



このコンテンツは公開から3年以上経過しており内容が古い可能性があります
最新情報については[サービス別資料](#)もしくはサービスのドキュメントをご確認ください

[AWS Black Belt Online Seminar]

AWS CodeCommit & AWS CodeArtifact

サービスカットシリーズ

Solutions Architect 松本 雅博
2020/10/20

AWS 公式 Webinar
<https://amzn.to/JPWebinar>



過去資料
<https://amzn.to/JPArchive>



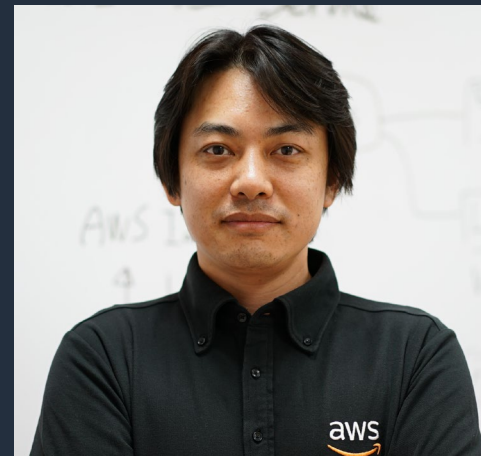
自己紹介

松本 雅博（まつもと まさひろ）

技術統括本部 西日本ソリューション部
ソリューションアーキテクト

関西を中心に、西日本のお客様をご支援
好きなサービス

- Code シリーズ
- AWS CloudFormation, AWS Cloud Development Kit



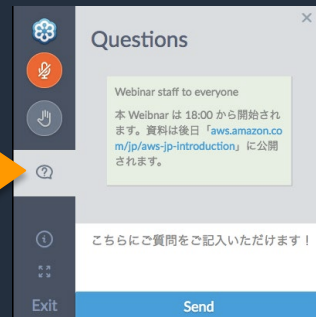
AWS Black Belt Online Seminar とは

「サービス別」「ソリューション別」「業種別」のそれぞれのテーマに分かれて、アマゾンウェブ サービス ジャパン株式会社が主催するオンラインセミナーシリーズです。

質問を投げることができます！

- 書き込んだ質問は、主催者にしか見えません
- 今後のロードマップに関するご質問はお答えできませんのでご了承下さい

- ① 吹き出しをクリック
- ② 質問を入力
- ③ Sendをクリック



Twitter ハッシュタグは以下をご利用ください
#awsblackbelt

内容についての注意点

- 本資料では2020年10月20日現在のサービス内容および価格についてご説明しています。最新の情報はAWS公式ウェブサイト(<http://aws.amazon.com>)にてご確認ください。
- 資料作成には十分注意しておりますが、資料内の価格とAWS公式ウェブサイト記載の価格に相違があった場合、AWS公式ウェブサイトの価格を優先とさせていただきます。
- 価格は税抜表記となっております。日本居住者のお客様には別途消費税をご請求させていただきます。
- AWS does not offer binding price quotes. AWS pricing is publicly available and is subject to change in accordance with the AWS Customer Agreement available at <http://aws.amazon.com/agreement/>. Any pricing information included in this document is provided only as an estimate of usage charges for AWS services based on certain information that you have provided. Monthly charges will be based on your actual use of AWS services, and may vary from the estimates provided.

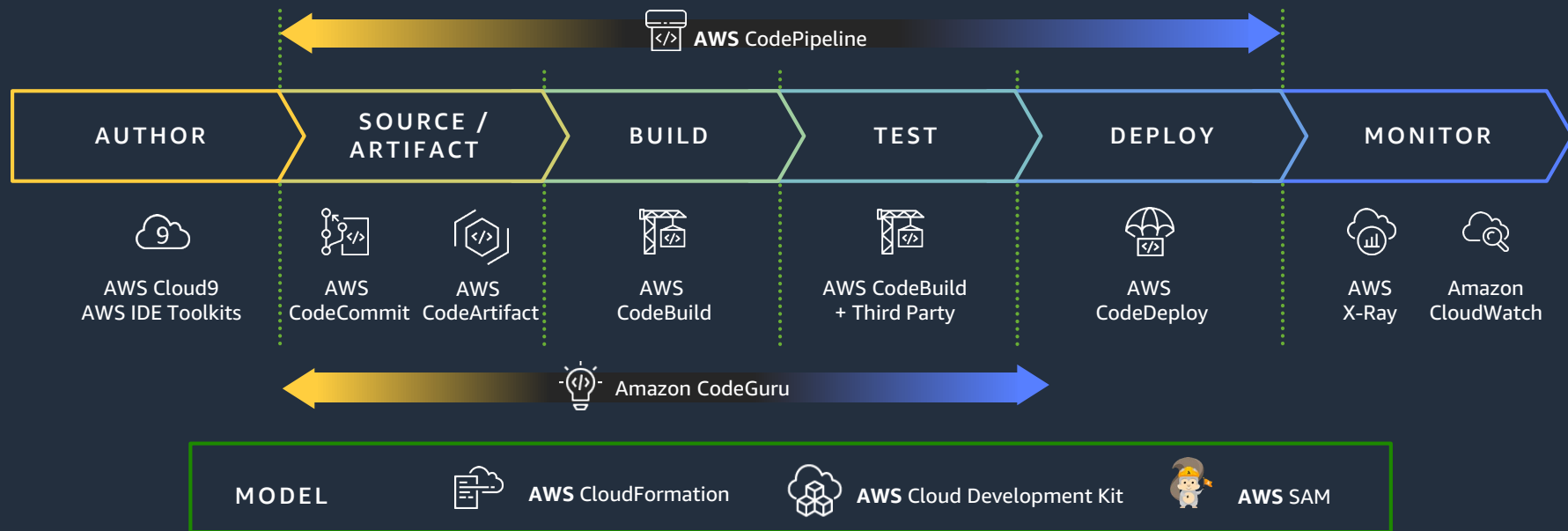
本セミナーの概要

- 本セミナーで学習できること
 - AWS CodeCommit
 - AWS CodeArtifact
- 対象者
 - アーキテクトの方
 - 技術者の方

本日のアジェンダ

- ソフトウェア開発に関連する AWS サービス
- AWS CodeCommit
- AWS CodeArtifact
- まとめ

ソフトウェア開発に関連する AWS サービス



自社でホストするツールの課題

- 開発者ごとの高価なライセンス費用
- ハードウェア保守コスト
- サポートスタッフのコスト
- 保存や管理できるデータ量やファイルタイプの制限
- 管理できるブランチの数やバージョン履歴の数の制限

クラウドにおけるソース管理に求められるもの



Secure



Fully
managed



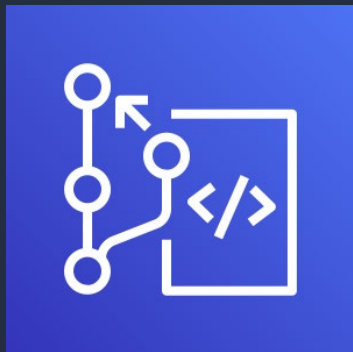
High
availability



Store
anything

AWS CodeCommit

AWS CodeCommit

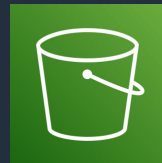


- ✓ **セキュア**で**スケーラブル**な**マネージドGit互換ソース管理**
- ✓ 標準的な Git ツールが利用可能
- ✓ Amazon S3 のスケーラビリティ、可用性、堅牢なストレージを利用
- ✓ 利用者固有のキーを使用した暗号化
- ✓ **レポジトリサイズの上限なし**
- ✓ コンテンツの直接編集をサポート
- ✓ プルリクエスト機能に対応
- ✓ VPC エンドポイントをサポート

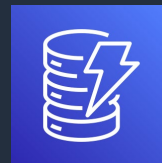
AWS CodeCommit 概要



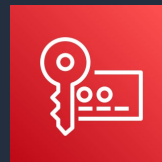
SSH or HTTPS



Git オブジェクトは
Amazon S3で管理



Git インデックスは
Amazon DynamoDB
で管理



暗号化キーは
AWS KMSで管理



リポジトリの作成

- リポジトリ名を指定するだけ
- 説明とタグはオプション
- Amazon CodeGuru Reviewer for Java との連携設定も可能

デベロッパー用ツール > CodeCommit > リポジトリ > リポジトリを作成

リポジトリを作成

コードを格納して共有する安全なリポジトリを作成します。リポジトリ名とリポジトリの説明を入力し始めます。リポジトリ名は、そのリポジトリの URL に含まれています。

リポジトリの設定

リポジトリ名

最大 100 文字。この制限が適用されます。

説明 - オプション

最大 1,000 文字

タグ

Amazon CodeGuru Reviewer for Java を有効にする - オプション

このリポジトリ内のすべてのプルリクエストの Java コードの品質を改善するための推奨事項をご覧ください。

サービスにリンクされたロールが存在しない場合は、IAM に代わって作成されます。

キャンセル

CLIから

```
aws codecommit create-repository --repository-name myfirstrepo ¥  
--repository-description "My first repository"
```

CodeCommit を利用するには



最も簡単な方法： CodeCommit の HTTPS 接続で Git 認証情報を設定する

- IAM で Git ユーザー名とパスワードを生成して使用する
- 様々な IDE と互換性がある

その他の接続方法：

- SSH 接続 (IAM と SSH キーを紐付ける)
- git-remote-codecommit (IAM のクレデンシアルを利用する)
- 開発ツールからの接続 (AWS Toolkit を利用する)

詳細は以下の URL を参照

<https://docs.aws.amazon.com/codecommit/latest/userguide/setting-up.html>



HTTPS 接続と Git 認証の手順

1. AWS CodeCommit にアクセスするユーザを作成
 2. IAM に静的なユーザー名とパスワードを生成
 3. 生成した認証情報を Git のユーザー名、パスワード認証で利用
- AWS CodeCommit は Git 1.7.9 以上をサポート
 - AWS CodeCommit は Curl 7.33 以上が必要
 - curl update 7.41.0 をHTTPSで利用する場合は既知の障害あり

以下を参照：

<https://docs.aws.amazon.com/codecommit/latest/userguide/troubleshooting.html>

HTTPS 接続 – IAM からの Git 認証情報の作成



- 認証情報 タブを選択

The screenshot shows the AWS IAM console interface. On the left is a navigation menu for 'Identity and Access Management (IAM)' with options like 'ダッシュボード', 'アクセス管理', 'グループ', 'ユーザー', 'ロール', 'ポリシー', and 'ID プロバイダー'. The main area is titled 'ユーザー > developer01' and '概要'. It displays user details: 'ユーザーの ARN' (arn:aws:iam:::user/developer01), 'パス' (/), and '作成時刻' (2020-10-16 16:25 UTC+0900). Below this is a tabbed interface with '認証情報' highlighted in a red box. Other tabs include 'アクセス権限', 'グループ', 'タグ (1)', and 'アクセスアドバイザー'. The '認証情報' tab shows 'サインイン 認証情報'.

HTTPS 接続 – IAM からの Git 認証情報の作成



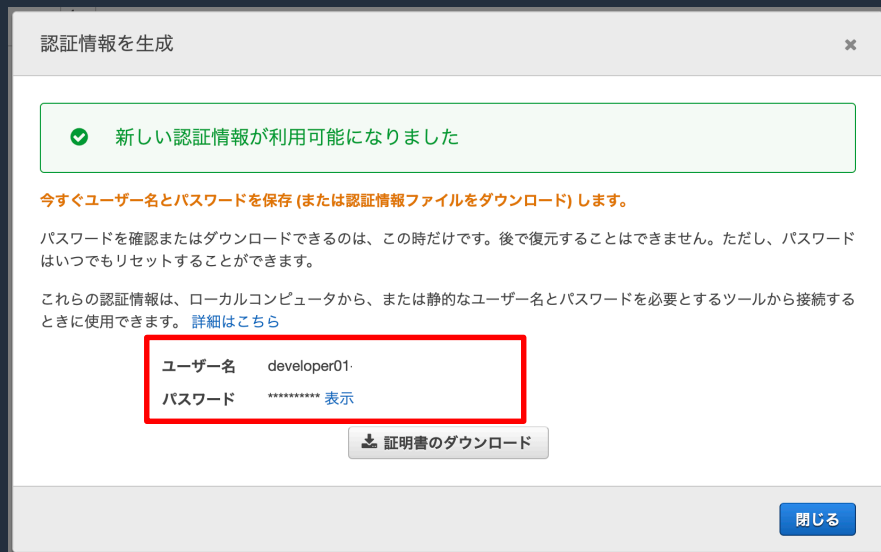
- 認証情報を生成をクリック

The screenshot shows the AWS IAM console interface. On the left, the 'Identity and Access Management (IAM)' sidebar is visible with a navigation menu including 'ダッシュボード', 'アクセス管理', 'グループ', 'ユーザー', and 'ロール'. The main content area is titled '結果がありません' (No results) and contains a section for 'AWS CodeCommit の HTTPS Git 認証情報'. Below this section, there is a button labeled '認証情報を生成' (Generate authentication information) which is highlighted with a red rectangular box. Below the button, a message states '認証情報が生成されていません。' (Authentication information has not been generated).

HTTPS 接続 – IAM からの Git 認証情報の作成



- 認証情報が作成される
- パスワードはこのタイミングでしか確認できない
- 認証情報のダウンロードを忘れずに



HTTPS 接続 – IAM からの Git 認証情報の作成



- パスワードを忘れてしまった場合、この画面よりリセット可能

Identity and Access Management (IAM)

ダッシュボード

▼ アクセス管理

- グループ
- ユーザー**
- ロール
- ポリシー
- ID プロバイダー
- アカウント設定

AWS CodeCommit の HTTPS Git 認証情報

AWS CodeCommit リポジトリへの HTTPS 接続の認証に使用できるユーザー名とパスワードを生成します。最大 2 セットの認証情報を生成して保存できます。 [詳細はこちら](#)

認証情報を生成 アクション ▼

	ユーザー名	アクション	ステータス	作成
<input checked="" type="radio"/>	developer01-at	無効化 パスワードのリセット 削除	有効	2020-10-16 16:41 UTC+0900

Amazon Keyspaces (for Apache Cassandra) の認証情報

Amazon Keyspaces への認証に使用できるユーザー名とパスワードを生成します。Amazon Keyspaces に対して最大 2 セットの認証情報を生

Local Git から CodeCommit への接続



- リポジトリの URL を取得する

デベロッパー用ツール **CodeCommit**

デベロッパー用ツール > CodeCommit > リポジトリ

リポジトリ 情報

	名前	説明	最終更新日時	URL のクローン
<input type="radio"/>	blackbelt-demo	BlackBelt 用	7 時間前	<input type="button" value="🔗 HTTPS"/> <input type="button" value="🔗 SSH"/> <input type="button" value="🔗 HTTPS (GRC)"/>
<input type="radio"/>	myfirstrepo	My first repository	9 時間前	<input type="button" value="🔗 HTTPS"/> <input type="button" value="🔗 SSH"/> <input type="button" value="🔗 HTTPS (GRC)"/>

デベロッパー用ツール **CodeCommit**

デベロッパー用ツール > CodeCommit > リポジトリ > myfirstrepo

myfirstrepo

myfirstrepo 情報

名前

- HTTPS のクローン
- SSH のクローン
- HTTPS のクローン (GRC)
- 接続のステップ

Local Git から CodeCommit への接続



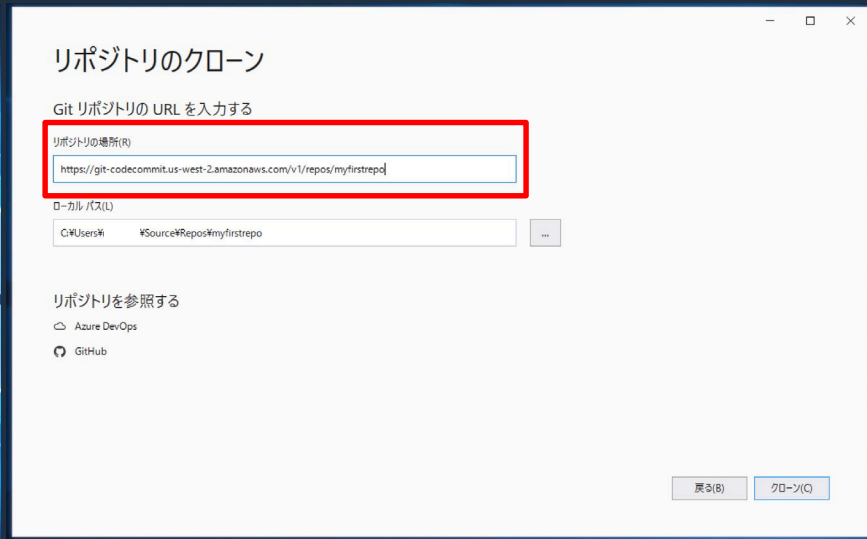
- 作成した認証情報を利用して接続

```
$ git clone https://git-codecommit.us-west-2.amazonaws.com/v1/repos/myfirstrepo
Cloning into 'myfirstrepo'...
Username for 'https://git-codecommit.us-west-2.amazonaws.com': developer01-at-
Password for 'https://developer01-at-          @git-codecommit.us-west-2.amazonaws.com':
warning: You appear to have cloned an empty repository.
```

Visual Studio からの接続



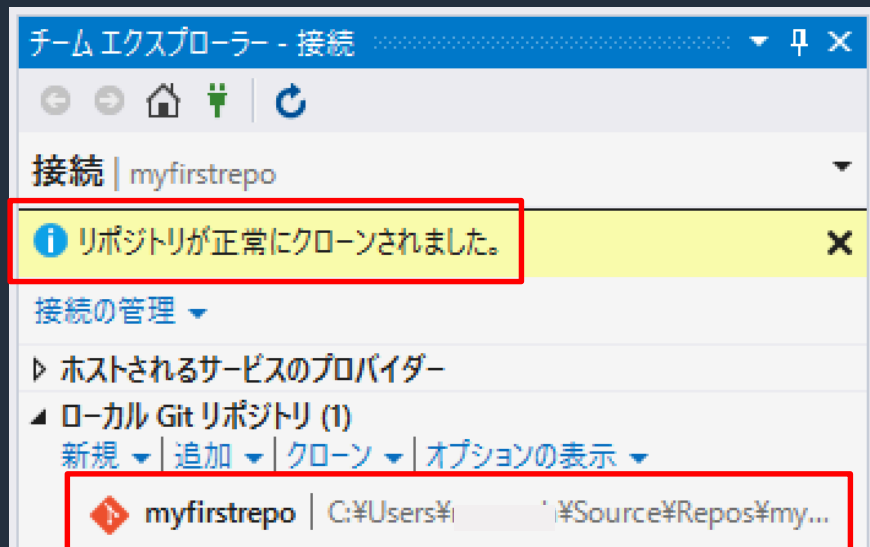
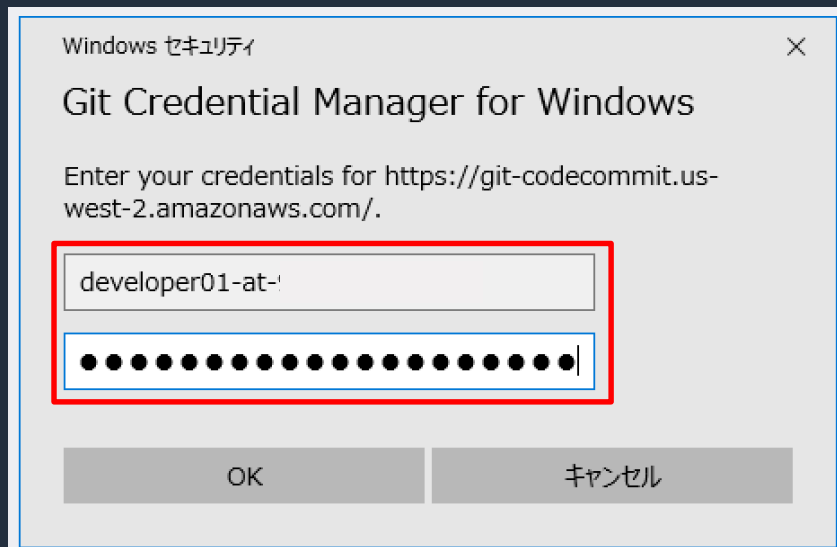
- リポジトリのクローンをクリックし、URL を入力する



Visual Studio からの接続



- 認証情報を入力する

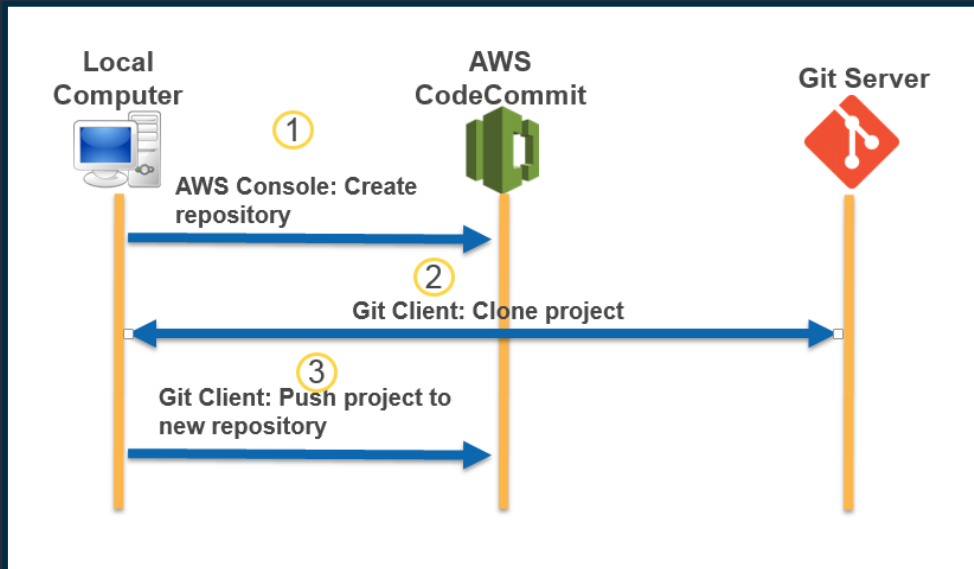


他の Git リポジトリからのマイグレーション



他の Git リポジトリからのマイグレーションは非常にシンプル

1. CodeCommit にリポジトリ を作成
2. 他の Git リポジトリからローカルにクローン
3. CodeCommit へ Push



参考資料 : git を学ぶ



Pro Git

<https://git-scm.com/book/ja/v2> (日本語)

Git Cheat sheet

<https://training.github.com/downloads/ja/github-git-cheat-sheet.pdf> (日本語)

Git Immersion

<http://gitimmersion.com/>

Git Reference

<https://git-scm.com/docs>

git はHTTPプロトコルを利用する場合、デフォルトでは毎回ユーザー名、パスワードの入力を求めるが、`git config --global credential.helper` を利用することでクレデンシャル情報をキャッシュまたは保存することが可能

<https://git-scm.com/book/en/v2/Git-Tools-Credential-Storage>



コンテンツの参照・編集・追加

- AWS CodeCommit コンソールでリポジトリのコンテンツを閲覧
 - ブランチやタグを切り替えて表示可能
- ファイルの編集、追加も可能
 - リポジトリ内のファイルを直接編集
 - ファイルの作成や、ローカルからのアップロードも可能

リポジトリのコンテンツ参照



- マネジメントコンソールから、リポジトリを選択

デベロッパー用ツール ×

CodeCommit

▼ ソース • CodeCommit

- 開始方法
- リポジトリ**
- 承認ルールテンプレート
- ▶ アーティファクト
 - CodeArtifact
- ビルド • CodeBuild

デベロッパー用ツール > CodeCommit > リポジトリ

リポジトリ 情報

< 1 > ⚙️

	名前 ▼	説明	最終更新日時 ▼	URL のクローン
<input type="radio"/>	blackbelt-demo	BlackBelt 用	8 時間前	<input type="button" value="🔗 HTTPS"/> <input type="button" value="🔗 SSH"/> <input type="button" value="🔗 HTTPS (GRC)"/>
<input type="radio"/>	myfirstrepo	My first repository	10 時間前	<input type="button" value="🔗 HTTPS"/> <input type="button" value="🔗 SSH"/> <input type="button" value="🔗 HTTPS (GRC)"/>

リポジトリのコンテンツ参照



- リポジトリ内を参照することが可能
- これ以降の表示内容は aws-samples/aws-cdk-examples

デベロッパー用ツール > CodeCommit > リポジトリ > blackbelt-demo

blackbelt-demo

通知 ▼ custom-logical-names ▼ ブリクエストの作成 URL のクローン ▼

blackbelt-demo / typescript / amplify-console-app 情報 ファイルの追加 ▼

名前
..
cdk.json
index.ts
package.json
README.md
tconfig.json



デベロッパー用ツール > CodeCommit > リポジトリ > blackbelt-demo

blackbelt-demo

通知 ▼ custom-logical-names ▼ ブリクエストの作成 URL のクローン ▼

blackbelt-demo / typescript / amplify-console-app / index.ts 情報 編集

```
1 import cdk = require('@aws-cdk/core');
2 import { CfnApp, CfnBranch } from '@aws-cdk/aws-amplify';
3
4 export class AmplifyConsoleAppCdkStack extends cdk.Stack {
5   constructor(scope: cdk.Construct, id: string, props?: cdk.StackProps) {
6     super(scope, id, props);
7
8     const amplifyApp = new CfnApp(this, 'test-app', {
9       name: 'your-amplify-console-app-name',
10      repository: 'https://github.com/<the-rest-of-the-repository-url>',
11      oauthToken: '<your-github-oauth-token>'
12    });
13
14    new CfnBranch(this, 'MasterBranch', {
15      appId: amplifyApp.attrAppId,
16      branchName: 'master' // you can put any branch here (careful, it will listen
17        to changes on this branch)
18    });
19  }
```

リポジトリのコンテンツ編集



- 参照だけでなく、ファイルの編集も可能

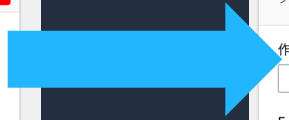
デベロッパー用ツール > CodeCommit > リポジトリ > blackbelt-demo

blackbelt-demo

通知 ▼ custom-logical-names ▼ ブリクエストの作成 URL のクローン ▼

blackbelt-demo / typescript / amplify-console-app / index.ts 情報 **編集**

```
1 import cdk = require('@aws-cdk/core');
2 import { CfnApp, CfnBranch } from '@aws-cdk/aws-amplify';
3
4 export class AmplifyConsoleAppCdkStack extends cdk.Stack {
5   constructor(scope: cdk.Construct, id: string, props?: cdk.StackProps) {
6     super(scope, id, props);
7
8     const amplifyApp = new CfnApp(this, 'test-app', {
9       name: 'your-amplify-console-app-name',
10      repository: 'https://github.com/<the-rest-of-the-repository-url>',
11      oauthToken: '<your-github-oauth-token>'
12    });
13
14    new CfnBranch(this, 'MasterBranch', {
```



デベロッパー用ツール > CodeCommit > リポジトリ > blackbelt-demo > ファイル

ファイルの編集

custom-logical-names ▼

blackbelt-demo / typescript / amplify-console-app / index.ts 情報

```
1 import cdk = require('@aws-cdk/core');
2 import { CfnApp, CfnBranch } from '@aws-cdk/aws-amplify';
3
4 export class AmplifyConsoleAppCdkStack extends cdk.Stack {
5   constructor(scope: cdk.Construct, id: string, props?: cdk.StackProps) {
```

custom-logical-names に対する変更のコミット

ファイル: blackbelt-demo/typescript/amplify-console-app/index.ts

作成者名

Eメールアドレス

メッセージのコミット - オプション
コミットメッセージを指定しないと、デフォルトのコミットメッセージが使用されます。

キャンセル **変更のコミット**

リポジトリのコンテンツ追加



- 参照だけでなく、ファイルの追加も可能

デベロッパー用ツール > CodeCommit > リポジトリ > blackbelt-demo

blackbelt-demo

通知 ▼ custom-logical-names ▼ プルリクエストの作成 URL のクローン ▼

blackbelt-demo 情報

ファイルの追加 ▲

- ファイルの作成
- ファイルのアップロード

名前
.github
csharp
java
python
scripts
typescript



デベロッパー用ツール > CodeCommit > リポジトリ > blackbelt-demo > ファイル

ファイルの作成

custom-logical-names ▼

blackbelt-demo 情報

1	⚙️
---	----

デベロッパー用ツール > CodeCommit > リポジトリ > blackbelt-demo > ファイル

ファイルのアップロード

custom-logical-names ▼

blackbelt-demo 情報

Name	Size	Actions
ファイルのアップロード アップロードするファイルを選択します。		
<input type="button" value="Choose file"/>		

コミット履歴の表示



- AWS CodeCommit コンソールでリポジトリのコミット履歴を表示
 - `git rebase` コマンドでリベースを行った場合は、リポジトリの履歴が変更されるので注意
 - ブランチやタグを切り替えて表示可能
 - 直前のコミットとの差分比較（分割表示または結合表示が可能）
 - Commit IDをコピー可能。コマンドラインのコミットの比較で利用可能
 - `</>`マークをクリックすると対象のソースコードを表示

コミット履歴の表示



- ブランチやタグを切り替えて表示することが可能

The screenshot shows the AWS CodeCommit interface for the repository 'blackbelt-demo'. The left sidebar contains navigation options, with 'コミット' (Commits) highlighted. The main content area shows the commit history table. A dropdown menu is open, showing the current branch 'custom-logical-names' and a list of other branches including 'master', '/c#-lambda', '/fix-build', '/mergify-update', 'r/add-java-ation-load-er', and 'cs/csharp-unpin-dependencies'. A blue arrow points from the dropdown in the main view to the expanded dropdown menu.

デベロッパー用ツール ×

デベロッパー用ツール > CodeCommit > リポジトリ > blackbelt-demo > コミット

blackbelt-demo

コミット | コミットビジュアルライザー | コミットの比較

コミット 情報

custom-logical-names ▼

Q | フィルタするタイプ

ブランチ

- custom-logical-names
デフォルトブランチ
- master
- /c#-lambda
- /fix-build
- /mergify-update
- r/add-java-ation-load-er
- cs/csharp-unpin-dependencies

コミット ID	メッセージのコミット	コミット日	作成者	アクション
efafb821	Merge branch 'master' into custom-logical-names	9 か月前		ID をコピーする 参照
42f652a5	update ts version to *	9 か月前		ID をコピーする 参照
ec418493	fix: update CDK versions for Java examples to latest (#217)	10 か月前		ID をコピーする 参照
22a34b59	Update README.md (#213)	10 か月前		ID をコピーする 参照



コミット履歴の表示

- 直前のコミットとの差分比較

デベロッパー用ツール > CodeCommit > リポジトリ > blackbelt-demo > コミット > efafb8219ad8e19e55b52002ac74800156022bd6

efafb8219ad8e19e55b52002ac74800156022bd6 をコミット コミット ID のコピー 参照

▼ 詳細

作成者	コミット日	親コミット
GitHub noreply@github.com	9 か月前	42f652a5d580ed1049c55b7041ebf6b5c70be3a9 ec4184933f98f07437deb13f2387563ab9d43d5c

メッセージのコミット
Merge branch 'master' into custom-logical-names

< 2の1ページ目 > ファイルに移動 コメントを非表示 空白の変更を非表示 統合 分割

▼ README.md ファイルコンテンツを参照 ファイルに関するコメント

```

*** @ -38,7 +38,7 @@
38 38 | [ecs-service-with-task-placement](https://github.com/aws-samples/aws-cdk-examples/tree/master/typescript/ecs/ec
39 39 | [ecs-service-with-advanced-alb-config](https://github.com/aws-samples/aws-cdk-examples/tree/master/typescript/ec
40 40 | [ecs-service-with-task-networking](https://github.com/aws-samples/aws-cdk-examples/tree/master/typescript/ecs/ec
41 - | [fargate-application-load-balanced-service](https://github.com/aws-samples/aws-cdk-examples/tree/master/typescri
41 + | [fargate-application-load-balanced-service](https://github.com/aws-samples/aws-cdk-examples/tree/master/typescri
42 - | [fargate-service-with-auto-scaling](https://github.com/aws-samples/aws-cdk-examples/tree/master/typescript/ecs/f

```

デベロッパー用ツール > CodeCommit > リポジトリ > blackbelt-demo > コミット > efafb8219ad8e19e55b52002ac74800156022bd6

efafb8219ad8e19e55b52002ac74800156022bd6 をコミット コミット ID のコピー 参照

▼ 詳細

作成者	コミット日	親コミット
GitHub noreply@github.com	9 か月前	42f652a5d580ed1049c55b7041ebf6b5c70be3a9 ec4184933f98f07437deb13f2387563ab9d43d5c

メッセージのコミット
Merge branch 'master' into custom-logical-names

< 2の1ページ目 > ファイルに移動 コメントを非表示 空白の変更を非表示 統合 分割

▼ README.md ファイルコンテンツを参照 ファイルに関するコメント

```

*** @ -38,7 +38,7 @@
38 | [ecs-service-with-task-placement]
38 | [ecs-service-with-task-placement]
38 | [ecs-service-with-task-placement]
38 | [ecs-service-with-task-placement] | Starting a container ECS with task
38 | [ecs-service-with-task-placement] | Starting a container ECS with task
38 | [ecs-service-with-task-placement] | Starting a container ECS with task
39 | [ecs-service-with-advanced-alb-config]
39 | [ecs-service-with-advanced-alb-config]
39 | [ecs-service-with-advanced-alb-config]
39 | [ecs-service-with-advanced-alb-config]
39 | [ecs-service-with-advanced-alb-config]
39 | [ecs-service-with-advanced-alb-config]

```



コミット履歴の表示



- コミットビジュアライザーを選択

デベロッパー用ツール > CodeCommit > リポジトリ > blackbelt-demo > コミット

blackbelt-demo

コミット | **コミットビジュアライザー** | コミットの比較

コミットビジュアライザー 情報 nija-at/c#-lambda ▼

a0c58db3	Fixed up per PR comments	10 か月前
04f7902f	feat: c# project with a c# lambda function handler	10 か月前
d6b90452	fix(java/resource-overrides): example uses method toPrettyString() which doesn't...	10 か月前
11e2605b	Upgrade typescript version to resolve implicitly issue (#159)	11 か月前
f4a6096e	update cdk version numbers (#171)	11 か月前
c99faf1a	Merge pull request #188 from zechariahks/feat-jobpoller-static-site-examples	11 か月前
26c9cc04	Merge branch 'master' into feat-jobpoller-static-site-examples	11 か月前
70765eac	Merge pull request #1 from MrArnoldPalmer/pr/188	11 か月前
70537a3f	feat: Format Java Code (#199)	11 か月前
fc47974b	added the static site and stepfunctions java examples info to ReadMe	11 か月前



プルリクエストの作成

プルリクエストとは？

- リポジトリのユーザに対する作業内容の通知
- プルリクエストを起点として、議論やコミュニケーションを機会を提供
- コードレビューや変更箇所の精査ができ、品質改善につなげる

プルリクエストの作成



ターゲットブランチとソースブランチを選択して作成

- タイトルと説明を入力

デベロッパー用ツール > CodeCommit > リポジトリ > PullRequestDemo > プルリクエスト > プルリクエストの作成

プルリクエストの作成

ターゲット: develop | ソース: feature03 | 比較 | キャンセル

Mergeable
現在 feature03 と develop の間に競合はありません。このプルリクエストは AWS CodeCommit コンソールにマージして閉じることができます。

詳細 プルリクエストの作成

タイトル
最大 150 文字

説明 - オプション マークダウンをプレビューする [詳細](#)

変更 | **コミット**

コミット

コミット ID	メッセージのコミット	コミット日	作成者	アクション
30ef3e60	Edited README.md	3 分前	Masahiro Matsumoto	ID をコピーする 参照

キャンセル プルリクエストの作成

デベロッパー用ツール > CodeCommit > リポジトリ > PullRequestDemo > プルリクエスト > 6

6: 機能 #003 を実装しました

プルリクエストを閉じる マージ

未解決 | 承認ルールがありません | マージの競合なし | ターゲット: develop << ソース: feature03 | 作成者: | 承認: 0

[詳細](#) | [アクティビティ](#) | [変更](#) | [コミット](#) | [承認](#)

詳細

詳細の編集

このプルリクエストには説明がありません。

プルリクエストへのコメント

以下に対してコメントを追加可能

- プルリクエスト
- ファイル
- コードの行

絵文字によるリアクションも可能に



プルリクエストのマージ



3つのマージ戦略をサポート

- fast-forward
- 3-way merge
- Squash merge

デベロッパー用ツール > CodeCommit > リポジトリ > PullRequestDemo > プルリクエスト > 6 > マージ

プルリクエストのマージ 6: 機能 #003 を実装しました


リクエスト詳細のマージ

プルリクエスト: 6 機能 #003 を実装しました


ターゲット develop << ソース feature03

マージ戦略 情報
現在のプルリクエストがターゲットブランチにマージされる方法を決定します


早送りマージ
`git merge --ff-only`
ブランチをマージし、送信先ブランチポイントを送信元ブランチの先端に移動します。これは Git でのデフォルトのマージ戦略です。



スカッシュしてマージ
`git merge --squash`
ソースブランチからのすべてのコミットをターゲットブランチで1つのマージコミットに結合します。



3ウェイマージ
`git merge --no-ff`
マージコミットを作成し、個別のソースコミットをターゲットブランチに追加します。



マージ後にソースブランチ feature03 を削除しますか？

キャンセル **プルリクエストのマージ**

プルリクエストの承認ルールワークフロー



コードをマージする前に満たさなければならないルール要件を定義し、高品質のコード変更のみがマージされるようにすることに役立つ

- 承認の総数
- 特定ユーザーからの承認

任意のリポジトリ、ブランチへ適用可能

デベロッパーツール > CodeCommit > 承認ルールテンプレート > 承認ルールテンプレートを作成

承認ルールテンプレートを作成

承認ルールテンプレート

承認ルールテンプレート
ApprovalRule 01

説明 - オプション
シニアプログラマー2名の承認が必要

必要な承認の数
2

承認グループのメンバー - オプション
承認グループのメンバーが指定されている場合、これらのメンバーからの承認のみがこのルールを満たすための条件としてカウントされます。ワイルドカードを使用して、1つの承認を持つ複数の承認者を指定できます。

承認者のタイプ	情報	値	
MAM ユーザー名または引き受けた...		SeniorProgrammer_*	削除

追加

ブランチフィルター - オプション
ブランチフィルターを使用して、特定のブランチ名がフィルターリスト内の名前と一致する場合はのみ、このテンプレートをプルリクエストに適用します。

ブランチ名
develop

削除

追加

▼ 関連付けられたリポジトリ

リポジトリ - オプション

PullRequestDemo X

キャンセル 作成

Amazon CodeGuru Reviewer との連携



- CodeGuru Reviewer と連携したレビューの実施も可能
- 詳細は 2020/8/4 の Amazon CodeGuru を参照

aws

[AWS Black Belt Online Seminar]

Amazon CodeGuru

サービスカットシリーズ

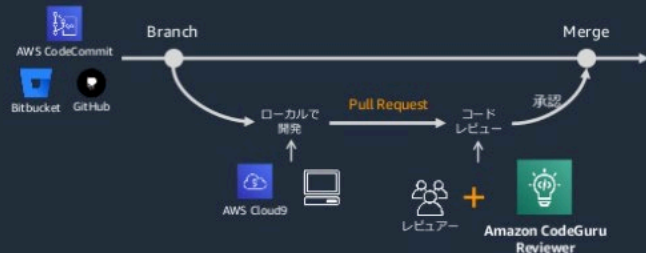
Solutions Architect
Yumiko Kanasugi
2020/8/4

AWS 公式 Webinar <https://amzn.to/JPWebinar>

過去資料 <https://amzn.to/JPArchive>

aws

CodeGuru Reviewer の位置付け



Amazon CodeGuru Reviewer でコードレビューの負担を軽減

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.



AWS CloudTrail を利用した API コールログ取得



AWS CodeCommit は、AWS CloudTrailと連携可能

- コンソール、git クライアント、AWS CLI から発行される CodeCommit API をキャプチャーしてS3 バケットに保存可能

```
{
  "eventVersion":"1.05",
  "userIdentity": {
    "type":"IAMUser", "principalId":"AIDACKCEVSQ6C2EXAMPLE",
    "arn":"arn:aws:iam::444455556666:user/Mary_Major", "accountId":"444455556666",
    "accessKeyId":"AKIAIOSFODNN7EXAMPLE", "userName":"Mary_Major"
  },
  "eventTime":"2016-12-14T17:57:36Z",
  "eventSource":"codecommit.amazonaws.com",
  ....
}
```



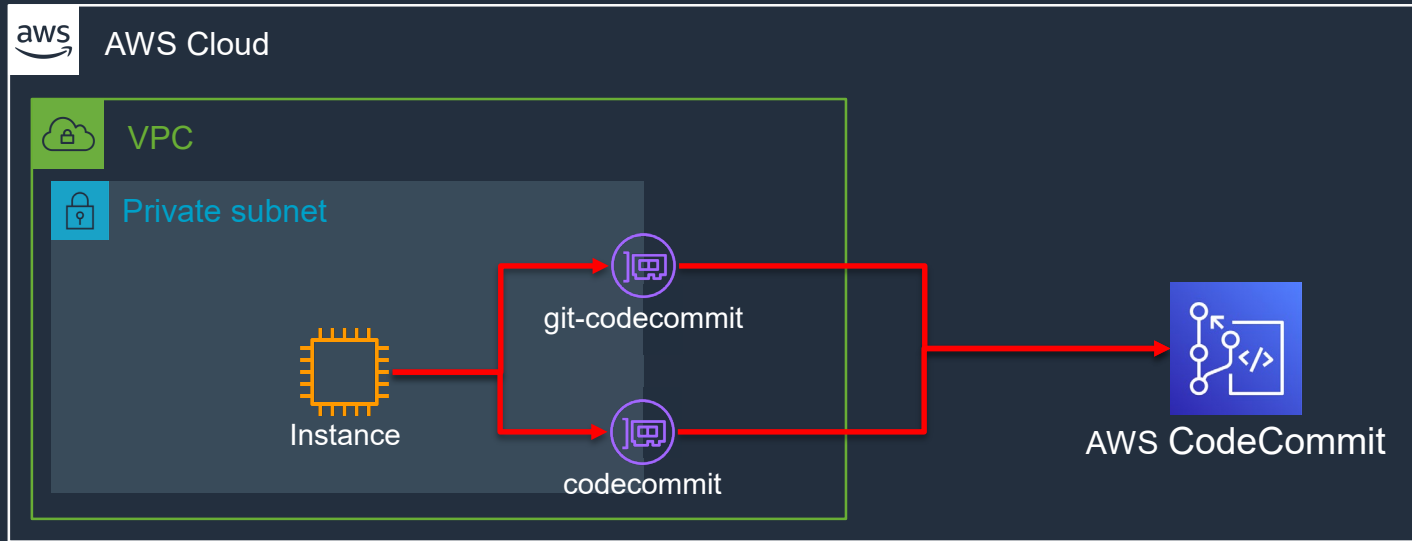
ブランチレベルのアクセス許可

- リポジトリを変更できるユーザーの制御だけでなく、リポジトリ内のブランチを変更できるユーザーを制御可能
- IAM ポリシーの Condition で操作の行えるブランチを制限
- 開発チーム以外にはプルリクを master ブランチへ マージさせないなど、ブランチへのアクセスを柔軟に制御できる

VPC エンドポイントをサポート



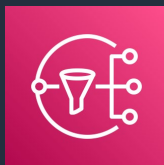
- Git 操作と CodeCommit API 操作の2種類のエンドポイント
- 連邦情報処理標準(FIPS) 準拠オプションもあり



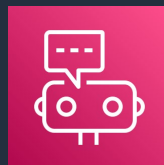


リポジトリイベントの通知

- 従来のトリガーよりも細かな条件設定が可能
 - プルリクエスト、承認、ブランチ・タグ操作
- Amazon SNS によるトピックの通知
- AWS Chatbot と連携した Slack への通知
- 詳細は以下を参照
 - <https://aws.amazon.com/jp/blogs/news/receive-aws-developer-tools-notifications-over-slack-using-aws-chatbot/>

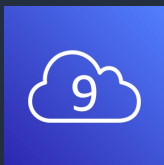


Amazon Simple
Notification Service

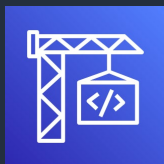


AWS Chatbot

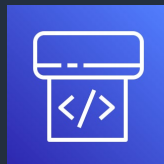
AWS サービスとの連携



AWS Cloud9



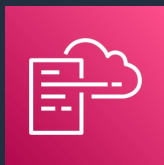
AWS CodeBuild



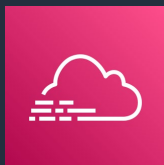
AWS CodePipeline



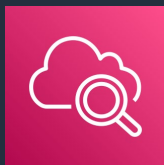
AWS CodeStar



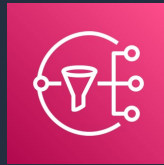
AWS CloudFormation



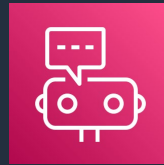
AWS CloudTrail



Amazon CloudWatch



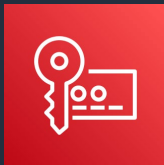
Amazon Simple
Notification Service



AWS Chatbot



AWS Amplify



AWS Key
Management Service



AWS Elastic Beanstalk



AWS Lambda



Amazon CodeGuru

AWS CodeCommit 制限事項



リポジトリの数	アカウントごとに1,000まで
単一のPushの参照数	最大4,000（create, delete, update を含む）。リポジトリ内の全体の参照数は無制限。
リポジトリ名	1~100文字まで。 .gitで終わる名前をつけることは出来ない。以下の文字は含むことができない。 ! ? @ # \$ % ^ & * () + = { } [] ¥ / > < ~ ` ' " ; :
Git blob サイズ	ファイルの数や単一のコミットでのファイルの合計サイズは無制限。メタデータは6MB以下、単一のblobファイルのサイズは2GBまで。
Commit Visualizer の ブランチ数	35 ブランチ/ページ。

料金



アクティブユーザー : コンソールや CLI、SDK を使ってアクセスする AWS Identity (例 : IAM ユーザーや IAM ロールなど) のこと。

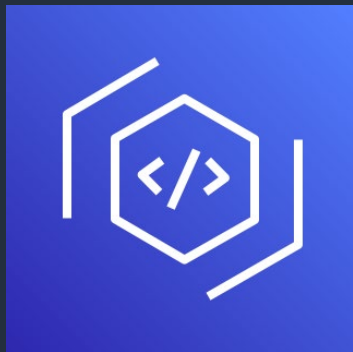
最初の 5 アクティブ ユーザーまで	最初の 5 を超えるアクティブ ユーザー (1 人当たり)
無料	1ドル/月
<ul style="list-style-type: none">無制限のリポジトリ数月に 50GB のストレージ10,000 Git リクエスト/月	<ul style="list-style-type: none">無制限のリポジトリ数月に 10GB のストレージ/ユーザー2,000 Git リクエスト/月/ユーザー

上記の制限を超えた場合

- ストレージ : \$0.06 / GB / 月
 - Git リクエスト : \$0.001 / リクエスト / 月
- が費用としてかかる。

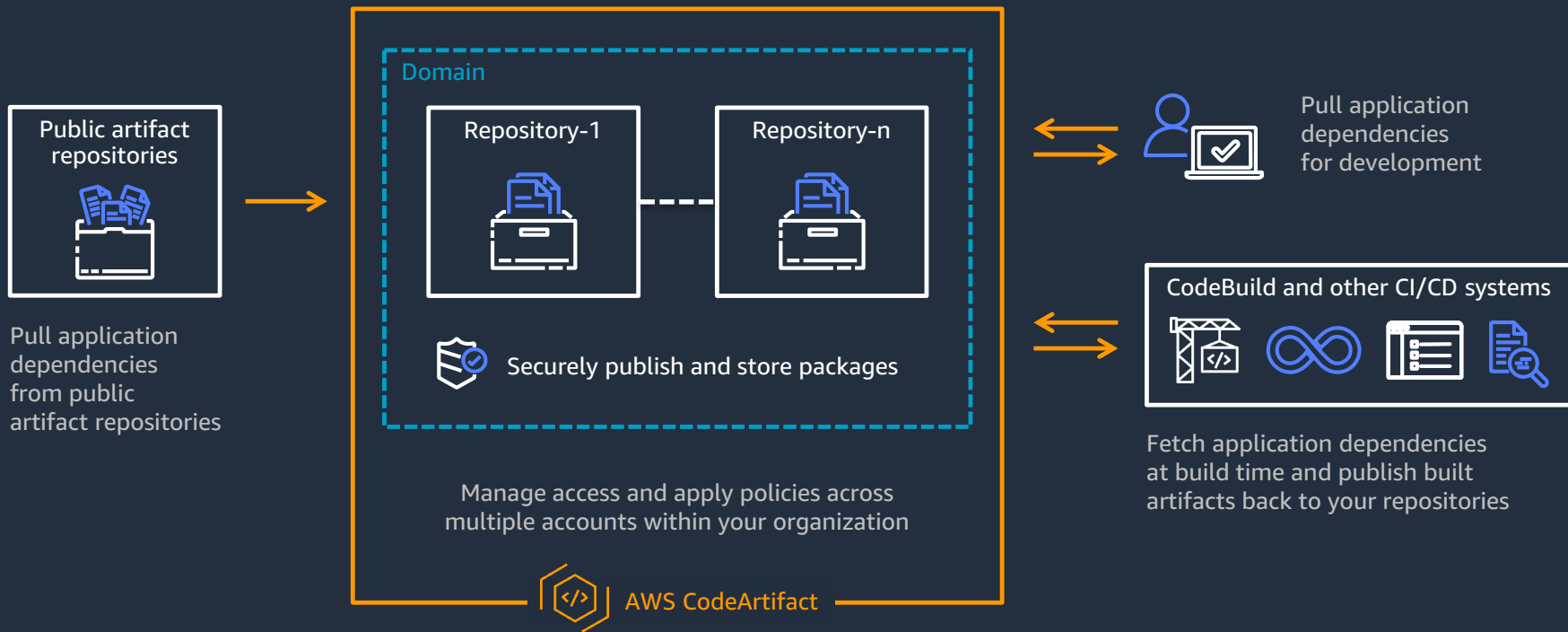
AWS CodeArtifact

AWS CodeArtifact



- ✓ **セキュア**で**スケーラブル**な**マネージドアーティファクト管理**
- ✓ Maven, Gradle, npm, yarn, twine, pip などの一般的なビルドツール、パッケージマネージャーから利用可能
- ✓ npm パブリックリポジトリ、Maven Central, Python Package Index (PyPi) などからパッケージ取得可能
- ✓ AWS Key Management Service (KMS) による暗号化をサポート
- ✓ リポジトリはポリグロットでサポートされている任意のパッケージを含めることができる
- ✓ VPC エンドポイントをサポート

AWS CodeArtifact 概要





ドメイン

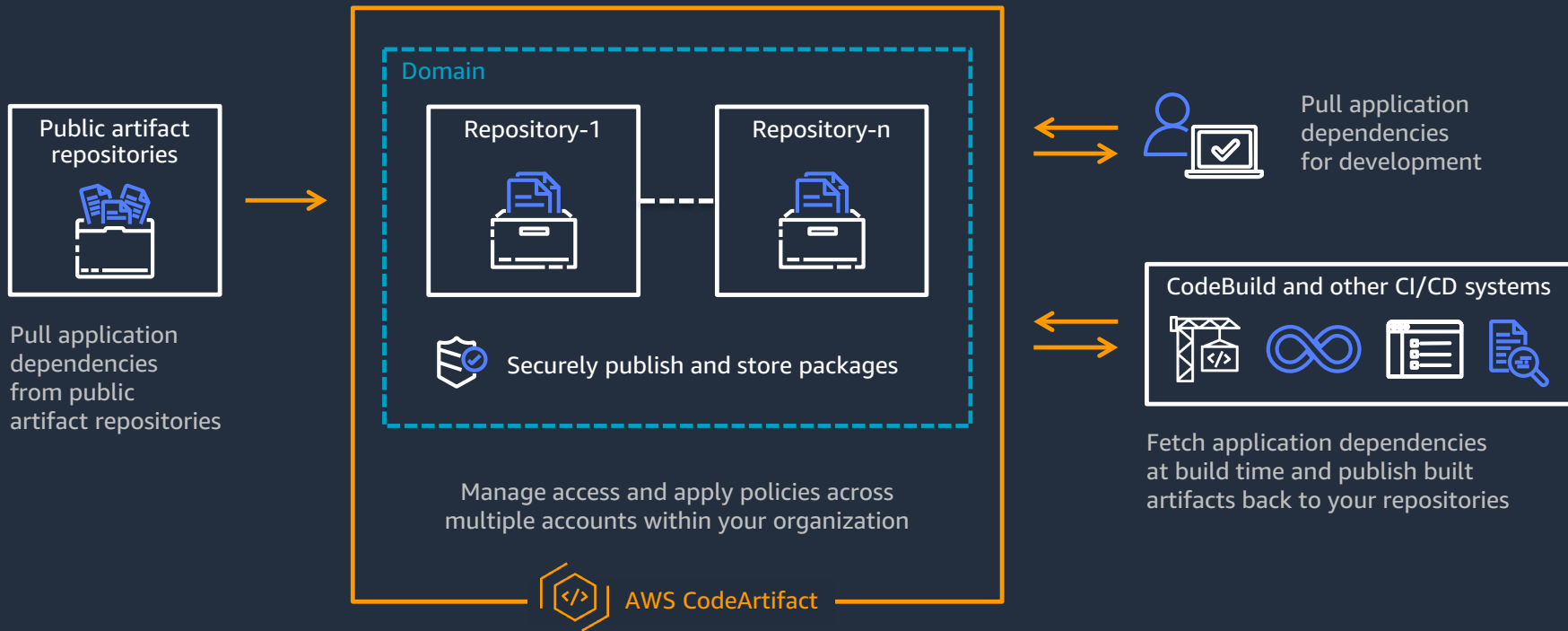
- パッケージとメタデータを格納する単位
 - パッケージが複数のリポジトリに存在していても、**保存はドメインごとに一度だけ**
 - ドメイン内の全てのアセット、メタデータは単一のキーで暗号化
- リポジトリを集約する
 - クロスアカウントに対応、別アカウントのリポジトリも格納可能
- 複数作成できるが、単一の本番用とテスト用を作成することを推奨
- アカウント内で一意であること



リポジトリ

- ドメインのメンバー
 - 必ず1つのドメインに所属する
 - 作成後に他のドメインへ移動できない
- **ドメインに格納されたパッケージにアクセスするためのエンドポイント**
- パッケージバージョンのセットが含まれる
- ポリグロット（多言語対応）
- リポジトリは他のリポジトリのアップストリームになることができる

AWS CodeArtifact 概要 (再掲)



リポジトリの作成



以下を入力

- リポジトリ名
- リポジトリの説明
- パブリックアップストリーム

デベロッパー用ツール > CodeArtifact > リポジトリ > リポジトリを作成

Step 1
リポジトリを作成

Step 2
ドメインを選択

Step 3
確認および作成

リポジトリを作成 情報

リポジトリ






リポジトリ名
リポジトリ名には、通常、プロジェクト名またはチーム名が含まれます。

英数字で始まる必要があります。使用できる文字は、英数字、アンダースコア (_)、ダッシュ (-)、またはヒリオド (.) です。

リポジトリの説明 - オプション

1000 文字以内

パブリックアップストリームリポジトリ - オプション
CodeArtifact パブリックアップストリームリポジトリは、Maven Central Repository や NPM などの公式パッケージ機関にリポジトリを接続する中間リポジトリです。

<input type="checkbox"/>	 maven-central-store Provides Maven artifacts from Maven Central Repository.
<input type="checkbox"/>	 google-android-store Provides Maven artifacts from Google Android.
<input type="checkbox"/>	 gradle-plugins-store Provides Maven artifacts from Gradle plugins.
<input checked="" type="checkbox"/>	 npm-store Provides npm artifacts from npm, Inc.
<input checked="" type="checkbox"/>	 pypi-store Provides PyPI artifacts from PyPA.

キャンセル

リポジトリの作成



以下を入力

- ドメインのアカウント
- ドメイン名
- 暗号化に利用するキー

デベロッパーツール > CodeArtifact > リポジトリ > リポジトリを作成

Step 1
リポジトリを作成

Step 2
ドメインを選択

Step 3
確認および作成

ドメインを選択 情報

ドメイン
ドメインは、CodeArtifact で作成されたリポジトリのグループです。各リポジトリは単一のドメインに属します。

AWS アカウント
CodeArtifact ドメインを所有する AWS アカウントを選択します。ドメインとリポジトリは、現在 アジアパシフィック (東京) に設定されているのと同じ AWS リージョン内に存在している必要があります。

この AWS アカウント () 異なる AWS アカウント

このアカウントにはドメインがありません
このリージョン アジアパシフィック (東京) にドメインがありません。すぐにドメインまたは アカウント全体での CodeArtifact ドメインの使用に関する詳細 [📄](#) を作成します。

ドメイン名
ドメイン名には連字、会社名が含まれます。

my-domain

英数字で始まる必要があります。使用できる文字は、小文字の英数字またはダッシュ (-) です。

ドメイン URL
my-domain-1.codeartifact.ap-northeast-1.amazonaws.com

追加設定
カスタマーマスターキー

カスタマーマスターキー
CodeArtifact が提供する AWS マネージド型 CMK を使用するか、独自の CMK を指定して、ドメインのアーティファクトを暗号化します。

AWS マネージド型キー
キーエイリアス「aws/codeartifact」を使用してアカウントに作成されたキー。

カスタマーマネージド型キー
ユーザーが作成および管理するキー。

キャンセル 前の **次へ**

リポジトリの作成

入力内容の確認

デベロッパー用ツール > CodeArtifact > リポジトリ > リポジトリを作成

Step 1
リポジトリを作成

Step 2
ドメインを選択

Step 3
確認および作成

確認および作成 情報

パッケージフロー

パッケージが my-domain を介して外部接続から my-repository に流れる方法を確認します。

```
graph LR; subgraph ExternalConnections [外部接続]; direction TB; A[publicnpmjs]; B[publicpypi]; end; subgraph UpstreamRepositories [アップストリームリポジトリ]; direction TB; C[npm-store]; D[pypi-store]; end; subgraph Domain [ドメイン: my-domain]; direction TB; E[リポジトリ my-repository]; end; A --> C; B --> D; C --> E; D --> E;
```

ステップ 1: リポジトリを作成 編集

リポジトリ

リポジトリ名 my-repository	リポジトリの説明 My first repository
アップストリームリポジトリ npm-store, pypi-store	

ステップ 2: ドメインを選択 編集

ドメイン

ドメイン名 my-domain	カスタマーマスターキー AWS マネージド型キー (alias/aws/codeartifact)
AWS アカウント番号	

キャンセル 前の リポジトリを作成

リポジトリの作成



作成したリポジトリと、アップストリームを表す追加リポジトリがドメイン内に作成される

デベロッパー用ツール > CodeArtifact > ドメイン > my-domain

my-domain 情報 削除 ドメインポリシーを適用

ドメイン

▶ 詳細
ドメイン所有者、ポリシー、暗号化キー、ARN および保存されたデータ。

リポジトリ (3) リポジトリを削除 接続手順の表示 リポジトリを作成

🔍 < 1 > ⚙️

	リポジトリ名	リポジトリの説明	リポジトリ管理者
<input type="radio"/>	my-repository	My first repository	
<input type="radio"/>	npm-store	Provides npm artifacts from npm, Inc.	
<input type="radio"/>	pypi-store	Provides PyPI artifacts from PyPA.	

パッケージマネージャの設定



パッケージマネージャ毎の設定方法をマネジメントコンソールで確認可能

- AWS CLI を使用した設定を**推奨**
- トークンの**有効期間は 12 時間**

接続手順

△ このリポジトリに接続する前に、AWS CLI をインストールして AWS 認証情報を設定する必要があります。AWS CodeArtifact は以下の CLI バージョンでサポートされています。

- 1.18.83 以降。AWS CLI バージョン 1 をインストール [🔗](#)。
- 2.0.21 以降。AWS CLI バージョン 2 をインストール [🔗](#)。

npm

▼ 推奨設定: AWS CLI を使用して設定する

1. この AWS CLI CodeArtifact コマンドを使用して npm クライアントを設定します (ログイン認証は 12 時間で失効します)。

```
aws codeartifact login --tool npm --repository my-repository --domain my-domain
```

📄 コピー

▶ 手動設定: リポジトリにプッシュしてリポジトリから取得する

完了

```
aws codeartifact login --tool npm ¥  
  
--repository ${リポジトリ名} ¥  
  
--domain ${ドメイン名} ¥  
  
--domain-owner ${ドメインのアカウントID}
```

パッケージの取得



例：AWS Cloud Development Kit(AWS CDK) のインストール

```
npm install -g aws-cdk
```

AWS CDK と依存するパッケージが npm パブリックリポジトリからダウンロードされ、リポジトリに追加される

デベロッパー用ツール > CodeArtifact > リポジトリ > my-repository

my-repository 情報 削除 リポジトリポリシーを適用 編集

リポジトリ My first repository

▶ 詳細
ドメイン、ポリシー、ARN、およびアップストリームリポジトリ。

パッケージ 接続手順の表示

🔍 < 1 > ⚙️

パッケージ	パッケージタイプ	最新バージョン
表示するパッケージがありません。		

まず、ローカルのパッケージマネージャーを設定して、このテーブルのパッケージを表示します。

接続手順の表示



デベロッパー用ツール > CodeArtifact > リポジトリ > my-repository

my-repository 情報 削除 リポジトリポリシーを適用 編集

リポジトリ My first repository

▶ 詳細
ドメイン、ポリシー、ARN、およびアップストリームリポジトリ。

パッケージ 接続手順の表示

🔍 < 1 2 3 4 5 ... > ⚙️

パッケージ	パッケージタイプ	最新バージョン
agent-base	npm	4.2.1
ajv	npm	6.12.5
ansi-styles	npm	4.2.1
archiver	npm	5.0.2

パッケージの取得



例：Pillow のインストール

```
pip3 install Pillow
```

CodeArtifact は **ポリグロット(多言語対応)**
サポートされる任意のタイプのパッケージを格納できる

The screenshot shows the AWS CodeArtifact console interface for a repository named 'my-repository'. The breadcrumb navigation is 'デベロッパー用ツール > CodeArtifact > リポジトリ > my-repository'. The repository name 'my-repository' is displayed with a '情報' (Info) link and buttons for '削除' (Delete), 'リポジトリポリシーを適用' (Apply Repository Policy), and '編集' (Edit). Below this, there is a section for 'リポジトリ' (Repository) with the name 'My first repository'. A '詳細' (Details) section is expanded, showing 'ドメイン、ポリシー、ARN、およびアップストリームリポジトリ。' (Domain, policy, ARN, and upstream repository). The 'パッケージ' (Packages) section is active, showing a search bar with 'pi' and a '接続手順の表示' (Show connection steps) button. Below the search bar is a table of packages:

パッケージ	パッケージタイプ	最新バージョン
pify	npm	3.0.0
pillow	pypi	7.2.0

パッケージの登録



例：独自のパッケージ

```
npm publish
```

デベロッパー用ツール > CodeArtifact > リポジトリ > my-repository

my-repository 情報

[削除](#) [リポジトリポリシーを適用](#) [編集](#)

リポジトリ My first repository

▶ **詳細**
ドメイン、ポリシー、ARN、およびアップストリームリポジトリ。

パッケージ

[接続手順の表示](#)

< 1 > ⚙️

パッケージ	パッケージタイプ	最新バージョン
my-package	npm	2.0.0

デベロッパー用ツール > CodeArtifact > リポジトリ > my-repository > my-package バージョン

my-package バージョン 情報

[削除](#)

< 1 > ⚙️

パッケージのバージョン	パッケージのステータス
<input type="radio"/> 2.0.0	Published
<input type="radio"/> 1.0.1	Published
<input type="radio"/> 1.0.0	Published



AWS CodeBuild からの利用例

CodeBuild でのビルド時にも利用可能

CodeBuild のサービスロールに以下の権限が必要

```
pre_build:
  commands:
    - aws codeartifact login --tool npm ...
build:
  commands:
    - npm install
```

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [ "codeartifact:GetAuthorizationToken",
                  "codeartifact:GetRepositoryEndpoint",
                  "codeartifact:ReadFromRepository"
                ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "sts:GetServiceBearerToken",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "sts:AWSServiceName": "codeartifact.amazonaws.com"
        }
      }
    }
  ]
}
```

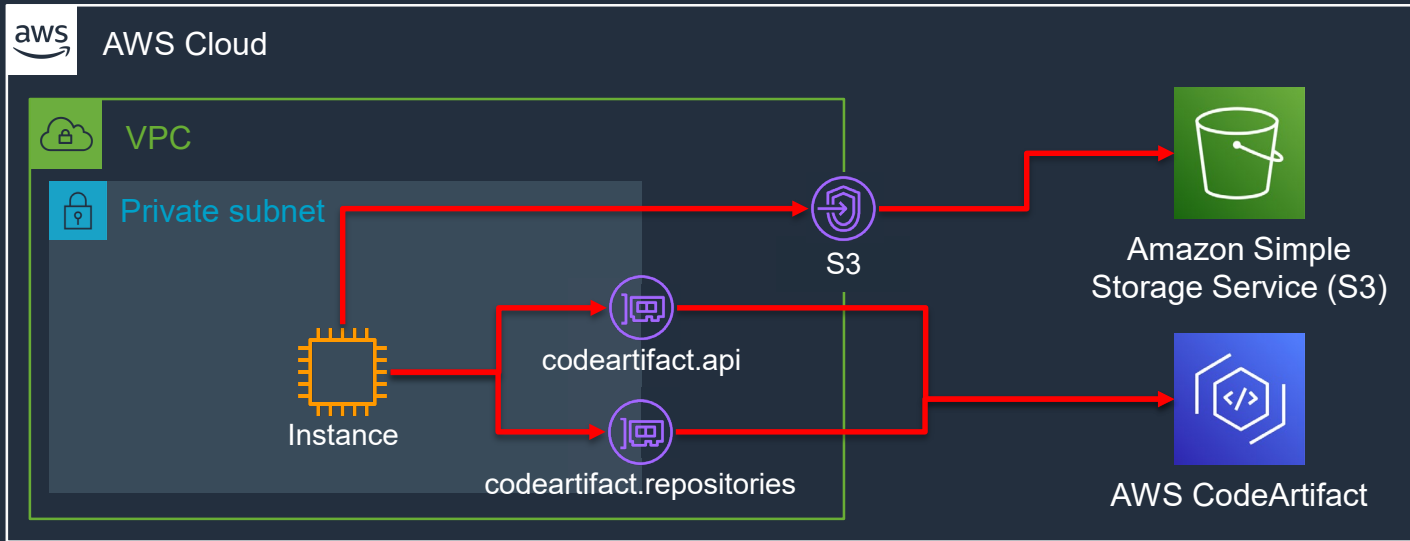
詳細は以下を参照 :

https://docs.aws.amazon.com/ja_jp/codeartifact/latest/ug/using-npm-packages-in-codebuild.html

VPC エンドポイントをサポート



VPC エンドポイント経由での利用には CodeArtifact だけでなく、**S3 の VPC エンドポイントも必要**





VPC エンドポイントをサポート

VPC エンドポイント経由で CodeArtifact を利用する際は、パッケージマネージャの設定時に**エンドポイント URL の指定が必要**

```
aws codeartifact login --tool npm ¥  
  
--repository ${リポジトリ名} ¥  
  
--domain ${ドメイン名} ¥  
  
--domain-owner ${ドメインのアカウントID} ¥  
  
--endpoint-url ${VPC エンドポイント}
```


AWS CloudTrail を利用した APIコールログ取得



GetAuthorizationToken API の例

```
GetAuthorizationToken API
{
  "eventVersion": "1.05",
  "userIdentity": { "type": "AssumedRole", "principalId": "AIDACKCEVSQ6C2EXAMPLE", ...,
    "sessionContext": {
      "attributes": {...}, "sessionIssuer": {...}
    }
  },
  "eventSource": "codeartifact.amazonaws.com",
  "eventName": "GetAuthorizationToken",
  ...
}
```

詳細は以下を参照 :

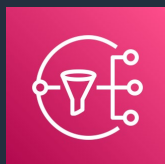
https://docs.aws.amazon.com/ja_jp/codeartifact/latest/ug/codeartifact-information-in-cloudtrail.html

Amazon EventBridge との連携



リポジトリ内のイベント（新しいパッケージバージョンの作成など）をトリガーに処理を自動化できる

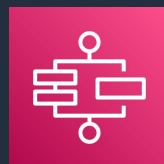
- Amazon SNS による通知
- AWS Lambda や AWS Step Functions の起動
- AWS CodePipeline の起動



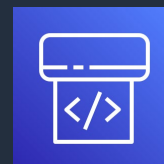
Amazon Simple
Notification Service



AWS Lambda



AWS Step Functions



AWS CodePipeline

AWS CodeArtifact の制限事項

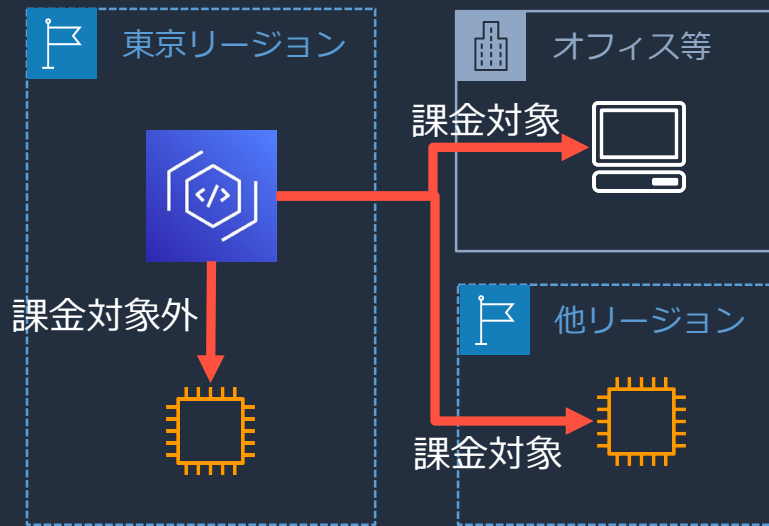


ドメインの数	アカウントごとに 10 個まで
リポジトリの数	ドメインごとに 1,000 個まで
アセットの数	パッケージバージョンごとに 100 個まで
アセットのサイズ	最大 1 GB
パッケージメタデータ ファイルのサイズ	最大 100 KB
設定できるアップスト リームリポジトリの数	リポジトリ毎に 10 個まで

利用料金

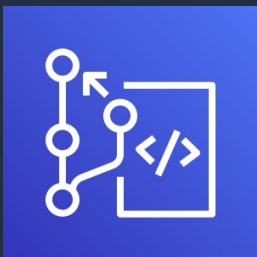


- 料金（東京リージョン）
 - ストレージサイズ
 - \$0.055/GB
 - リクエスト数
 - \$0.06/10,000 リクエスト
 - データ転送量（アウトバウンド）
 - インターネットへ \$0.144/GB
 - 別リージョンへ \$0.09/GB
- 無料枠
 - 2GB のストレージ（月）
 - 100,000 回分のリクエスト（月）



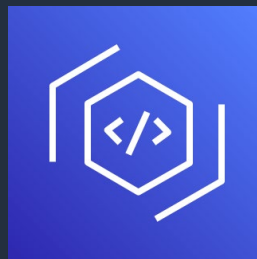
まとめ

まとめ



AWS CodeCommit は、

- セキュア
- スケーラブル
- フルマネージドマネージド
- Git 互換ソース管理



AWS CodeArtifact は、

- セキュア
- スケーラブル
- フルマネージドマネージド
- アーティファクト管理

初期費用無料、従量課金で簡単に使い始めることができる

Q&A

お答えできなかったご質問については

AWS Japan Blog 「<https://aws.amazon.com/jp/blogs/news/>」にて

後日掲載します。

AWS の日本語資料の場所「AWS 資料」で検索



日本担当チームへお問い合わせ サポート 日本語 ▾ アカウント ▾

コンソールにサインイン

製品 ソリューション 料金 ドキュメント 学習 パートナー AWS Marketplace その他 🔍

AWS クラウドサービス活用資料集トップ

アマゾン ウェブ サービス (AWS) は安全なクラウドサービスプラットフォームで、ビジネスのスケールと成長をサポートする処理能力、データベースストレージ、およびその他多種多様な機能を提供します。お客様は必要なサービスを選択し、必要な分だけご利用いただけます。それらを活用するために役立つ日本語資料、動画コンテンツを多数ご提供しております。(本サイトは主に、AWS Webinar で使用した資料およびオンデマンドセミナー情報を掲載しています。)

[AWS Webinar お申込 »](#)

[AWS 初心者向け »](#)

[業種・ソリューション別資料 »](#)

[サービス別資料 »](#)

<https://amzn.to/JPArchive>



AWS Well-Architected 個別技術相談会

毎週“W-A個別技術相談会”を実施中

- AWSのソリューションアーキテクト(SA)に
対策などを相談することも可能

• 申込みはイベント告知サイトから

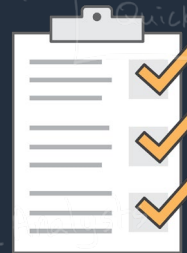
(<https://aws.amazon.com/jp/about-aws/events/>)

AWS イベント

で[検索]



AWS Well-Architected



ご視聴ありがとうございました

AWS 公式 Webinar
<https://amzn.to/JPWebinar>



過去資料
<https://amzn.to/JPArchive>

