



AWS DevOps祭り 2018

AWS Developer Tools Update

Atsushi Fukui
Solutions Architect
DevOps Specialist
Amazon Web Services Japan K.K.

2018.12.03

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

❖ 技術統括本部レディネスソリューション部

❖ ソリューション アーキテクト

Dev & Opsスペシャリスト

❖ 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

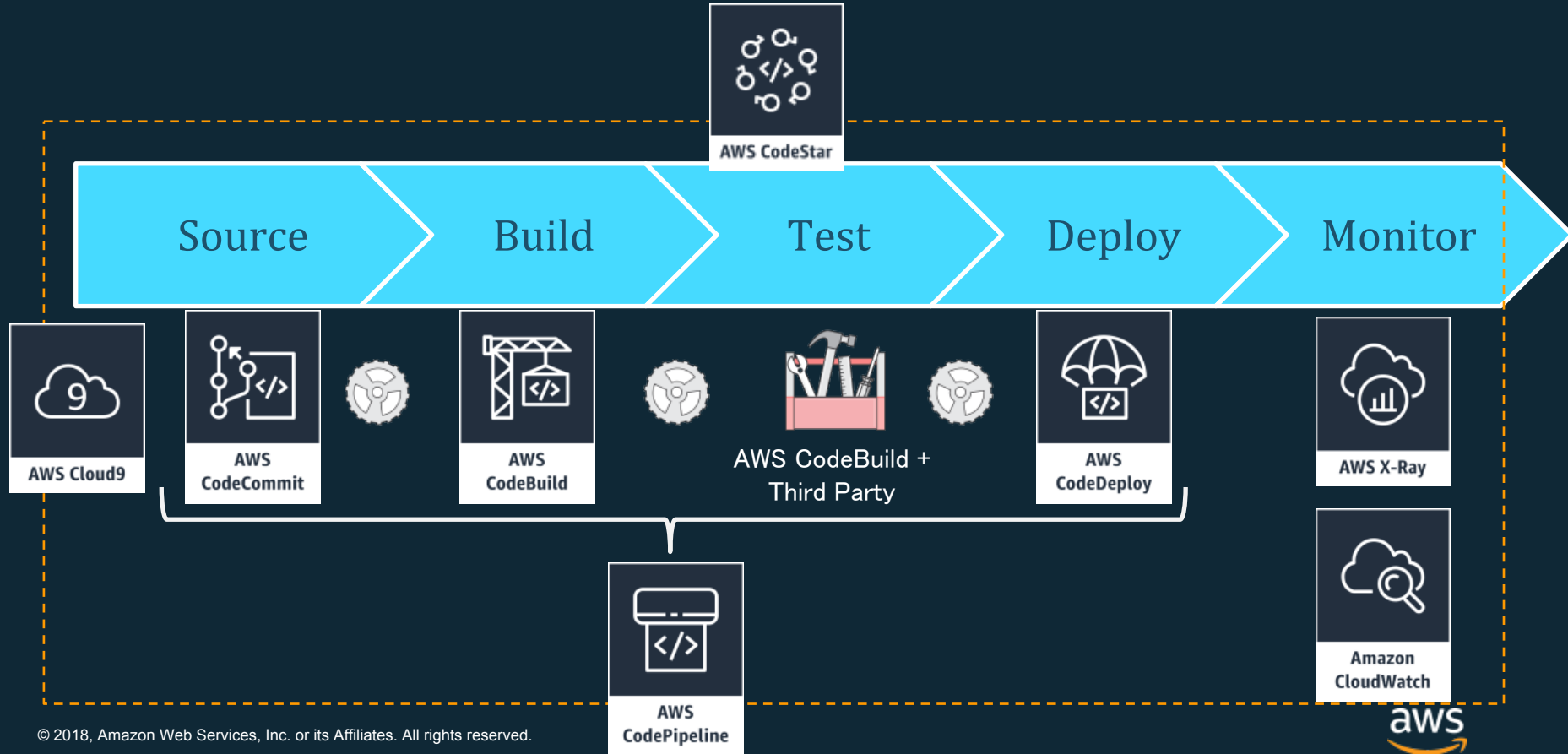
❖ AWS Code シリーズ、AWS Cloud9、Amazon ECS/EKS、AWS Lambda



このセッションの内容

- AWS Developer Tools サービスとは
- re:Invent 2018で発表された最新アップデート
- re:Invent 2018以前に発表されたアップデート

AWS Developer Tools とは



AWS Cloud9



- クラウドベースの統合された開発環境 (IDE)
- ブラウザのみでコーディング、実行、デバッグ
- リアルタイムのペアプログラミングを可能にするチームでの環境共有
- AWSへのダイレクトなターミナルアクセス
- サーバーレスの優れた開発環境を提供: ローカルテストとSDK, ライブラリ、プラグインが事前設定されたデプロイ環境

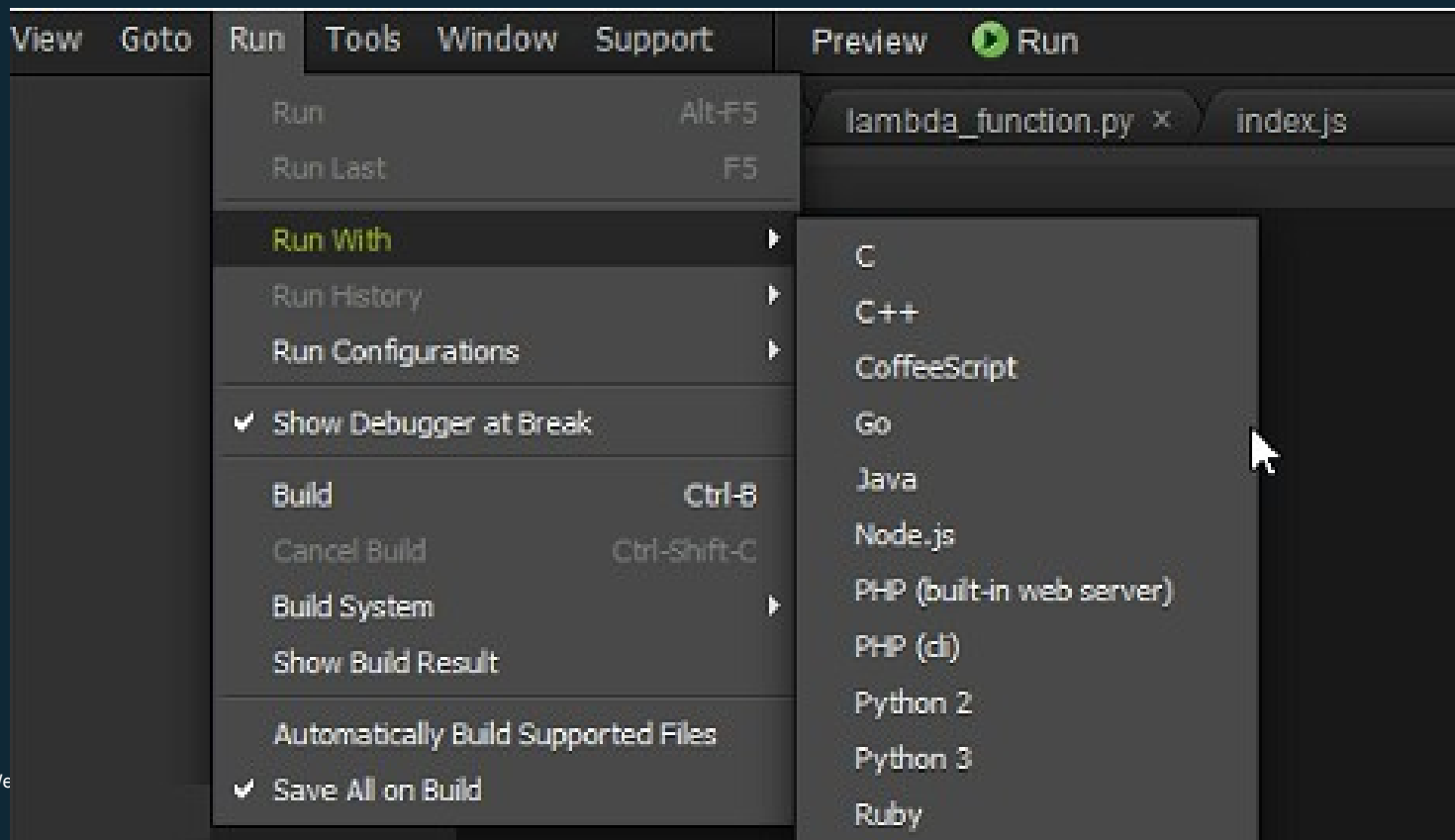
フル機能のエディタ

The image displays the Cloud9 IDE interface with two code panels. The left panel shows a JavaScript file named `index.js` with a dropdown menu for code completion under the `handlers` object. The right panel shows a Python file named `lambda_function.py` with a red error message: "expected an indented block (<string>, line 11)".

Annotations in the image include:

- Multiple panels**: Points to the split view of the IDE.
- Code hinting**: Points to the error message in the Python code.
- Code autocomplete**: Points to the dropdown menu in the JavaScript code.

幅広いランタイムの選択肢



フル機能のデバッガ

Breakpoint

The screenshot shows the VS Code debugger interface. The code editor displays the following JavaScript code:

```
1 //a javascript program to add two variables
2
3 function addVars (first, second) {
4   var sum = first + second;
5   return sum;
6 }
7
8 var firstVar = 1;
9 var secondVar = 2;
10
11
12 var total = addVars(firstVar, secondVar);
13
14 console.log(total);
15
```

A red dot breakpoint is set on line 9. The local variables panel shows the following variables:

Variable	Value	Type
__dirname	"/home/ec2-user/environment/NodeJS"	string
__filename	"/home/ec2-user/environment/NodeJS/debugging.js"	string
addVars	function ()	function
exports	[Object]	object
firstVar	1	number
module	[Object]	object
require	function ()	function
secondVar	undefined	undefined
this	[Object]	object
total	undefined	undefined
Scope		global

The call stack shows the current function call: debugging.js:9.

Debugging controls

Debugging panel

AWS CodeStar



- AWS上でのアプリケーションの素早い開発、ビルド、デプロイ
- 数分でAWS上での開発を開始
- 安全にチームを横断して作業
- ソフトウェア デリバリーを容易に管理
- 多様なプロジェクトテンプレートから選択

AWS CodeStarのテンプレート

The screenshot displays the AWS CodeStar console interface. At the top, there are navigation elements: 'Services', 'Resource Groups', a user profile 'cdhuil @ 0025-6330-0498', 'Select a Region', and 'Support'. Below this is a progress bar with three steps: 'Select Template' (highlighted in blue), 'Setup Tools', and 'Start Coding'.

The main content area is titled 'Select an application template to get started. [Help me choose](#)'. On the left, there is a 'Refine by' sidebar with the following sections:

- Application category: Web application, Web service
- Programming languages: Ruby, Node.js, Java, Python, PHP, HTML 5
- AWS services: AWS Elastic Beanstalk, Amazon EC2, AWS Lambda

The main area displays a grid of application templates. Each template card includes an icon, the application name, a category (Web application), and the AWS service used for deployment. The visible templates are:

- Ruby on Rails** (Web application):
 - Option 1: AWS Elastic Beanstalk (runs in a managed application environment)
 - Option 2: Amazon EC2 (runs on virtual servers that you manage)
- Java Spring** (Web application):
 - Option 1: AWS Elastic Beanstalk (runs in a managed application environment)
 - Option 2: Amazon EC2 (runs on virtual servers that you manage)
- Node.js** (Web application):
 - Option 1: AWS Lambda (running serverless)
 - Option 2: AWS Elastic Beanstalk (runs in a managed application environment)
 - Option 3: Amazon EC2 (runs on virtual servers that you manage)
- Python (Django)** (Web application):
 - Option 1: AWS Elastic Beanstalk (runs in a managed application environment)
- Express.js** (Web application):
 - Option 1: Amazon EC2 (runs on virtual servers that you manage)
- PHP (Laravel)** (Web application):
 - Option 1: AWS Elastic Beanstalk (runs in a managed application environment)

At the bottom of the console, there are links for 'Feedback', 'English', and a footer with copyright information: '© 2008 - 2017, Amazon Web Services, Inc. or its affiliates. All rights reserved.' along with 'Privacy Policy' and 'Terms of Use' links.

AWS CodeStar Project

チームwiki

Cloud9 Environment

管理コンソール
へのリンク

The screenshot shows the AWS CodeStar Project console for a project named 'demo-py-lmd'. The interface is divided into several sections:

- Team wiki tile:** A section for creating and editing team wiki pages. It includes a list of instructions for setting up the project.
- Commit history:** A table showing recent commits to the 'master' branch. The commits are:
 - refactoring (943dd16)
 - for api gateway (5ba05a9)
 - adjust api gateway param (10c4e8f)
 - update lambda versioning (73f3849)
 - modify return json (33a1efb)
- Application activity:** A graph showing application activity over time, with a y-axis from 0.0 to 1.00 and an x-axis from 14:00 to 13:00.
- AWS Cloud9 environments:** A section for managing Cloud9 environments, with a 'See my environments' button.
- Application endpoints:** A section for managing application endpoints, with a URL: `https://b70wh2xl8.execute-api.us-east-1.amazonaws.com/Prod/`.
- Continuous deployment:** A section for managing continuous deployment, with a 'Release change' button and a pipeline history showing stages: Source, Build, and Deploy.

API GW endpoint

CodePipeline ステージ

CodeCommit 履歴

CloudWatch モニタ

AWS CodeCommit



- セキュアでスケーラブルなマネージドGitソース管理
- スタンダードなGit tool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
- カスタマ特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
- Post commit hooks で SNS/Lambdaを呼び出せる
- Pull Request サポート
- ブランチごとの権限管理
- マネジメント コンソール上での編集、コミット

AWS CodeBuild



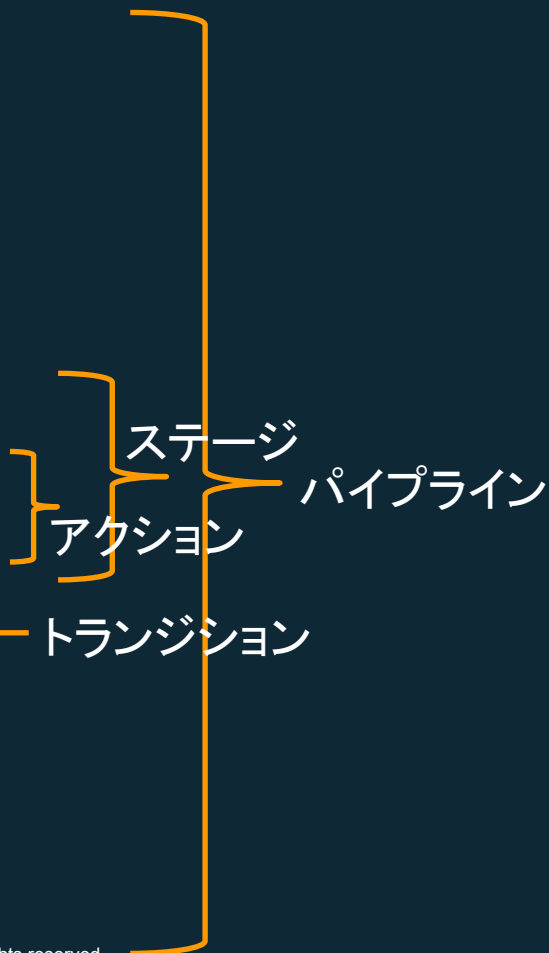
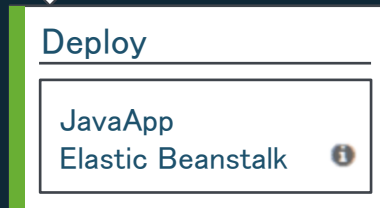
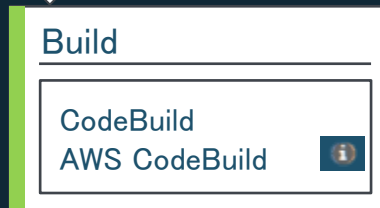
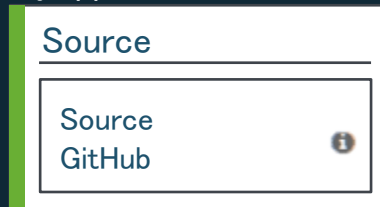
- 完全なマネージドのビルドサービスでソースコードのコンパイル、実行、テスト、ソフトウェア パッケージの生成をサポート
- 継続的なスケールと同時複数ビルドの実行
- Dockerイメージによってニーズにマッチするカスタムなビルド環境を構築可能
- 利用したコンピュータ リソース/分のみ支払い
- CodePipelineやJenkinsとの統合が可能
- VPC内のリソースへのアクセス、VPCエンドポイントの提供
- ローカル環境での実行とデバッグのサポート

AWS CodePipeline



- アプリケーションのすばやく信頼できるアップデートを可能にする**継続的デリバリサービス**
- ソフトウェアリリースプロセスの**モデル化と見える化**
- コードが変更されるたびにコードをビルド、テスト、デプロイ
- サードパーティツールやAWSとの統合

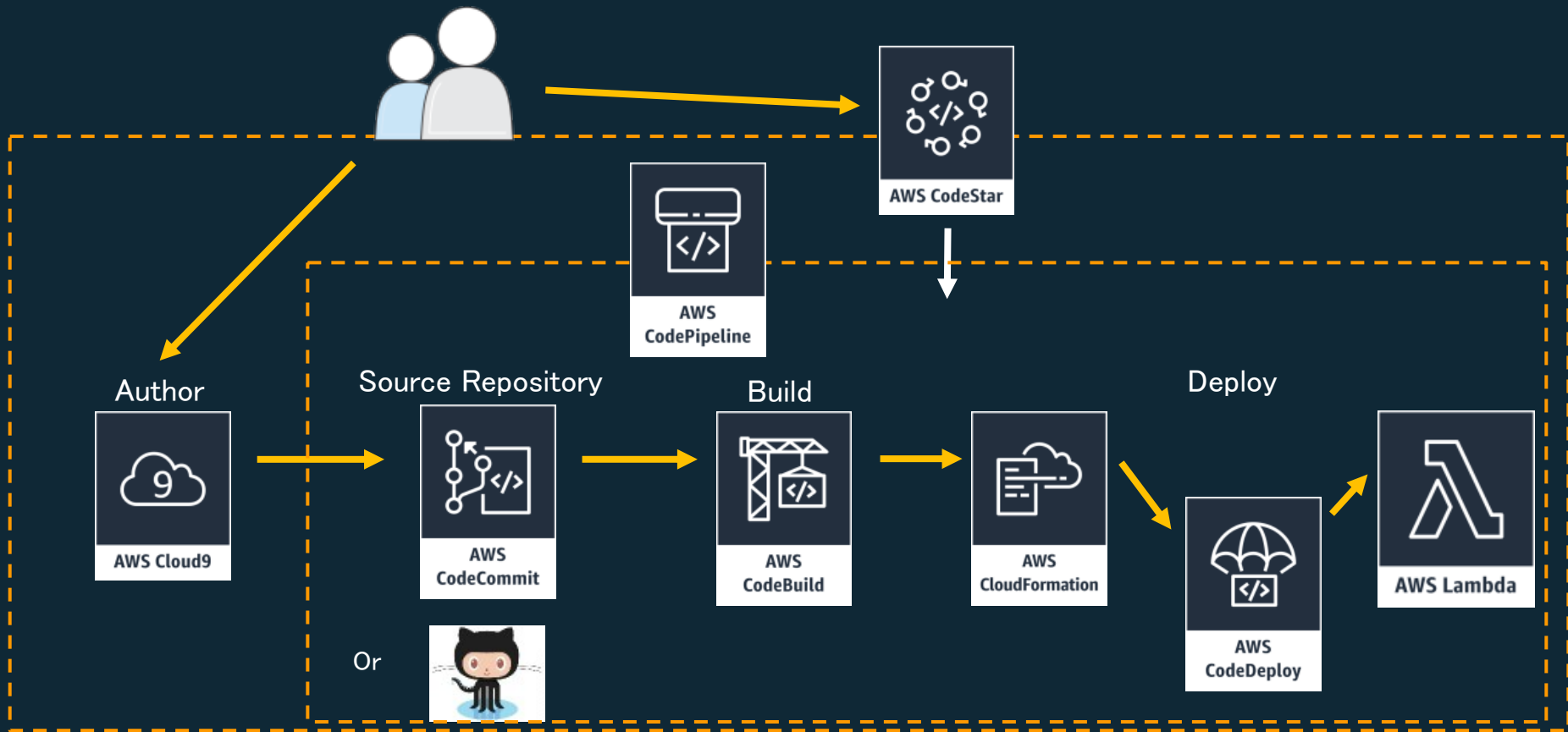
MyApplication



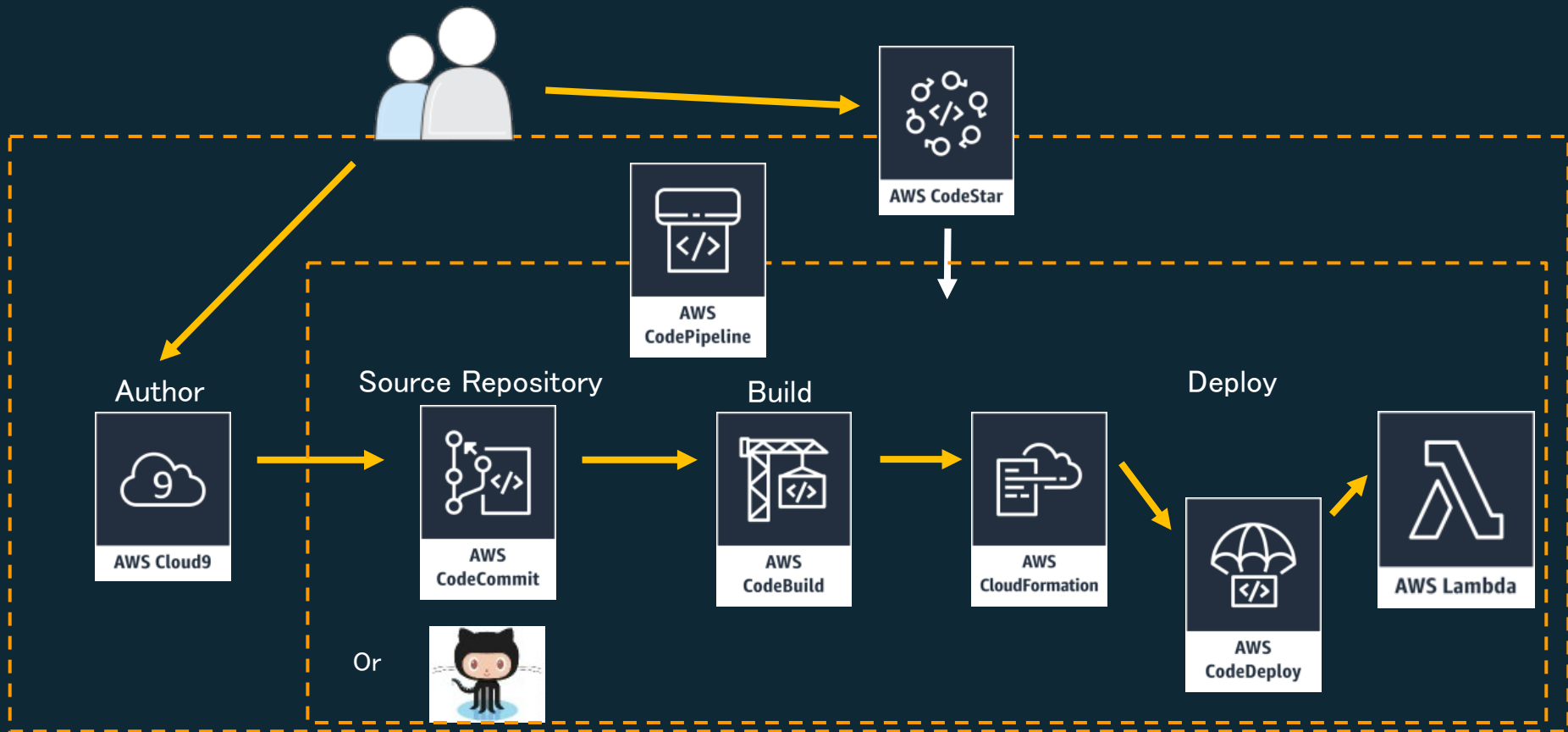
Demo

AWS CodeStarによるAWS Lambda ファンクションのデプロイメントの自動化

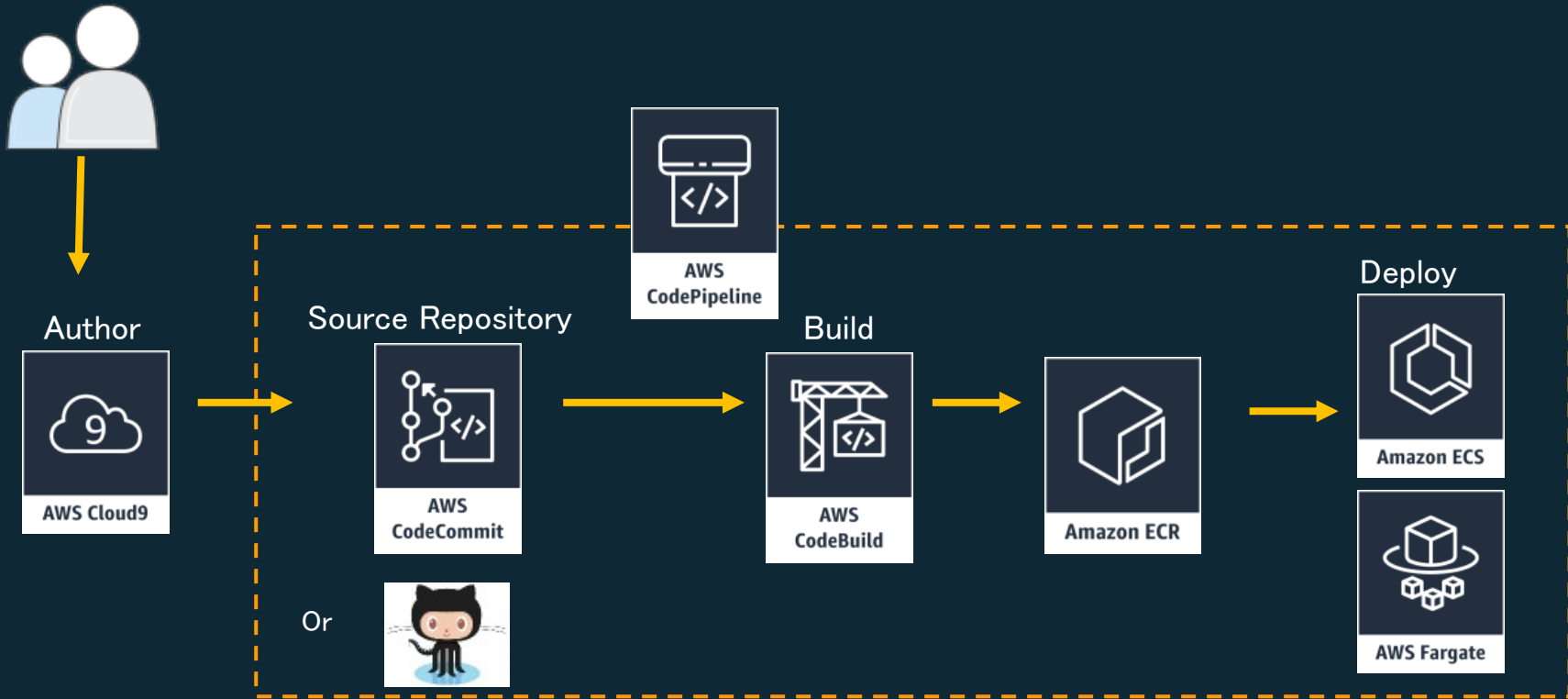
AWS CodeStarがサーバーレスアプリのCI/CDを自動生成



AWS CodeStarがサーバーレスアプリのCI/CDを自動生成



AWS CodePipelineがECS/Fargateへデプロイ

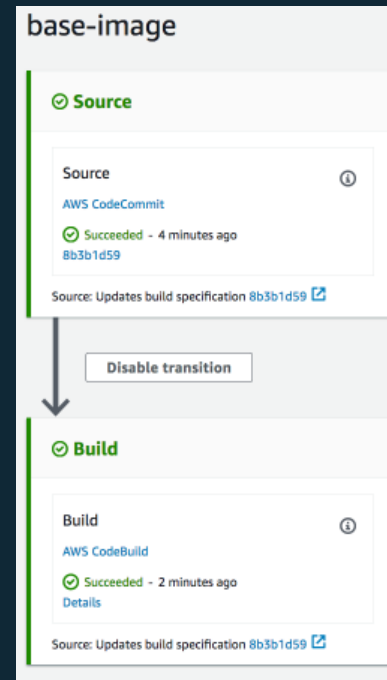


AWS Developer Tools Service Update

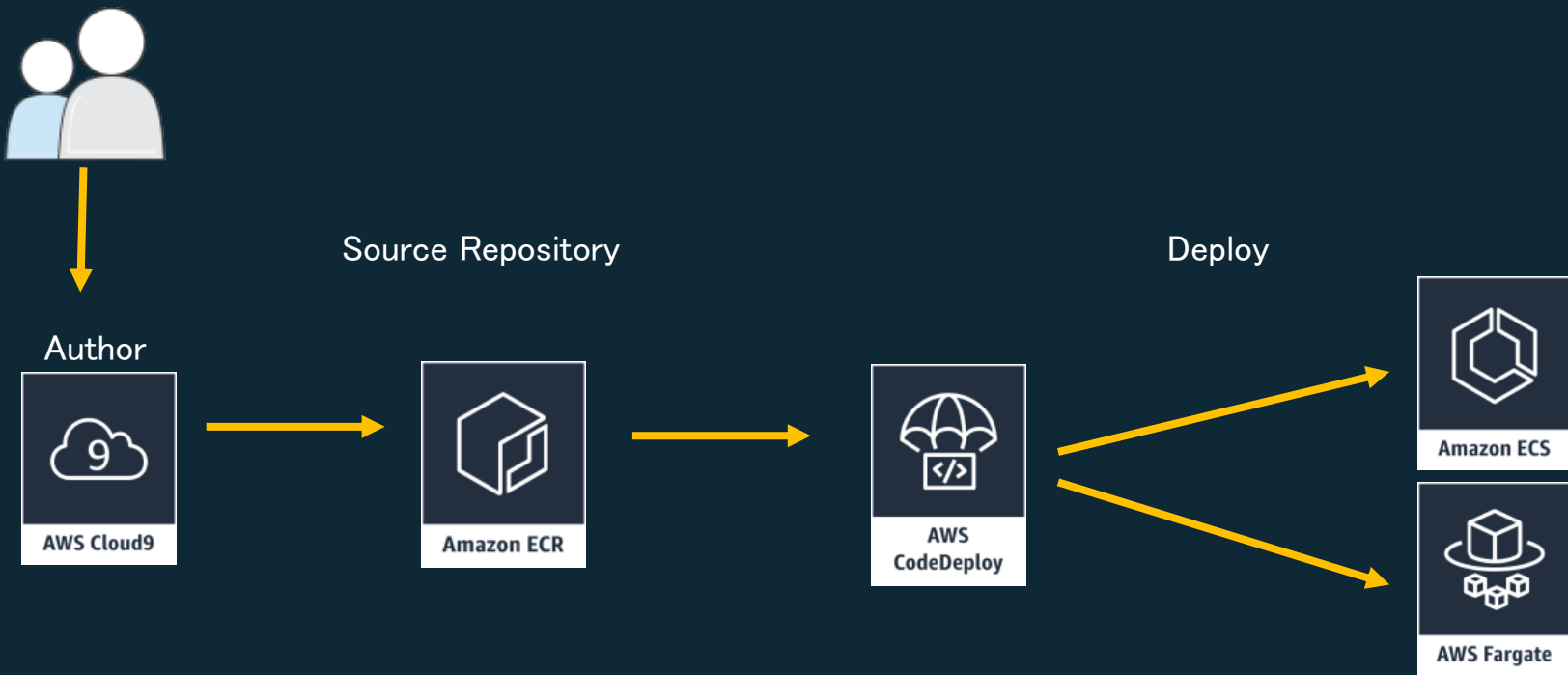
re:Invent 2018で発表されたアップ デート

AWS CodeDeployがECS/FargateへのBlue/Greenデプロイをサポート

- AWS CodeDeployでAmazon ECSとAWS FargateへのBlue/Greenデプロイメントが可能に
- これにより以下のフローを実装できるようになった
 - ECRにイメージをプッシュ
 - CodePipelineが起動しパイプラインを開始
 - CodeBuildでビルドを実行
 - CodeDeployで本番環境にBlue/Greenデプロイ
- 東京を含む各リージョンで利用可能



AWS CodeDeployがECS/Fargateへデプロイ



Demo

Amazon ECS CodeDeploy IAM Role

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "ecs:DescribeServices",
        "ecs:CreateTaskSet",
        "ecs:UpdateServicePrimaryTaskSet",
        "ecs>DeleteTaskSet",
        "elasticloadbalancing:DescribeTargetGroups",
        "elasticloadbalancing:DescribeListeners",
        "elasticloadbalancing:ModifyListener",
        "elasticloadbalancing:DescribeRules",
        "elasticloadbalancing:ModifyRule",
        "lambda:InvokeFunction",
        "cloudwatch:DescribeAlarms",
        "sns:Publish",
        "s3:GetObject",
        "s3:GetObjectMetadata",
        "s3:GetObjectVersion"
      ],
      "Resource": "*",
      "Effect": "Allow"
    }
  ]
}
```


- Amazon ECS blue/greenデプロイを実現するためには、CodeDeployサービスに下記のアクセス許可を与える必要がある
 - Amazon ECSアップデート
 - ELBアップデート
 - Lambdaファンクションの実行
 - LoudWatchアラームの記述
 - SNSパブリッシュ
 - S3オブジェクトの取得


AWS CodeDeploy コンソール

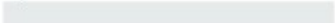
d-6BCOF7NVW

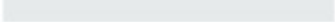
Stop deployment Stop and roll back deployment

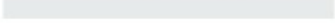
Deployment status

Step 1:
Deploying replacement task set
 In progress



Step 2:
Test traffic route setup Not started


Step 3:
Rerouting production traffic to replacement task set Not started


Step 4:
Wait 1 hour 0 minutes Not started


Step 5:
Terminate original task set Not started


Traffic shifting progress

Original	Replacement
 100%	 0%
Original task set serving traffic	Replacement task set not serving traffic

AWS CodeDeploy コンソール

d-6BCOF7NVW Stop deployment Stop and roll back deployment

Deployment status

Step 1:
Deploying replacement task set

Step 2:
Test traffic route setup

Step 3:
Rerouting production traffic to replacement task set

Step 4:
Wait 1 hour 0 minutes

Step 5:
Terminate original task set

Task set activity

Task set ID	Environment	Task set status	Traffic	Desired count	Running count	Pending count
ecs-svc/9223370493085028225	Replacement	PRIMARY	100%	2	2	0
ecs-svc/9223370493086082291	Original	ACTIVE	0	2	2	0

Deployment lifecycle events

Event	Duration	Status	Start time	End time
BeforeInstall	less than one second	✔ Succeeded	Dec 3, 2018 1:55 AM	Dec 3, 2018 1:55 AM
Install	2 minutes 3 seconds	✔ Succeeded	Dec 3, 2018 1:55 AM	Dec 3, 2018 1:57 AM
AfterInstall	less than one second	✔ Succeeded	Dec 3, 2018 1:57 AM	Dec 3, 2018 1:57 AM
AllowTestTraffic	less than one second	✔ Succeeded	Dec 3, 2018 1:57 AM	Dec 3, 2018 1:57 AM
AfterAllowTestTraffic	less than one second	✔ Succeeded	Dec 3, 2018 1:57 AM	Dec 3, 2018 1:57 AM
BeforeAllowTraffic	less than one second	✔ Succeeded	Dec 3, 2018 1:57 AM	Dec 3, 2018 1:57 AM
AllowTraffic	less than one second	✔ Succeeded	Dec 3, 2018 1:57 AM	Dec 3, 2018 1:57 AM
AfterAllowTraffic	less than one second	✔ Succeeded	Dec 3, 2018 1:57 AM	Dec 3, 2018 1:57 AM

AWS CodeDeploy コンソール

The screenshot displays the AWS CodeDeploy console for deployment **d-6BCOF7NVW**. The interface is divided into several sections:

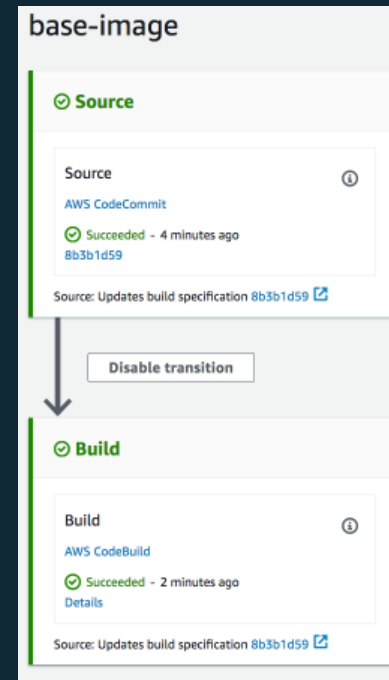
- Deployment status:** Shows the overall progress of the deployment. The current step is **Step 4: Wait 1 hour 0 minutes**, which is in a **Waiting** state. Previous steps (Step 1 to Step 3) are **Completed** and **Succeeded**.
- Task set activity:** Lists the task sets used in the deployment. Two task sets are shown, both with IDs `ecs-svc/9223370493085028225` and `ecs-svc/9223370493086082291`.
- Deployment lifecycle events:** A table showing the sequence of events during the deployment. The events include `BeforeInstall`, `Install`, `AfterInstall`, `AllowTestTraffic`, `AfterAllowTestTraffic`, `BeforeAllowTraffic`, `AllowTraffic`, and `AfterAllowTraffic`.
- Traffic shifting progress:** A bar chart showing the percentage of traffic shifted from the original task set to the replacement task set. The original task set is at **0%** (not serving traffic), and the replacement task set is at **100%** (serving traffic).

Buttons for **Stop deployment**, **Stop and roll back deployment**, and **Terminate original task set** are visible at the top of the console.



AWS CodePipelineがECRに対応

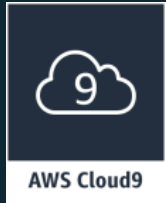
- AWS CodePipelineでAmazon ECRをソースプロバイダとして利用できるようになった
- ECRにイメージをプッシュしてAWS CodePipelineを起動しパイプラインを開始可能に
- 東京を含む各リージョンで利用可能



AWS CodePipelineがECRに対応



Author



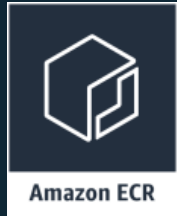
AWS Cloud9



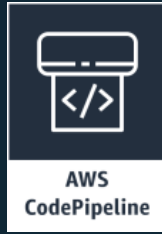
Source Repository



AWS CodeCommit

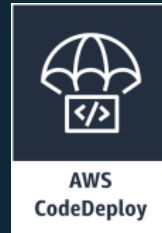


Amazon ECR

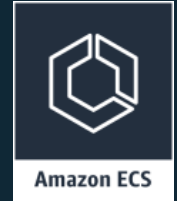


AWS CodePipeline

Deploy



AWS CodeDeploy



Amazon ECS



AWS Fargate

Demo

AWS CodePipelineの設定

- ALB、ECSのタスク定義、サービス作成に加えて。。。
- taskdef.jsonとappspec.yamlの作成
 - Placeholderの設定あり
- AWS CodeCommitに上記ファイルを登録
- CodePipelineでパイプラインの作成
- Source : CodeCommit
- Deploy:ECS(Blue/Green)
- パイプラインの編集でSource : ECRを追加
- taskdef.jsonとappspec.yaml、ECR Imageの取得を設定
- 詳細は下記のURLを参照

<https://docs.aws.amazon.com/codepipeline/latest/userguide/tutorials-ecs-ecr-codedeploy.html#tutorials-ecs-ecr-codedeploy-taskdefinition>

AWS CodePipelineコンソール

Action name
Choose a name for your action

No more than 100 characters

Action provider

 ▼

AWS CodeCommit

Repository name

Choose a repository that you have already created where you have pushed your source code.

 ▼

Branch name

Choose a branch of the repository

 ▼

Change detection options

Choose a detection mode to automatically start your pipeline when a change occurs in the source code.

Amazon CloudWatch Events (recommended)
Use Amazon CloudWatch Events to automatically start my pipeline when a change occurs

AWS CodePipeline
Use AWS CodePipeline to check periodically for changes

Output artifacts

Choose a name for the output of this action.

No more than 100 characters

AWS CodePipelineコンソール

Action name
Choose a name for your action

Source

No more than 100 characters

Action provider

AWS CodeCommit

AWS CodeCommit

Repository name
Choose a repository that you have access to

aspnetcoremvc-repo

Branch name
Choose a branch of the repository

master

Change detection options
Choose a detection mode for your code.

Amazon CloudWatch (recommended)
Use Amazon CloudWatch to automatically start a pipeline when a change occurs

Output artifacts
Choose a name for the output of this action.

SourceArtifact

No more than 100 characters

Action name
Choose a name for your action

Image

No more than 100 characters

Action provider

Amazon ECR

Amazon ECR

Repository name
Choose an Amazon ECR repository as the source location.

aspnetcoremvc

Image tag - optional
Choose the image tag that triggers your pipeline when a change occurs in the image repository.

latest

If an image tag is not selected, defaults to latest

Output artifacts
Choose a name for the output of this action.

MyImage

No more than 100 characters

AWS CodePipelineコンソール

<p>Action name Choose a name for your action</p> <input type="text" value="Source"/> <p>No more than 100 characters</p>	<p>Action name Choose a name for your action</p> <input type="text" value="Image"/> <p>No more than 100 characters</p>	<p>Action name Choose a name for your action</p> <input type="text" value="Deploy"/> <p>No more than 100 characters</p>
<p>Action provider</p> <input type="text" value="AWS CodeCommit"/>	<p>Action provider</p> <input type="text" value="Amazon ECR"/>	<p>Action provider</p> <input type="text" value="Amazon ECS (Blue/Green)"/>
<p>AWS CodeCommit</p>	<p>Amazon ECR</p>	<p>Amazon ECS (Blue/Green)</p>
<p>Repository name Choose a repository that you have access to</p> <input type="text" value="aspnetcoremvc-repo"/>	<p>Repository name Choose an Amazon ECR repository as the source location</p> <input type="text" value="aspnetcoremvc"/> <input type="button" value="Create"/>	<p>AWS CodeDeploy application name Choose one of your existing applications, or create a new one in AWS CodeDeploy.</p> <input type="text" value="AppECS-aspnetcoremvc-cluster-aspnetcoremvc-svc"/> <input type="button" value="Create application"/>
<p>Branch name Choose a branch of the repository</p> <input type="text" value="master"/>	<p>Image tag - optional Choose the image tag that triggers your pipeline when a new image is pushed to the repository</p> <input type="text" value="latest"/> <p>If an image tag is not selected, defaults to latest</p>	<p>AWS CodeDeploy deployment group Choose one of your existing deployment groups, or create a new one in AWS CodeDeploy.</p> <input type="text" value="DgpECS-aspnetcoremvc-cluster-aspnetcoremvc-svc"/>
<p>Change detection options Choose a detection mode for your code.</p> <p><input checked="" type="radio"/> Amazon CloudWatch (recommended) Use Amazon CloudWatch to automatically start change occurs</p>	<p>Output artifacts Choose a name for the output of this action.</p> <input type="text" value="MyImage"/> <p>No more than 100 characters</p>	<p>Amazon ECS task definition Choose the input artifact where your Amazon ECS task definition file is stored. If other than the default file path, specify the path and filename of your task definition file.</p> <input type="text" value="SourceArtifact"/> <input type="text" value="taskdef.json"/> <p>The default path is taskdef.json.</p>
<p>Output artifacts Choose a name for the output of this action.</p> <input type="text" value="SourceArtifact"/> <p>No more than 100 characters</p>		

AWS CodePipelineコンソール

Action name
Choose a name for your action

Source

No more than 100 characters

Action provider

AWS CodeCommit

AWS CodeCommit

Repository name
Choose a repository that you have access to

aspnetcoremvc-repo

Branch name
Choose a branch of the repository

master

Change detection options
Choose a detection mode for your code.

Amazon CloudWatch (recommended)
Use Amazon CloudWatch to automatically start a change occurs

Output artifacts
Choose a name for the output of this action.

SourceArtifact

No more than 100 characters

Action name
Choose a name for your action

Image

No more than 100 characters

Action provider

Amazon ECR

Amazon ECR

Repository name
Choose an Amazon ECR repository as the source location

aspnetcoremvc

Image tag - optional
Choose the image tag that triggers your pipeline when a new image is pushed to the repository. If an image tag is not selected, defaults to latest

latest

Output artifacts
Choose a name for the output of this action.

MyImage

No more than 100 characters

Action name
Choose a name for your action

Deploy

No more than 100 characters

Action provider

Amazon ECS (Blue/Green)

Amazon ECS (Blue/Green)

AWS CodeDeploy application name
Choose one of your existing applications, or create a new one

AppECS-aspnetcoremvc-cluster-aspnetcoremvc

AWS CodeDeploy deployment group
Choose one of your existing deployment groups, or create a new one

DgpECS-aspnetcoremvc-cluster-aspnetcoremvc

Amazon ECS task definition
Choose the input artifact where your Amazon ECS task definition file path, specify the path and filename of your task definition file.

SourceArtifact

The default path is taskdef.json.

AWS CodeDeploy AppSpec file
Choose the input artifact where your AWS CodeDeploy AppSpec file is stored. If other than the default file path, specify the path and filename of your AppSpec file.

SourceArtifact

appspec.yaml

Dynamically update task definition image - optional
You can provide an input artifact and a placeholder name for the container definition image that will be used to dynamically update a task definition. You can specify multiple input artifacts and placeholders.

Input artifact with placeholder text in the task definition

MyImage

Placeholder text in the task definition

IMAGE_1_NAME

Remove

Add

Input artifacts
Choose an input artifact for this action. [Learn more](#)

SourceArtifact

Remove

MyImage

Remove

No more than 100 characters

Add

AWSの新サービス・新機能は、95%以上、お客様の要望に基づいており、開発者に必要なツールを提供

IDEs



AWS Cloud9



AWS Toolkit for PyCharm

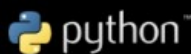


AWS Toolkit for IntelliJ



AWS Toolkit for VS Code

Languages



elixir

COBOL

AWS Lambda Runtime API

Management

AWS Lambda layers

AWS Nested apps

Websocket support for API Gateway

ALB support for Lambda

Coordination / Orchestration

Step Functions
API Connectors

Managed Streaming for Kafka

AWS Toolkits for Popular IDEs

開発者が普段使っているIDEに、AWS Toolkitsが対応

- PyCharm 用 AWS Toolkitsの一般公開を開始。VS Codeと IntelliJ 用 AWS Toolkitを開発中。
- Python、Java、ノード、.NET など、任意のIDE および言語でのコード作成、ステップスルーのデバッグ、デプロイをして、サーバーレスアプリケーションを簡単に開発できます。
- これらのツールキットはオープンソースであり、サーバーレスだけでなく、すべての AWS 向けの開発を支援することを目的としています。JetBrainsマーケットプレイスでPyCharmをチェックし、GitHubで、フィードバックを提供したり、コントリビュートしたりしてください！



AWS Toolkit
for PyCharm

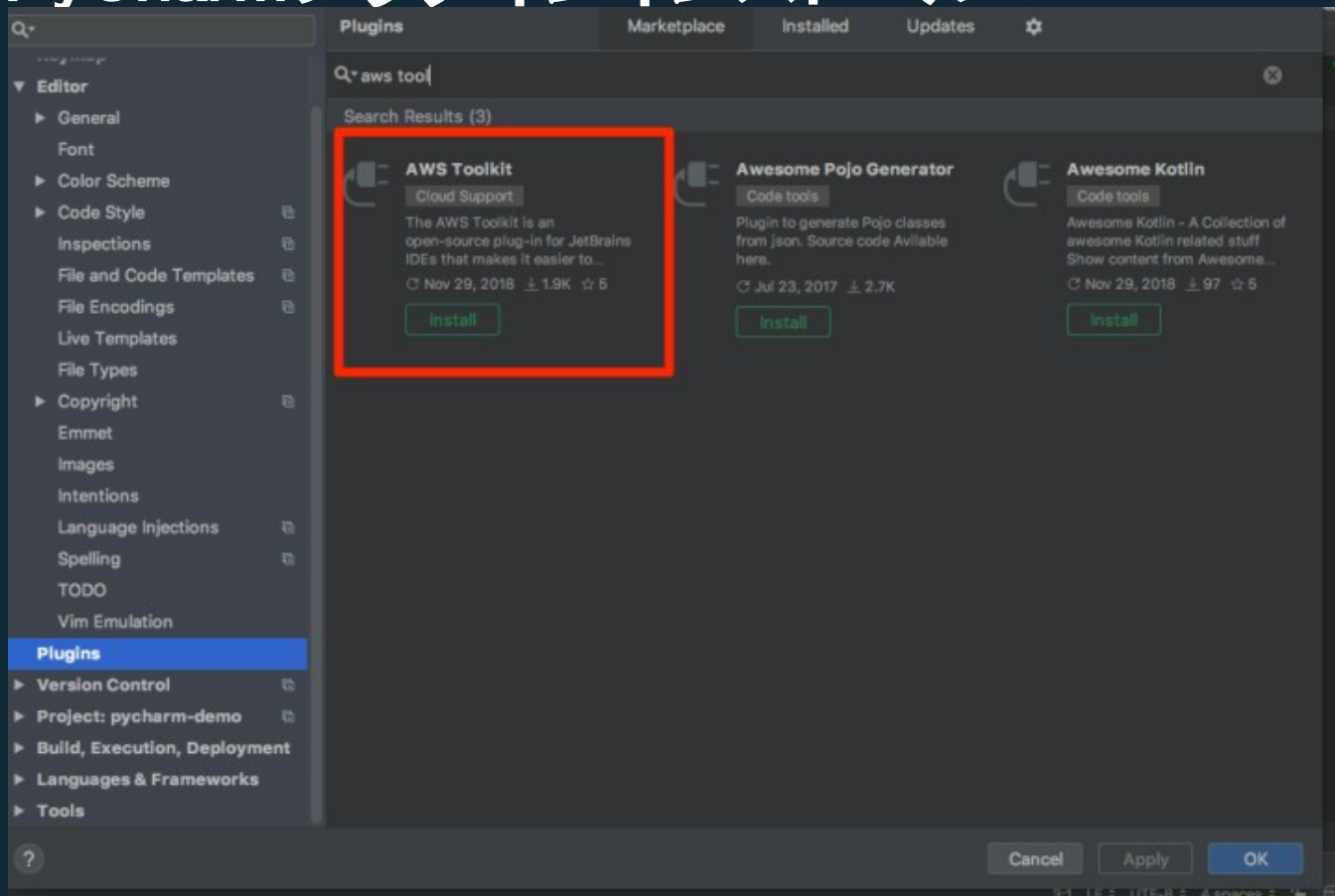


AWS Toolkit
for IntelliJ



AWS Toolkit
for VS Code

PyCharmプラグインインストール



re:Invent 2018前に発表されたアッ プデート

AWS CodeCommit が CLI と SDK で新規ファイルとフォルダーアクションをサポート



- AWS CodeCommitでAWS CLI と SDK を用いて直接ファイルの削除、ファイルの内容取得、フォルダへのアクセスを可能に
- これまで、これらのオペレーションには Git クライアントのインストールと設定が必要だったが、今回、CLI または SDK を用いてどのような CodeCommit レポジトリでのこうしたアクションを素早く行って時間を節約できるようになった
- AWS CodeCommit は完全マネージド型のソースコントロールサービスで、企業によるセキュアかつ高スケーラブルなプライベート Git リポジトリのホストを容易に

<https://aws.amazon.com/jp/about-aws/whats-new/2018/09/aws-codecommit-supports-new-file-and-folder-actions-via-the-cli/>

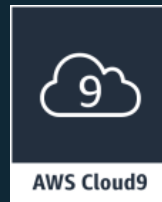
AWS CodeBuild で複数の入力ソースと出力アーティファクトを伴うビルドプロジェクト生成が可能に



- AWS CodeBuildが複数の入力ソースと出力アーティファクトを伴うビルドプロジェクトの生成をサポート
- プロジェクトは Amazon S3、AWS CodeCommit、GitHub、GitHub Enterprise または Bitbucket からのひとつ以上のソースを用いて、複数のアーティファクト群をひとつ以上の Amazon S3 buckets にアップロード可能に
- プロジェクトに何も入力ソースが無いようにも設定可能
- AWS CodePipeline 統合を CodeBuild と共に用いて、複数の入出力アーティファクトでのパイプラインを CodeBuild プロジェクトに作成可能

<https://aws.amazon.com/jp/about-aws/whats-new/2018/08/aws-codebuild-adds-ability-to-create-build-projects-with-multiple-input-sources-and-output-artifacts/>

AWS Cloud9 で TypeScript のサポートを開始



- AWS Cloud9 が、TypeScript プログラム言語のサポートを開始し TypeScript プロジェクトの開発が容易に
- TypeScript のサポート機能には、オートコンプリション、アイコンの余白、リファレンスの検索、定義への移動、シンボルへの移動が含まれ、すばやいコーディングを助け、エラーを防ぐ
- 利用を開始するには、新しい TypeScript をインポートもしくはスタートし、Cloud9 環境でワンタイムパッケージインストールを完了し、コーディングを開始する

<https://aws.amazon.com/jp/about-aws/whats-new/2018/10/aws-cloud9-now-supports-typescript/>

AWS CodeBuild でBitbucket プルリクエストのビルドをサポート



- AWS CodeBuild で、Atlassian Bitbucketのプルリクエストによるビルドが可能に
- CodeBuild を使用してアプリケーションコードを編集、ビルドしているチームでの共同作業が容易に
- コードの変更をプッシュした際に自動的にビルドのトリガーとなる、Bitbucket からの webhook を受け付けるように、AWS CodeBuildを設定することが可能に

<https://aws.amazon.com/jp/about-aws/whats-new/2018/10/aws-codebuild-now-supports-building-bitbucket-pull-requests/>

AWS Elastic Beanstalk コンソールが Network Load Balancer をサポート



AWS Elastic
Beanstalk

- AWS Elastic Beanstalk は AWS Elastic Beanstalk コンソールを通じて、Network Load Balancer の作成をサポート
- これまで Network Load Balancer の作成には、AWS Elastic Beanstalk CLI (コマンドラインインターフェイス) を使用する方法しかなかった
- 可用性の高い Elastic Beanstalk を構成する場合、Elastic Beanstalk コンソールにおいて、Classic Load Balancer および Application Load Balancer に加え、Network Load Balancer を選択可能
- 高可用性の Elastic Beanstalk を構成する際に、AWS Elastic Beanstalk コンソールでは旧世代の Classic Load Balancer に代わって、Application Load Balancer の使用がデフォルトに

https://aws.amazon.com/jp/about-aws/whats-new/2018/10/aws_elastic_beanstalk_console_supports_network_load_balancer/

AWS CodePipelineの実行がより高速に、かつステージごとのパイプラインアクション数の増加もサポート



- AWS CodePipelineがパイプラインアクション間の遷移時間を削減
- これによりパイプラインの実行時間が短縮され、ビルドとテストの結果が高速化、より迅速に反復することが可能に
- すべてのアクションタイプに対してステージあたりのアクションのデフォルト制限を50に引き上げ
- 以前は、連続アクションと並列アクションの両方で10の制限を含め、ステージあたり20アクションのデフォルト制限があった
- 新しい上限を設定することで、ステージごとのアクション数の制限を気にせずに、より複雑なパイプラインを作成することが可能

<https://aws.amazon.com/about-aws/whats-new/2018/11/aws-codepipeline-now-executes-faster-and-supports-more-pipeline-actions-per-stage/>

4 7 AWS CodePipeline でクロスリージョンアクションがサポート可能に



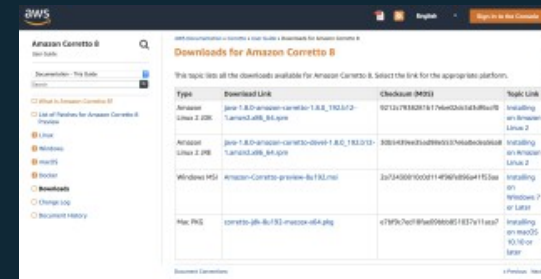
- AWS CodePipeline で、単一のパイプラインから、複数のリージョンでのデプロイ、構築、テストなどのアクションを容易に実行可能に
- 以前は、CodePipeline からリージョンでアクションを実行するには、そのリージョンにパイプラインを設定する必要があった
- AWS コマンドラインインターフェイス (AWS CLI) または AWS CloudFormation を使用して、AWS CodePipeline で簡単にクロスリージョンアクションを設定可能
- 例えば、クロスリージョンデプロイを設定するには、デプロイステージにアクションを追加し、デプロイ先のリージョンを指定する

<https://aws.amazon.com/jp/about-aws/whats-new/2018/11/aws-codepipeline-now-supports-cross-region-actions/>

LTS付きのOpenJDKディストリビューション、 Amazon Correttoを発表

NEW

- 無料で利用でき、マルチプラットフォームをサポート。パフォーマンス改善とセキュリティ対応を長期にわたって提供(LTS)
- Amazon Linux 2/Windows/MacOS/Docker ImageをサポートしたCorretto 8がプレビュー中。GAは2019年1-3月予定で、UbuntuとRHELをサポートする計画
- Corretto 8は2023年6月まで、Corretto 11は2024年8月までのサポートを最低限提供する



<https://docs.aws.amazon.com/corretto/latest/corretto-8-ug/downloads-list.html>

お願い

ぜひお試し頂き、フィードバックを頂けると幸いです。