メールの送信」を参照してください。	2024 年 5 月 7 日
E メールオーケストレーション	Amazon Pinpoint は、Amazon SES リソースを使用して E メールを送信する方法を更新しました。詳細については、Amazon SESロール」を参照してください。	2024 年 4 月 30 日
Amazon Pinpoint のユーザーガイドドキュメントを更新しました	SMS および音声リソース管理トピックが Amazon Pinpoint SMS ユーザーガイド にリダイレクトされるようになりました。詳細については、Amazon Pinpoint SMS ユーザーガイド」を参照してください。	2024 年 2 月 8 日
Amazon Pinpoint のクォータ	[最大休業日数] ルール、[休業日の最大長] ルール名、[休業日の開始時刻と終了時刻の間の最大日数] ルール、および [最大営業時間数] ルール	2023 年 12 月 19 日

	のクォータを追加しました 詳細については、「Amazon Pinpoint のクォータ」を参照してください。	
Amazon Pinpoint のユーザーガイドドキュメントを更新しました	Amazon Pinpoint SMS と音声リソースを作成、設定、管理する方法に関する最新情報を入手するには、新しい「Amazon Pinpoint SMS ユーザーガイド」を参照してください。	2023 年 11 月 16 日
Amazon Pinpoint のクォータ	、 UpdateEndpointsBatch、、 UpdateEndpoint、および のクォータを更新しました PutEvents DeleteEndpoint GetEndpoint。詳細については、「Amazon Pinpoint のクォータ」を参照してください。	2023 年 9 月 22 日
Amazon Pinpoint のクォータ	CreateEmailTemplate、、 CreateSmsTemplate、 、、 CreatePushTemplate CreateInAppTemplate、 CreateVoiceTemplate、および のクォータを更新しました UpdateEmailTemplate UpdateSmsTemplate UpdatePushTemplate UpdateInAppTemplate UpdateVoiceTemplate CreateImportJob。詳細については、「Amazon Pinpoint のクォータ」を参照してください。	2023 年 9 月 12 日

ジャーニーおよびキャンペーンの実行メトリクス	ジャーニーとキャンペーンに新しい分析メトリクスを追加しました。詳細については、「ジャーニーおよびキャンペーンの実行メトリクス」を参照してください。	2023 年 4 月 25 日
Amazon Pinpoint 用のインターフェイス VPC エンドポイントの作成	Amazon Pinpoint がインターフェイス VPC エンドポイントをサポートしました。詳細については、「Amazon Pinpoint 用のインターフェイス VPC エンドポイントの作成」を参照してください。	2023 年 4 月 11 日
転送時の暗号化	2023 年 3 月 22 日から、Amazon Pinpoint は TLS 1.0 をサポートしなくなりますが、TLS 1.2 以降は引き続き使用できます。詳細については、「転送中の暗号化」を参照してください。	2023 年 3 月 20 日
Amazon Pinpoint のクォータ	キャンペーン、ジャーニー、アプリ内メッセージのクォータの増加をリクエストするプロセスを更新しました。詳細については、「Amazon Pinpoint のクォータ」を参照してください。	2022 年 12 月 16 日
リージョナルな可用性	Amazon Pinpoint は、米国東部 (オハイオ) リージョンで利用可能になりました。	2022 年 10 月 5 日

IAM ロールの例の更新	セキュリティのベストプラクティスとの整合性を高めるため、ドキュメント全体でいくつかの IAM ロールの例を更新しました。	2022 年 5 月 27 日
SMS および音声 API バージョン 2	Amazon Pinpoint に、SMS と音声メッセージを送信するための専用 API が追加されました。この API には設定セット、プール、オプトアウトリストなどの新機能が含まれており、SMS と音声メッセージをトランザクションで送信するお客様に役立ちます。詳細については、「Amazon Pinpoint SMS および音声 API の使用」を参照してください。	2022 年 4 月 1 日
ワンタイムパスワードの作成および検証	Amazon Pinpoint に、ワンタイムパスワード (OTP) を生成し、SMS メッセージとしてユーザーに送信する機能が追加されました。また、ユーザーがアプリケーションやサイトにワンタイムパスワードを入力する際に、そのコードを検証するための API も含まれています。詳細については、「Sending and validating One-Time Passwords (OTPs)」を参照してください。	2021 年 11 月 26 日

アプリケーション内のメッセージ	Amazon Pinpoint の [in-app messaging] 機能をユーザーのアプリケーションに統合するための情報を追加しました。	2021 年 11 月 10 日
コード例	Amazon Pinpoint の一般的なオペレーションのコード例のライブラリを追加しました。	2021 年 11 月 3 日
プロジェクトクォータ	Amazon Pinpoint プロジェクトの最大数は 100 のままですが、 でサービス制限引き上げリクエストを開くことで、このクォータを増やすことができるようになりました AWS Support。	2021 年 10 月 11 日
Lambda ポリシーの更新。	特定のLambda許可ポリシーに `AWS:SourceAccount` 条件を含める必要があります。この要件に対応するために、「Creating custom channels in Amazon Pinpoint」と「Customizing segments with AWS Lambda」トピックのサンプルポリシーを更新しました。	2021 年 10 月 7 日
UpdateEndpoint	Amazon Pinpoint UpdateEndpoint API が によって口グに記録されるようになりました CloudTrail。	2020 年 11 月 16 日

カスタム属性	Amazon Pinpoint は、E メールメッセージテンプレートで 250 の属性をサポートします。「クォータ」を参照してください。	2020 年 9 月 18 日
リージョナルな可用性	Amazon Pinpoint は、アジアパシフィック (東京) リージョン、欧州 (ロンドン) リージョン、およびカナダ (中部) リージョンでご利用いただけるようになりました。Amazon Pinpoint SMS および音声 API は、これらのリージョンではご利用いただけませんのでご注意ください。	2020 年 9 月 10 日
リージョナルな可用性	Amazon Pinpoint は、アジアパシフィック (東京) リージョンでご利用いただけるようになりました。Amazon Pinpoint SMS および音声 API は、これらのリージョンではご利用いただけませんのでご注意ください。	2020 年 9 月 2 日
キャンペーンイベント	キャンペーンイベントの `delivery_type` パラメータに、新しいキャンペーンイベントの情報を追加しました。	2020 年 8 月 2 日

リージョナルな可用性	Amazon Pinpoint は、アジアパシフィック (ソウル) リージョンでご利用いただけるようになりました。このリージョンでは、Amazon Pinpoint API は、音声または SMS をサポートしていないのでご注意ください。	2020 年 7 月 31 日
リージョナルな可用性	Amazon Pinpoint が AWS GovCloud (US) リージョンで利用可能になりました。	2020 年 4 月 30 日
カスタムチャンネル	Lambda 関数や webhooks を利用したカスタムチャンネルの作成に関する情報を更新しました。	2020 年 4 月 23 日
機械学習	レコメンダーモデルからパーソナライズされたレコメンデーションを取得し、オプションで AWS Lambda 関数を使用してそれらのレコメンデーションを強化する方法に関する情報を追加しました。	2020 年 3 月 4 日
セキュリティ	Amazon Pinpoint の様々なセキュリティコントロールと機能についての情報を提供する、「セキュリティの章」を追加しました。	2020 年 2 月 4 日

ジャーニー	Amazon Pinpoint ジャーニーを使用して、プロジェクトのメッセージングアクティビティを実行する自動ワークフローを開発することに関する情報を追加しました。また、ジャーニーに適用されるメトリクスのサブセットの分析データのクエリに関する情報も追加しました。	2019 年 10 月 31 日
分析	キャンペーンとトランザクションメッセージの分析データをクエリする方法を説明する手順と、クエリ結果の使用に関する情報を追加しました。	2019 年 10 月 17 日
分析	トランザクション E メールおよび SMS メッセージに適用されるメトリクスのサブセットの分析データのクエリに関する情報を追加しました。	2019 年 9 月 6 日
コード例	Amazon Pinpoint がサポートするすべてのサービスを使用して、トランザクションのプッシュ通知を送信するために使用できるコード例を追加しました。	2019 年 7 月 30 日
分析	プロジェクト (アプリケーション) およびキャンペーンに適用されるメトリックのサブセットについての分析データのクエリに関する情報を追加しました。	2019 年 7 月 24 日

セグメント	Salesforce や Marketo などの外部システムから Amazon Pinpoint に顧客データをインポートするためのソリューションを説明する「チュートリアル」を追加しました。	2019 年 5 月 14 日
リージョナルな可用性	Amazon Pinpoint が、 AWS アジアパシフィック (ムンバイ) およびアジアパシフィック (シドニー) リージョンで利用可能になりました。	2019 年 4 月 25 日
Amazon Pinpoint で Postman を使用する	Postman を使って Amazon Pinpoint API とやりとりする方法を説明した「チュートリアル」を追加しました。	2019 年 4 月 8 日
タグ付け	Amazon Pinpoint リソースのタグ付けに関する情報を追加しました。	2019 年 2 月 27 日
SMS 登録	「チュートリアルの章」を追加し、SMS ユーザーの登録を処理するソリューションの作成方法を示す「チュートリアル」を追加しました。	2019 年 2 月 27 日
コード例	E メール、SMS、音声メッセージをプログラムで送信する方法を紹介する、複数のプログラミング言語による コード例を追加しました。	2019 年 2 月 6 日

以前の更新

次の表は、2018年12月までの Amazon Pinpointデベロッパーガイドの各リリースにおける重要な変更点を説明したものです。

変更	説明	日付
リージョナルな可用性	Amazon Pinpoint が AWS 米国西部 (オレゴン) および欧州 (フランクフルト) リージョンで利用可能になりました。	2018 年 12 月 21 日
音声チャンネル	新しい Amazon Pinpoint 音声チャンネルを使用して音声メッセージを作成し、それらのメッセージを電話でお客様に配信できます。現時点では、Amazon Pinpoint SMS および 音声 API を使用してのみ音声メッセージを送信できます。	2018 年 11 月 15 日
欧州 (アイルランド) での可用性	Amazon Pinpoint が AWS 欧州 (アイルランド) リージョンで利用可能になりました。	2018 年 10 月 25 日
イベント API	Amazon Pinpoint API を使用してイベントを記録し、そのイベントをエンドポイントと関連付けます。	2018 年 8 月 7 日
エンドポイントを定義、検索するためのコード例	エンドポイントを定義、更新、削除、検索する方法を示すコード例を追加しています。 AWS CLI、 AWS SDK for Java、および Amazon Pinpoint API の例を示します。詳細については、	2018 年 8 月 7 日

変更	説明	日付
	「Amazon Pinpoint へ対象の定義」および「Amazon Pinpoint の対象者データへアクセス」を参照してください。	
エンドポイントエクスポートのアクセス許可	Amazon Pinpoint エンドポイントを Amazon S3 バケットにエクスポートすることを許可する IAM ポリシーを設定します。	2018 年 5 月 1 日
SMS 用の電話番号の確認	Amazon Pinpoint API を使用して電話番号を確認し、SMS メッセージの有効な宛先であるかどうかを判断します。	2018 年 4 月 23 日
Amazon Pinpoint の統合に関するトピックの更新	AWS SDKs またはライブラリを使用して、Amazon Pinpoint を Android、iOS、またはアプリケーションと統合します。JavaScript	2018 年 3 月 23 日
AWS CloudTrail ログ記録	を使用した Amazon Pinpoint API コールのログ記録に関する情報を追加しました CloudTrail。	2018 年 2 月 6 日
サービスクォータを更新	E メールクォータに関する情報を追加し、「クォータ」を更新しました。	2018 年 1 月 19 日

変更	説明	日付
Amazon Pinpoint 拡張機能のパブリックベータ版	AWS Lambda 関数を使用してセグメントをカスタマイズしたり、カスタムメッセージングチャネル を作成したりします。	2017 年 11 月 28 日
プッシュ通知のペイロードのクォータ	このクォータには、モバイルプッシュメッセージのペイロードサイズが含まれます。	2017 年 10 月 25 日
サービスクォータを更新	SMS と E メールチャンネル情報を「クォータ」に追加しました。	2017 年 10 月 9 日
ADM および Baidu モバイルプッシュ	Baidu および ADM モバイルプッシュチャンネルからのプッシュ通知を処理するようにアプリケーションコードを更新しました。	2017 年 9 月 27 日
Amazon Cognito ユーザープールでのユーザー ID と認証イベント。	Amazon Cognito ユーザープールを使用してモバイルアプリケーションでユーザーのサインインを管理する場合、Amazon Cognito はエンドポイントにユーザー ID を割り当て、Amazon Pinpoint に認証イベントを報告します。	2017 年 9 月 26 日
ユーザー ID	エンドポイントにユーザー ID を割り当てて、個々のユーザーからのアプリケーションの使用状況をモニタリングします。AWS Mobile SDK および SDK for Java の例を示しています。	2017 年 8 月 31 日

変更	説明	日付
認証イベント	認証イベントを報告し、ユーザーがアプリケーションに対して認証する頻度を確認します。例は、アプリケーションでのイベントのレポート で参照できます。	2017 年 8 月 31 日
サンプルイベントの更新	サンプルイベントには、E メールと SMS アクティビティについて Amazon Pinpoint がストリーミングするイベントが含まれます。	2017 年 6 月 08 日
Android セッション管理	AWS Mobile Hub サンプルアプリケーションにより提供されているクラスを使用した Android アプリケーションのセッションを管理します。	2017 年 4 月 20 日
更新された収益化イベントサンプル	収益化イベントの報告に関するサンプルコードが更新されました。	2017 年 3 月 31 日
イベントストリーム	Amazon Pinpoint を設定してアプリケーションとキャンペーンイベントを Kinesis ストリームに送ることができます。	2017 年 3 月 24 日
アクセス許可	アカウントの AWS ユーザーとモバイルアプリのユーザーに Amazon Pinpoint へのアクセスを許可する方法については、Amazon Pinpoint で IAM が機能する仕組み「」を参照してください。	2017 年 1 月 12 日

変更	説明	日付
Amazon Pinpoint の一般への可用性	本リリースでは、Amazon Pinpoint を導入しました。	2016 年 12 月 1 日

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。