



AWS  
**Black Belt**  
Online Seminar

AWS Black Belt Online Seminar

Developer Tools

# AWS Code Services

AWS CodeCommit、AWS CodeBuild 編

アマゾン ウェブ サービス ジャパン株式会社

ソリューション アーキテクト 福井 厚

2017.03.22

# 自己紹介

## ❖ 名前

- ❖ 福井 厚 (ふくい あつし) fatsushi@

## ❖ 所属

- ❖ アマゾン ウェブ サービス ジャパン株式会社
- ❖ 技術統括本部エンタープライズ ソリューション部
- ❖ ソリューション アーキテクト

## ❖ 前職

- ❖ エンタープライズ アプリケーション開発コンサルタント

## ❖ 好きなAWSサービス

- ❖ AWS Code シリーズ、AWS IoT、Lambda (C#)



# 内容についての注意点

- 本資料では2017年3月22日時点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が東京リージョンを使用する場合、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.

# Agenda

- ❖ CI/CDとAWS Code Series
- ❖ AWS CodeCommit
- ❖ AWS CodeBuild
- ❖ まとめ



# ソフトウェアの動きは加速している

❖ ソフトウェアの作成と配布はかつてないほどスピードが要求されている

- ほとんどあるいはまったく資金調達せずにスタートアップが巨大企業に対抗できる
- ダウンロードひとつで数百万人のユーザーにすぐにソフトウェアを配布できる
- 多くの要求に応えるには機敏性が最も重要



# ソフトウェア配布モデルは大きく様変わりしている

## かつてのソフトウェア配布モデル



## 新しいソフトウェア配布モデル



# 機敏な動きに必要なツールとは？

- ❖ この新しいソフトウェア駆動の世界でソフトウェアをリリースするのに必要なツールとは
  - ソフトウェア開発のリリースプロセスの流れを管理するツール
  - コードの不具合や潜在的な問題を正しくテスト／検査するツール
  - アプリケーションをデプロイするツール

# リリースプロセスの4つの主なフェーズ

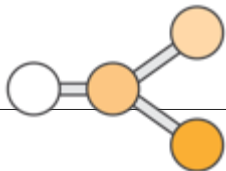
ソース

ビルド

テスト

運用

- ソースコードを  
チェックイン
- バージョン管理、  
ブランチ管理
- 新しいコードの  
ピアレビュー
- Java、C#などの  
コードのコンパイル
- ユニットテスト
- スタイルチェッカー
- コードメトリック
- コンテナイメージの  
作成
- 他のシステムと  
の統合テスト
- ロードテスト
- UIテスト
- 侵入テスト
- 本番環境に  
デプロイ





# リリースプロセスのレベル

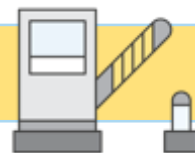
継続的インテグレーション(CI)、継続的デプロイメント(CD)を実現



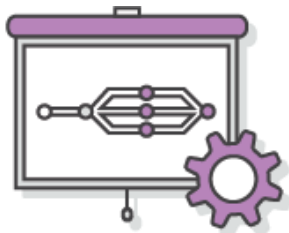
継続的インテグレーション

継続的デリバリ

継続的デプロイメント



# 継続的デリバリのメリット



ソフトウェアの  
リリースプロセスを  
自動化



開発者の  
生産性を改善



バグをすばやく  
検出して対処



アップデートの  
配信を高速化

# AWS Code シリーズ



**AWS CodePipeline**



**AWS CodeDeploy**



**AWS CodeCommit**



**AWS CodeBuild**

A close-up photograph of a single red apple with a yellowish-green stem. The apple has a simple smiley face drawn on its surface with red markers. The face consists of two small red dots for eyes, a red curved line for a smile, and a red equals sign (=) for a nose. The background is a soft, out-of-focus light green.

継続的デリバリ

=

開発者がより幸せに！

# AWS Code シリーズ

ソフトウェア リリース フェーズ



# AWS Code シリーズ

ソフトウェア リリース フェーズ



AWS CodeCommit

# AWS Code シリーズ

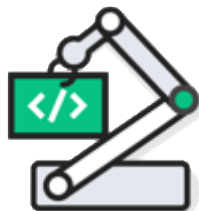
ソフトウェア リリース フェーズ

Source

Build

Test

Production



AWS CodeBuild

# AWS Code シリーズ

ソフトウェア リリース フェーズ

Source

Build

Test

Production



Third Party  
Tooling



**AWS** CodeBuild



# AWS Code シリーズ

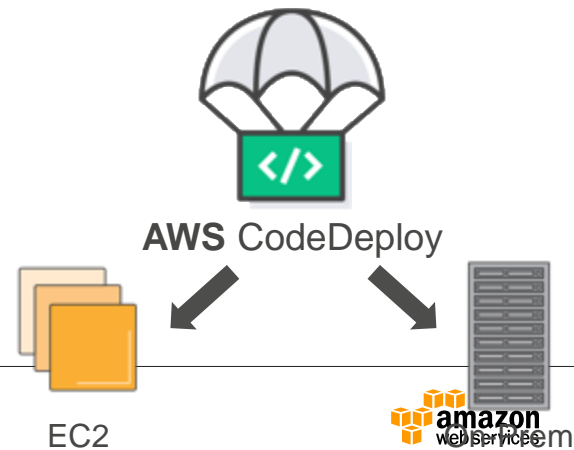
ソフトウェア リリース フェーズ



AWS CodeDeploy

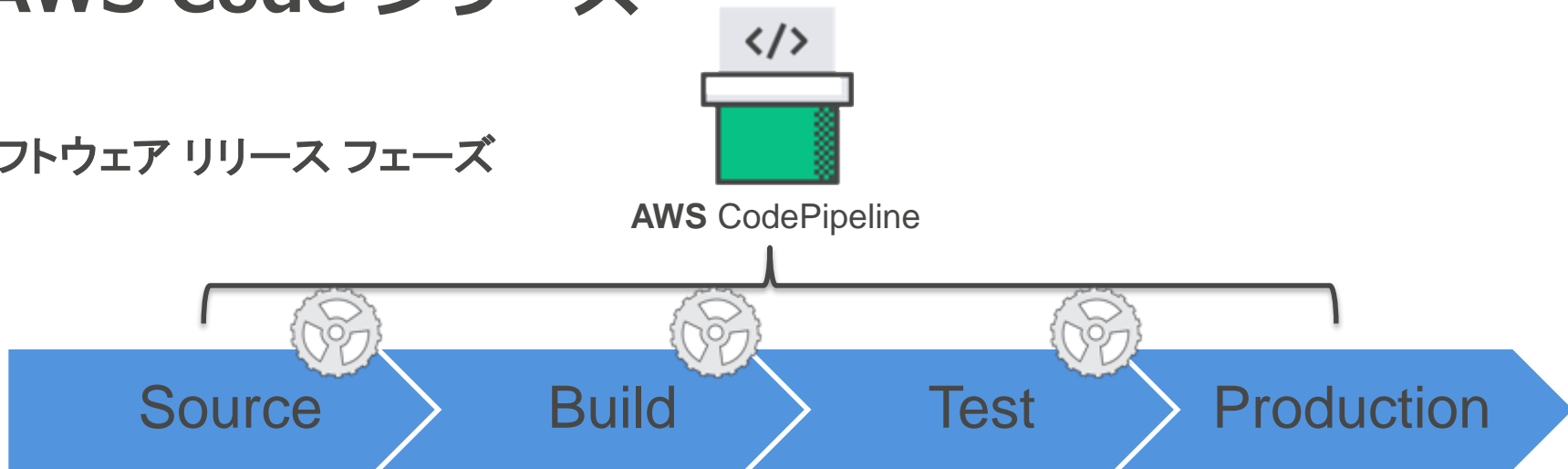
# AWS Code シリーズ

ソフトウェア リリース フェーズ



# AWS Code シリーズ

ソフトウェア リリース フェーズ



# AWS Code シリーズ

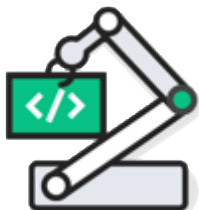
ソフトウェア リリース フェーズ



AWS CodePipeline



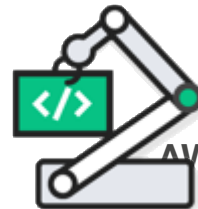
AWS CodeCommit



AWS CodeBuild



Third Party  
Tooling



AWS CodeBuild



AWS CodeDeploy

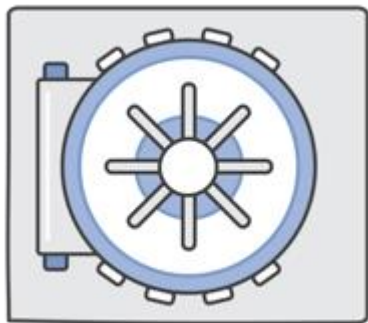
# Storing your code

```
<li><a href="index.html">Home</a></li>
<li><a href="home-events.html">Home Events</a></li>
<li><a href="multi-col-menu.html">Multiple Column Menu on Larger Viewports</a>
<li class="has-children"> <a href="#" class="current">Header Options</a>
  <ul>
    <li><a href="tall-button-header.html">Tall Button Header</a></li>
    <li><a href="image-logo.html">Image Logo</a></li>
    <li class="active"><a href="tall-logo.html">Tall Logo Images</a>
  </ul>
</li>
<li class="has-children"> <a href="#">Carousels</a>
  <ul>
    <li><a href="variable-width-slider.html">Variable Image Width Slider</a></li>
    <li><a href="testimonial-slider.html">Testimonial Slider</a></li>
    <li><a href="featured-work-slider.html">Featured Work Slider</a></li>
    <li><a href="equal-column-slider.html">Equal Column Slider</a></li>
    <li><a href="video-slider.html">Video Slider</a></li>
    <li><a href="mini-bootstrap-carousel.html">Mini Slider</a></li>
  </ul>
</li>
```

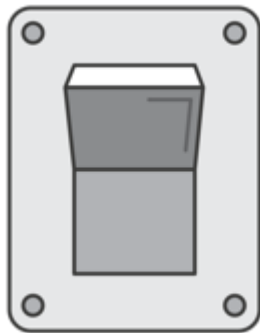
# 自社でホストするバージョン管理ツールの課題

- ❖ 開発者ごとの高価なライセンス料金
- ❖ 高いハードウェア保守コスト
- ❖ 高いサポート 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

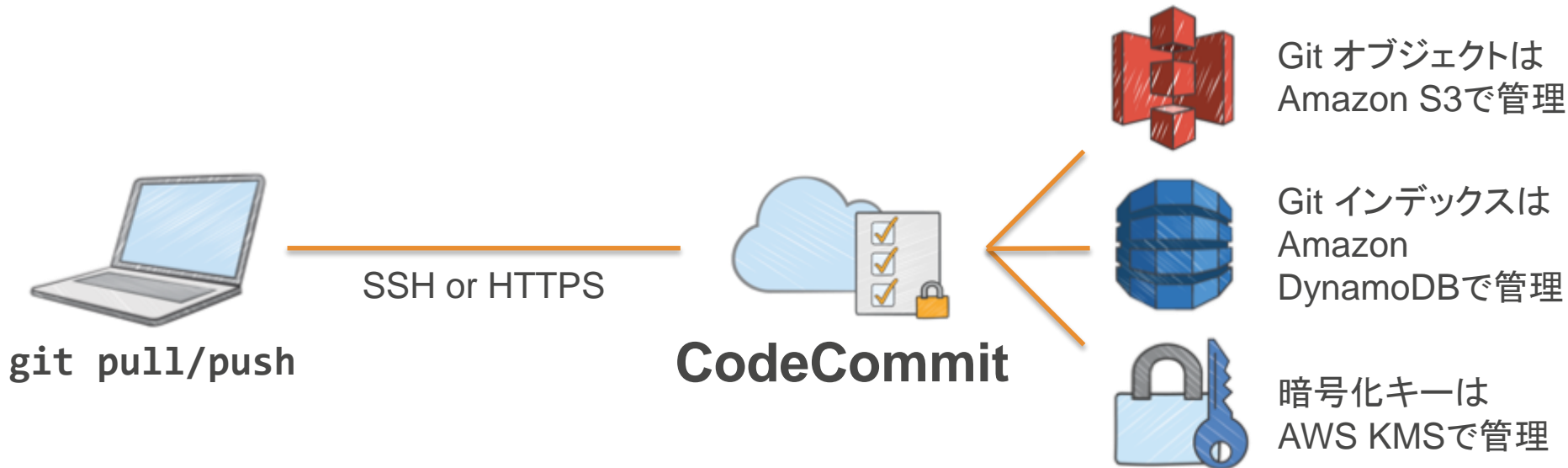
# AWS CodeCommit



- ❖ AWSによるフルマネージドなサービス
  - ❖ 高可用性、耐久性及びハードウェア、ソフトウェアを管理する負荷の削減
- ❖ セキュアにコードを管理
  - ❖ AWS CodeCommitは送信時及び保存時に暗号化
- ❖ 容易にスケール
  - ❖ 多数のファイルやブランチ、履歴を保持可能
- ❖ どのようなファイルも保存可能
  - ❖ 保存できるファイルのサイズやタイプに制限はない
- ❖ 他のAWSサービスやサードパーティーと連携可能
  - ❖ IAM Roleによる他のサービスとの連携が容易
- ❖ 他のリモートレポジトリからの移行が容易
  - ❖ 他のGitベース レポジトリからシンプルに移行可能
- ❖ 既存のGitツールが利用可能

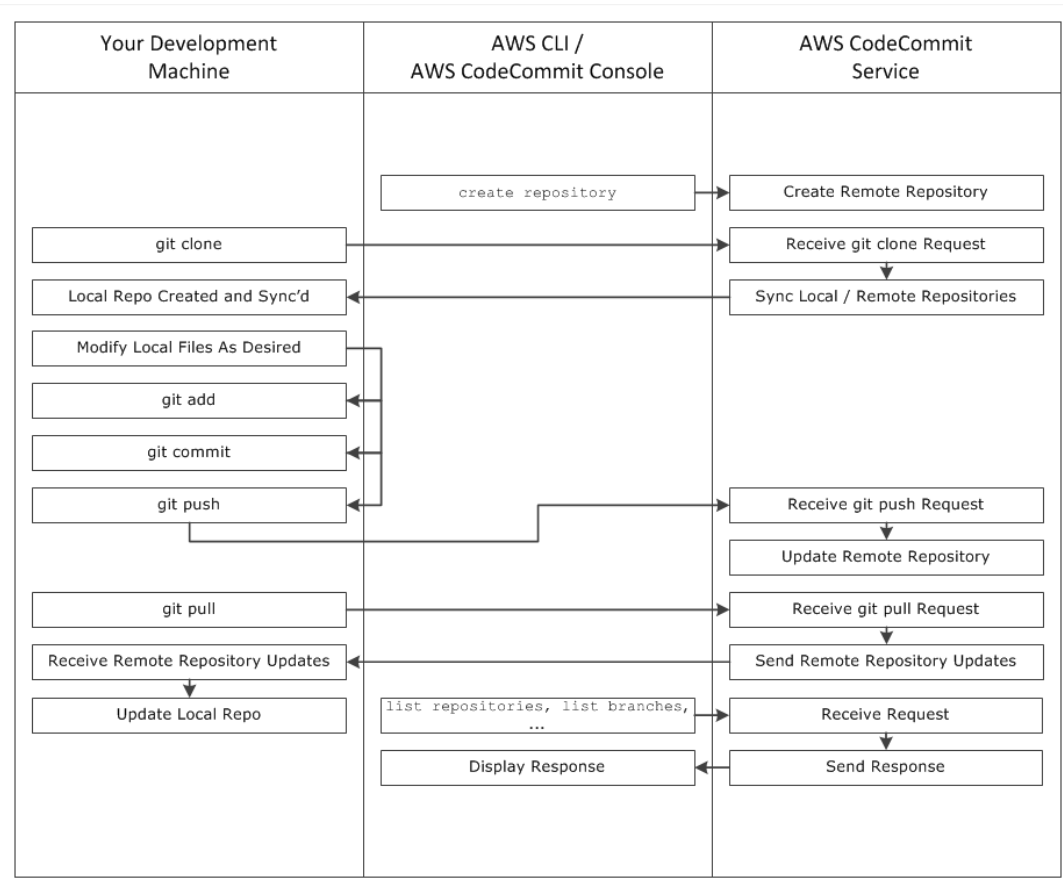


# AWS CodeCommit

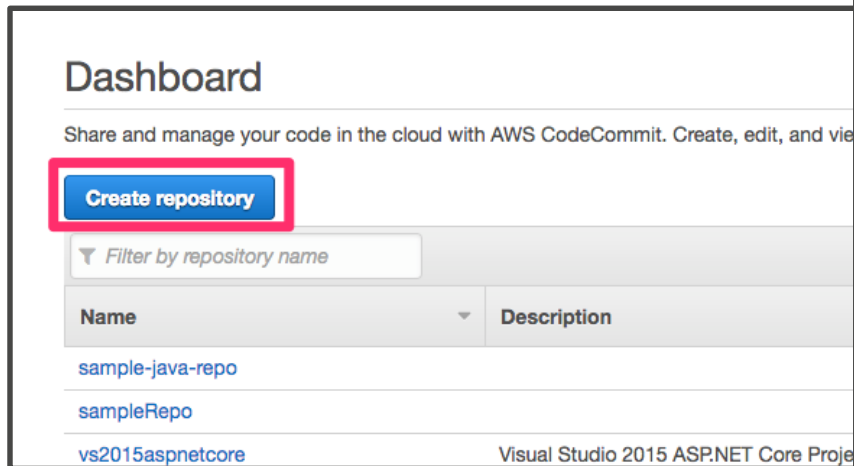


# CodeCommit 利用の流れ

1. AWS CLIかAWS CodeCommit コンソールを利用してAWS CodeCommit リポジトリを作成
2. 開発マシンから **git clone** で特定のリポジトリの名前を指定。AWS CodeCommitリポジトリと接続されたローカル リポジトリを作成
3. 開発マシン上のローカル レポジトリを編集し、**git add**、**git commit**、**git push**することでAWS CodeCommit リポジトリに変更を送信
4. 他のユーザーの変更をダウンロードするために **git pull** を実行しAWS CodeCommitリポジトリとローカル リポジトリを同期。これによってファイルの最新バージョンで作業していることを確実にする



# リポジトリの作成



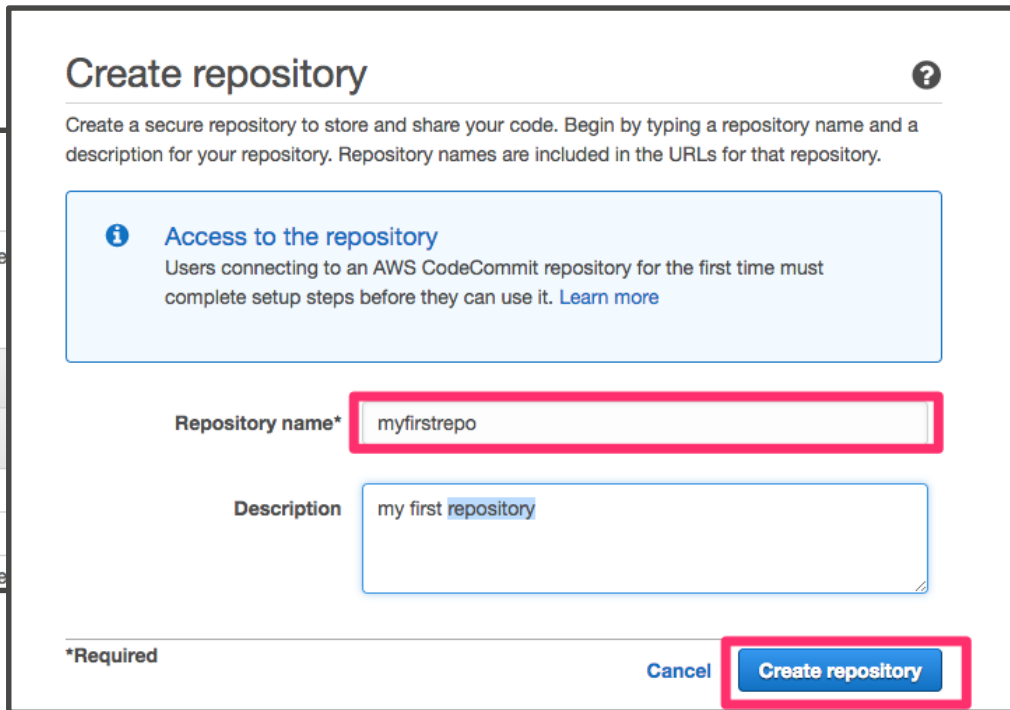
**Dashboard**

Share and manage your code in the cloud with AWS CodeCommit. Create, edit, and view your repositories.

**Create repository**

Filter by repository name

Name	Description
<a href="#">sample-java-repo</a>	
<a href="#">sampleRepo</a>	
<a href="#">vs2015aspnetcore</a>	Visual Studio 2015 ASP.NET Core Project



## Create repository ?

Create a secure repository to store and share your code. Begin by typing a repository name and a description for your repository. Repository names are included in the URLs for that repository.

**Access to the repository**  
Users connecting to an AWS CodeCommit repository for the first time must complete setup steps before they can use it. [Learn more](#)

**Repository name\***

**Description**

\*Required Cancel **Create repository**

## AWS CLIで実行

```
$ aws codecommit create-repository --repository-name MyDemoRepo \  
  --repository-description "My demonstration repository"
```

# AWS CodeCommit を利用するには

- ❖ 最も簡単な方法：AWS 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
  - ❖ AWS CodeCommitでサポートしているすべてのOSで動作
  - ❖ 統合開発環境（IDE）やその他のツールと互換性がある
- ❖ その他の接続方法
  - ❖ SSH接続（IAMとSSHキーの紐付け、Linux, Mac、Unix、Windows設定）
  - ❖ 開発ツールからの接続（Visual Studio, Eclipseなど）
  - ❖ 詳細は以下のURLを参照  
<http://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IAM User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 version 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があり

以下を参照：

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

# IAM User の作成

## Add user



### Set user details

You can add multiple users at once with the same access type and permissions. [Learn more](#)

User name\*

[+ Add another user](#)

IAM Userの名前を入力

### Select AWS access type

Select how these users will access AWS. Access keys and autogenerated passwords are provided in the last step. [Learn more](#)

- Access type\*  **Programmatic access**  
Enables an **access key ID** and **secret access key** for the AWS API, CLI, SDK, and other development tools.
- AWS Management Console access**  
Enables a **password** that allows users to sign-in to the AWS Management Console.

- Console password\*  Autogenerated password  
 Custom password

Show password

Require password reset  User must create a new password at next sign-in

IAM Userのパスワードを入力

\* Required

Cancel

Next: Permissions

# ポリシーのアタッチ

Add user

1

Details

2

Permissions

3

Review

4

Complete

Set permissions for codecommit-user



Add user to group



Copy permissions from existing user



Attach existing policies directly

既存ポリシーのアタッチ

Attach one or more existing policies directly to the user or create a new policy. [Learn more](#)

Create policy

Refresh

Filter: Policy type

CodeCommit

	Policy name	Type	Attache
<input type="checkbox"/>	<a href="#">AWSCodeCommitPowerUse...</a>	Customer managed	
<input type="checkbox"/>	<a href="#">AWSCodeCommitPowerUser</a>	AWS managed	0 Provider
<input checked="" type="checkbox"/>	<a href="#">AWSCodeCommitFullAccess</a>	AWS managed	0 Provider
<input type="checkbox"/>	<a href="#">AWSCodeCommitReadOnly</a>	AWS managed	0 Provider

以下のポリシーを選択

- AWSCodeCommitFullAccess
- IAMUserSSHKeys
- IAMSelfManageServiceSpecificCredentials
- IAMReadOnlyAccess

# ポリシーのアタッチ

## Add user



Details



Permissions



Review



Complete

## Review

Review your choices. After you create the user, you can view and download the autogenerated password and access key.

### User details

User name codecommit-user

AWS access type Programmatic access

Console password type Custom password

Require password reset No

### Permissions summary

The following policies will be attached to the user shown below.

Type	Name
Managed policy	<a href="#">AWSCodeCommitFullAccess</a>
Managed policy	<a href="#">IAMUserSSHKeys</a>
Managed policy	<a href="#">IAMSelfManageServiceSpec</a>
Managed policy	<a href="#">IAMReadOnlyAccess</a>

## Add user



Details



Permissions



Review



Complete



### Success

You successfully created the users shown below. You can view and download user security credentials. You can also email users instructions for signing in to the AWS Management Console. This is the last time these credentials will be available to download. However, you can create new credentials at any time.

Users with AWS Management Console access can sign-in at: <https://fatsushi.signin.aws.amazon.com/console>

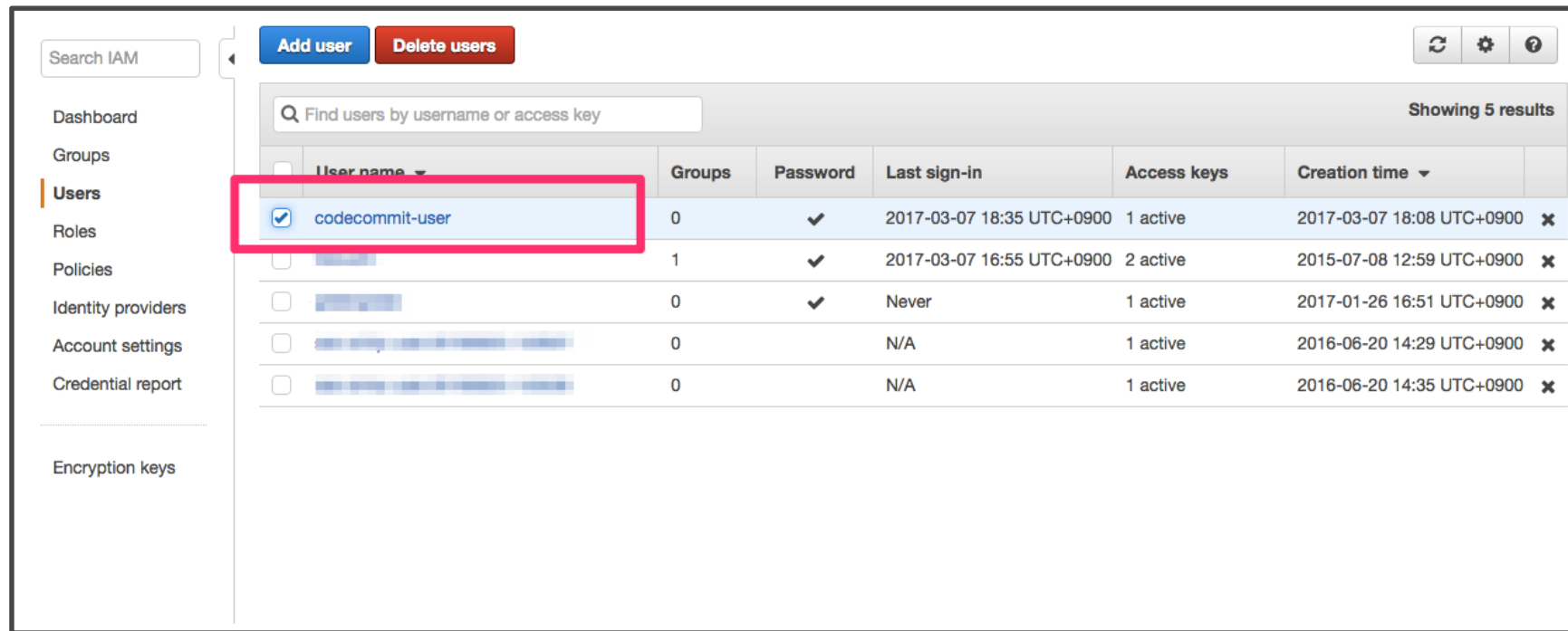
Download .csv

	User	Access key ID	Secret access key	Email login instructions
▶	✓ codecommit-user	████████████████████	***** Show	<a href="#">Send email</a>

Close



# Git ユーザー名、パスワードの生成



The screenshot displays the AWS IAM console interface. On the left, a navigation menu includes 'Dashboard', 'Groups', 'Users', 'Roles', 'Policies', 'Identity providers', 'Account settings', 'Credential report', and 'Encryption keys'. The 'Users' section is selected. At the top, there are buttons for 'Add user' and 'Delete users', along with search and settings icons. A search bar contains the text 'Find users by username or access key'. Below this, a table lists five users. The first user, 'codecommit-user', is highlighted with a red rectangular box. The table columns are: 'User name', 'Groups', 'Password', 'Last sign-in', 'Access keys', and 'Creation time'. The 'codecommit-user' row shows 0 groups, a password checkmark, a last sign-in of '2017-03-07 18:35 UTC+0900', 1 active access key, and a creation time of '2017-03-07 18:08 UTC+0900'. The other four users have their names redacted with blue bars.

User name	Groups	Password	Last sign-in	Access keys	Creation time
<input checked="" type="checkbox"/> codecommit-user	0	✓	2017-03-07 18:35 UTC+0900	1 active	2017-03-07 18:08 UTC+0900
<input type="checkbox"/> [redacted]	1	✓	2017-03-07 16:55 UTC+0900	2 active	2015-07-08 12:59 UTC+0900
<input type="checkbox"/> [redacted]	0	✓	Never	1 active	2017-01-26 16:51 UTC+0900
<input type="checkbox"/> [redacted]	0	N/A	N/A	1 active	2016-06-20 14:29 UTC+0900
<input type="checkbox"/> [redacted]	0	N/A	N/A	1 active	2016-06-20 14:35 UTC+0900

# Git ユーザー名、パスワードの生成

The screenshot shows the AWS IAM console interface. On the left is a navigation sidebar with a search bar labeled 'Search IAM' and a list of menu items: Dashboard, Groups, Users (highlighted with an orange bar), Roles, Policies, Identity providers, Account settings, Credential report, and Encryption keys. The main content area is titled 'SSH keys for AWS CodeCommit' and includes a sub-header 'Use SSH public keys to authenticate access to AWS CodeCommit repositories. [Learn more](#)'. Below this is a button labeled 'Upload SSH public key'. A table with columns 'SSH key ID', 'Uploaded', and 'Status' is shown, containing the text 'No results'. The next section is titled 'HTTPS Git credentials for AWS CodeCommit' with a sub-header 'Generate a user name and password you can use to authenticate HTTPS connections to AWS CodeCommit repositories. You can generate and store up to 2 sets of credentials. [Learn more](#)'. A button labeled 'Generate' is highlighted with a red rectangular box.

# Git ユーザー名、パスワードの生成

Git credentials generated ×

IAM has generated a user name and password for you to use when authenticating to AWS CodeCommit. You can use these credentials when connecting to AWS CodeCommit from your local computer and from tools that require a static user name and password. [Learn more](#)

**User name** codecommit-user-at [REDACTED]

**Password** [REDACTED] [Show](#)

This is the only time the password will be available to view, copy, or download. We recommend downloading these credentials and storing the file in a secure location. You can reset the password in IAM at any time.

[Download credentials](#) [Close](#)

# Git ユーザー名、パスワードの生成

The screenshot shows the AWS IAM console interface. On the left is a navigation sidebar with options: Search IAM, Dashboard, Groups, Users (highlighted), Roles, Policies, Identity providers, Account settings, Credential report, and Encryption keys. The main content area is titled "SSH keys for AWS CodeCommit" and includes a sub-header "Use SSH public keys to authenticate access to AWS CodeCommit repositories. [Learn more](#)". Below this is a button "Upload SSH public key" and an empty table with columns "SSH key ID", "Uploaded", and "Status", containing the text "No results".

The second section is titled "HTTPS Git credentials for AWS CodeCommit" and includes a sub-header "Generate a user name and password you can use to authenticate HTTPS connections to AWS CodeCommit repositories. You can generate and store up to 2 sets of credentials. [Learn more](#)". Below this is a button "Generate".

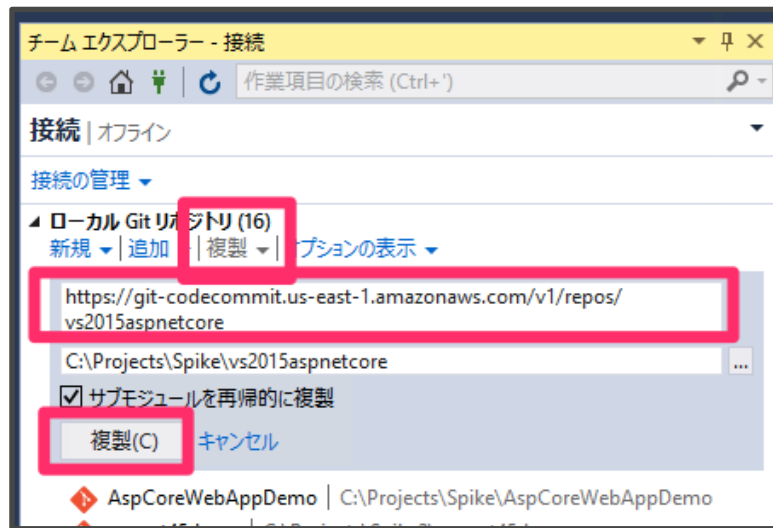
A table below the "Generate" button shows the generated credentials, highlighted with a red border:

User name	Created	Status	Manage	
codecommit-user-at-██████████	2017-03-07 18:47 UTC+0900	Active	<a href="#">Reset Password</a>   <a href="#">Make inactive</a>	✕

# Local GitからCodeCommit への接続

- ❖ git clone や git push 実行時にIAMコンソールで生成しダウンロードしたユーザー名、パスワードを指定
- ❖ 開発ツールにIAMコンソールで生成したユーザー名、パスワードを登録

# Visual Studio からの接続



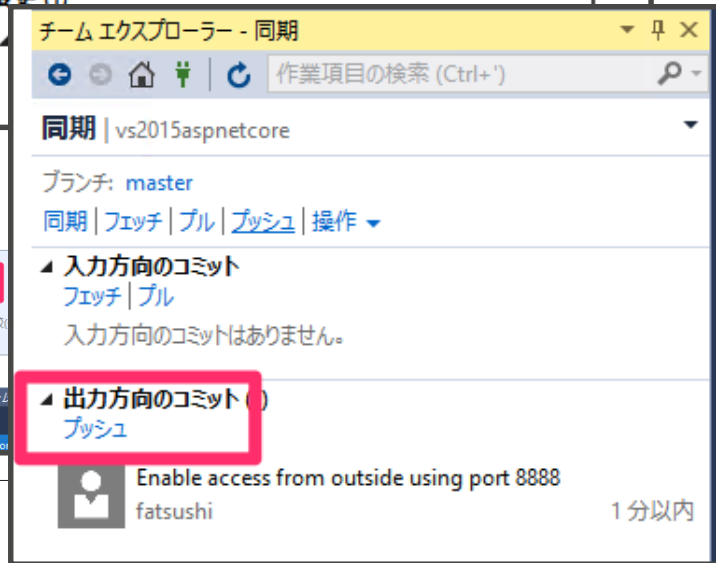
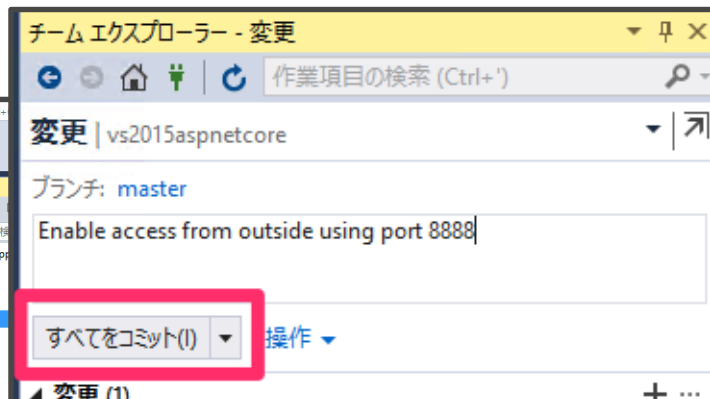
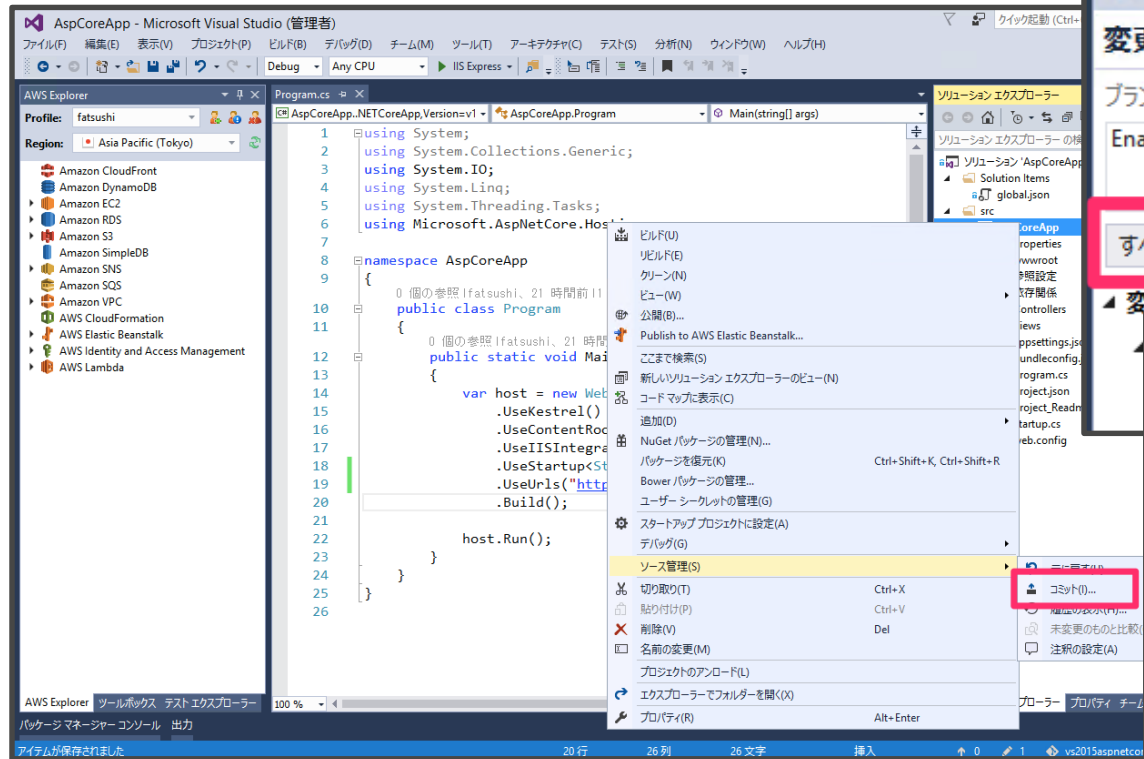
# Visual Studio からの接続

The image shows a composite screenshot of Visual Studio with three main panels:

- Left Panel (Team Explorer):** Titled "チーム エクスプローラー - 接続". It shows a list of local Git repositories. The repository "vs2015aspnetcore" is selected. A red box highlights the "複製(C)" (Copy) button in the context menu.
- Center Panel (Visual Studio IDE):** Shows the "Visual Studio" logo and a list of recent files. The file "vs2015aspnetcore" is highlighted in a red box.
- Right Panel (Team Explorer):** Titled "チーム エクスプローラー - 接続". It shows the same repository list as the left panel. A red box highlights the "vs2015aspnetcore" repository.

At the bottom of the screenshot, the Windows taskbar shows the system clock as 4:08 PM on 3/19/2017.

# Visual Studio からの接続





# 参考資料 : git を学ぶ

- ❖ Pro Git  
<https://git-scm.com/book/ja/v2> (日本語)
- ❖ Git Cheat sheet  
<https://services.github.com/on-demand/downloads/ja/github-git-cheat-sheet.pdf>  
(日本語)
- ❖ Git Immersion  
<http://gitimmersion.com/>
- ❖ Git Reference  
<http://gitref.org/index.html>
- ❖ 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  
以下を参照  
<https://git-scm.com/book/en/v2/Git-Tools-Credential-Storage>

# リポジトリのコンテンツ参照

## Dashboard ?

Share and manage your code in the cloud with AWS CodeCommit. Create, edit, and view details about your code repositories.

[Create repository](#) ↻

« < 1 to 6 of 6 Repositories > » Repositories per page 10 ▾

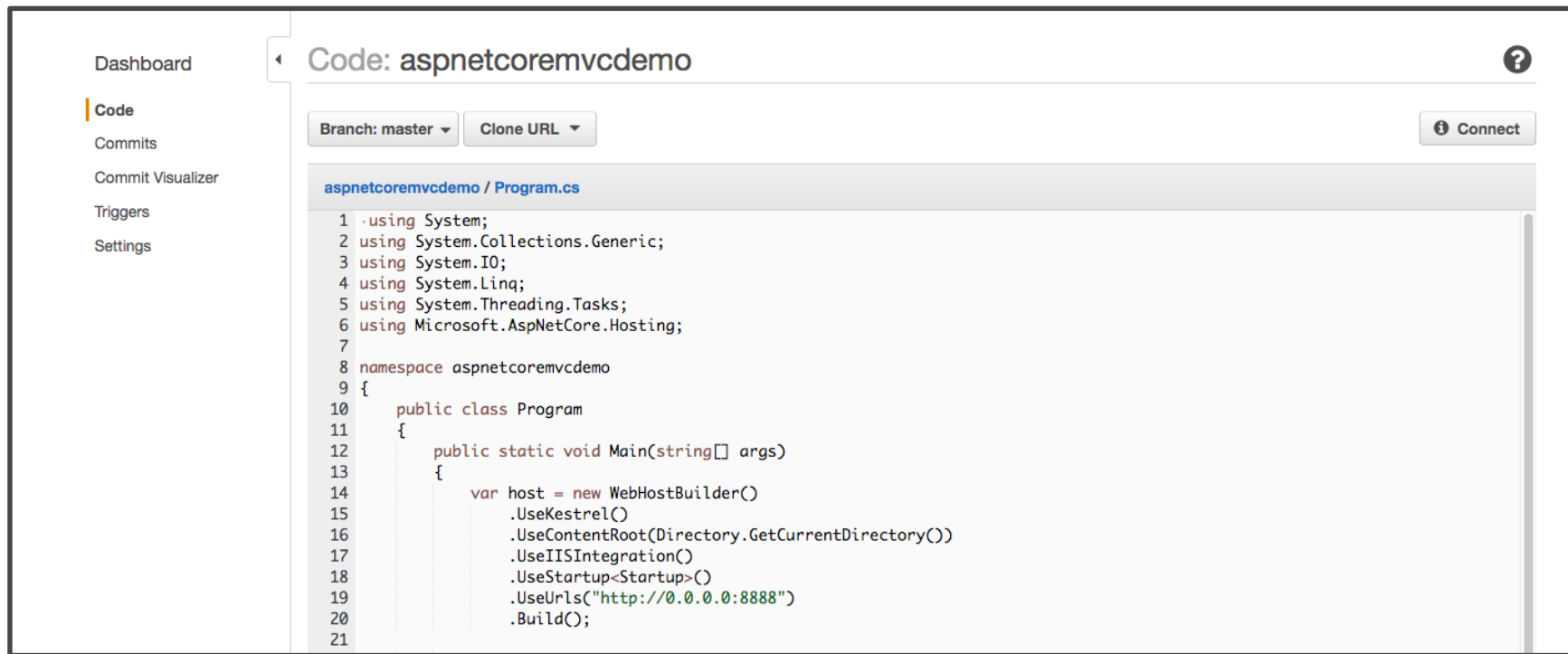
Name ▾	Description	Last updated ▾	URL
<a href="#">vs2015aspnetcore</a>	Visual Studio 2015 ASP.NET Core Project	Mar 19, 2017 7:24:36 AM UTC	<a href="#">🔗</a>
<a href="#">aspnetcoremvcdemo</a>	ASP.NET Core MVC Demo	Mar 17, 2017 9:29:02 AM UTC	<a href="#">🔗</a>
<a href="#">ecs-cd-aspnetcoremvcdemo</a>	ECS Continuous Deployment for ASP.NET Core MVC App	Mar 11, 2017 5:34:57 AM UTC	<a href="#">🔗</a>
<a href="#">AspNetCoreWebAppDemo</a>	ASP.NET Core Web Application Demo	Mar 8, 2017 9:40:32 AM UTC	<a href="#">🔗</a>
<a href="#">CodeBuildDemoDockerPHP</a>	Code Build Demo for Docker PHP sample	Mar 8, 2017 5:54:03 AM UTC	<a href="#">🔗</a>
<a href="#">MyDemoRepo</a>	My Demonstration Repository	May 9, 2016 12:36:47 AM UTC	<a href="#">🔗</a>

« < 1 to 6 of 6 Repositories > » Repositories per page 10 ▾

# リポジトリのコンテンツ参照

The screenshot displays the AWS CodeCommit console interface for a repository named 'aspnetcoremvcdemo'. On the left, a navigation sidebar includes 'Dashboard', 'Code', 'Commits', 'Commit Visualizer', 'Triggers', and 'Settings'. The main area shows the repository's file structure under the heading 'Code: aspnetcoremvcdemo'. At the top right of the main area is a 'Connect' button. Below the repository name, there are controls for 'Branch: master' and 'Clone URL'. The file list includes folders like 'Controllers', 'Views', and 'wwwroot', and files such as '.bowerrc', 'appsettings.Development.json', 'appsettings.json', 'aspnetcoremvcdemo.csproj', 'bower.json', 'bundleconfig.json', 'Dockerfile', 'Program.cs', and 'Startup.cs'. The 'Program.cs' file is highlighted with a red rectangular box.

# リポジトリのコンテンツ参照



The screenshot shows a GitHub repository interface. On the left is a navigation sidebar with links for Dashboard, Code (selected), Commits, Commit Visualizer, Triggers, and Settings. The main area is titled 'Code: aspnetcoremvcdemo' and includes a 'Branch: master' dropdown and a 'Clone URL' button. A 'Connect' button is in the top right. The code file 'aspnetcoremvcdemo / Program.cs' is displayed with the following content:

```
1 using System;
2 using System.Collections.Generic;
3 using System.IO;
4 using System.Linq;
5 using System.Threading.Tasks;
6 using Microsoft.AspNetCore.Hosting;
7
8 namespace aspnetcoremvcdemo
9 {
10     public class Program
11     {
12         public static void Main(string[] args)
13         {
14             var host = new WebHostBuilder()
15                 .UseKestrel()
16                 .UseContentRoot(Directory.GetCurrentDirectory())
17                 .UseIISIntegration()
18                 .UseStartup<Startup>()
19                 .UseUrls("http://0.0.0.0:8888")
20                 .Build();
21 }
```

# 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（スプリット表示または統合表示が可能）
  - ❖ コミット履歴のユーザー名にマウスカーソルを合わせるとEメールアドレスを表示
  - ❖ 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

Dashboard

Code

**Commits**

Commit Visualizer

Triggers



Settings

## Commits: aspnetcoremvcdemo



Review the history of commits to this repository, browse the code at specific commits, and compare changes from the parent commit.

Branch: master ▾



▼ Commits on March 20th, 2017

- ▶ add develop branch  
fatsushi authored 4 hours ago 4fd6423c  

▼ Commits on March 17th, 2017

- ▶ modify dockerfile  
fatsushi authored 3 days ago 374ab865  

▼ Commits on March 11th, 2017

- ▶ delete Dockerfile and buildspec.yml  
fatsushi authored 9 days ago 0a9fe6c6  

# コミット履歴の表示

```
Program.cs </>
1  using System;
2  using System.Collections.Generic;
3  using System.IO;
4  using System.Linq;
5  using System.Threading.Tasks;
... @ -11,6 +12,8 @@
11 {
12     public static void Main(string[]
args)
13     {
14         var host = new
WebHostBuilder()
15             .UseKestrel()
16             .UseContentRoot(Directory.GetCurrentDirector
y())
...
1  using System;
2  using System.Collections.Generic;
3 + using System;
4  using System.IO;
5  using System.Linq;
6  using System.Threading.Tasks;
... @ -11,6 +12,8 @@
12 {
13     public static void Main(string[]
args)
14     {
15 +         //add log for develop branch
16 +         Console.WriteLine("log
comment");
17         var host = new
WebHostBuilder()
18             .UseKestrel()
19             .UseContentRoot(Directory.GetCurrentDirector
y())
...
```

# コミット グラフの表示

## ❖ Commit Visualizer を選択

The screenshot displays the AWS CodeCommit Commit Visualizer interface. The main view shows a commit graph for the 'aspnetcoremvcdemo' repository on the 'master' branch. The graph consists of a vertical line of green dots representing commits, with a blue dot representing the current commit. A tooltip is visible over one of the commits, showing the commit message 'add develop branch' and the commit ID '4fd6423c'. The interface includes a sidebar with navigation options: Dashboard, Code, Commits, Commit Visualizer (selected), Triggers, and Settings. The top navigation bar shows the repository name and a search input field for commit IDs.

Dashboard

Commit Visualizer: aspnetcoremvcdemo

Branch: master

Type a full commit ID

ok

modify log

John add log

March 20th, 2017

fatsushi

add develop branch

[View differences between this commit and its parent](#)

**Commit ID**  
4fd6423c

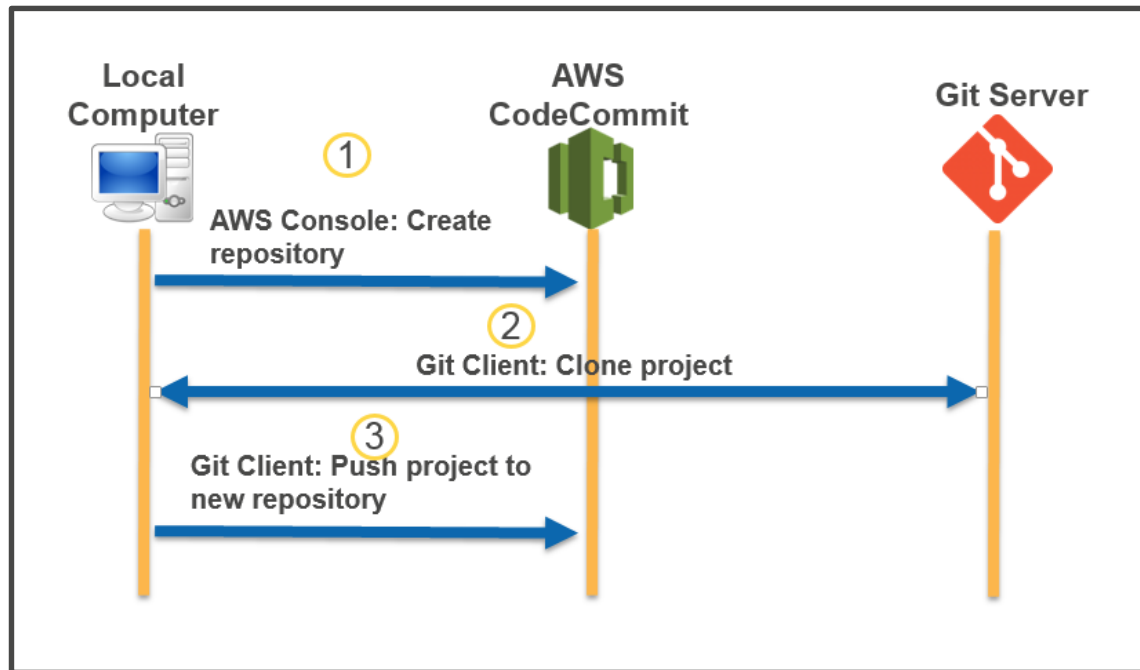
**Parent ID**  
374ab86



# 他のGitリポジトリからのマイグレーション

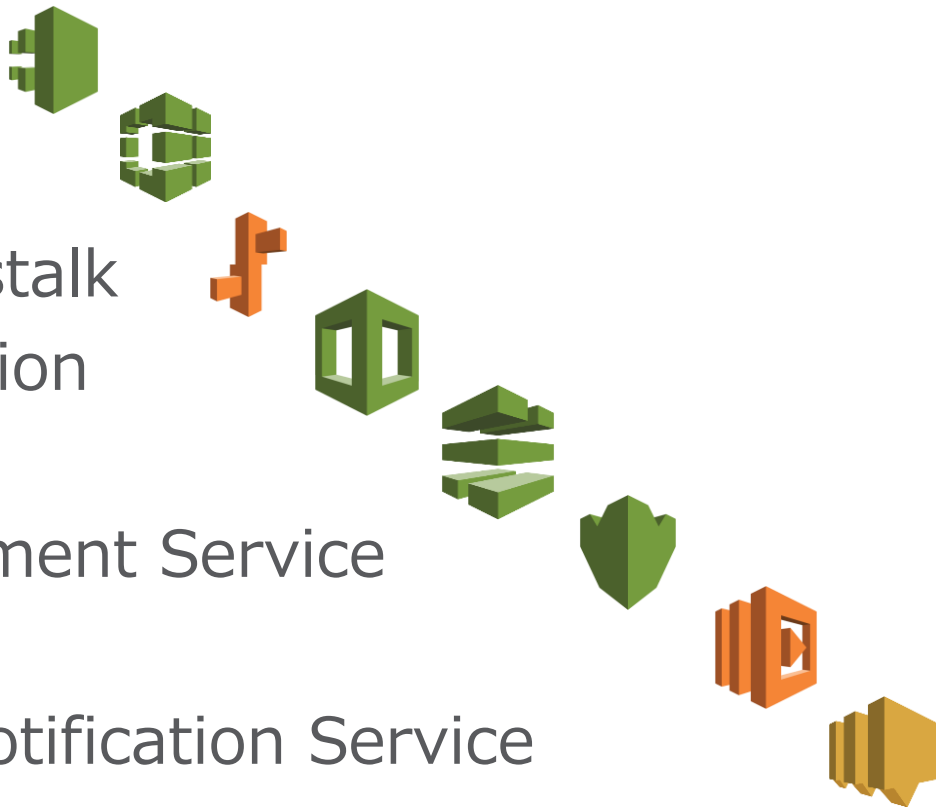
❖ 他のGit リポジトリからのマイグレーションは非常にシンプル

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にPush



# AWSサービスとの連携

- ❖ AWS CloudTrail
- ❖ AWS CodeBuild
- ❖ AWS Elastic Beanstalk
- ❖ AWS CloudFormation
- ❖ AWS CodePipeline
- ❖ AWS Key Management Service
- ❖ AWS Lambda
- ❖ Amazon Simple Notification Service



# AWS CloudTrailを利用したAPIコール ログ取得

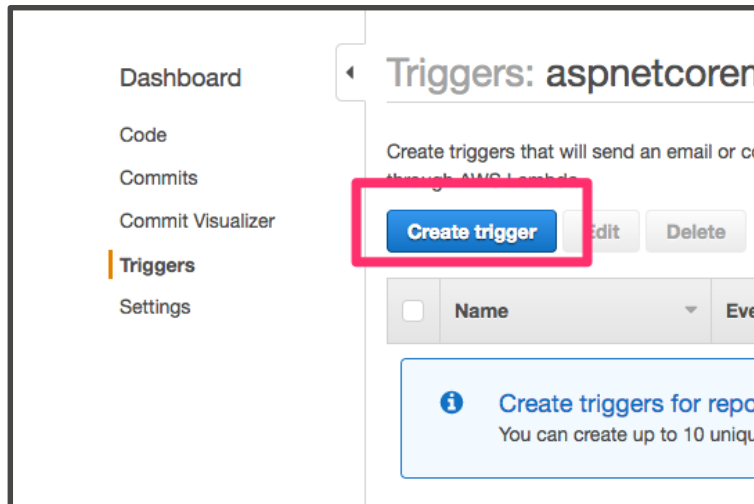
- ❖ AWS CodeCommit は、AWS CloudTrailと連携してコンソール、git クライアント、AWS CLI から発行されるCodeCommit API をキャプチャーしてS3 バケットに保存可能

```
{
  "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",
  ....
}
```

# AWS CodeCommit リポジトリ トリガー

- ❖ AWS CodeCommit リポジトリのトリガーを構成  
(コードのPush時など)
  - ❖ Amazon SNSをによるトピックの通知
  - ❖ AWS Lambda の実行
  - ❖ 関連リソースは CodeCommit と同じリージョンで作成する必要がある
- ❖ 作成したトリガーはテストすることができる
  - ❖ コンソールとAWS CLIの両方で可能

# トリガーの作成



### Create trigger

Triggers are actions taken in response to repository events. Triggers can be configured to send notifications to Amazon Simple Notification Service (SNS) or to an AWS Lambda function. Choose a trigger below to get started. [Learn more](#)

Trigger name\* myfirsttrigger

Events\* Choose one or more events

Branch names\* Push to existing branch Create branch or tag Delete branch or tag

Service

You can configure your trigger to use an existing SNS topic or Lambda function.

Send to\*  Amazon SNS  AWS Lambda

SNS topic\* Select or type an SNS topic ⓘ

Custom data For example, an IRC channel ID # ⓘ

AWS CodeCommit must have permission to publish to Amazon SNS topics from this trigger. [Learn more](#)

\*Required

Cancel Create

# 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。
リポジトリ内のトリガーの数	10以内
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。

# CodeCommit の価格

最初の5 アクティブ ユーザー	最初の5以上のアクティブ ユーザー(一人当たり)
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

- \$0.06 / GB / 月
- \$0.001 / Git リクエスト

# AWS CodeCommit 関連Blog ポスト

- ❖ [Using AWS CodeCommit with Git Repositories in Multiple AWS Accounts](#)
- ❖ [Using AWS CodeCommit and GitHub Credential Helpers](#)
- ❖ [Using AWS CodeCommit from Eclipse](#)
- ❖ [AWS CodeCommit with Amazon EC2 Role Credentials](#)
- ❖ [AWS CodeCommit and SourceTree Setup Tutorial with SSH Keys](#)
- ❖ [Integrating AWS CodeCommit with Jenkins](#)
- ❖ [Integrating AWS CodeCommit with Review Board](#)



A photograph of an automotive assembly line. In the center, a car body is being processed by several orange robotic arms. The background shows more of the factory floor with various equipment and structural elements. The text "Build & test your application" is overlaid in white, bold, sans-serif font across the middle of the image.

**Build & test your  
application**

# AWS CodeBuild



## ❖フルマネージド

❖AWS CodeBuild はセットアップ、パッチ、アップデート、サーバー管理の手間が不要

## ❖オン デマンド

❖ビルドしたいニーズに合わせて自動でスケール。利用した分数のみのお支払い

## ❖すぐに使える

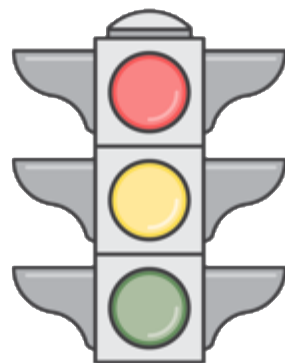
❖最もポピュラーな開発言語用の構成済みのビルド環境を提供。最初にビルドスクリプトを作成するだけ。

# どのように動作するのか?

1. ソースコードのダウンロード
2. 一時的なテナ内でbuildspecで構成されたコマンドを実行（ビルドごとに新規に作成される）
3. マネジメント コンソールとCloudWatch Logs にビルドの出力結果が送信される
4. 生成されたアーティファクトをS3バケットにアップロード

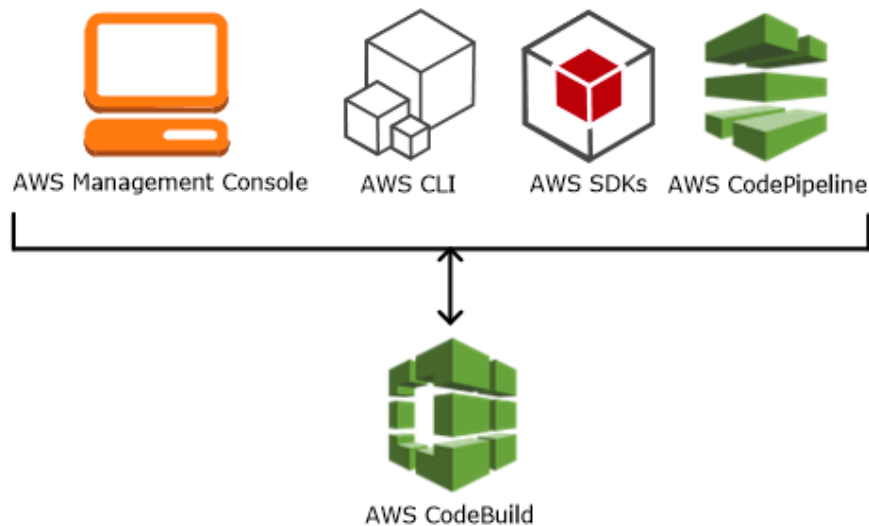
# CodeBuildでどのようにリリースプロセスを自動化するか？

- CI/CDのためのAWS CodePipelineとの統合
- 容易な自動化（API/CLI）
- ビルド環境を持ち込み可能
  - 必要なツールを含むDockerイメージの作成
- オープンソース Jenkins プラグイン
  - CodeBuildをJenkins Master のWorkerとして利用



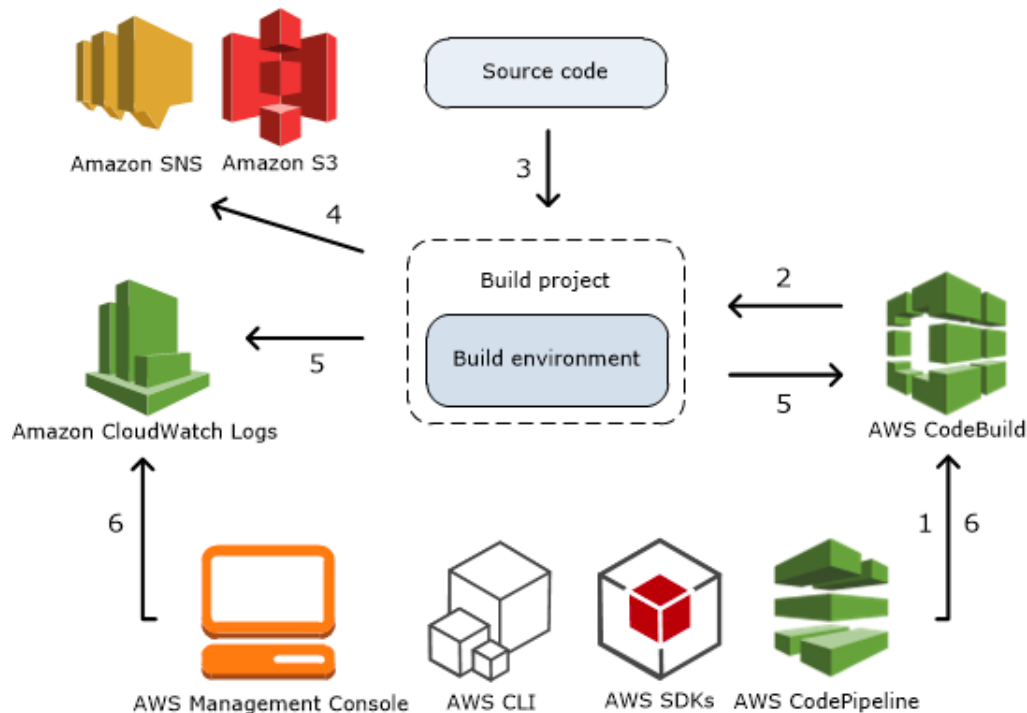
# AWS CodeBuild の実行

- ❖ AWS CodeBuild のコンソール
- ❖ AWS CodePipeline のコンソール
- ❖ AWS CLI
- ❖ AWS SDK



# AWS CodeBuild のコンセプト

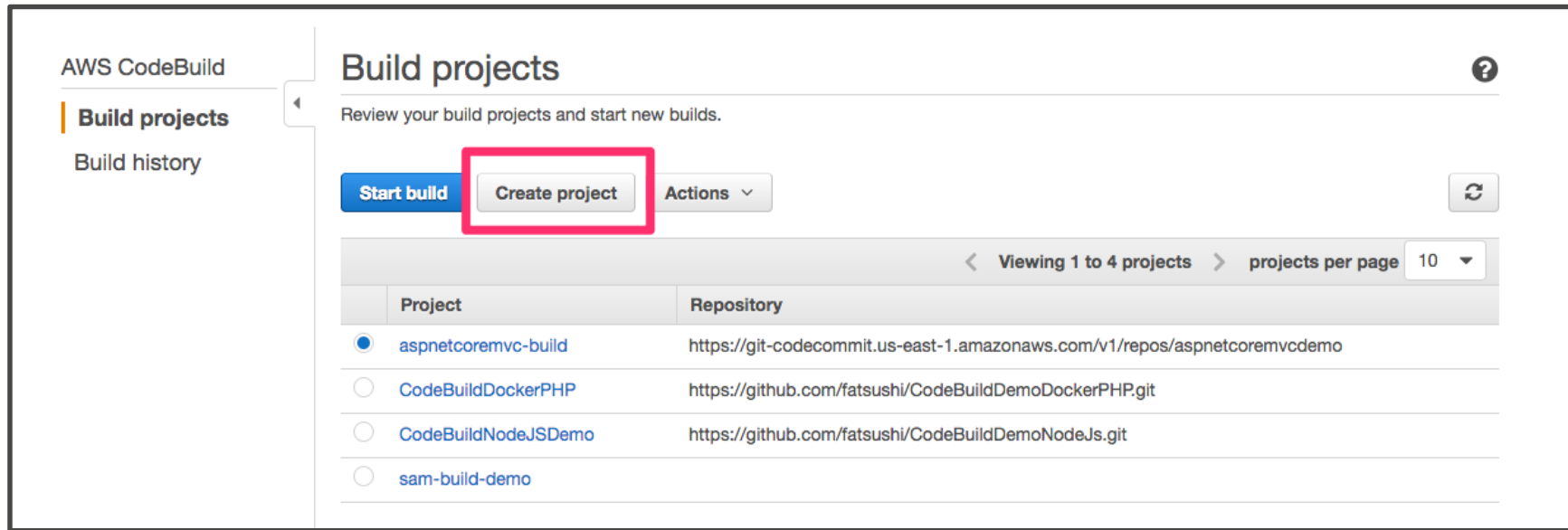
1. CodeBuild にビルド プロジェクトを作成
2. AWS CodeBuild はビルド プロジェクトに基づいてビルド環境を構築
3. AWS CodeBuildはソースコードをダウンロード
4. ビルドのアウトプットをS3へアップロード
5. ビルド実行中、AWS CodeBuild と AWS CloudWatch Logs に情報を送信
6. ビルド実行中、AWS CodeBuild コンソール、AWS CLI、AWS SDKs、AWS APIでビルド情報を取得



# AWS CodeBuild を実行するまえに

- ❖ ソースコードはどこから取得？
  - AWS CodeCommit、Amazon S3、GitHub
- ❖ 実行すべきビルド コマンドと実行順序は？
  - buildspec にダウンロードしたソースをどのようにアプトプットするかを記載
- ❖ どのようなランタイムとツールがビルドに必要な？
  - 言語はJava, Ruby, Python, Node.js? ビルドツールは Maven, Ant, コンパイラ実行? ビルドはGit, AWS CLIが必要? それとも他のツール?
  - AWS CodeBuildは Docker Imageを利用したビルド環境でビルドを実行する  
<http://docs.aws.amazon.com/codebuild/latest/userguide/build-env-ref.html#build-env-ref-available>

# AWS CodeBuild プロジェクトの作成



The screenshot shows the AWS CodeBuild console interface. On the left, there is a navigation menu with 'AWS CodeBuild' at the top, followed by 'Build projects' (which is selected and highlighted with an orange bar) and 'Build history'. The main content area is titled 'Build projects' and includes a subtitle 'Review your build projects and start new builds.' Below this, there are three buttons: 'Start build' (blue), 'Create project' (white with a red border), and 'Actions' (grey with a dropdown arrow). To the right of these buttons is a refresh icon. Below the buttons is a table with columns 'Project' and 'Repository'. The table shows four projects, with the first one selected (indicated by a blue radio button). The table also includes pagination controls: '< Viewing 1 to 4 projects >' and 'projects per page 10' with a dropdown arrow.

Project	Repository
<input checked="" type="radio"/> <a href="#">aspnetcoremvc-build</a>	<a href="https://git-codecommit.us-east-1.amazonaws.com/v1/repos/aspnetcoremvdemo">https://git-codecommit.us-east-1.amazonaws.com/v1/repos/aspnetcoremvdemo</a>
<input type="radio"/> <a href="#">CodeBuildDockerPHP</a>	<a href="https://github.com/fatsushi/CodeBuildDemoDockerPHP.git">https://github.com/fatsushi/CodeBuildDemoDockerPHP.git</a>
<input type="radio"/> <a href="#">CodeBuildNodeJSDemo</a>	<a href="https://github.com/fatsushi/CodeBuildDemoNodeJs.git">https://github.com/fatsushi/CodeBuildDemoNodeJs.git</a>
<input type="radio"/> <a href="#">sam-build-demo</a>	



# AWS CodeBuild プロジェクトの作成

## Create project

Step 1: Configure project  
Step 2: Review

### Configure your project

Specify settings for your build project.

Project name\*

Description [Add description](#)

Source: What to build

Source provider\*

Repository\*

Environment: How to build

Environment image\*  Use an image managed by AWS CodeBuild  
 Specify a Docker image

Operating system\*

Runtime\*

Version\*

Build specification  Use the buildspec.yml in the source code root directory  
 Insert build commands

Artifacts: Where to put the artifacts from this build project

Artifacts type\*

Service role

Specify a service role that enables AWS CodeBuild to call dependent AWS services on your behalf. [Learn more.](#)

Create a service role in your account  
 Choose an existing service role from your account

Role name\*

▶ Show advanced settings

\*Required

[Cancel](#) [Continue](#)

# AWS CodeBuild の実行

AWS CodeBuild

**Build projects**

Build history

## Build projects

Review your build projects and start new builds.

**Start build** Create project Actions

Refresh

AWS CodeBuild

Build projects

**Build history**

## Start new build

Choose the build project you want to use. Optionally, you can build a specific version of the source code, and you can override any of the build project's settings for this build only.

Project name\* CodeBuildNodeJSDemo

Source provider GitHub

Repository https://github.com/fatsushi/CodeBuildDemoNodeJs.git

Source version *Type your source version*

▶ Show advanced options

▶ Environment variables

\*Required

Cancel **Start build**

# AWS CodeBuild の実行

AWS CodeBuild

- Build projects
- Build history**

## CodeBuildNodeJSDemo:ba60e9c2-d82e-4077-afd6-d293f4adda7b Succeeded ?

Review your build details as it progresses.

### Build

**Build ARN** `arn:aws:codebuild:us-east-1:786032344772:build/CodeBuildNodeJSDemo:ba60e9c2-d82e-4077-afd6-d293f4adda7b`

**Build project** [CodeBuildNodeJSDemo](#)

**Source provider** GitHub

**Repository** <https://github.com/fatsushi/CodeBuildDemoNodeJs.git>

**Start time** Mar 2, 2017 2:23:25 AM UTC

**End time** Mar 2, 2017 2:24:50 AM UTC

**Status** Succeeded

**Initiator** fatsushi

▶ Build details

### Phase details

	Name	Status	Duration	Completed
▶	SUBMITTED	<span>Succeeded</span>		Mar 2, 2017 2:23:26 AM UTC
▶	PROVISIONING	<span>Succeeded</span>	1 min, 2 secs	Mar 2, 2017 2:24:29 AM UTC
▶	DOWNLOAD_SOURCE	<span>Succeeded</span>	4 secs	Mar 2, 2017 2:24:34 AM UTC
▶	INSTALL	<span>Succeeded</span>	9 secs	Mar 2, 2017 2:24:44 AM UTC
▶	PRE_BUILD	<span>Succeeded</span>		Mar 2, 2017 2:24:44 AM UTC

# AWS CodeBuild の履歴

AWS CodeBuild

## Build Project: CodeBuildNodeJSDemo

Build projects | Build history

Back Delete Edit project

Project

Project name CodeBuildNodeJSDemo  
Description AWS CodeBuild Project  
Source provider GitHub  
Repository https://github.com/fatsushi/codebuild-nodejs-demo  
Artifacts upload location fatsushi-codebuild-artifacts

Project details

Build history

Stop Start build

	Build run	Submitter
<input type="checkbox"/>	<a href="#">CodeBuildNodeJSDemo...</a>	fatsushi
<input type="checkbox"/>	<a href="#">CodeBuildNodeJSDemo...</a>	fatsushi
<input type="checkbox"/>	<a href="#">CodeBuildNodeJSDemo...</a>	fatsushi
<input type="checkbox"/>	<a href="#">CodeBuildNodeJSDemo...</a>	fatsushi

End time 3 minutes ago  
Status **Succeeded**  
Initiator fatsushi

Build details

Phase details

Name	Status	Duration	Completed
SUBMITTED	<b>Succeeded</b>		4 minutes ago
PROVISIONING	<b>Succeeded</b>	1 min, 5 secs	3 minutes ago
DOWNLOAD_SOURCE	<b>Succeeded</b>	6 secs	3 minutes ago
INSTALL	<b>Succeeded</b>	12 secs	3 minutes ago
PRE_BUILD	<b>Succeeded</b>		3 minutes ago
BUILD	<b>Succeeded</b>		3 minutes ago
POST_BUILD	<b>Succeeded</b>		3 minutes ago
UPLOAD_ARTIFACTS	<b>Succeeded</b>	1 sec	3 minutes ago
FINALIZING	<b>Succeeded</b>	3 secs	3 minutes ago
COMPLETED	<b>Succeeded</b>		

## Build logs

Showing the last 20 lines of build log below. [View entire log](#)

```
[Container] 2017/03/20 16:00:35  
[Container] 2017/03/20 16:00:35 Phase complete; PRE_BUILD Success: true  
[Container] 2017/03/20 16:00:35 Phase complete; BUILD Success: true
```

# buildspec.yml Example

```
version: 0.1

environment_variables:
  plaintext:
    JAVA_HOME: "/usr/lib/jvm/java-8-openjdk-amd64"

phases:
  install:
    commands:
      - apt-get update -y
      - apt-get install -y maven
  pre_build:
    commands:
      - echo Nothing to do in the pre_build phase...
  build:
    commands:
      - echo Build started on `date`
      - mvn install
  post_build:
    commands:
      - echo Build completed on `date`

artifacts:
  type: zip
  files:
    - target/messageUtil-1.0.jar
  discard-paths: yes
```

- ビルドの各フェーズで利用される環境変数
- 各フェーズで実行するコマンドの指定
  - “insutall”では環境を準備するためのパッケージのインストールやコマンドの実行など
  - ”pre build“では構文チェックやコマンドの実行など
  - “build”ではビルドツールやコマンドの実行など
  - “post build”ではテスト実行やレジトリへのコンテナイメージの配布
- アーティファクトを作成しS3に保存

# buildspec.yml (1/2)

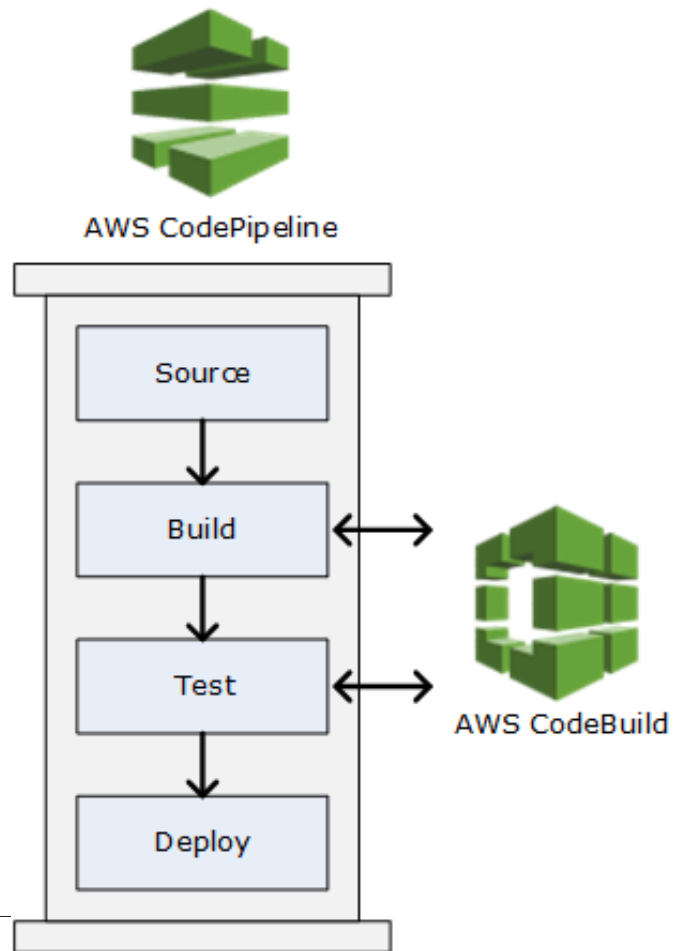
			説明
version			必須。0.1 の指定を推奨。
environment _variables	plaintext		オプション。カスタムの環境変数をキー/値で指定。同名の環境変数が既に存在する場合は上書きするので注意。CODEBUILD_で始まる環境変数は内部利用のため予約されている。
phases	install	command s	オプション。インストールフェーズで実行するコマンドを指定。例えば単体テストで利用するMochaやRSpecのインストールなど。
	pre_build	command s	オプション。ビルド前に実行するコマンドを指定。例えばAmazon ECRへのログインやnpmの依存関係のインストールなど。
	build	command s	オプション。ビルドで実行するコマンドを指定。例えばMochaやRSpecの実行など。記述された順番にひとつずつコマンドを実行。
	post_build	command s	オプション。ビルド後に実行するコマンドを指定。例えばビルドした成果物をWARやJARにパッケージ化するためにMavenを実行したり DockerイメージをECRにPushしたり、その後SNSにビルド状態を通知するなど。

# buildspec.yml (2/2)

			説明
artifacts	files		オプション。CodeBuildがビルド出力の場所を知り、Amazon S3へ成果物をアップロードする準備を行うために利用。 設定例: my-file.jar (単一ファイル) my-subdirectory/* (ディレクトリ以下のファイル) my-subdirectory/**/* (ディレクトリ以下のすべてのディレクトリとファイル) **/* (すべてのファイル)
	discard-paths		オプション。ビルド 出力成果物内のパスを保持するかどうかを指定。No(デフォルト)はpath情報を保持。
	base-directory		オプション。一つ以上のトップレベル ディレクトリを指定。指定されたディレクトリからの相対パスでマッチするファイル/ディレクトリを出力。

# AWS CodePipeline との連携

- ❖ CodePipeline の Build または Test アクションに追加可能
- ❖ 継続的インテグレーションの一部としてAWS CodeBuildを利用



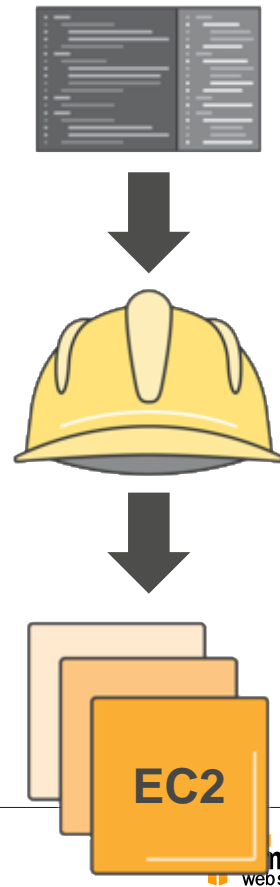


# 注意点

- 前のフェーズで実行に失敗した場合は、次のフェーズのコマンドは実行されない
- コマンドは個別の環境で実行されるため、直前のコマンドで設定した環境変数やディレクトリの変更は引き継がれない
  - シェル スクリプトを記述することでまとまったコマンドとして実行可能

# Building Your Code

- ❖ コードの「ビルド」は、一般に、コンパイル済みのバイナリを要求する言語を指す：
- ❖ .NET言語：C#、F#、Visual Basicなど
- ❖ Java言語とJVM言語：Java、Scala、JRuby
- ❖ Go
- ❖ iOS言語：Swift、Objective-C
- ❖ Dockerコンテナイメージを作成するプロセスも画像の「ビルド」と呼ぶ



# ビルド不要

- ❖ 最近の多くの言語はビルドが不要：
- ❖ 以下の言語はインタプリタ型言語と呼ばれる：
  - PHP
  - Ruby
  - Python
  - Node.js
- ❖ コードをデプロイするだけ！

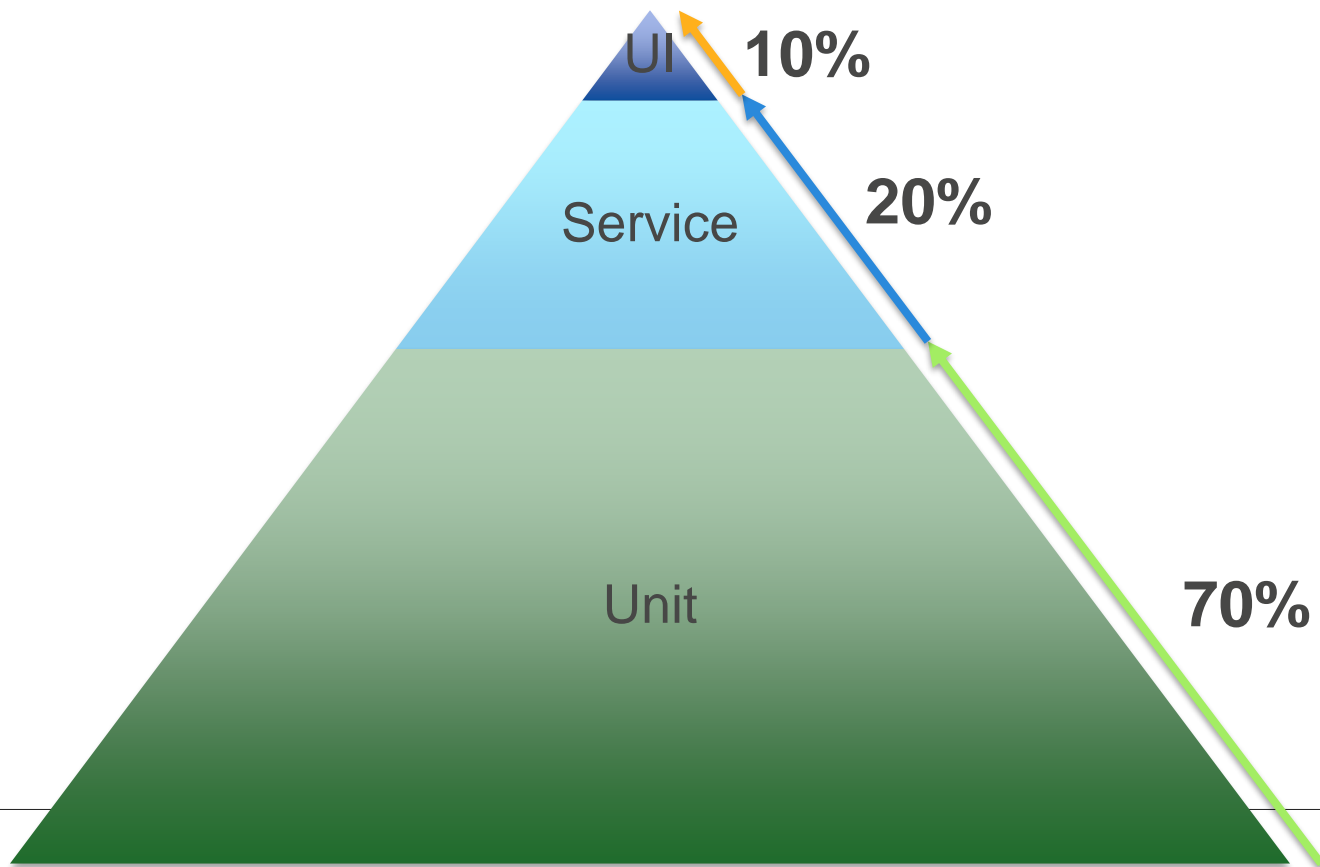


# コードのテスト

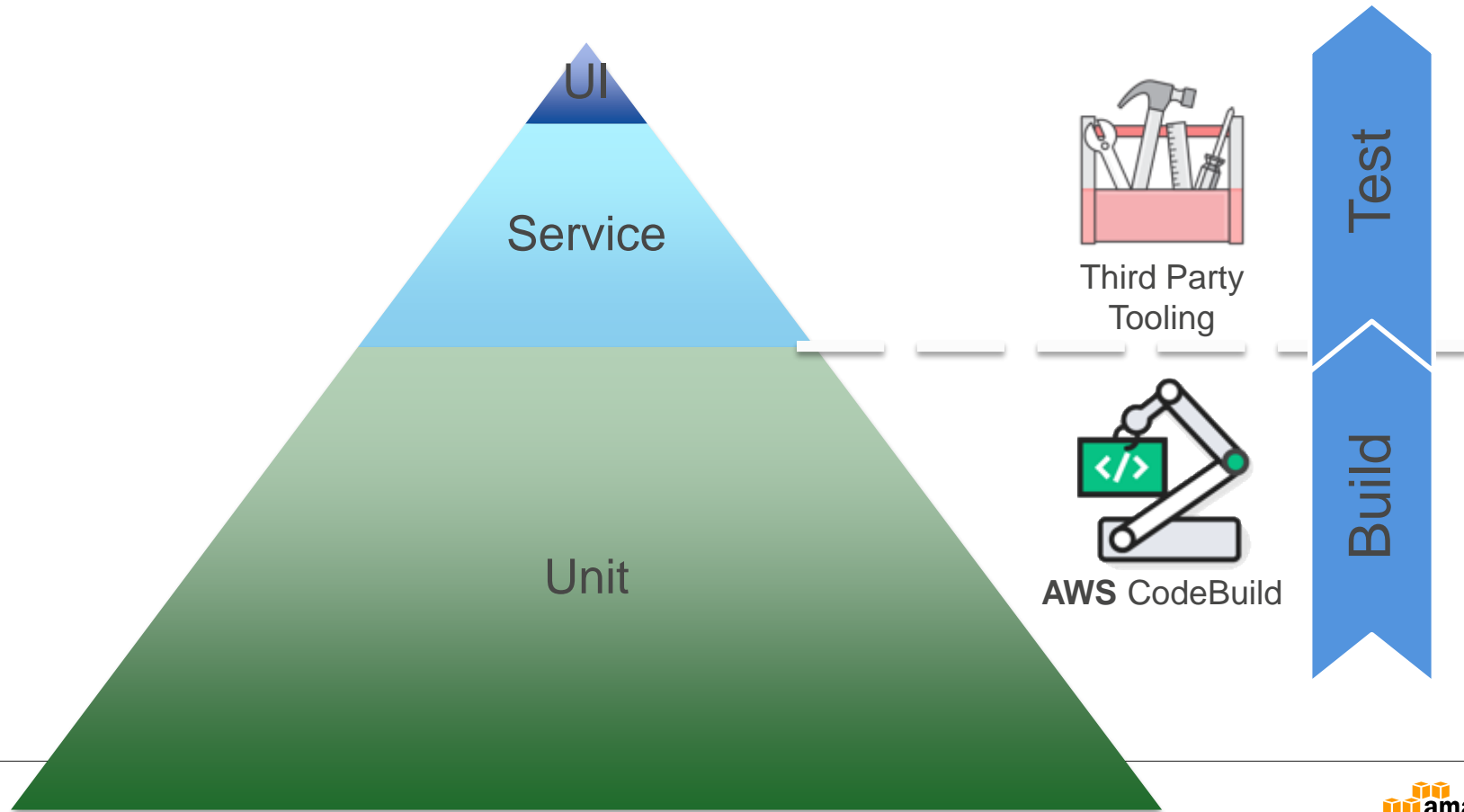
- ❖ テストは科学であると同時に芸術でもある！
- コードをテストする目的：
  - 期待どおりに機能することを確認
  - プログラミングの構文エラーを捕捉
  - コーディング パターンの標準化
  - アプリケーションの不適切な使用法やロジックの欠陥によるバグを削減
  - アプリケーションをよりセキュアにするため



# テストに焦点を当てる場所:



# サービスとリリース ステップがどのテストと関連するか？



# 価格

- 分単位の課金
- メモリとCPUリソースの量によって3つの異なるタイプを選択

Compute instance type	Memory (GB)	vCPU	Price per build minute (\$)
build.general1.small	3	2	0.005
build.general1.medium	7	4	0.010
build.general1.large	15	8	0.020

- build.general1.smallで100分/月の無料枠

A large, intricate wooden siege engine, possibly a trebuchet or a similar catapult, is mounted on a stone wall. The structure is made of dark, weathered wood and features a complex framework of beams and supports. A long, thick wooden arm extends from the structure, ending in a large, circular, lattice-like frame. The engine is positioned on a paved area, and the background shows a stone wall, a grassy area, and a view of the ocean under a cloudy sky. A few people are visible in the distance, walking along the wall. The overall scene suggests a historical or educational site.

# Deploying your applications



# AWS CodeDeploy



- ❖ **自動化されたデプロイメント**
  - ❖ 開発、テスト、本番環境への完全に自動化されたデプロイメント。AWS CodeDeployは数千台の環境にスケール可能
- ❖ **最小のダウンタイム**
  - ❖ インプレース デプロイメントではAmazon EC2インスタンスに対してローリング アップデートが可能
  - ❖ ブルー/グリーン デプロイメントでは最新のリリースをインスタンスにデプロイしてトラフィックを即時またはテスト完了後すぐに新しい環境にリルート
- ❖ **停止とロールバック**
  - ❖ 自動的またはマニュアルで停止とロールバックが可能
- ❖ **センターコントロール**
  - ❖ AWS CodeDeployコンソール及びAWS CLIでステータスのトラックが可能
- ❖ **容易な適用**

A black and white historical photograph showing several workers in a deep trench. They are working with a large, corrugated metal pipe that runs diagonally across the frame. One worker in the foreground is using a tool to adjust the pipe's position. Another worker is visible further down the trench, and a third person's legs are seen standing on top of the pipe in the background. The trench walls are rough and uneven. The overall scene depicts a manual construction or installation process.

# Orchestrating build and deploy with a pipeline

# AWS CodePipeline



- ❖ リリース プロセスを自動化
  - ❖ ソースリポジトリからビルド、テスト、デプロイメントまで完全に自動化されたリリース プロセス
- ❖ 一環したリリース プロセスを確立
- ❖ 改善された品質でデリバリをスピードアップ
  - ❖ リリース プロセスを自動化し、インクリメンタルな機能追加を可能とすることで素早くリリース
- ❖ お気に入りのツールを利用可能
  - ❖ 既存のソース、ビルド、デプロイメントのツールをパイプラインに組み込み
- ❖ 進捗状況が一目瞭然

# まとめ

- ❖ AWS CodeCommit はプライベートで無制限のリポジトリを容易に構築可能
- ❖ AWS CodeBuild はフルマネージドなビルド環境を提供
- ❖ これらを活用することで高品質なソフトウェアをインクリメンタルな開発手法で素早くリリース！

# aws.amazon.com/devops



Menu



[AWS re:Invent](#)

[Products](#) ▾

[Solutions](#)

[Pricing](#)

[More](#) ▾

[English](#) ▾

[My Account](#) ▾

[Sign In to the Console](#)



## DevOps and AWS

Tooling and infrastructure resources for DevOps practitioners

[Get Started with AWS](#)

[What is DevOps?](#)

[DevOps Blog](#)

[Partner Solutions](#)

[Resources](#)

AWS provides a set of flexible services designed to enable companies to more rapidly and reliably build and deliver products using AWS and DevOps practices. These services simplify provisioning and managing infrastructure, deploying application code, automating software release processes, and monitoring your application and infrastructure performance.

DevOps is the combination of cultural philosophies, practices, and tools that increases an organization's ability to deliver applications and services at high velocity: evolving and improving products at a faster pace than organizations using traditional software development and infrastructure management

[AWS DevOps Blog](#)

## Building a Cross-Region/Cross-Account Code Deployment Solution on AWS

by BK Chaurasiya | on 01 NOV 2016 | in [How-To](#) | [Permalink](#) | [Comments](#)

Many of our customers have expressed a desire to build an end-to-end release automation workflow solution that can deploy changes across multiple regions or different AWS accounts.

In this post, I will show you how you can easily build an automated cross-region code deployment solution using [AWS CodePipeline](#) (a [continuous delivery](#) service), [AWS CodeDeploy](#) (an automated application deployment service), and [AWS Lambda](#) (a serverless compute service). In the Taking This Further section, I will also show you how to extend what you've learned so that you can create a cross-account deployment solution.

We will use AWS CodeDeploy and AWS CodePipeline to create a multi-pipeline solution running in two regions (Region A and Region B). Any update to the source code in Region A will trigger validation and deployment of source code changes in the pipeline in Region A. A successful processing of source code in all of its AWS CodePipeline stages will invoke a Lambda function as a custom action, which will copy the source code into an S3 bucket in Region B. After the source code is copied into this bucket, it will trigger a similar chain of processes into the different AWS CodePipeline stages in Region B. See the following diagram



Search the DevOps Blog

Search

Resources

[AWS Developer Tools](#)

[AWS Management Tools](#)

[DevOps and AWS](#)

[Resources for DevOps on AWS](#)

# AWS CodePipeline, AWS CodeBuild, Amazon ECR, AWS CloudFormationを利用したAmazon ECSへの継続的デプロイメント

## ECS Continuous Deployment

View progress and manage your pipeline.

**Edit** **Release change**

**Source**

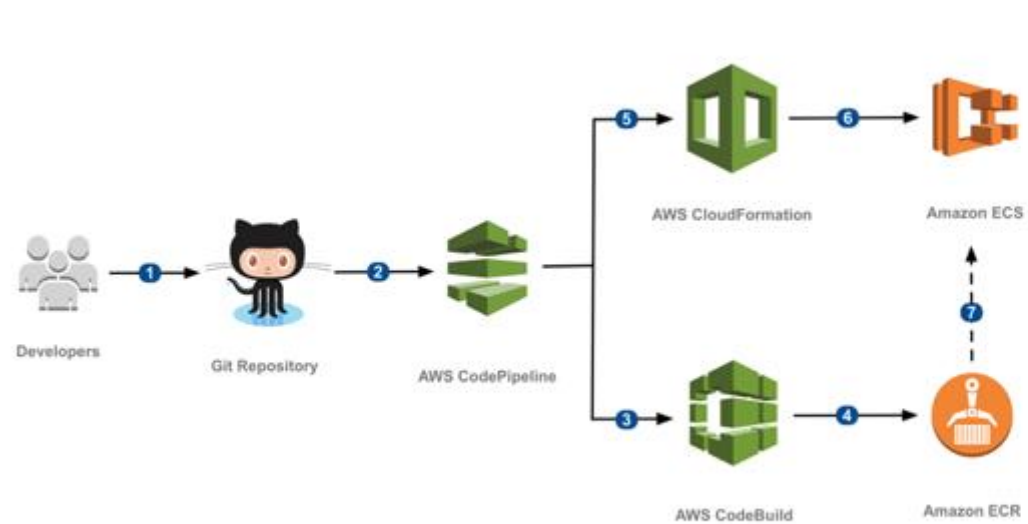
- App: **GitHub** (Succeeded 8 min ago)
- Template: **Amazon S3** (Succeeded 8 min ago)
- App: Update index.php  
Template: Amazon S3 version id: 6F0ppvxH0WtXq4Rdtrsy...

**Build**

- Build: **AWS CodeBuild** (Succeeded 3 min ago)
- App: Update index.php  
Template: Amazon S3 ver...

**Deploy**

- Deploy: **AWS CloudFormation** (Succeeded just now)
- App: Update index.php  
Template: Amazon S3 ver...



- 1 Developers continually integrate their changes together into a main branch hosted within a source code repository system such as GitHub.
- 2 AWS CodePipeline polls the source code repository and triggers an execution of the continuously delivery pipeline when a new revision is found.
- 3 AWS CodePipeline sends the new revision to AWS CodeBuild which builds a Docker container image from the source code.
- 4 AWS CodeBuild pushes the newly built Docker container image tagged with the build ID to an Amazon ECR repository.
- 5 AWS CodePipeline initiates an update of the AWS CloudFormation stack which defines the Amazon ECS task definition and service.
- 6 AWS CloudFormation creates a new task definition revision referencing the newly built image and updates the Amazon ECS service.
- 7 Amazon ECS fetches the new container from Amazon ECR and replaces the old task with the new one which completes the deployment.



<http://amzn.to/2I1PMYh>



