



# [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

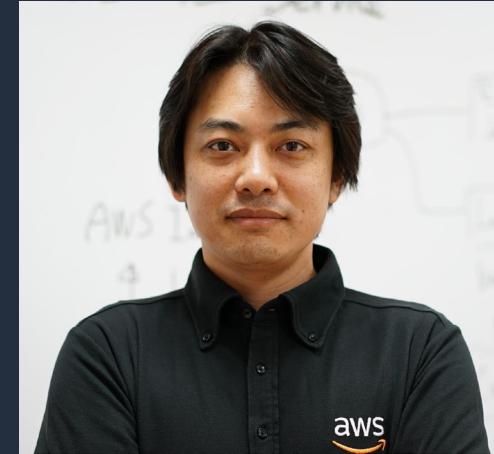
松本 雅博（まつもと まさひろ）

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



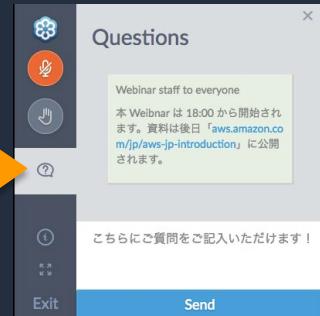
# AWS Black Belt Online Seminar とは

「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、Amazon 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います。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.

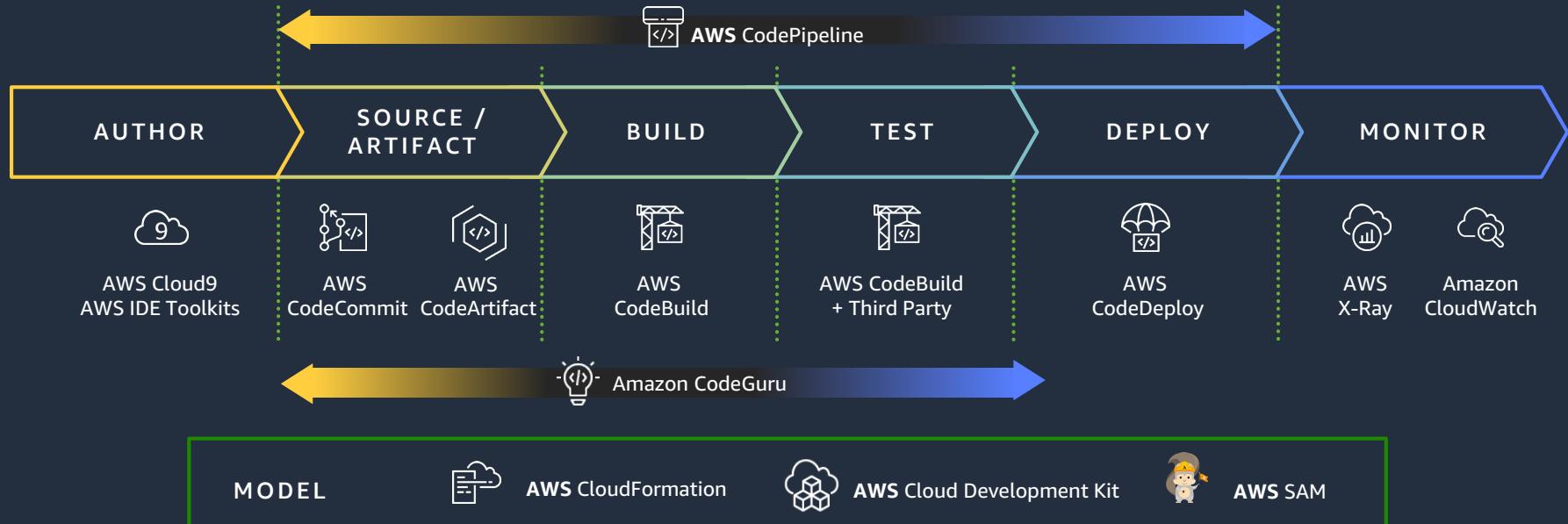
# 本セミナーの概要

- 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する AWS サービス
- AWS CodeCommit
- AWS CodeArtifact
- まとめ

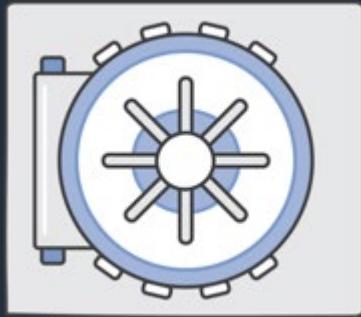
# ソフトウェア開発に関する AWS サービス



# 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



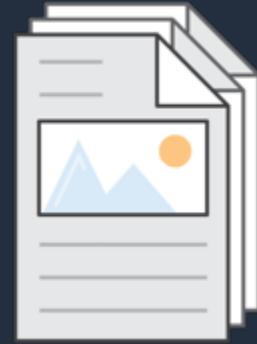
Secure



Fully  
managed



High  
availability



Store  
anything

# AWS CodeCommit

# AWS CodeCommit



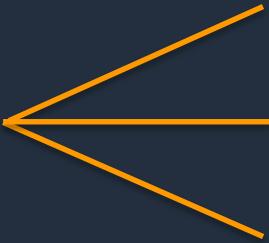
- ✓ セキュアでスケーラブルなマネージドGit互換ソース管理
- ✓ 標準的な Git ツールが利用可能
- ✓ Amazon S3 のスケーラビリティ、可用性、堅牢なストレージを利用
- ✓ 利用者固有のキーを使用した暗号化
- ✓ レポジトリサイズの上限なし
- ✓ コンテンツの直接編集をサポート
- ✓ プルリクエスト機能に対応
- ✓ VPC エンドポイントをサポート

# AWS CodeCommit 概要



SSH or HTTPS

git pull/push



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  
  

説明 - オプショナル  
 最大 1,000 文字

タグ

Amazon CodeGuru Reviewer for Java を有効にする - オプショナル  
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。

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 のクレデンシャルを利用する)
- 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 Identity and Access Management (IAM) console. On the left, the navigation pane is visible 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 'ユーザー' option is currently selected. The main content area displays details for a user named 'developer01'. The ARN is listed as 'arn:aws:iam:::user/developer01'. The password field is empty ('/'). The creation date is '2020-10-16 16:25 UTC+0900'. Below this, there are tabs for 'アクセス権限', 'グループ', 'タグ (1)', '認証情報' (which is highlighted with a red box), and 'アクセスアドバイザー'. A sub-section titled 'サインイン認証情報' is shown.

# 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

Identity and Access Management (IAM)

結果がありません

### 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**

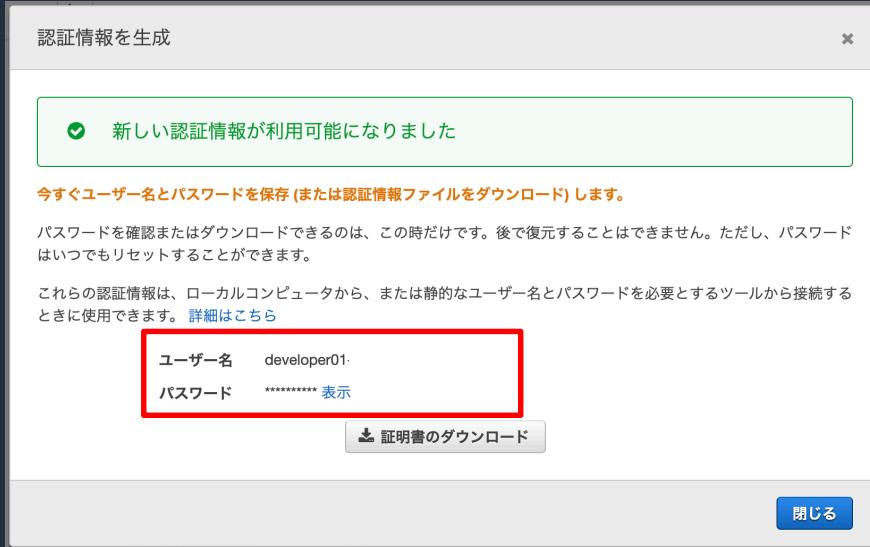
認証情報が生成されていません。

Amazon Keyspaces (for Apache Cassandra) の認証情報

# 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	ステータス	作成
developer01-a	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成できます。[詳細はこちら](#)

無効化  
パスワードのリセット  
削除

# 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**

▼ 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 C 通知 URL のクローン リポジトリの表示 リポジトリの削除 リポジトリを作成

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	<span>HTTPS</span> <span>SSH</span> <span>HTTPS (GRC)</span>
<input type="radio"/> myfirstrepo	My first repository	9 時間前	<span>HTTPS</span> <span>SSH</span> <span>HTTPS (GRC)</span>

デベロッパー用ツール ×

**CodeCommit**

▼ ソース・CodeCommit

- 開始方法
- リポジトリ
- コード**
- プルリクエスト

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

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

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



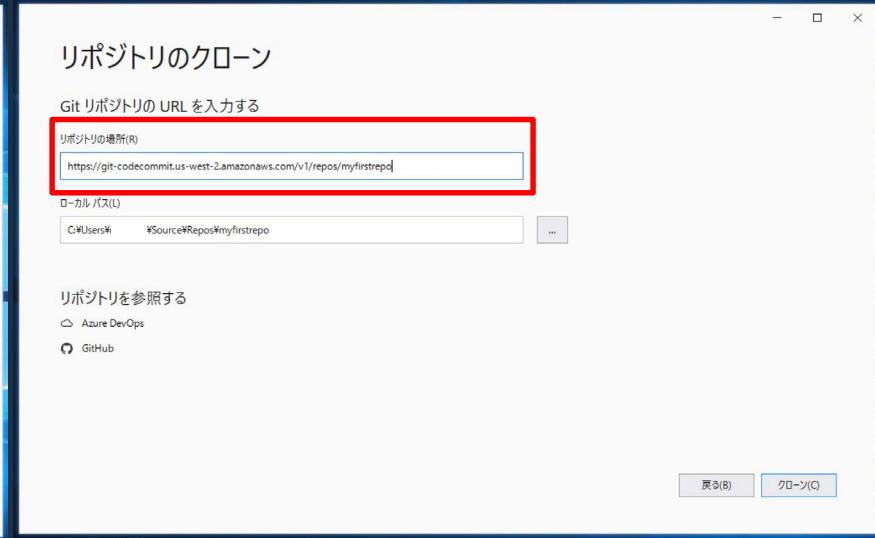
- 作成した認証情報を利用して接続

```
$ 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



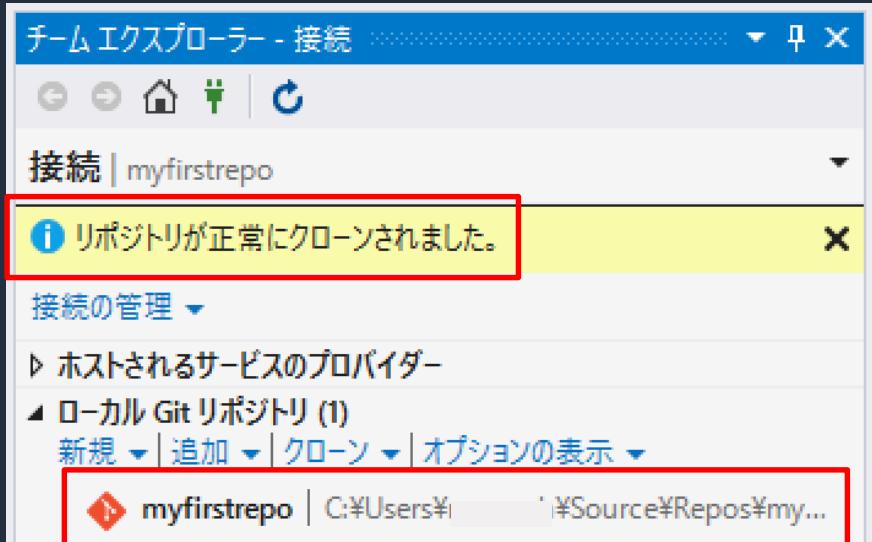
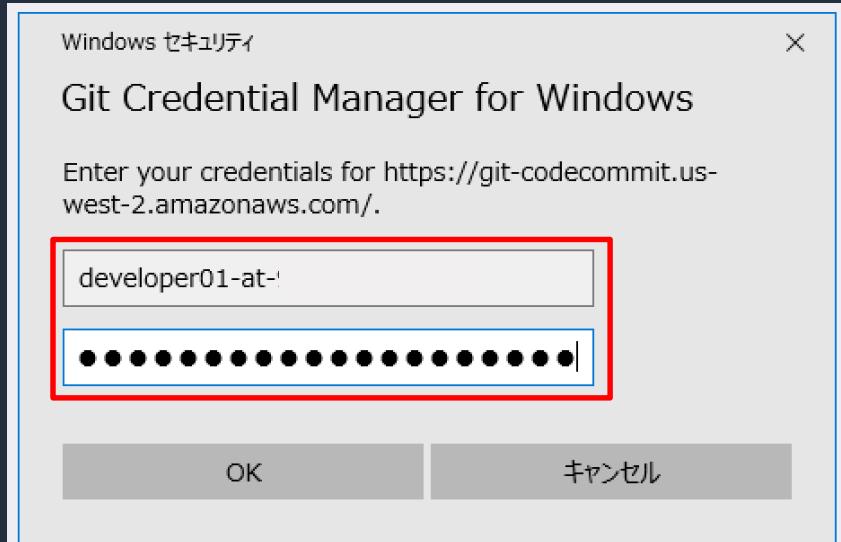
- リポジトリのクローンをクリックし、 URL を入力する



# Visual Studio からの接続



- 認証情報を入力する

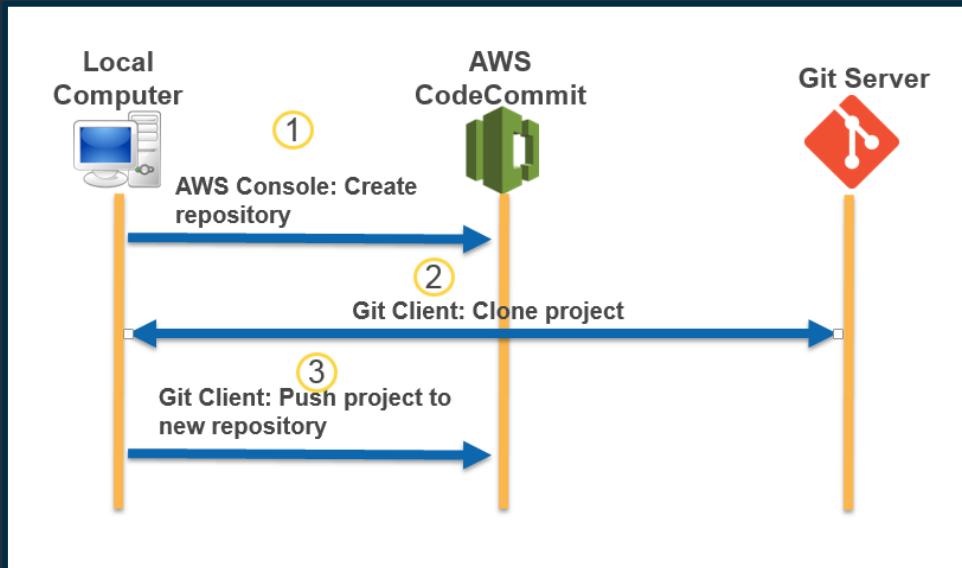




# 他の Git リポジトリからのマイグレーション

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ヤ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

The screenshot shows the AWS CodeCommit console interface. On the left, there's a navigation sidebar with sections like 'デベロッパー用ツール' (Developer Tools), 'CodeCommit', 'ソース・CodeCommit' (Sources - CodeCommit), '開始方法' (Getting Started), 'リポジトリ' (Repository) which is highlighted in orange, '承認ルールテンプレート' (Approval Rule Template), 'アーティファクト' (Artifact), 'CodeArtifact', and 'ビルド・CodeBuild'. The main area is titled 'デベロッパー用ツール > CodeCommit > リポジトリ' and shows a list of repositories. The first repository, 'blackbelt-demo', has its name highlighted with a red box. The table columns are '名前' (Name), '説明' (Description), '最終更新日時' (Last Updated), and 'URL のクローン' (Clone URL). Each row provides three clone options: HTTPS, SSH, and HTTPS (GRC). The second repository listed is 'myfirstrepo'.

名前	説明	最終更新日時	URL のクローン
blackbelt-demo	BlackBelt 用	8 時間前	<a href="#">HTTPS</a> <a href="#">SSH</a> <a href="#">HTTPS (GRC)</a>
myfirstrepo	My first repository	10 時間前	<a href="#">HTTPS</a> <a href="#">SSH</a> <a href="#">HTTPS (GRC)</a>

# 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 ブルリクエストの作成 URL のクローン ファイルの追加

blackbelt-demo / typescript / amplify-console-app 情報

名前

..

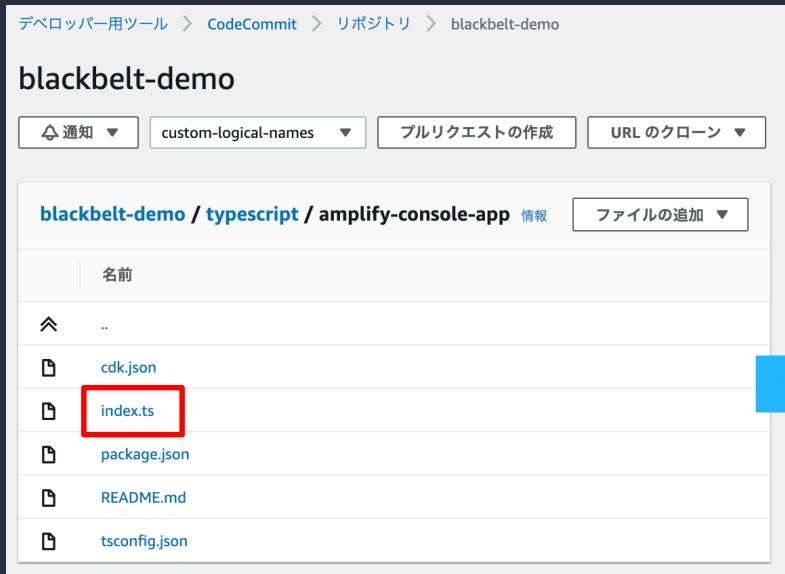
cdk.json

**index.ts**

package.json

README.md

tsconfig.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 ブルリクエストの作成 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  }
20}
```



# 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 ▾ ブラウザの作成 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) {
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', {
```

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 - オプショナル  
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

名前

- .github
- csharp
- java
- python
- scripts
- typescript

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



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

blackbelt-demo 情報

custom-logical-names

ファイルのアップロード

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 links for CodeCommit, including 'ソース' (Sources), 'コミット' (Commits) which is highlighted with a red box, and 'ブランチ' (Branches). The main area displays a table of commits with columns for Commit ID, Message, Date, Author, and Actions. The dropdown menu at the top right of the table is set to 'custom-logical-names'. A large blue arrow points from this dropdown to a detailed view of the dropdown menu on the right, which lists 'custom-logical-names' and 'master'.

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		<a href="#">ID をコピーする</a> <a href="#">参照</a>
42f652a5	update ts version to *	9か月前		<a href="#">ID をコピーする</a> <a href="#">参照</a>
ec418493	fix: update CDK versions for Java examples to latest (#217)	10か月前		<a href="#">ID をコピーする</a> <a href="#">参照</a>
22a34b59	Update README.md (#213)	10か月前		<a href="#">ID をコピーする</a> <a href="#">参照</a>



# 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 GitHub noreply@github.com	コミット日 9か月前	親コミット 42f652a5d580ed1049c55b7041ebf6b5c7abe3a9 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/ecs
39 39 | [ecs-service-with-advanced-alb-config](https://github.com/aws-samples/aws-cdk-examples/tree/master/typescript/ecs/
40 40 | [ecs-service-with-task-networking](https://github.com/aws-samples/aws-cdk-examples/tree/master/typescript/ecs/ec
41 41 - | [fargate-application-load-balanced-service](https://github.com/aws-samples/aws-cdk-examples/tree/master/typescri
41 41 + | [fargate-service-with-autoscaling](https://github.com/aws-samples/aws-cdk-examples/tree/master/typescript/fargat
42 42 | [fargate-service-with-task-placement](https://github.com/aws-samples/aws-cdk-examples/tree/master/typescript/fargat
```

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 GitHub noreply@github.com	コミット日 9か月前	親コミット 42f652a5d580ed1049c55b7041ebf6b5c7abe3a9 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-advanced-alb-config](https://github.com/aws-samples/aws-cdk-
examples/tree/master/typescript/ecs/ecs-service-with-
task-placement/) | Starting a container ECS with task
placement specifications |
39 | [ecs-service-with-task-placement]
39 | [ecs-service-with-advanced-alb-config]
39 | [ecs-service-with-task-placement]
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

ninja-at/c#/lambda ▾

コミット	メッセージ	時間
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 zechariahs/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  ソース  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

説明 - オプショナル  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	<input type="button" value="ID をコピーする"/> <input type="button" value="参照"/>

キャンセル

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

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

The screenshot shows a GitHub pull request interface. At the top, there are two buttons: 'ファイルコンテンツを参照' (View file contents) and 'ファイルに関するコメント' (Comments on file). Below this is a code editor window displaying XML-like code. A specific line of code is highlighted in green: `+ <PackageReference Include="Amazon.CDK" Version="*" />`. To the right of the code editor, there is a comment section. The first comment is from 2 minutes ago, with a 'comment' icon, '編集' (Edit) button, and '削除' (Delete) button. Below the comment, it says '行単位にコメントを書くことも可能' (You can also write comments at the line level). A dropdown menu is open, showing a list of emoji reactions: None, 🌟, 😊, 🎉, ❤️, 😢, 😅, 😮, 😳, and 🎉. The list is ordered by most recent reactions. The code editor shows several other lines of code with similar patterns of green highlighting and emoji reactions.

# 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な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

子ページ一覧ツール > 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 - オプショナル  
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

承認ルールのメンバー - オプショナル  
承認フルのメンバーが既定でされている場合、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つの名前を持つ複数の承認者を組合せます。

承認者のタイプ  
 IAM 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 - オプショナル  
 ブランチフィルターを使用して、送信先ブランチ名がフィルタリスト内の名前と一致する場合にのみ、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 - オプショナル  
 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 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::44445556666:user/Mary_Major", "accountId": "44445556666",  
        "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

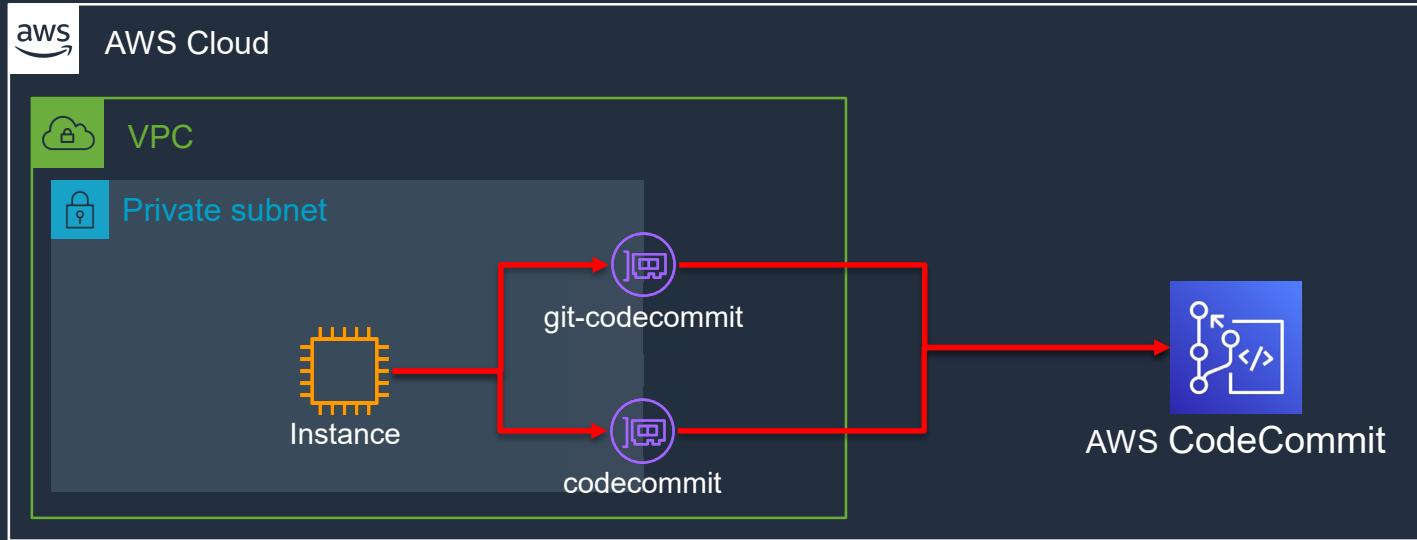


- リポジトリを変更できるユーザーの制御だけでなく、リポジトリ内のブランチを変更できるユーザを制御可能
- IAM ポリシーの Condition で操作の行えるブランチを制限
- 開発チーム以外にはプルリクを master ブランチへ マージさせないなど、ブランチへのアクセスを柔軟に制御できる

# VPC エンドポイントをサポート



- Git 操作用と CodeCommit API 操作用の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



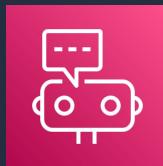


# リポジトリイベントの通知

- ・ 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

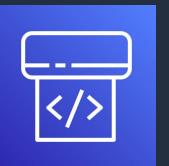
# AWS サービスとの連携



AWS Cloud9



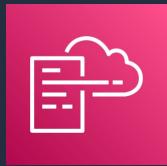
AWS CodeBuild



AWS CodePipeline



AWS CodeStar



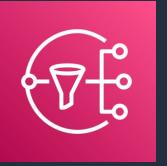
AWS CloudFormation



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。



**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"><li>無制限のリポジトリ数</li><li>月に 50GB のストレージ</li><li>10,000 Git リクエスト/月</li></ul>	<ul style="list-style-type: none"><li>無制限のリポジトリ数</li><li>月に 10GB のストレージ/ユーザー</li><li>2,000 Git リクエスト/月/ユーザー</li></ul>

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。

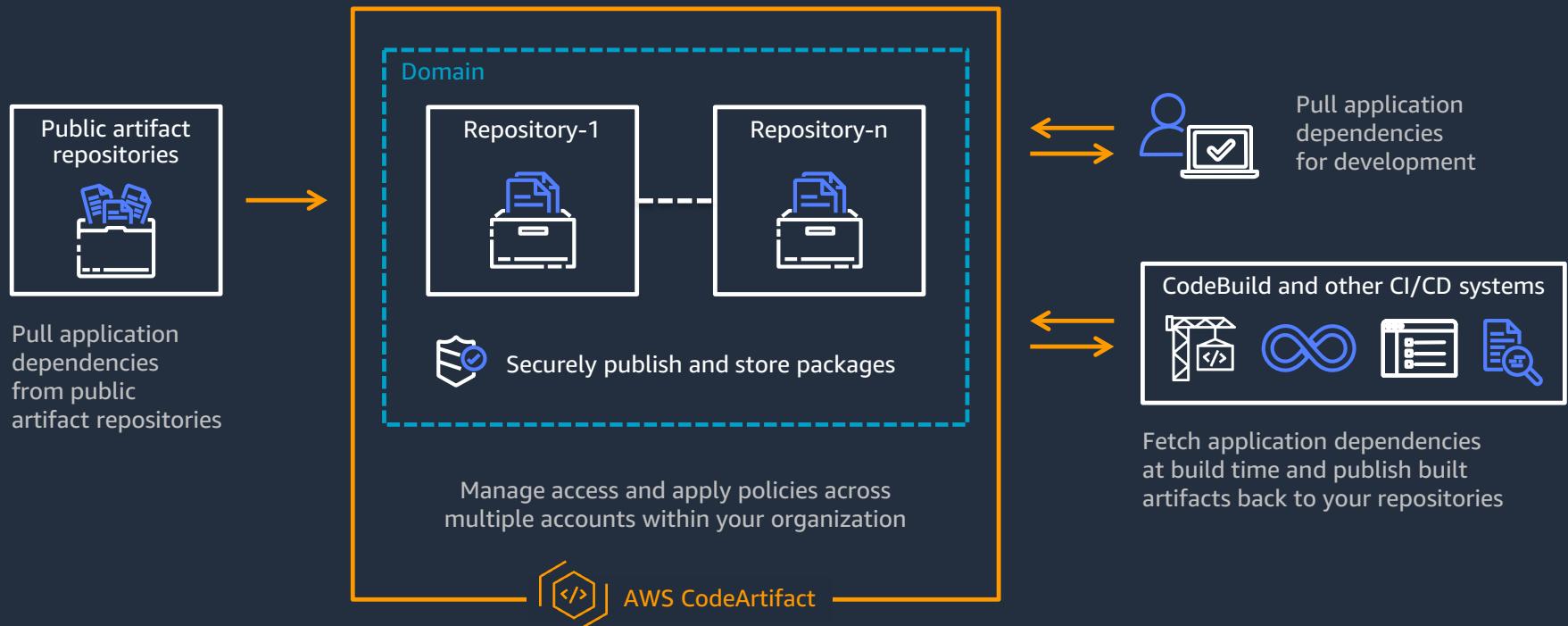
# AWS CodeArtifact

# 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 Pacage 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



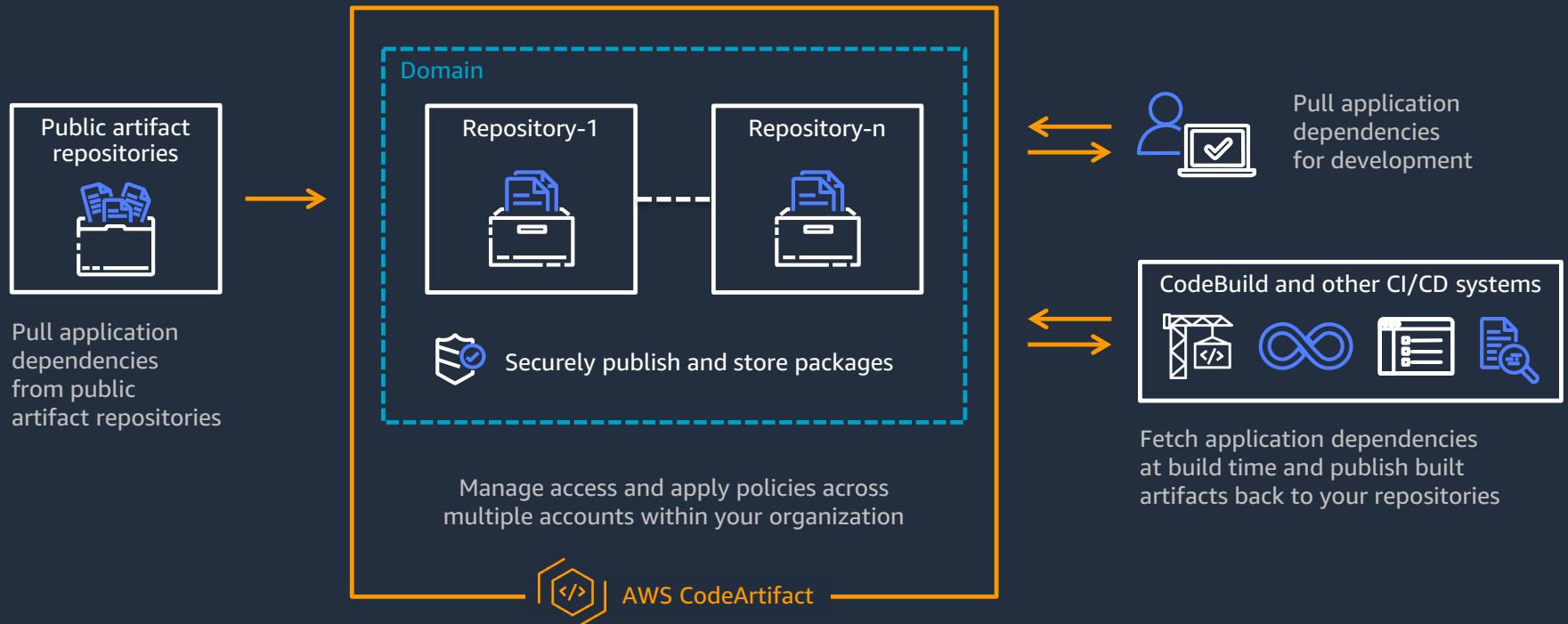
- パッケージとメタデータを格納する単位
  - パッケージが複数のリポジトリに存在していても、**保存はドメインごとに一度だけ**
  - ドメイン内の全てのアセット、メタデータは单一のキーで暗号化
- リポジトリを集約する
  - クロスアカウントに対応、別アカウントのリポジトリも格納可能
- 複数作成できるが、单一の本番用とテスト用を作成することを推奨
- アカウント内で一意であること

# 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。  
my-repository

文字数に制限があります。使用できる文字は、英数字、半角スペース、ハイフン (-)、タクシュー (.)、またはヒラオド ( ) です。

リポジトリの説明 - オプショナル  
My first repository

1000 文字以内

パブリックアップストリームリポジトリ - オプショナル  
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 Next

# 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

### 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**  
アカウント 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**  
ドメイン名には通常、会社名が含まれます。

**ドメイン URL**  
my-domain- [.codeartifact.ap-northeast-1.amazonaws.com](https://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、ドメインのデータを暗号化します。

AWS マネージド型キー  
キーは「/aws/codeartifact」を独自の CMK を指定して、ドメインのデータを暗号化します。  
 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 前へ **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。

外部接続 ドメイン: my-domain

public:npmjs アップストリームリポジトリ [?]

public:pypi アップストリームリポジトリ [?]

npm-store pypi-store リポジトリ my-repository

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

検索 ソート

リポジトリ名	リポジトリの説明	リポジトリ管理者
my-repository	My first repository	
npm-store	Provides npm artifacts from npm, Inc.	
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 [\[リンク\]](#)。
- 2.0.21 以降。AWS CLI バージョン 2 をインストール [\[リンク\]](#)。

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

The screenshot shows the AWS CodeArtifact console interface. At the top, there are navigation links: デベロッパー用ツール > CodeArtifact > リポジトリ > my-repository. Below this, there's a header for the repository named "my-repository" with a "情報" button. A large blue arrow points from the left screenshot to the right one.

The main area is titled "パッケージ" (Packages). It includes a search bar and a "接続手順の表示" (Show connection steps) button. A table lists packages with columns: パッケージ (Package), パッケージタイプ (Package Type), and 最新バージョン (Latest Version). A red box highlights the first row of the table, which is empty. Below the table, a message says: "表示するパッケージがありません。" (No packages to display.) and "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。" (First, set up your local package manager to display the packages in this table.)

The screenshot shows the same AWS CodeArtifact console interface after the package has been added. The repository "my-repository" now contains several packages. A red box highlights the first four rows of the table:

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. At the top, there's a breadcrumb navigation: デベロッパー用ツール > CodeArtifact > リポジトリ > my-repository. Below the navigation, there's a header for 'my-repository' with tabs for '情報', '削除', 'リポジトリポリシーを適用', and '編集'. A button labeled 'リポジトリ' with the sub-label 'My first repository' is also present. The main area has a section titled '詳細' with a note: '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。' Below this is a 'パッケージ' section with a search bar containing 'pi'. A table lists two 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

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

Q my

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
2.0.0	Published
1.0.1	Published
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"  
        }  
      }  
    }  
  ]  
}
```

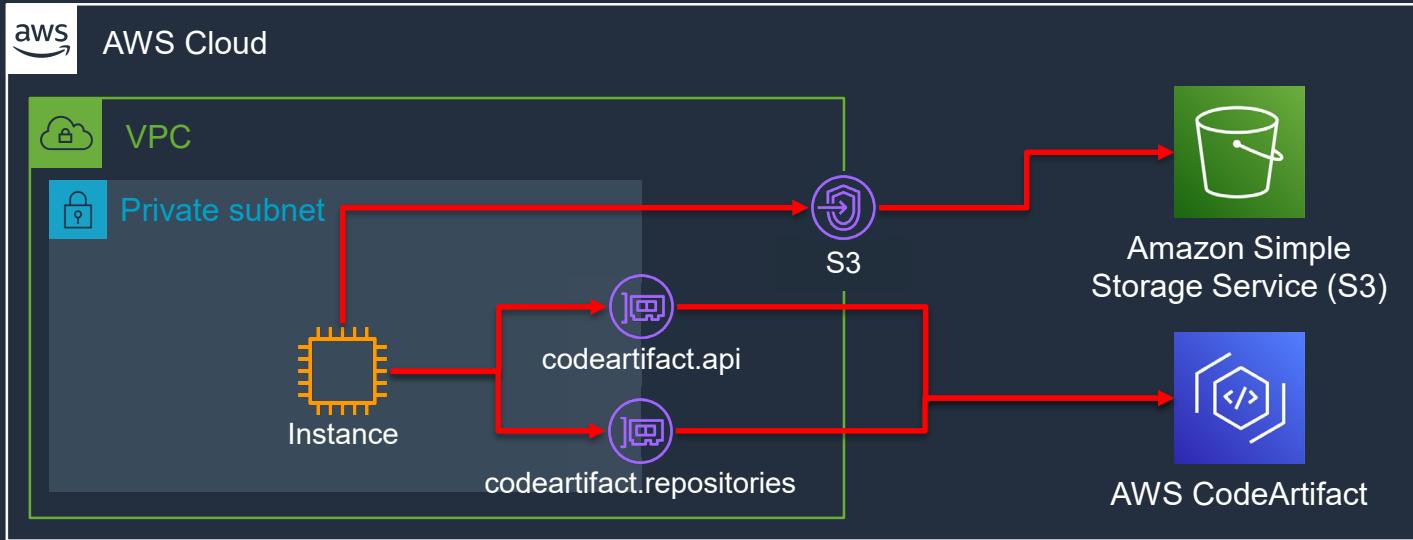
詳細は以下を参照 :

[https://docs.aws.amazon.com/ja\\_jp/codeartifact/latest/ug/using-npm-packages-in-codebuild.html](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](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

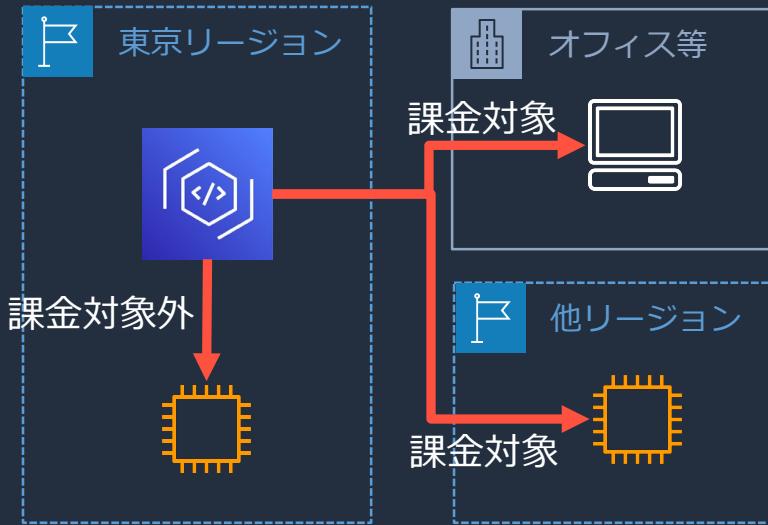


ドメインの数	アカウントごとに 10 個まで
リポジトリの数	ドメインごとに 1,000 個まで
アセットの数	パッケージバージョンごとに 100 個まで
アセットのサイズ	最大 1 GB
パッケージメタデータ ファイルのサイズ	最大 100 KB
設定できるアップスト リームリポジトリの数	リポジトリ毎に 10 個まで

# 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

The screenshot shows the AWS Japan Language Resources homepage. At the top, there's a navigation bar with the AWS logo, search bar, and links for '日本担当チームへお問い合わせ' (Contact Support), 'サポート' (Support), '日本語' (Japanese), 'アカウント' (Account), and 'コンソールにサインイン' (Sign in to the console). Below the navigation bar is a menu with links for '製品' (Products), 'ソリューション' (Solutions), '料金' (Pricing), 'ドキュメント' (Documentation), '学習' (Learning), 'パートナー' (Partners), 'AWS Marketplace' (AWS Marketplace), 'その他' (Other), and a search icon. The main content area features a large title 'AWS クラウドサービス活用資料集トップ' (Top of the AWS Cloud Service Utilization Document Collection) and a descriptive paragraph about the service. At the bottom, there are four buttons: 'AWS Webinar お申込' (Apply for AWS Webinar), 'AWS 初心者向け' (For AWS beginners), '業種・ソリューション別資料' (Industry and solution-specific documents), and 'サービス別資料' (Service-specific documents).

<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]

# 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>

