



AWS CDK 入門 勉強会

Mizuki Ugajin

Partner Solutions Architect
Amazon Web Services Japan G.K.
2022/11/18

手動操作（マネジメントコンソール）

操作手順書が別途必要

The screenshot shows the AWS Lambda Run Command interface. At the top, there are buttons for "Run a command" and "Actions". Below is a search bar labeled "Filter by attributes". A table lists several commands:

Command ID	Instance ID	Document name	Status	Requested date	Comment
65555b90-ee60-45...	i-8fd6aa30	AWS-RunPowerSh...	Success	October 21, 2015 at...	Listing services
65555b90-ee60-45...	i-d583f76a	AWS-RunPowerSh...	Success	October 21, 2015 at...	Listing services
65555b90-ee60-45...	i-8ed6aa31	AWS-RunPowerSh...	Success	October 21, 2015 at...	Listing services
ca4b10c6-cee1-437...	i-d583f76a	AWS-RunPowerSh...	Success	October 20, 2015 at...	getting list of pr...
561e5f4a-27d2-419...	i-d583f76a	AWS-RunPowerSh...	Success	October 20, 2015 at...	ipconfig on the b...

Below the table, it says "Command ID: 65555b90-ee60-4520-9dc3-e42e94445469 Instance ID: i-8fd6aa30". The "Output" tab is selected. The output details are as follows:

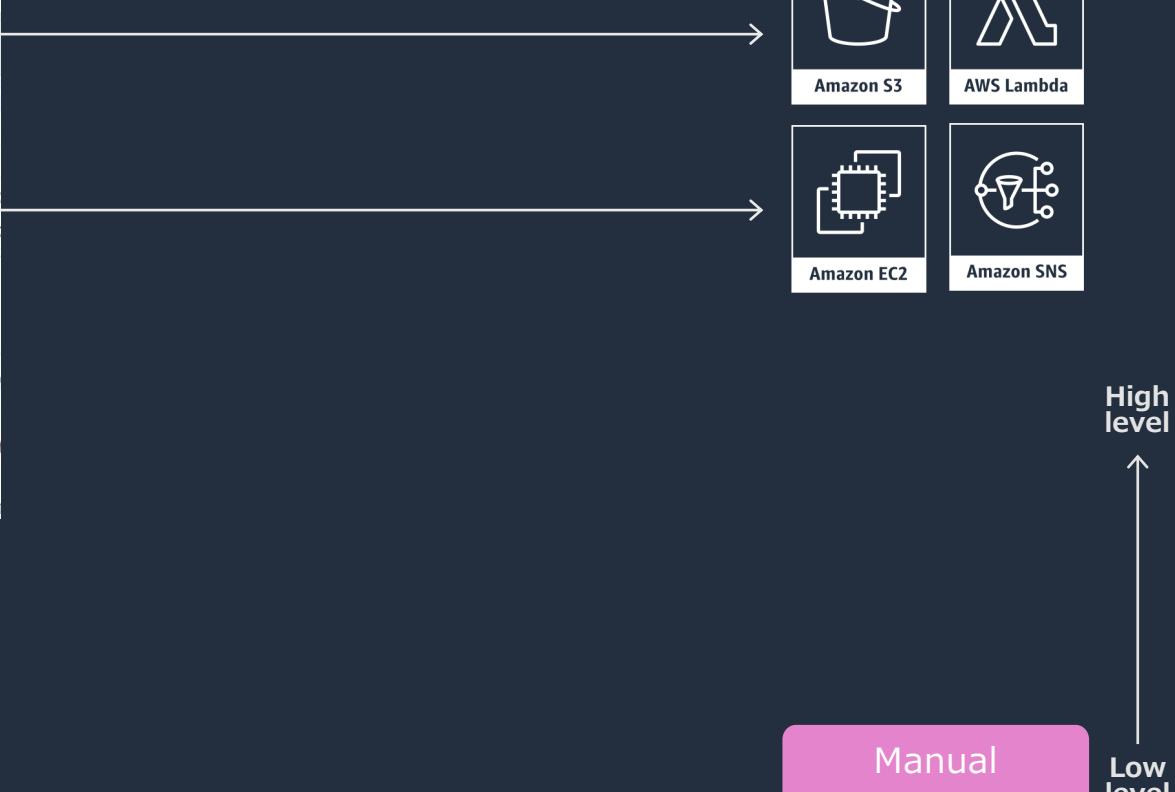
Command ID	Document name	Date requested	Output S3 bucket
65555b90-ee60-4520-9dc3-e42e94445469	AWS-RunPowerShellScript	October 21, 2015 at 3:56:59 PM UTC-7	run-command-test

On the right, there is a summary of the command parameters:

Instance ID	Status	Comment
i-8fd6aa30	Success	Listing services

Document parameters:

- Execution role: arn:aws:lambda:us-east-1:123456789012:role/lambdaBasicExecutionRole
- Region: us-east-1
- Timeout: 300 seconds
- Memory: 128 MB
- Version: \$LATEST



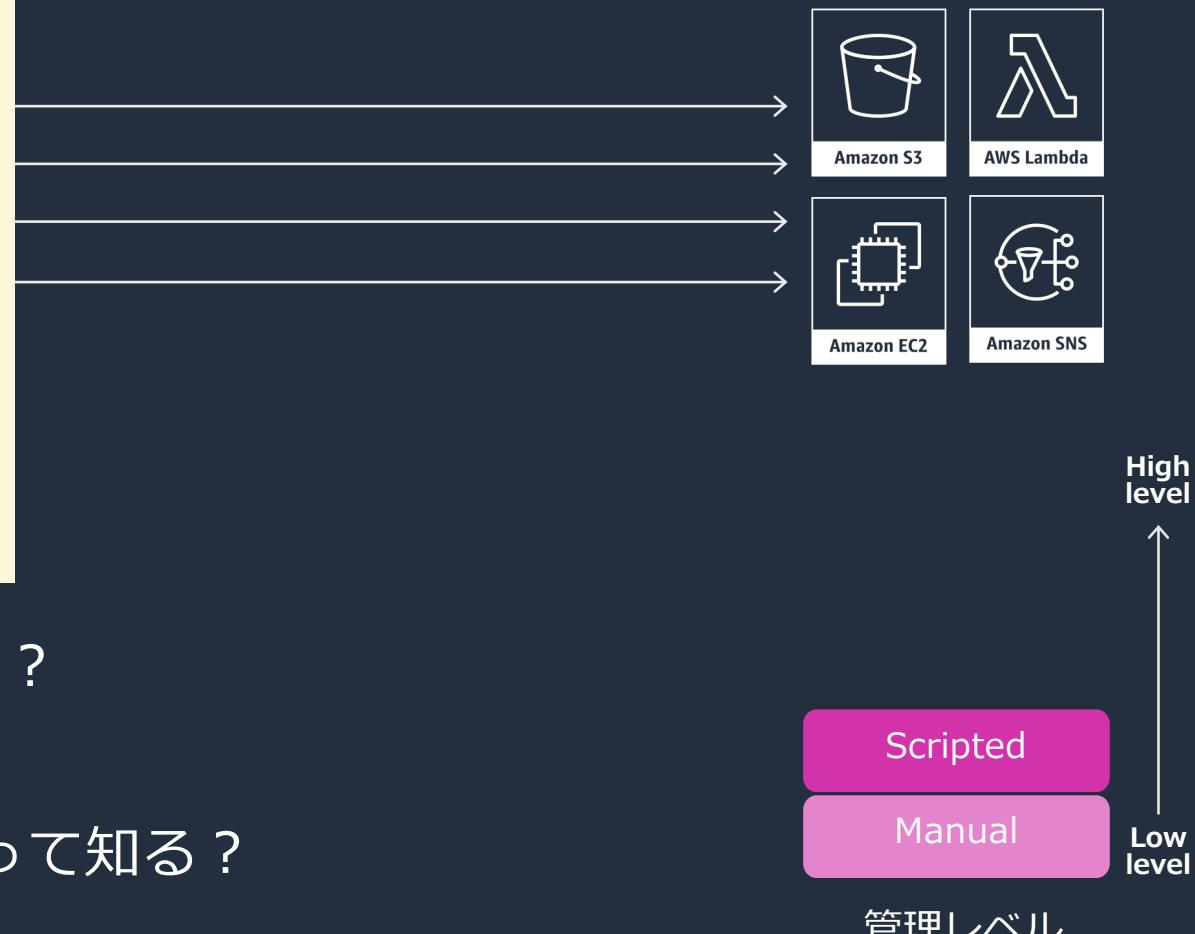
スクリプト (SDK, CLI)

操作手順の定義が可能

```
require 'aws-sdk-ec2'

ec2 = Aws::EC2::Resource.new(region: 'us-west-2')

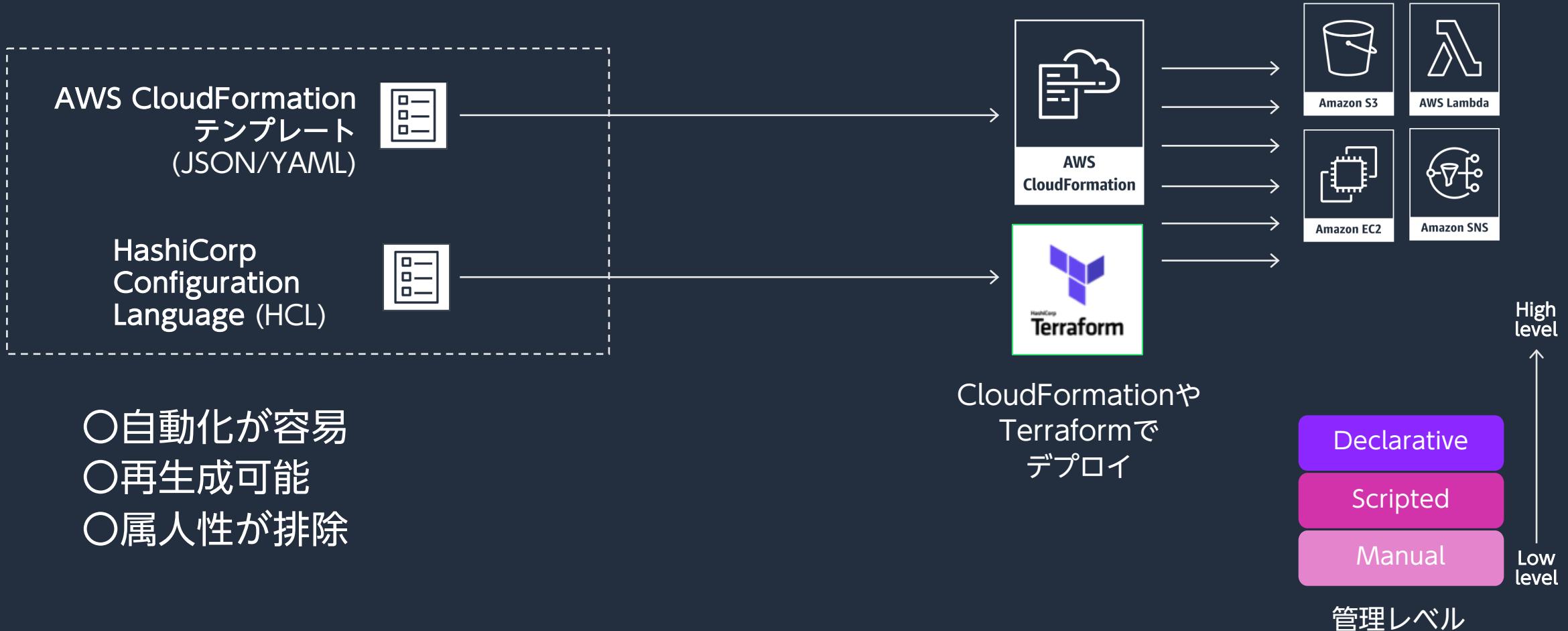
instance = ec2.create_instances({
  image_id: 'IMAGE_ID',
  min_count: 1,
  max_count: 1,
  key_name: 'MyGroovyKeyPair',
  security_group_ids: ['SECURITY_GROUP_ID'],
  instance_type: 't2.micro',
  placement: {
    availability_zone: 'us-west-2a'
  },
  subnet_id: 'SUBNET_ID',
  iam_instance_profile: {
    arn: 'arn:aws:iam::' + ACCOUNT_ID + ':instance-profile/aws-opsworks-ec2-role'
  }
})
```



- ・ ? APIコールが失敗したら何が起こる？
- ・ ? どうやってアップデートする？
- ・ ? リソースが準備完了なのはどうやって知る？
- ・ ? どうやってロールバックする？

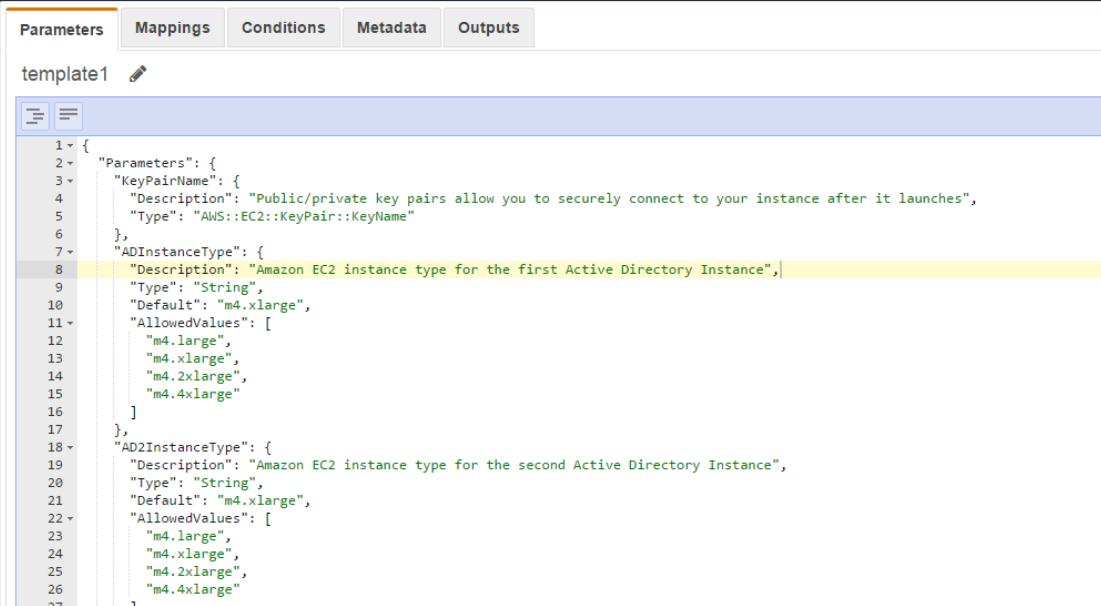
プロビジョニングツール(CloudFormationなど)

るべき状態の定義が可能



Infrastructure as Code

コードによってリソースをモデル化



```
1  {
2    "Parameters": {
3      "KeyPairName": {
4        "Description": "Public/private key pairs allow you to securely connect to your instance after it launches",
5        "Type": "AWS::EC2::KeyPair::KeyName"
6      },
7      "ADInstanceType": {
8        "Description": "Amazon EC2 instance type for the first Active Directory Instance",
9        "Type": "String",
10       "Default": "m4.xlarge",
11       "AllowedValues": [
12         "m4.large",
13         "m4.xlarge",
14         "m4.2xlarge",
15         "m4.4xlarge"
16       ]
17     },
18     "AD2InstanceType": {
19       "Description": "Amazon EC2 instance type for the second Active Directory Instance",
20       "Type": "String",
21       "Default": "m4.xlarge",
22       "AllowedValues": [
23         "m4.large",
24         "m4.xlarge",
25         "m4.2xlarge",
26         "m4.4xlarge"
27     }
28 }
```

- 可能な限り自動化
- アプリケーションとインフラストラクチャの両方
- 宣言的にインフラを定義
- セキュリティを含め注意深くインフラを設計
- 定義や設定をアプリケーションコードのごとく扱う
- バージョンコントロール
- アプリケーションの一部としてのインフラ
- ロールバックのためのプラン

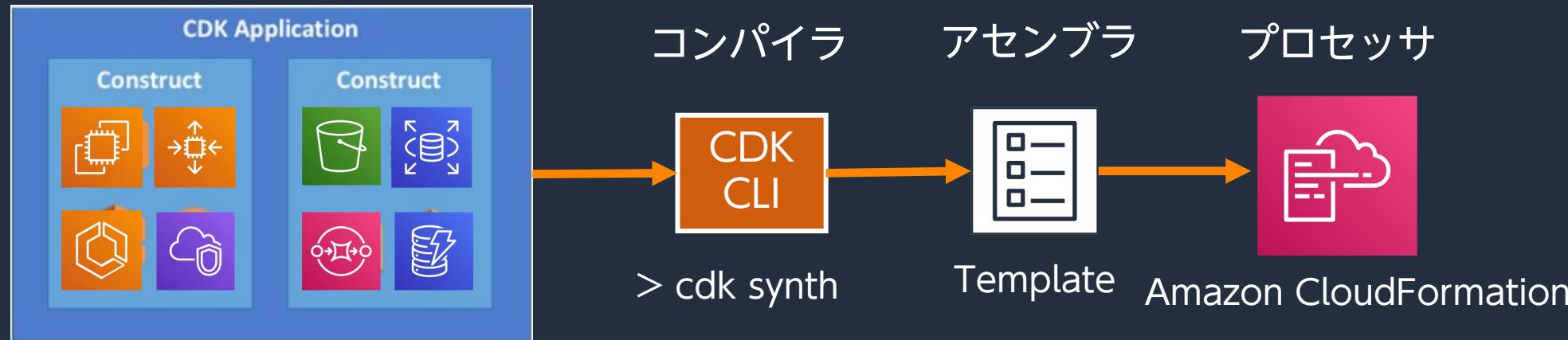


AWS Cloud Development Kit (CDK)



AWS CDK (AWS Cloud Development Kit)

- ・ インフラをPython, TypeScript等のコードで定義してAWS CloudFormationでプロビジョニング
- ・ オブジェクト指向的に記述&ループ等が使用可能
- ・ 推奨デフォルト値が設定済みのためコード量が少なく済む
- ・ 現在、JavaScript、TypeScript、Python、Java、C#に対応



AWS CDK (AWS Cloud Development Kit)

例) S3バケット作成

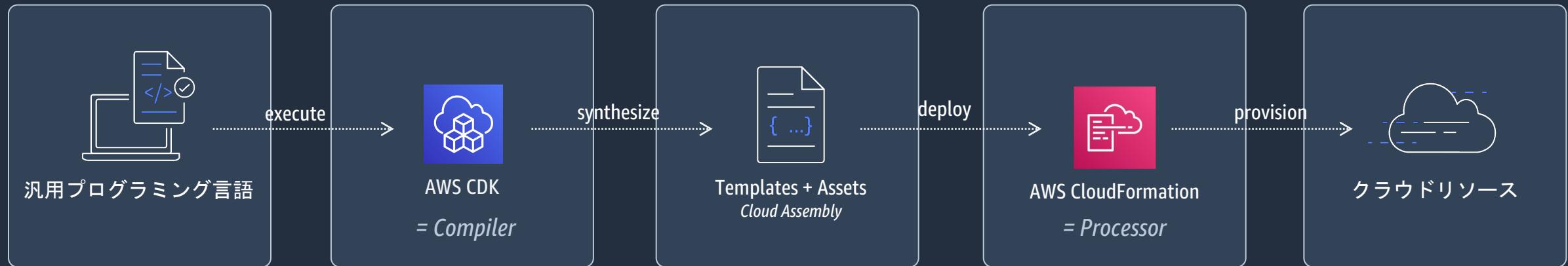
```
const cdk = require('@aws-cdk/core');
const s3 = require('@aws-cdk/aws-s3');

class MyStack extends cdk.Stack {
  constructor(parent, id, props) {
    super(parent, id, props);

    new s3.Bucket(this, 'MyFirstBucket', {
      versioned: true
    });
  }
}
```



AWS CDK のメリット



AWS CDK による抽象化のメリット

- ✓ 安全なデフォルト値とベストプラクティスによる抽象化
- ✓ コード量が少なく可読性が高い
- ✓ 一般のプログラミング言語の制御構文や抽象化を利用可能
- ✓ IDE やテキストエディタによる補完
- ✓ コンパイラや静的解析ツール、テストによる実行前のチェック
- ✓ リソースに加え、スタック間の依存関係を自動で解決
- ✓ インフラだけでなくアプリのコードや設定を含め一元管理



AWS CloudFormation のメリット

- ✓ あるべき状態が定義されている(宣言的)
- ✓ 人手を介さず、繰り返し実行可能
- ✓ サービスが幂等性、順序、ロールバックを担保
- ✓ バージョン管理が可能
- ✓ プルリクエストやコードレビューなど、アプリ開発のプラクティスを適用可能

AWS CDK のコンセプト



AWS CDK の概念

<https://docs.aws.amazon.com/cdk/v2/guide/home.html>

App

- ・ アプリケーション全体
- ・ 複数のAWSアカウント、リージョンにまたがることが可能

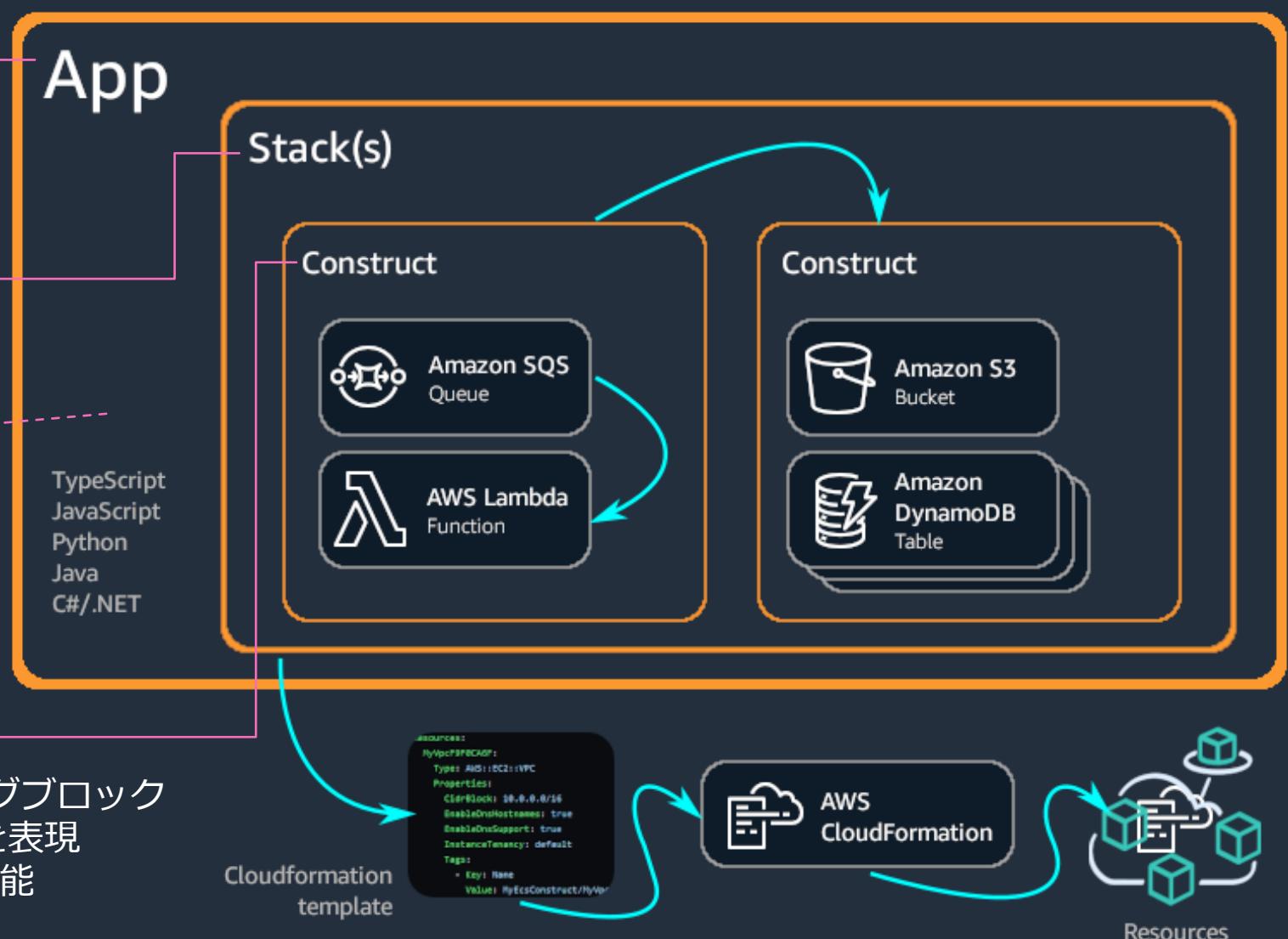
Stack

- ・ CloudFormation スタックに対応
- ・ デプロイ可能な最小単位

Stage (Optional)

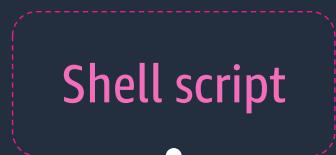
- ・ 複数の Stack をまとめて論理的なアプリ(サービス)のかたまりを表現
- ・ 1つの Stage を定義して環境ごとにインスタンス化することで複製が可能

aws



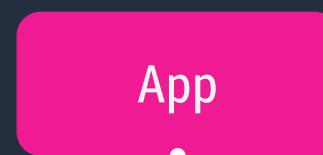
AWS CDK の概念 - CloudFormation との比較

AWS CloudFormation



- スタックを1つずつデプロイ
`$ aws cloudformation deploy HogeStack`
- 依存関係を考慮して
人がデプロイ順序を決める
- 複数のスタックを一括デプロイ
するためにはスクリプトや
Makefile, CodePipeline 等が必要
- CloudFormation Module による力
プセル化と再利用

AWS CDK



- 複数のスタックをまとめて
1つのアプリとしてデプロイ
(スタックを指定してデプロイも可能)
`$ cdk deploy --all`
- スタック間の依存関係を CDK が
解決してデプロイ順序を決定
- Construct による抽象化と再利用

AWS Constructs Library

AWS CDK が標準で提供する Construct のライブラリ

Patterns (L3)

- 複数のリソースを含む一般的な構成パターンを事前に定義したもの
- `aws-ecs-patterns.LoadBalancedFargateService` など

High-level constructs (L2)

- デフォルト値や便利なメソッドを定義したAWSリソースを表すクラス
- 例) クラス `s3` は メソッド `s3.Bucket.addLifecycleRule()` を持つ

Low-level constructs (L1)

- CloudFormationリソースおよびプロパティと1:1で対応（自動生成される）
- `CfnXXX` という名前（例：`s3.CfnBucket` は `AWS::S3::Bucket` を意味）
- すべてのプロパティを明示的に設定する必要がある



CloudFormation テンプレート

例: IAM User からの
読み取り専用アクセスを許可する S3 Bucket を作成

AWS CloudFormation
template language



Resources:

MyBucket:

Type: AWS::S3::Bucket

MyUser:

Type: AWS::IAM::User

MyUserPolicy:

Type: AWS::IAM::Policy

Properties:

PolicyDocument:

Statement:

- Action:

- s3:GetObject*
- s3:GetBucket*
- s3>List*

Effect: Allow

Resource:

- Fn::GetAtt: [MyBucket, Arn]
- Fn::Sub: "\${MyBucket.Arn}/*"

Version: "2012-10-17"

PolicyName: MyUserPolicy

Users:

- Ref: MyUser

L1 Constructs

CloudFormation テンプレートとほぼ1:1対応
型チェックや補完、ループなどは使用可能

AWS CloudFormation
resources



AWS CDK



"L1"

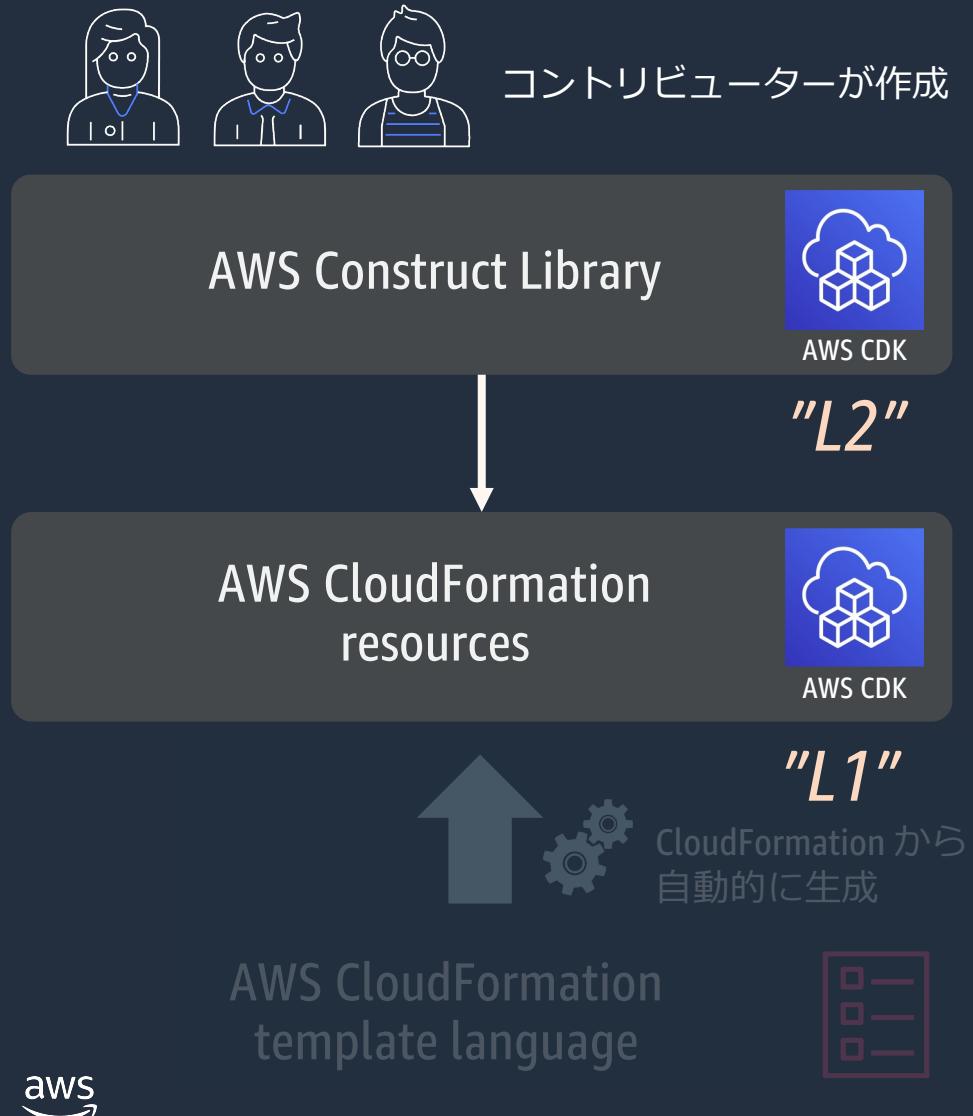
CloudFormation から
自動的に生成

AWS CloudFormation
template language



```
const bucket = new CfnBucket(this, 'MyBucket');
const user = new CfnUser(this, 'MyUser');
new CfnPolicy(this, 'MyUserPolicy', {
  policyName: 'MyUserPolicy',
  policyDocument: new PolicyDocument({
    statements: [new PolicyStatement({
      actions: [
        's3:GetObject*',
        's3:GetBucket*',
        's3>List*' ],
      resources: [
        bucket.bucketArn,
        `${bucket.bucketArn}/*` ]
    })]
  }),
  users: [user.userName],
});
```

L2 Constructs



```
const bucket = new s3.Bucket(this, 'MyBucket');

const user = new iam.User(this, 'MyUser');

bucket.grantRead(user);
```

grant() メソッドにより IAM Policy を自動生成
コードから意図が明確に

Construct Hub

<https://constructs.dev/>

- 1,000 以上のオープンソースのコンストラクトを公開
 - 公開されたコンストラクトを組み合わせることで、目的に合わせたアプリケーション環境をさらに迅速に構成できる
 - CDK, CDK8s, CDKtf などのタイプやバージョン、言語、パブリッシャーなどで検索可能

The screenshot shows the Construct Hub homepage and a detailed view of a specific construct.

Homepage: The main page features a search bar, filters for CDK Type (AWS CDK, CDK8s, CDKtf), Programming Language (TypeScript, Python, Java, .NET, Go), and Publisher (AWS, HashiCorp, Deno, Community). It displays a list of constructs, with the first few being:

- @cdktf/provider-kubernetes**: Prebuilt Kubernetes Provider for Terraform CDK (cdktf)
- @cdktf/provider-postgresql**: Prebuilt postgresql Provider for Terraform CDK (cdktf)
- aws-cdk/cdk-aurora-globaldatabase**: cdk-aurora-globaldatabase is an AWS CDK construct library that provides Cross Region Create Global Aurora RDS Databases.
- aws-cdk/cdk-dynamo-table-viewer**: An AWS CDK construct which exposes an endpoint with the contents of a DynamoDB table.
- CDK8s**: Cloud Development Kit for Kubernetes

Detailed View: A detailed view of the **cdk-spa-deploy** construct (v1.104.1) is shown. It includes documentation, dependencies (TypeScript, Vue, React, Angular, Websites, spa, website, deploy, cloudfront), and code samples. The code sample for the **HostedZoneConfig** initializer is as follows:

```
using com.cdkpatterns;

new HostedZoneConfig {
    string IndexDoc,
    string WebsiteFolder,
    string ErrorDocument,
    Behavior[] CFBehaviors = null,
    string ErrorCode = null,
    Role Role = null,
    string Subdomain = null
};
```

Footer: © 2022, Amazon Web Services, Inc. or its affiliates.



ハンズオン前準備



ハンズオン環境へのアクセス

- <https://dashboard.eventengine.run/login> へアクセス

Terms & Conditions:

1. By using the Event Engine for the relevant event, you agree to the [AWS Event Terms and Conditions](#) and the [AWS Acceptable Use Policy](#). You acknowledge and agree that are using an AWS-owned account that you can only access for the duration of the relevant event. If you find residual resources or materials in the AWS-owned account, you will make us aware and cease use of the account. AWS reserves the right to terminate the account and delete the contents at any time.
2. You will not: (a) process or run any operation on any data other than test data sets or lab-approved materials by AWS, and (b) copy, import, export or otherwise create derivative works of materials provided by AWS, including but not limited to, data sets.
3. AWS is under no obligation to enable the transmission of your materials through Event Engine and may, in its discretion, edit, block, refuse to post, or remove your materials at any time.
4. Your use of the Event Engine will comply with these terms and all applicable laws, and your access to Event Engine will immediately and automatically terminate if you do not comply with any of these terms or conditions.

5482-

事前に共有された Event Hash を入力 (入力済みの場合はスキップ)

This is the 12 or 16 digit hash that was given to you for this event or for a specific team.

クリック

✓ Accept Terms & Login



ハンズオン環境へのアクセス

OTPを選択し、メールアドレスを入力
→メールで受け取ったパスコードを入力してサインイン

Sign in with
Pick the sign-in method you prefer

Email One-Time Password (OTP)

Enter your personal or corporate email to receive a one-time password

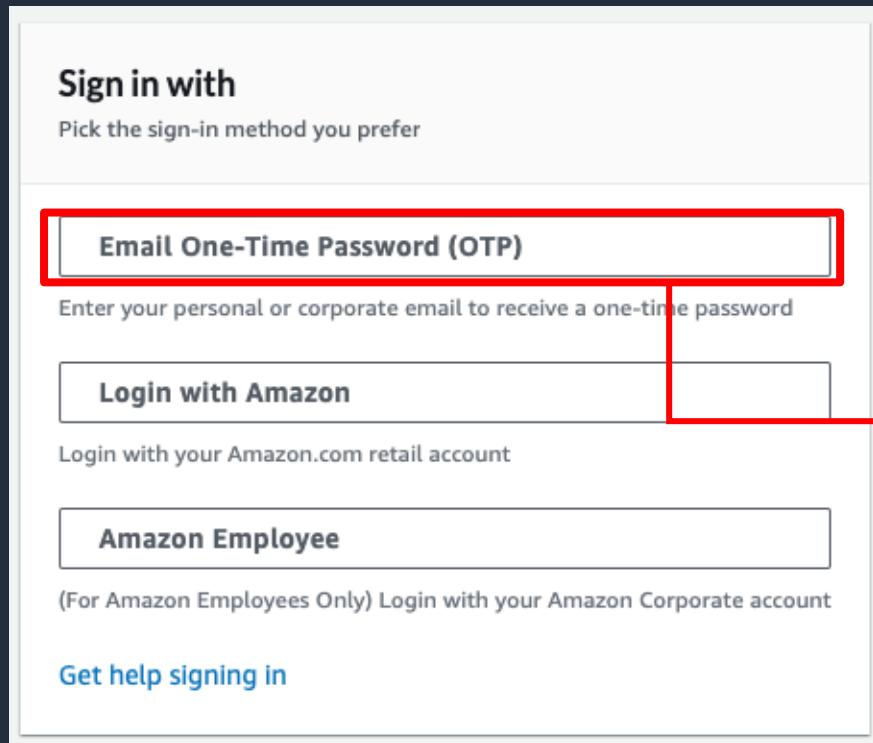
Login with Amazon

Login with your Amazon.com retail account

Amazon Employee

(For Amazon Employees Only) Login with your Amazon Corporate account

[Get help signing in](#)



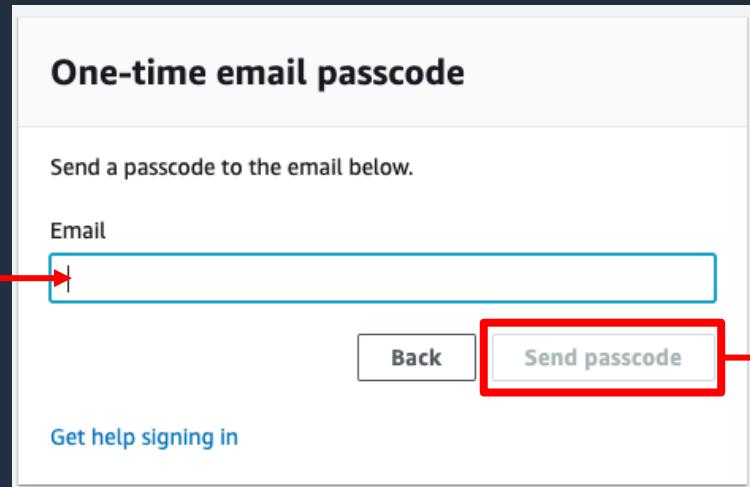
One-time email passcode

Send a passcode to the email below.

Email

[Back](#) Send passcode

[Get help signing in](#)



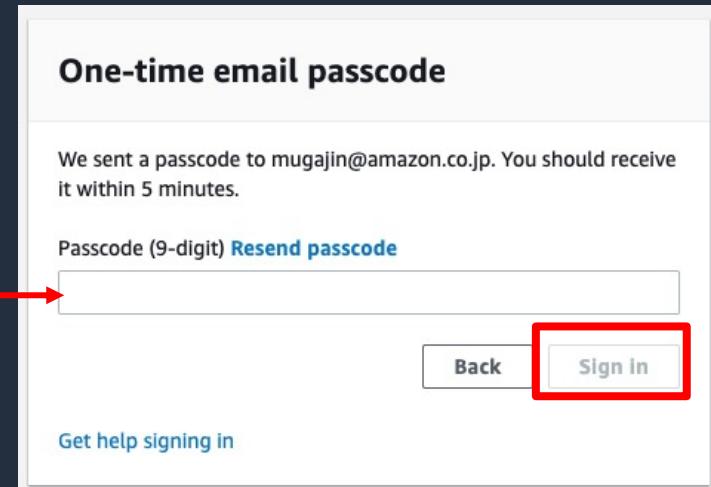
One-time email passcode

We sent a passcode to mugajin@amazon.co.jp. You should receive it within 5 minutes.

Passcode (9-digit) [Resend passcode](#)

[Back](#) Sign in

[Get help signing in](#)



メールアドレスを入力

メールで届いた
パスコードを入力

チーム名をご自身のお名前に変更

Team Dashboard

Event

Team Dashboard ヘリダイレクトされたら
Set Team Name をクリック

 Set Team Name  AWS Console  SSH Key

Event: AWS Core Services Workshop for [REDACTED]
Team Name: (Team Name Not Set Yet)

Event ID: 6f3d5b47e41946adb2210527a2e9e575
Team ID: ab7667526d2c45f7ad9fef00221ed8a3

チーム名をご自身のお名前に変更

お名前を設定していただくことで、
ハンズオン手順がわからなくなった場合のサポートがスムーズになります

Set Team Name

You will only be able to do this once! Choose your team name wisely!

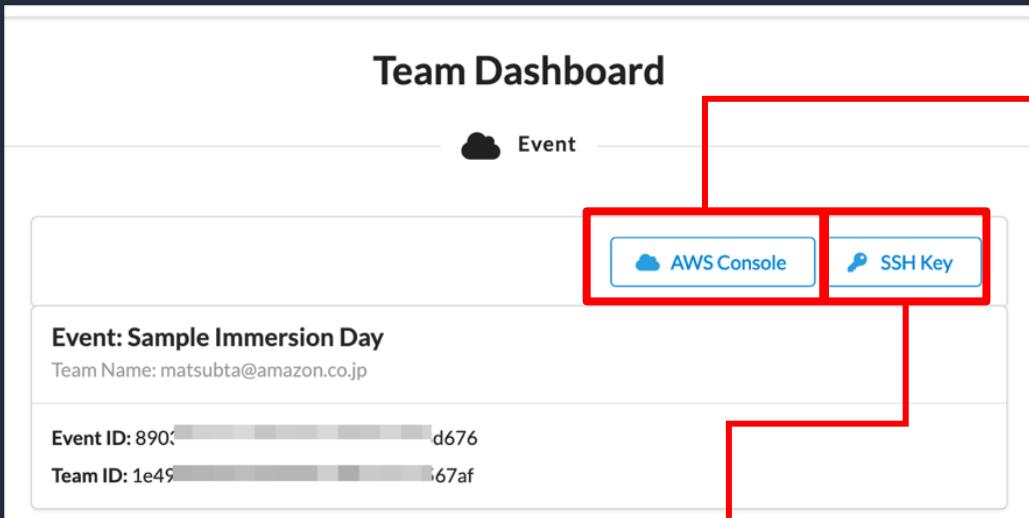
Name

Cancel Set Team Name



ハンズオン環境へのアクセス

表示されるチーム画面の「AWS Console > Open AWS Console」にて、払い出されたアカウントでAWS管理コンソールにログイン可能



The top half of the diagram shows the "AWS Console Login" page with a red box around the "Open AWS Console" button. A blue arrow points from the "AWS Console" button on the Team Dashboard to this button. To the right is a pink icon labeled "AWS 管理 コンソール". The bottom half shows the "AWS Management Console" dashboard with a blue box around the "AWS マネジメントコンソール" title bar. A blue arrow points from the "AWS Management Console" title bar to the "AWS Management Console" section of the "How do I use the AWS CLI?" page.



- AWS管理コンソールを起動し、イベント活動（ハンズオンやGameDayイベント）を開始
- ログイン後は、IAMユーザーの作成含め、制限内でAWSの利用が可能

AWS マネジメントコンソールへのアクセス

AWSコンソールが表示されたら準備OK!
ハンズオンを楽しんでいきましょう！

 Services ▾ [Option+S]   TeamRole/MasterKey @ 0816-0117-9476 ▾  Tokyo ▾ Support ▾

AWS Management Console

AWS services

▼ Recently visited services

 Elastic Container Registry	 EC2	 CloudWatch
 IAM	 Cloud9	 Elastic Kubernetes Service
 CodeBuild	 Elastic Container Service	 AWS Cloud Map
 CodePipeline	 VPC	 AWS Transfer Family
 CodeDeploy	 CloudFormation	 Secrets Manager

▶ All services

Stay connected to your AWS resources on-the-go

 AWS Console Mobile App now supports four additional regions. Download the AWS Console Mobile App to your iOS or Android mobile device. [Learn more](#)

Explore AWS

Introducing the New Amazon EKS Console

View and explore Kubernetes clusters and applications running anywhere.



ハンズオン環境についての注意事項

- 参加者それぞれに個別のAWSアカウントを発行しています
- この環境は**イベント終了後に削除**されます
 - 保存しておきたいファイル等は、**PCにダウンロード**しておいてください
- サポートへの問い合わせやサービス上限緩和申請は行わないでください
- ハンズオンへの参加以外の目的で利用しないでください

AWS Cloud9の準備

The screenshot shows the AWS search interface. The search bar at the top contains the query 'cloud9'. Below the search bar, there is a sidebar with a blue header containing the text '近日中に新しいAWS' and '2022年6月より、新 詳細はこちら または'.

The main content area displays the search results for 'cloud9' under the heading '「cloud9」の検索結果'. It includes a section titled 'サービス' with a count of 4, and a list of items: 'サービス (4)', '機能 (5)', 'ブログ (472)', 'ドキュメンテーション (35,808)', 'ナレッジ記事 (30)', and 'チュートリアル (2)'. To the right of this list, a callout bubble with the text 'クリック' points to the 'Cloud9' service entry.

The 'Cloud9' entry is highlighted with a star icon and includes a 'コードの記...' link. Below it, there are sections for '主な機能' and 'アカウント環境'. A large orange button labeled 'Create environment' is located on the right side of the page, with a callout bubble pointing to it and the text 'クリック'.

AWS Cloud9の準備

Details

Name 名前を入力

Ugajin

Limit of 60 characters, alphanumeric, and unique per user.

Description - *optional*

Limit 200 characters.

Environment type [Info](#)

Determines what the Cloud9 IDE will run on.

New EC2 instance
Cloud9 creates an EC2 instance in your account. The configuration of your EC2 instance cannot be changed by Cloud9 after creation.

Existing compute
You have an existing instance or server that you'd like to use.

AWS Cloud9の準備

New EC2 instance

Instance type Info
The memory and CPU of the EC2 instance that will be created for Cloud9 to run on.

t2.micro (1 GiB RAM + 1 vCPU)
Free-tier eligible. Ideal for educational users and exploration.

t3.small (2 GiB RAM + 2 vCPU)
Recommended for small web projects.

m5.large (2 GiB RAM + 2 vCPU)
Recommended for production and most general-purpose development.

Additional instance types
Explore additional instances to fit your need.

Platform Info
This will be installed on your EC2 instance. We recommend Amazon Linux 2.

Amazon Linux 2

Timeout
How long Cloud9 can be inactive (no user input) before auto-hibernating. This helps prevent unnecessary charges.

4 hours

Instance typeに t3.small を選ぶ

4 hoursを選ぶ

Create をクリック

Cancel Create



CDK を使った開発方法



AWS CDK の開発に必要なもの

Git



- Git のインストール <https://git-scm.com/>
- git-secrets のインストール <https://github.com/awslabs/git-secrets>
 - セキュリティ対策のため導入推奨 (Python 3 が必要)

IaC ツールチェーンと言語ランタイム



TypeScript 以外の言語で記述する場合
各言語のランタイムやコンパイラも必要

- AWS CLI v2 のインストール
<https://docs.aws.amazon.com/cli/latest/userguide/install-cliv2.html>
- Node.js のインストール <https://nodejs.org/en/>
- AWS CDK Toolkit のインストール (または npx で実行)

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

テキストエディタ / IDE



- (例) VSCode をインストール <https://code.visualstudio.com/>

(準備) AWS CLI の認証情報を設定

IAM ユーザーのアクセスキーを使用する場合

Console ➔

```
$ aws configure --profile <your-profile-name>
```

AWS SSO を使用する場合 (CDK v2.18.0~)

https://docs.aws.amazon.com/ja_jp/cli/latest/userguide/cli-configure-sso.html

Console ➔

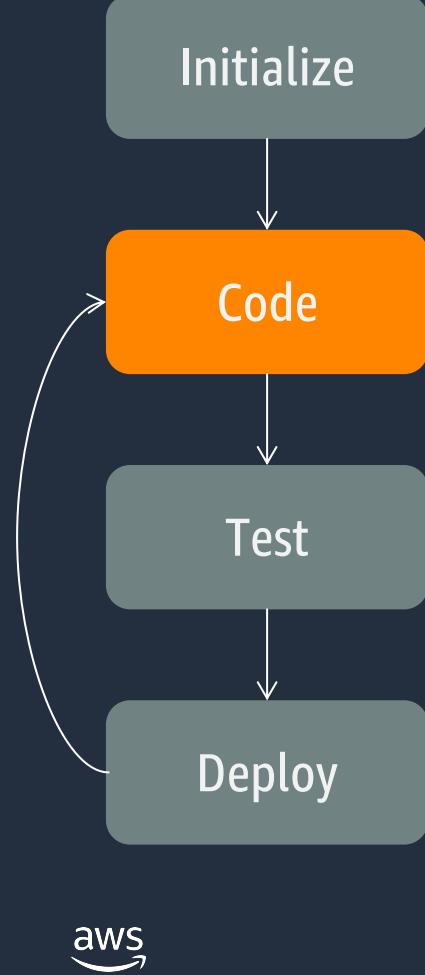
```
$ aws configure sso
```



AWS CDK での開発フロー 1/4 Initialize



AWS CDK での開発フロー 2/4 Code



IDE やテキストエディタを使用して CDK アプリをコーディング

cdk init で生成されるファイルの例

```
. └── README.md  
   ├── bin  
   │   └── cdk-sample.ts  
   ├── cdk.json  
   ├── jest.config.js  
   ├── lib  
   │   └── cdk-sample-stack.ts  
   ├── node_modules  
   ├── package-lock.json  
   ├── package.json  
   └── test  
       └── cdk-sample.test.ts  
tsconfig.json
```

コード例 (引数など一部省略)

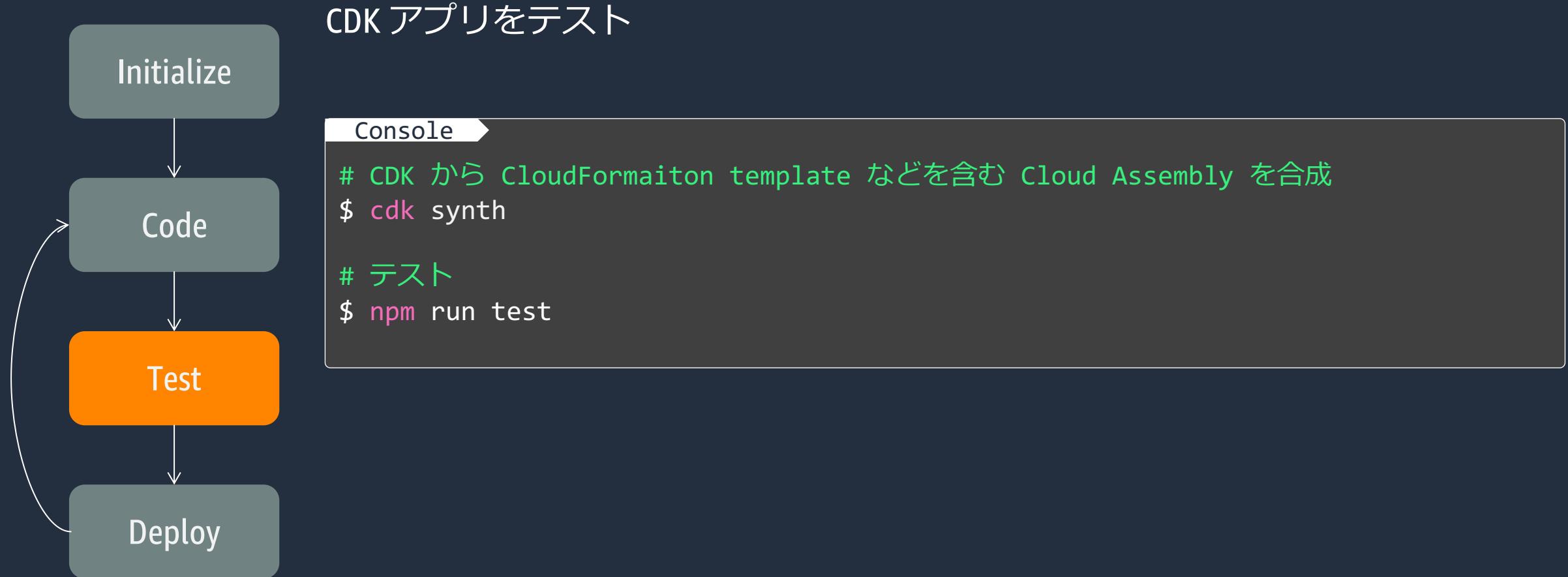
bin/cdk-sample.ts

```
# App を作成  
const app = new cdk.App();  
# App に Stack を追加  
new CdkSampleStack(app, "MyApp");
```

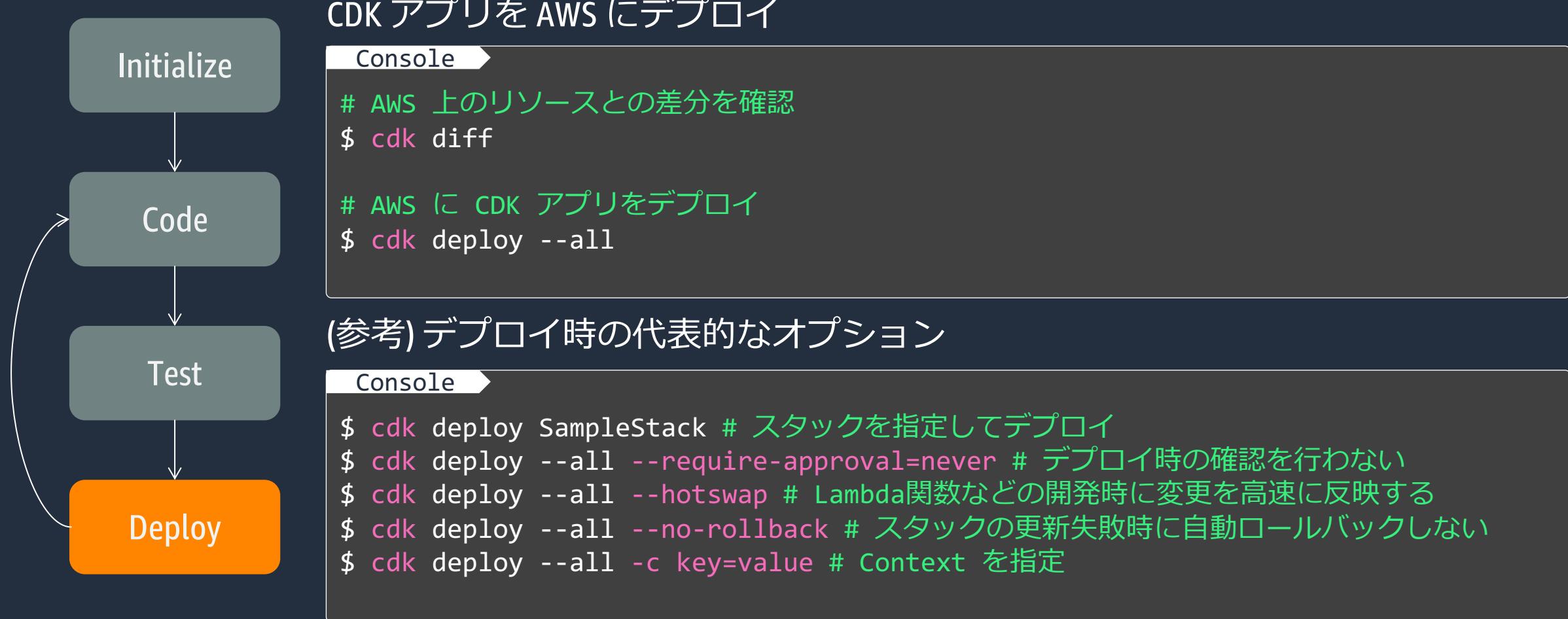
lib/cdk-sample-stack.ts

```
# Stack を定義  
export class CdkSampleStack extends Stack {  
    constructor() {  
        # コンストラクタ内で Construct を追加  
        new s3.Bucket(this, "HogeBucket");  
        new sqs.Queue(this, "HogeQueue");  
    }  
}
```

AWS CDK での開発フロー 3/4 Test



AWS CDK での開発フロー 4/4 Deploy



すばやく変更を反映: hotswap & no-rollback

`cdk deploy` と `cdk watch` (ファイルの変更を監視して自動デプロイ) に
デプロイ時間を短縮するためのオプションが追加

--hotswap (cdk watch ではデフォルトで有効)

- Lambda 関数、ECS コンテナイメージ、Step Functions ステートマシンを CloudFormation を経由せず API で直接デプロイすることで素早く反映
- CloudFormation スタックとドリフトが発生するため、開発用途でのみ使用
- Hotswap 対応リソースはこちらを参照

<https://github.com/aws/aws-cdk/blob/main/packages/aws-cdk/README.md#hotswap-deployments-for-faster-development>

--no-rollback

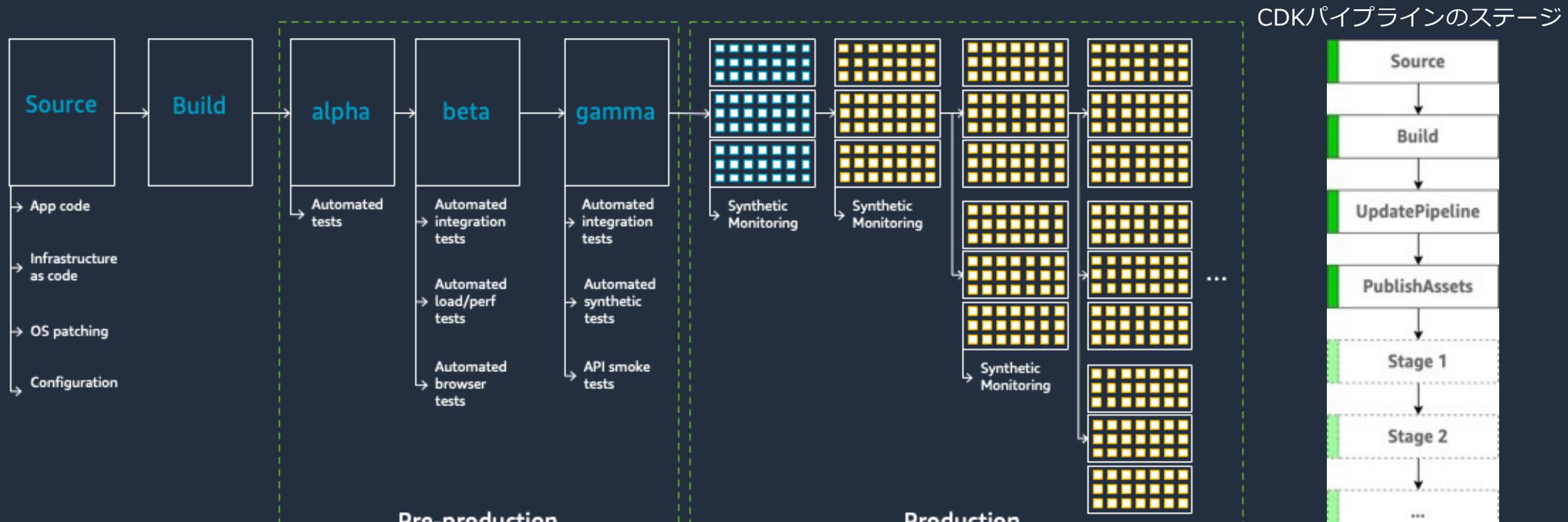
- デプロイ失敗時に CloudFormation を自動でロールバックしない
- Amazon RDS などの作成に時間がかかるリソースがある場合に有効

<https://aws.amazon.com/blogs/developer/increasing-development-speed-with-cdk-watch/>

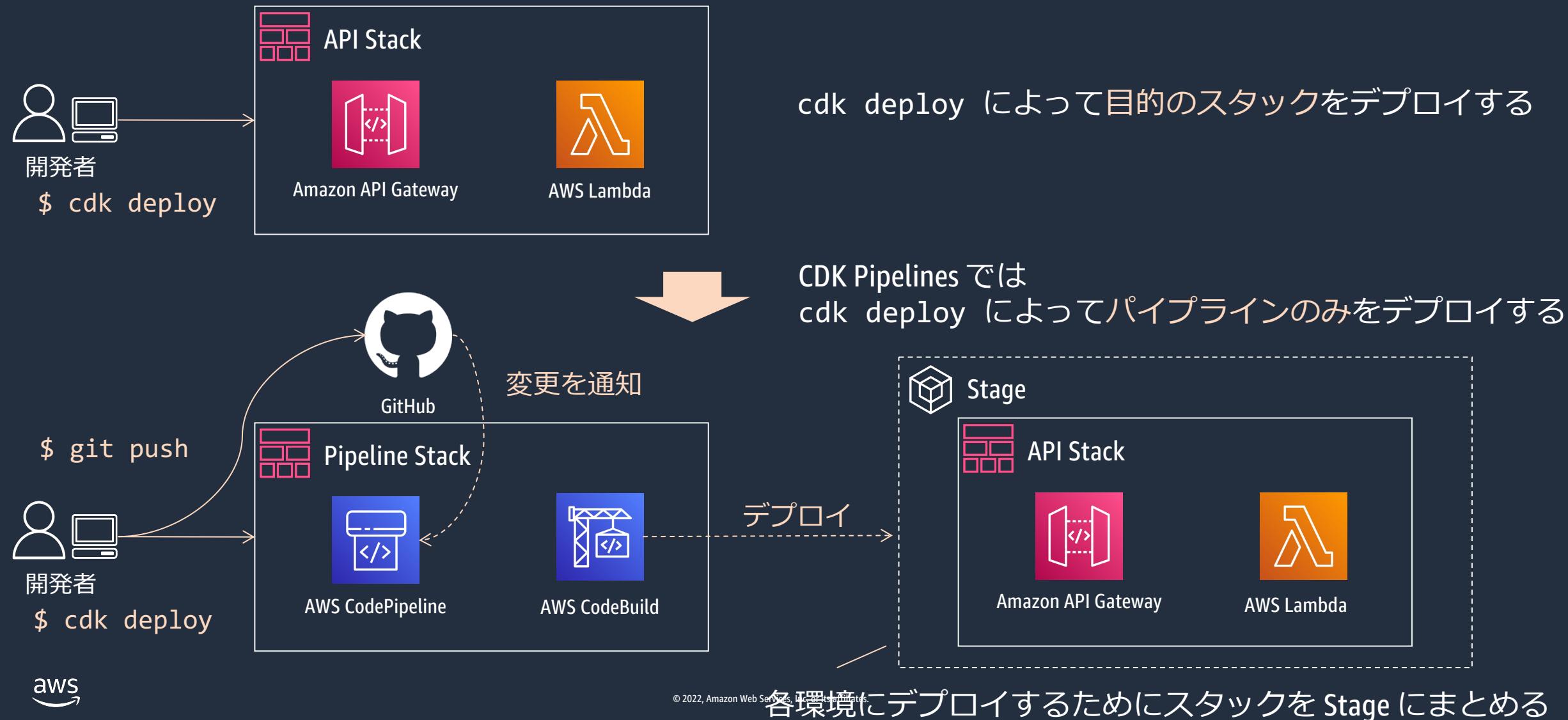
CDK Pipelines

CDKアプリケーションのCI/CDパイプラインのための高レベルライブラリ。

パイプラインは、`cdk deploy`を1回実行するだけで自己構成可能。その後、ソースコードに新しいCDKアプリケーションまたはステージを追加すると、パイプラインが自動的に更新される



既存の CDK アプリを CDK Pipelines に移行する



AWS CDK の学習をはじめる



AWS CDK のリソース

AWS CDK Developer Guide

https://docs.aws.amazon.com/ja_jp/cdk/v2/guide/home.html

AWS CDK のコンセプトや
ベストプラクティスなど
開発に役立つ情報を記載

API Reference

<https://docs.aws.amazon.com/cdk/api/v2/docs/aws-construct-library.html>

API の仕様はこちらで確認

CDK Workshop

<https://cdkworkshop.com/>
実際にコードを書きながら
CDK を学べるワークショップ
TypeScript, Python, .NET, Java に対応



AWS CDK Blackbelt



The slide features a dark teal background with a hexagonal geometric pattern. In the top left corner is the AWS logo. The title '[AWS Black Belt Online Seminar]' is in white, bold font. Below it, the subtitle 'AWS Cloud Development Kit (CDK)' is also in white, bold font. On the left side, there is Japanese text 'サービスカットシリーズ'. On the right side, there is information about the speaker: 'Senior Solutions Architect' and '大村 幸敬' (Miyatachi Yuki), along with the date '2020/03/03'. Below this, there is a link to an AWS webinar: 'AWS 公式 Webinar' and 'https://amzn.to/JPWebinar'. To the right of the link are two QR codes. One QR code links to the '過去資料' (Past Materials) page, with the URL 'https://amzn.to/JPArchive'. The other QR code links to the AWS logo. At the bottom left, there is a copyright notice: '© 2020, Amazon Web Services, Inc. or its Affiliates. All rights reserved.'

aws

[AWS Black Belt Online Seminar]

AWS Cloud Development Kit (CDK)

サービスカットシリーズ

Senior Solutions Architect
大村 幸敬
2020/03/03

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

過去資料
<https://amzn.to/JPArchive>

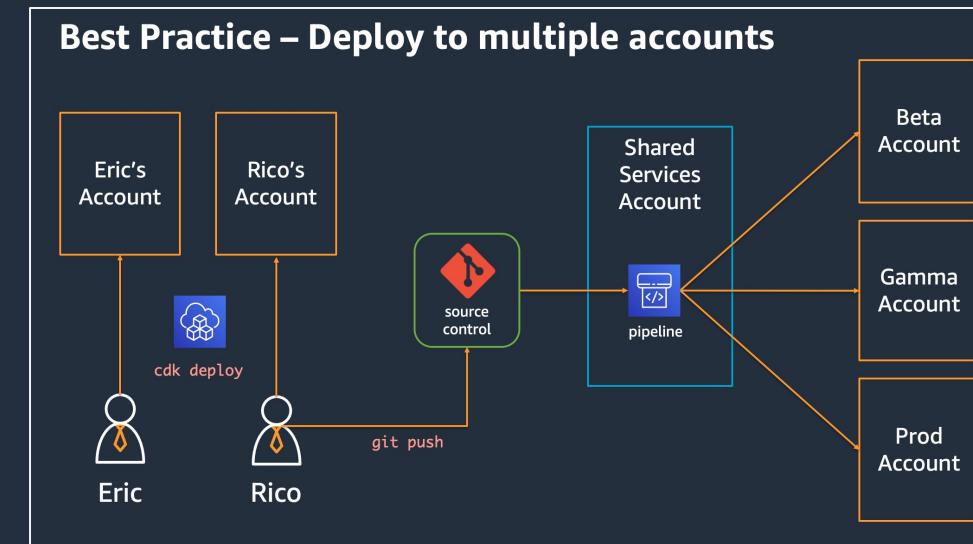
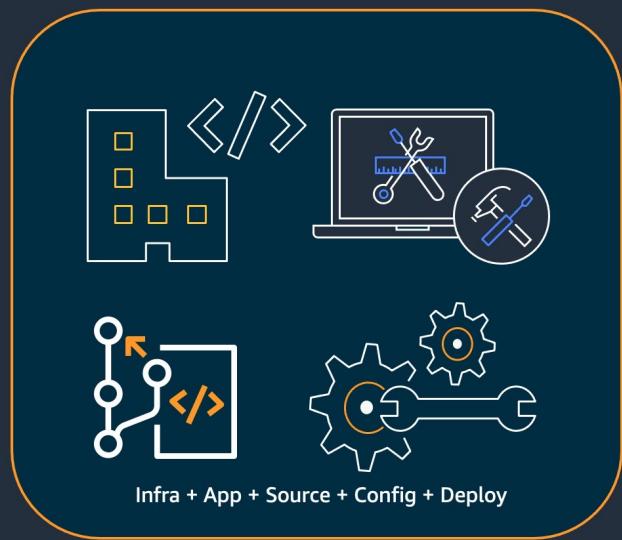
aws

<https://aws.amazon.com/jp/blogs/news/webinar-bb-aws-cloud-development-kit-cdk-2020/>

CDK のベストプラクティス

AWS CDKでクラウドアプリケーションを開発するためのベストプラクティス

<https://aws.amazon.com/jp/blogs/news/best-practices-for-developing-cloud-applications-with-aws-cdk/>





Thank you!