AWS Black Belt Online Seminar

AWS CloudFormation
#1 基礎編

福井 敦 (Fukui Atsushi)
Partner Sales Solutions Architect
2023/7

AWS Black Belt Online Seminar とは

- 「サービス別」「ソリューション別」「業種別」などのテーマに分け、アマゾン ウェブ サービス ジャパン合同会社が提供するオンラインセミナーシリーズです
- AWS の技術担当者が、AWS の各サービスやソリューションについてテーマごとに動画を公開します
- 以下の URL より、過去のセミナー含めた資料などをダウンロードすることができます
  - https://aws.amazon.com/jp/aws-jp-introduction/aws-jp-webinar-service-cut/
  - https://www.youtube.com/playlist?list=PLzWGOASvSx6FIwIC2X1nObr1KcMCBBlqY

ご感想は Twitter へ！ハッシュタグは以下をご利用ください
#awsblackbelt

内容についての注意点

- 本資料では 2023 年 07月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。AWS のサービスは常にアップデートを続けているため、最新の情報は AWS 公式ウェブサイト (https://aws.amazon.com/) にてご確認ください
- 資料作成には十分注意しておりますが、資料内の価格と AWS 公式ウェブサイト記載の価格に相違があった場合、AWS 公式ウェブサイトの価格を優先とさせていただきます
- 価格は税抜表記となっています。日本居住者のお客様には別途消費税をご請求させていただきます
- 技術的な内容に関しましては、有料の AWS サポート窓口へお問い合わせください
- 料金面でのお問い合わせに関しましては、カスタマーサポート窓口へお問い合わせください (マネジメントコンソールへのログインが必要です)

本セミナーの対象者

想定聴講者

- CloudFormation をこれから利用される方、概要をお知りになりたい方

前提知識

- AWS の概要を理解していること
- YAML、JSON のファイル形式について理解していること
- インフラ構成管理ツールのご利用経験があると望ましい（必須ではない）

ゴール

- CloudFormation を扱う上で必要な用語（テンプレート、スタック、変更セット）をご理解いただくこと
- CloudFormation の利用方法（スタックの作成方法とテンプレートの記述方法）についてイメージを掴んでいただくこと

自己紹介

名前

福井 敦 (Fukui Atsushi/@hukui)

所属

技術統括本部 Partner Sales Solutions Architect

経歴

国内 SIer で運用管理/インフラ設計構築を経験

好きなAWSサービス

AWS CloudFormation, AWS Systems Manager

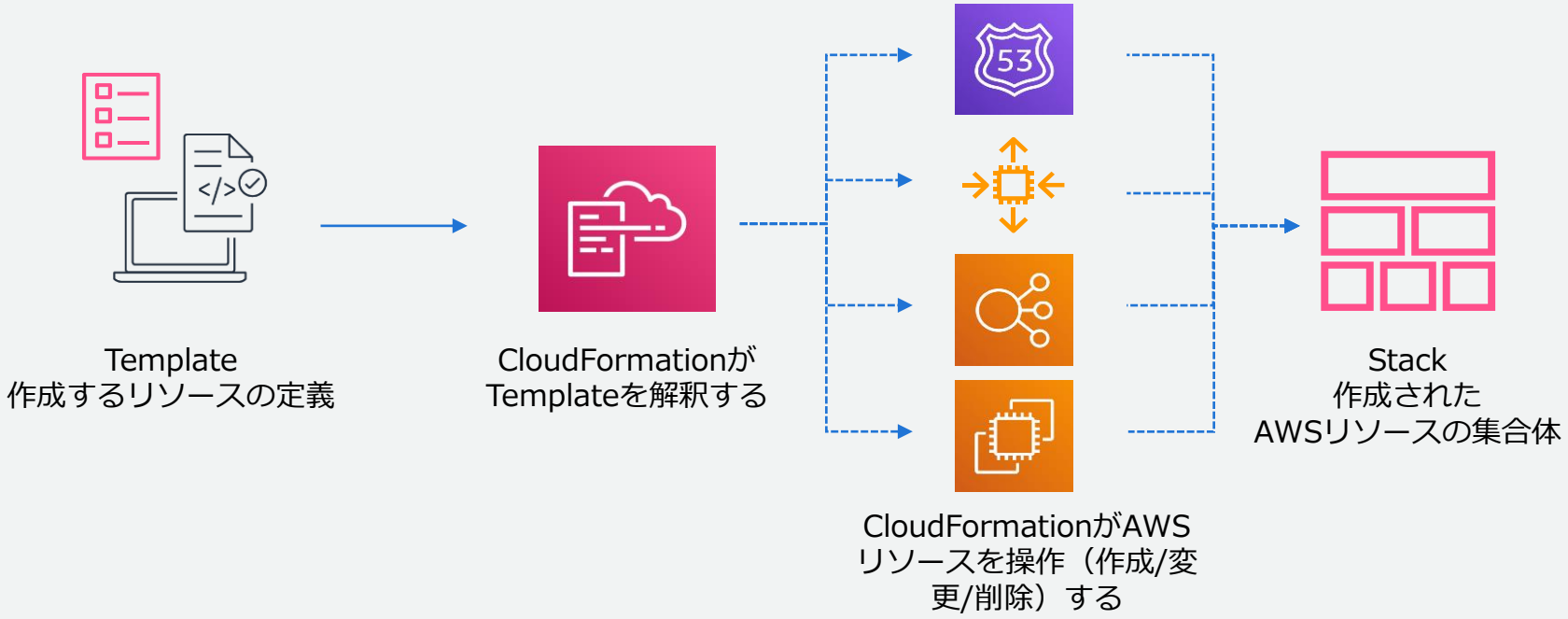
アジェンダ

1. AWS CloudFormation とは
2. CloudFormation を使った構成管理の流れ
3. テンプレートの構造
4. まとめ

AWS CloudFormation とは

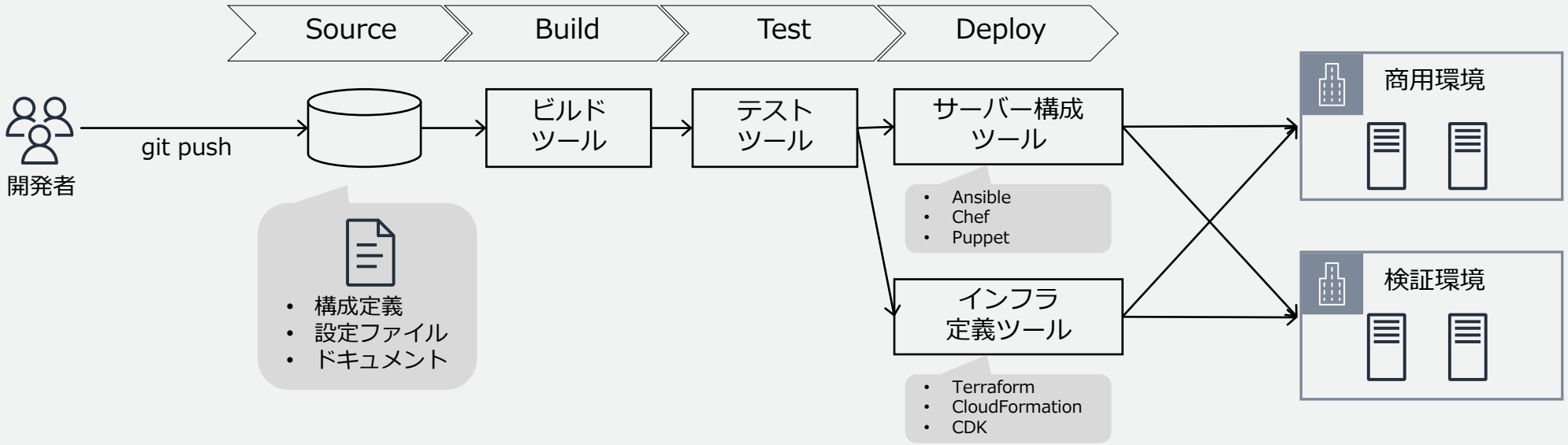
CloudFormation 概要

- あるべき状態を記載した設定ファイル (テンプレート) を元にAWS の構築を自動化できるサービス
- テンプレートにはリソースの設定情報を JSON や YAML 形式で記述する
- CloudFormation は作成したリソースの集合体をスタックという単位で管理する
- 追加料金は不要（プロビジョニングされた AWS リソース分の料金のみ発生）



Infrastructure as code (IaC)

- IaC とは、インフラをコードで定義し運用するアプローチのこと
  - コードで定義、作成し、コードの変更を通じてインフラストラクチャを更新することで複製や再構築が容易になり、手動のオペレーションを排除することができる
  - Version Control System※、CI/CD を活用するソフトウェアの開発手法をインフラストラクチャに適用し、開発効率の向上を計ることができる



※ Version Control Systems。Git や Subversion といったバージョン管理のソフトウェアを指す

CloudFormation 基本機能

作成 | 変更 | 削除

- テンプレートに定義された構成でスタックを自動作成
- 並列でリソースを作成し、依存関係がある場合は自動的に解決

メリット

✓構築作業を迅速化できる

✓手作業によるオペレーションミスを排除できる

✓一度テンプレートを作成すれば、検証環境の作成や別のリージョンへの展開などが容易

CloudFormation 基本機能

作成 | **変更** | 削除

- テンプレートで示された状態を目指し、現在の状態との差分を埋めるように働く
- 変更セットを作ることで差分の内容とリソースへの影響を事前に確認可能
- スタックの作成および変更中、アプリケーションの状態に問題が発生した場合にロールバックが可能

メリット

✓幂等性、ロールバックが担保される

✓変更前後のテンプレート（=テキストファイル）の違いを見れば、インフラストラクチャに対する変更内容を確認できる

注意： 手動で行った変更は CloudFormation の管理外になります。

更新動作について

CloudFormation は、現在のスタックの状態とアップロードされたテンプレートの違いに基づいてリソースを更新する。変更のないリソースは、更新中も中断されることなく実行され、変更のあるリソースは、以下のいずれかの動作をとる。どのリソースタイプに対してどのプロパティを更新するかによって動作が異なる。各プロパティの更新動作（Update requires）は「AWS リソースタイプのリファレンス」を参照。

中断を伴わない更新 – No interruption

リソースの使用を中断することなくリソースを更新する。物理IDに変更はない。

例：EC2のタグの変更

一時的な中断を伴う更新 – Some interruption

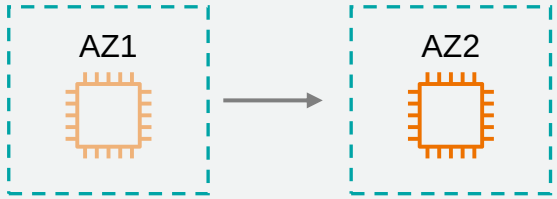
一部の実行を中断してリソースの更新を行う。物理IDに変更はない。

例：インスタンスタイプの変更

置換 - Replacement

リソースを再作成し、新しい物理 ID になる。まず置換先となる新しいリソースを作成してから古いリソースを削除する。

例：Availability Zone の変更



https://docs.aws.amazon.com/ja_jp/AWSCloudFormation/latest/UserGuide/using-cfn-updating-stacks-update-behaviors.html

CloudFormation 基本機能

作成 | 変更 | **削除**

- 依存関係を解決しつつリソースを全て削除
- RDS のようなデータストアはスナップショットの取得 / 保持が可能

メリット

✓削除作業を迅速化できる

✓手作業によるオペレーションミスを排除できる

CloudFormation の用語

CloudFormation

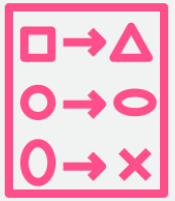
スタックの作成/変更/削除およびエラー検知とロールバックを行う

テンプレート

リソース、属性、依存関係についてあるべき状態を定義するもの

スタック

テンプレートからプロビジョニングされるリソースの集合のこと



変更セット

テンプレートの変更前後の差分と変更に伴う影響（無停止変更 / 再起動 / 再作成）を事前に確認するもの

CloudFormation の用語 - テンプレート

テンプレート

- 構築したいリソースの“設計図“。どんなリソースを構築するのか(状態)を記述
  - JSON / YAML フォーマットに対応

- JSON フォーマット

```
{
    "AWSTemplateFormatVersion": "2010-09-09",
    "Resources": {
        "S3Bucket": {
            "Type": "AWS::S3::Bucket",
            "Properties": {
                "AccessControl": "PublicRead",
                "WebsiteConfiguration": {
                    "IndexDocument": "index.html",
                    "ErrorDocument": "error.html"
                }
            },
            "DeletionPolicy": "Retain"
        },
        "BucketPolicy": {
            "Type": "AWS::S3::BucketPolicy"
```

- YAML フォーマット

```
AWSTemplateFormatVersion: 2010-09-09
Resources:
  S3Bucket:
    Type: 'AWS::S3::Bucket'
    Properties:
      AccessControl: PublicRead
      WebsiteConfiguration:
        IndexDocument: index.html
        ErrorDocument: error.html
    DeletionPolicy: Retain
  BucketPolicy:
    Type: 'AWS::S3::BucketPolicy'
    Properties:
      PolicyDocument:
        Id: MyPolicy
        Version: 2012-10-17
```

https://docs.aws.amazon.com/ja_jp/AWSCloudFormation/latest/UserGuide/gettingstarted.templatebasics.html

CloudFormation の用語 - スタック

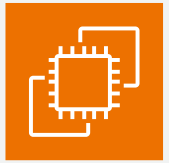
- テンプレートからプロビジョニングされるリソースの集合のことをスタックと呼ぶ
  - スタック単位でリソースの管理が可能
  - スタックの削除を実行すると、スタックに紐づくリソースが削除される
  - 使用するリソースおよびリソースの構築順は、テンプレートの依存関係から CloudFormation が自動的に決定

一例ですが、このような様々な種類のリソースを一度に作成/変更/削除が可能


S3


DB

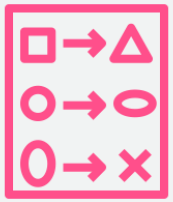

Web


App

Hosted Zone

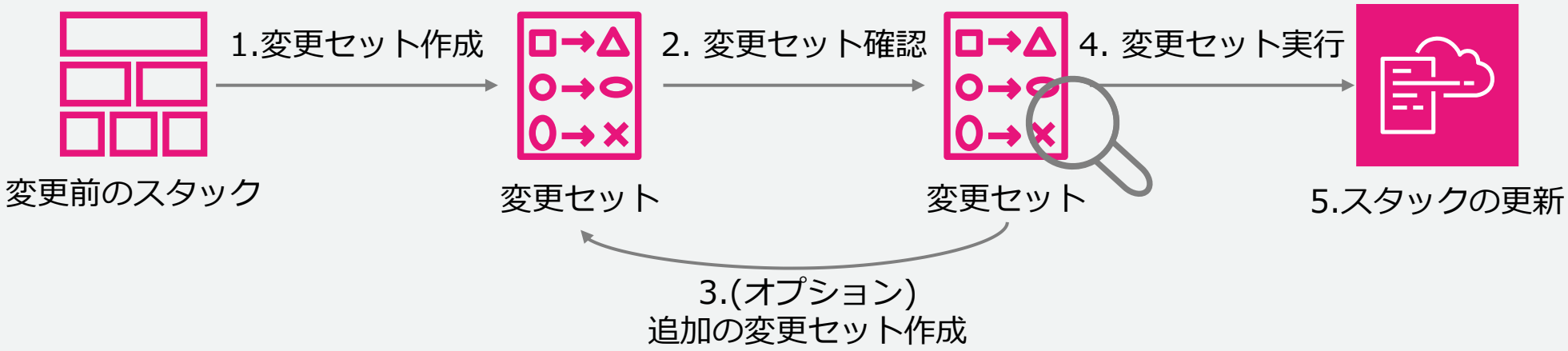
https://docs.aws.amazon.com/ja_jp/AWSCloudFormation/latest/UserGuide/stacks.html

CloudFormation の用語 - 変更セット



変更セット

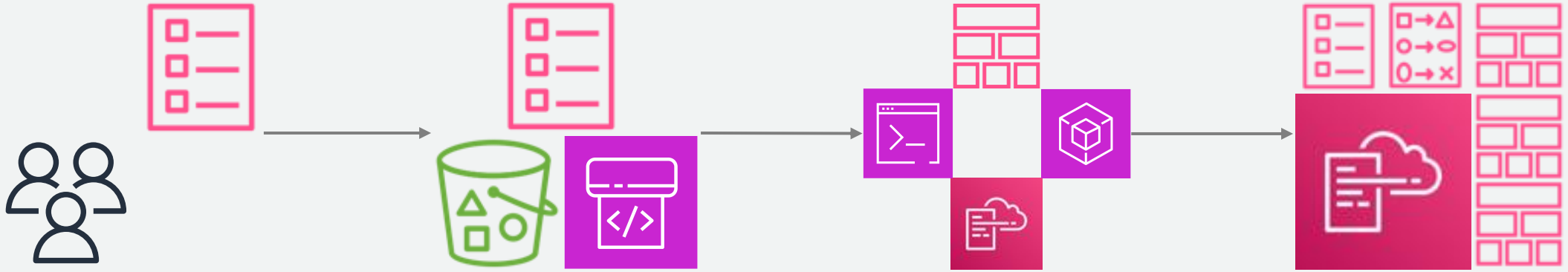
- 稼働中のリソースに与える影響をスタックの変更前に確認できる
  - スタックを更新する前に、続行するか他の変更セットを作成するかを検討できる



https://docs.aws.amazon.com/ja_jp/AWSCloudFormation/latest/UserGuide/using-cfn-updating-stacks-changesets.html

CloudFormation を使った構成管理の流れ

CloudFormation を使った構成管理の流れ



YAML/JSONで記述

AWS SAM や AWS CDK、サンプルテンプレートの利用も可

アップロード

ローカルファイルをマネジメントコンソールやS3 バケット、パイプライン経由でアップロード

スタックの作成

マネジメントコンソール、AWS CLI、AWS SDK、スタックセットを利用してスタックを作成する

リソースの作成と管理

スタックの詳細情報を確認し、以降スタックリソースを単一ユニットとして扱う

CloudFormation を使った構成管理の流れ

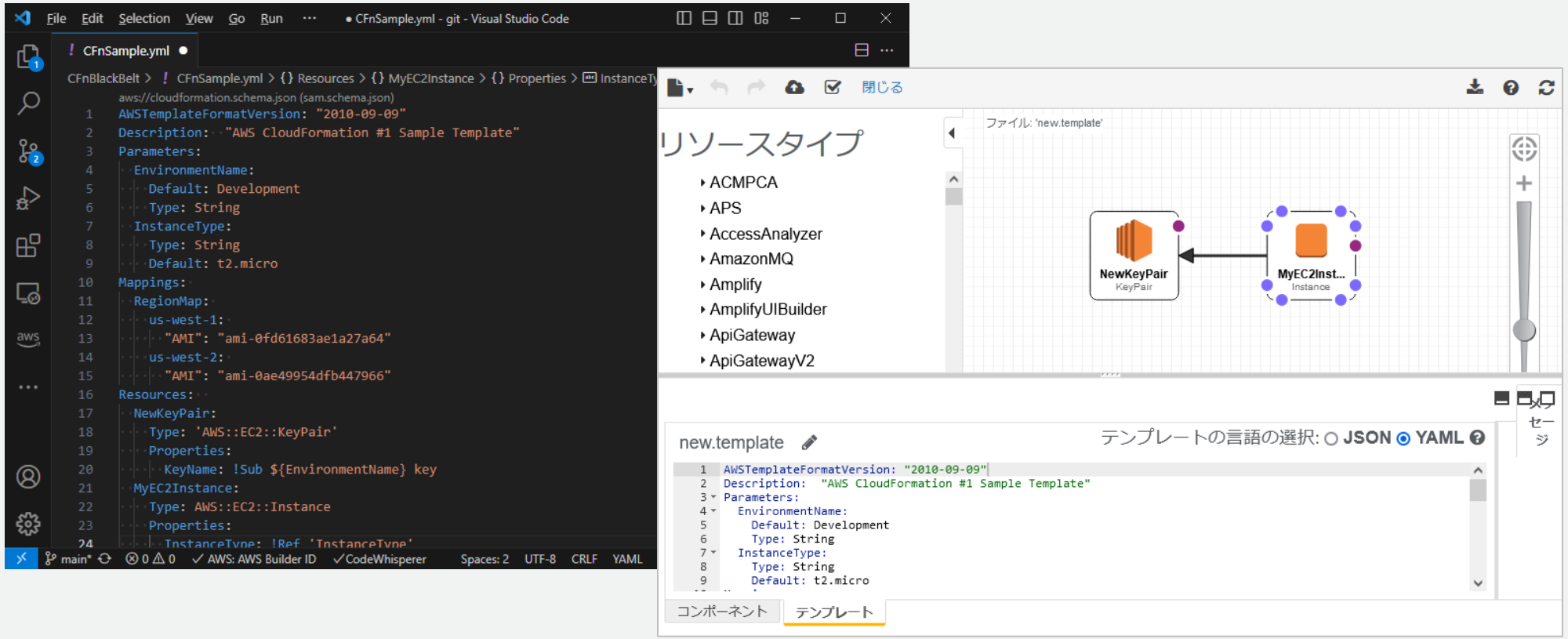
YAML/JSONで記述

アップロード

スタックの作成

リソースの作成と管理

- リファレンスを参考に、お好きなエディタでテンプレートを記述
  - （一例）左図が Visual Studio Code、右図が AWS CloudFormation デザイナー
  - テンプレートの記述内容は、後述



https://docs.aws.amazon.com/ja_jp/AWSCloudFormation/latest/UserGuide/working-with-templates-cfn-designer.html

CloudFormation を使った構成管理の流れ

- AWSのマネジメントコンソールにて、前述のテンプレートをアップロード
- ローカルのファイルを選択してアップロードするか、S3にある場合はそちらを選択することも可能

https://docs.aws.amazon.com/ja_jp/AWSCloudFormation/latest/UserGuide/cfn-using-console-create-stack-template.html

CloudFormation を使った構成管理の流れ

YAML/JSONで記述

アップロード

スタックの作成

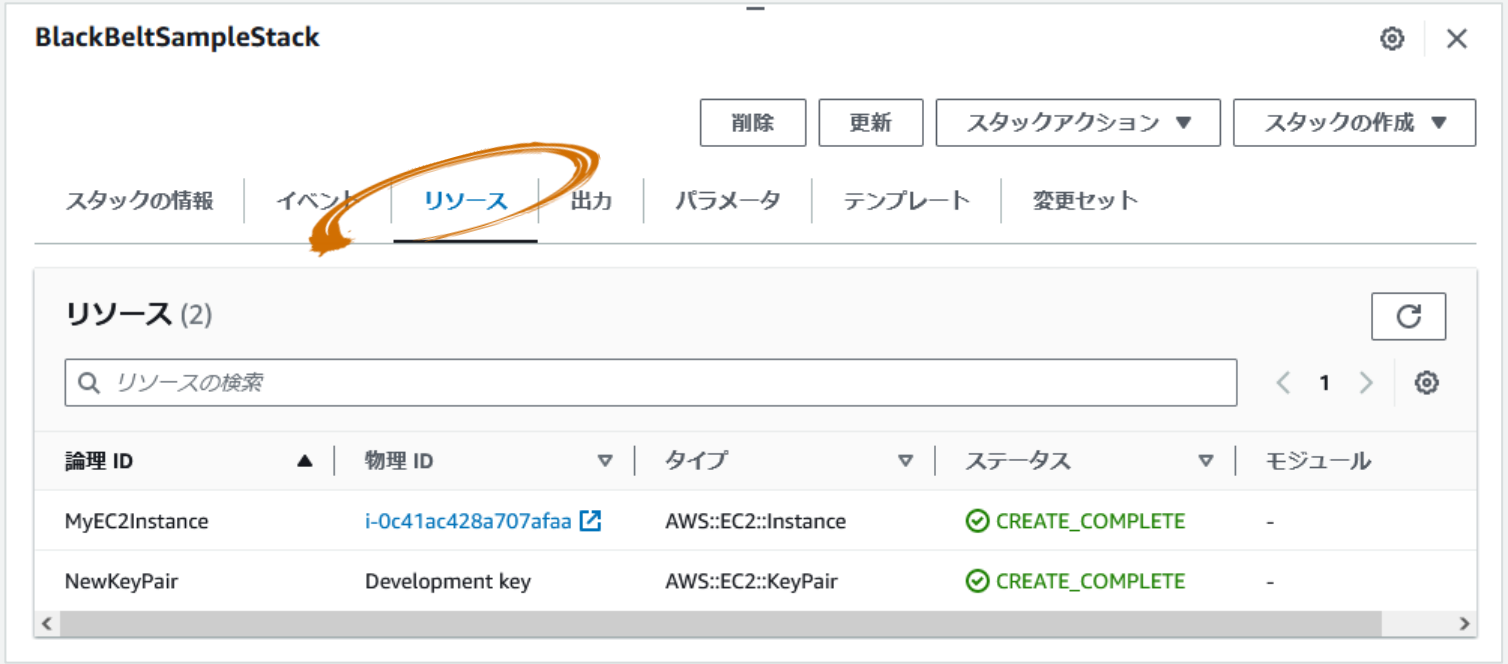
リソースの作成と管理

- レビューページで、スタックの設定を確認
- 必要に応じてIAMリソースが作成されることを承認して、スタックを作成

https://docs.aws.amazon.com/ja_jp/AWSCloudFormation/latest/UserGuide/cfn-using-console-create-stack-review.html

CloudFormation を使った構成管理の流れ

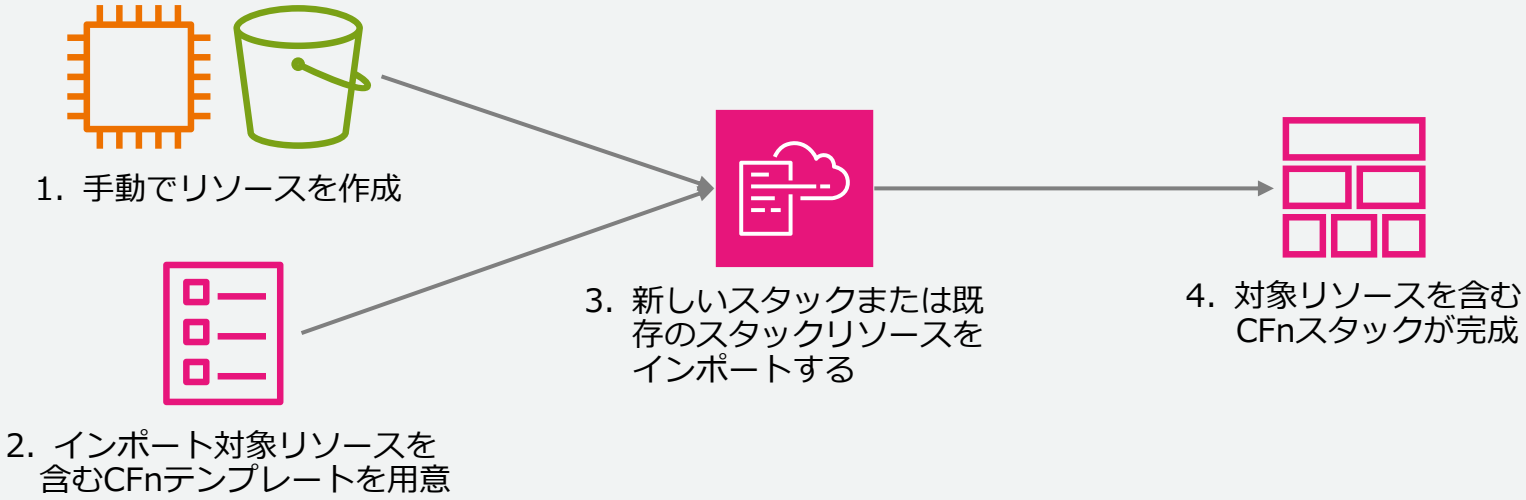
- スタック作成処理が完了すると、リソースタブで確認できる
- 更新する際は、再度テンプレートをアップロードして更新する



https://docs.aws.amazon.com/ja_jp/AWSCloudFormation/latest/UserGuide/cfn-using-console-create-stack-review.html

インポート機能について

- 手動で作成した AWS リソースをあとから CloudFormation スタックにインポートして管理可能
  - リソースをスタックの管理下から切り離したり、別のスタック管理下に移動することも可能
  - インポート対象のリソースの実際の設定と一致するようテンプレートを用意する
    - AWS 公式のツールではないが、Former2 でテンプレート作成の省力化が可能
      - https://github.com/iann0036/former2



https://docs.aws.amazon.com/ja_jp/AWSCloudFormation/latest/UserGuide/resource-import.html

テンプレートの構造

テンプレートの要素

- テンプレートは下表に示すセクションから構成される
  - Resources だけが必須のセクション
  - セクションの順序は任意だが、あるセクションの値が前のセクションの値を参照する場合があるため、下表の順序を推奨

順序	セクション名	説明	必須
1	AWSTemplateFormatVersion	テンプレートのバージョン	No
2	Description	テンプレートの説明文	No
3	Metadata	テンプレートに関する追加情報	No
4	Parameters	実行時にユーザ入力を求めるパラメータ	No
5	Rules	スタックの作成または更新前に入力されたパラメータを検証	No
6	Mappings	条件パラメータ値の指定に使用	No
7	Conditions	リソースが作成または設定される条件を登録	No
8	Transform	変換および拡張処理の呼出しに使用	No
9	Resources	スタックを構成するリソースを定義	**Yes**
10	Outputs	スタック構築後に出力させる値（DNS名やIPアドレスなど）	No

https://docs.aws.amazon.com/ja_jp/AWSCloudFormation/latest/UserGuide/template-anatomy.html

テンプレートの記述例

- 書き出しは、**AWSTemplateFormatVersion** と **Description** で始めましょう
  - 最新のフォーマットバージョンは 2010-09-09 であり、現時点で唯一の有効な値
  - Description セクションにテンプレートに関する説明を記載

順序	セクション名	説明
1	AWSTemplateFormatVersion	テンプレートのバージョン
2	Description	テンプレートの説明文
3	Metadata	テンプレートに関する追加情報
4	Parameters	実行時にユーザ入力を求めるパラメータ
5	Rules	スタックの作成または更新前にパラメータを検証
6	Mappings	条件パラメータ値の指定に使用
7	Conditions	リソースが作成または設定される条件を登録
8	Transform	変換および拡張処理の呼出しに使用
9	Resources	スタックを構成するリソースを定義
10	Outputs	スタック構築後に出力させる値

```
AWSTemplateFormatVersion: "2010-09-09"
Description:  "AWS CloudFormation #1 Sample Template"
```


https://docs.aws.amazon.com/ja_jp/AWSCloudFormation/latest/UserGuide/format-version-structure.html
https://docs.aws.amazon.com/ja_jp/AWSCloudFormation/latest/UserGuide/template-description-structure.html

テンプレートの記述例

- 必須の **Resources** セクションに作成したい EC2 インスタンスなどのリソースを定義しましょう
  - リソースタイプ毎に定められているプロパティを記述する
  - 利用可能なリソースタイプとプロパティはドキュメントを参照

順序	セクション名	説明
1	AWSTemplateFormatVersion	テンプレートのバージョン
2	Description	テンプレートの説明文
3	Metadata	テンプレートに関する追加情報
4	Parameters	実行時にユーザ入力を求めるパラメータ
5	Rules	スタックの作成または更新前にパラメータを検証
6	Mappings	条件パラメータ値の指定に使用
7	Conditions	リソースが作成または設定される条件を登録
8	Transform	変換および拡張処理の呼出しに使用
9	Resources	スタックを構成するリソースを定義
10	Outputs	スタック構築後に出力させる値

```yaml
AWSTemplateFormatVersion: "2010-09-09"
Description:  "AWS CloudFormation #1 Sample Template“

Resources:
  NewKeyPair:
    Type: 'AWS::EC2::KeyPair'
    Properties:
      KeyName: Development key
  MyEC2Instance:
    Type: AWS::EC2::Instance
    Properties:
      InstanceType: t2.micro
      ImageId: ami-0ae49954dfb447966
      KeyName: !Ref NewKeyPair
```


https://docs.aws.amazon.com/ja_jp/AWSCloudFormation/latest/UserGuide/aws-template-resource-type-ref.html

AWS::EC2::KeyPair

RSS

フィルタビュー All

Specifies a key pair for use with an Amazon Elastic Compute Cloud instance as follows:

- To import an existing key pair, include the `PublicKeyMaterial` property.
- To create a new key pair, omit the `PublicKeyMaterial` property.

When you import an existing key pair, you specify the public key material for the key. We assume that you have the private key material for the key. AWS CloudFormation does not create or return the private key material when you import a key pair.

When you create a new key pair, the private key is saved to AWS Systems Manager Parameter Store, using a parameter with the following name: `/ec2/keypair/{key-pair-id}`. For more information about retrieving private key, and the required permissions, see Create a key pair using AWS CloudFormation in the *Amazon EC2 User Guide*.

When AWS CloudFormation deletes a key pair that was created or imported by a stack, it also deletes the parameter that was used to store the private key material in Parameter Store.

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

YAML

```
Type: AWS::EC2::KeyPair
Properties:
  KeyFormat: String
  KeyName: String
  KeyType: String
  PublicKeyMaterial: String
  Tags:
    - Tag
```

Properties

`KeyFormat`

The format of the key pair.

Default: `pem`

Required: No

Type: String

Allowed values: `pem` | `ppk`

Update requires: Replacement

`KeyName`

A unique name for the key pair.

Constraints: Up to 255 ASCII characters

Required: Yes

Type: String

Update requires: Replacement

`KeyType`

The type of key pair. Note that ED25519 keys are not supported for Windows instances.

If the `PublicKeyMaterial` property is specified, the `KeyType` property is ignored, and the key type is inferred from the `PublicKeyMaterial` value.

Default: `rsa`

Required: No

Type: String

Allowed values: `ed25519` | `rsa`

Update requires: Replacement

`PublicKeyMaterial`

The public key material. The `PublicKeyMaterial` property is used to import a key pair. If this property is not specified, then a new key pair will be created.

Required: No

Type: String

Update requires: Replacement

`Tags`

The tags to apply to the key pair.

Required: No

Type: List of Tag

Update requires: Replacement

Return values

Ref

When you pass the logical ID of this resource to the intrinsic `Ref` function, `Ref` returns the name of the key pair.

For more information about using the `Ref` function, see Ref.

Fn::GetAtt

`KeyFingerprint`

If you created the key pair using Amazon EC2:

- For RSA key pairs, the key fingerprint is the SHA-1 digest of the DER encoded private key.
- For ED25519 key pairs, the key fingerprint is the base64-encoded SHA-256 digest, which is the default for OpenSSH, starting with OpenSSH 6.8 [5].

If you imported the key pair to Amazon EC2:

- For RSA key pairs, the key fingerprint is the MD5 public key fingerprint as specified in section 4 of RFC 4716.
- For ED25519 key pairs, the key fingerprint is the base64-encoded SHA-256 digest, which is the default for OpenSSH, starting with OpenSSH 6.8 [5].

`KeyPairId`

The ID of the key pair.

Examples

Create a new key pair and specify it when launching an instance

The following example omits the `PublicKeyMaterial` property to create a new key pair, and specifies the key pair when launching an instance.

YAML

```
Resources:
  MyKeyPair:
    Type: 'AWS::EC2::KeyPair'
    Properties:
      KeyName: MyKeyPair
  MyEC2Instance:
```

AWS::EC2::KeyPair

RSS

フィルタビュー All

All

JSON

YAML

Specifies a key pair for use with an Amazon Elastic Compute Cloud

follows:

- To import an existing key pair, include the PublicKeyMaterial property.

https://docs.aws.amazon.com/ja_jp/AWSCloudFormation/latest/UserGuide/aws-template-resource-type-ref.html

AWS::EC2::KeyPair

Specifies a key pair for use with an Amazon Elastic Compute Cloud instance as follows:

- To import an existing key pair, include the `PublicKeyMaterial` property.
- To create a new key pair, omit the `PublicKeyMaterial` property.

When you import an existing key pair, you specify the public key material for the key. AWS CloudFormation does not create or return the private key material when you import a key pair.

When you create a new key pair, the private key is saved to AWS Systems Manager Parameter Store, using a parameter with the following name: `/ec2/keypair/{key_pair_id}`. For more information about retrieving private key, and the required permissions, see Create a key pair using AWS CloudFormation in the *Amazon EC2 User Guide*.

When AWS CloudFormation deletes a key pair that was created or imported by a stack, it also deletes the parameter that was used to store the private key material in Parameter Store.

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

YAML

```
Type: AWS::EC2::KeyPair
Properties:
  KeyFormat: String
  KeyName: String
  KeyType: String
  PublicKeyMaterial: String
  Tags:
    - Tag
```

Properties

`KeyFormat`

The format of the key pair.

Default: `pem`

Required: No

Type: String

Allowed values: `pem` | `ppk`

Update requires: Replacement

`KeyName`

A unique name for the key pair.

Constraints: Up to 255 ASCII characters

Required: Yes

Type: String

Update requires: Replacement

`KeyType`

The type of key pair. Note that ED25519 keys are not supported for Windows instances.

If the `PublicKeyMaterial` property is specified, the `KeyType` property is ignored, and the key type is inferred from the `PublicKeyMaterial` value.

Default: `rsa`

Required: No

Type: String

Allowed values: `ed25519` | `rsa`

Update requires: Replacement

`PublicKeyMaterial`

The public key material. The `PublicKeyMaterial` property is used to import a key pair. If this property is not specified, then a new key pair will be created.

Required: No

Type: String

Update requires: Replacement

`Tags`

The tags to apply to the key pair.

Required: No

Type: List of Tag

Update requires: Replacement

Return values

Ref

When you pass the logical ID of this resource to the intrinsic `Ref` function, `Ref` returns the name of the key pair.

For more information about using the `Ref` function, see Ref.

Fn::GetAtt

`KeyFingerprint`

If you created the key pair using Amazon EC2:

- For RSA key pairs, the key fingerprint is the SHA-1 digest of the DER encoded private key.
- For ED25519 key pairs, the key fingerprint is the base64-encoded SHA-256 digest, which is the default for OpenSSH, starting with OpenSSH 6.8 [5].

If you imported the key pair to Amazon EC2:

- For RSA key pairs, the key fingerprint is the MD5 public key fingerprint as specified in section 4 of RFC 4716.
- For ED25519 key pairs, the key fingerprint is the base64-encoded SHA-256 digest, which is the default for OpenSSH, starting with OpenSSH 6.8 [5].

`KeyPairId`

The ID of the key pair.

Examples

Create a new key pair and specify it when launching an instance

The following example omits the `PublicKeyMaterial` property to create a new key pair, and specifies the key pair when launching an instance.

YAML

```
Resources:
  MyKeyPair:
    Type: 'AWS::EC2::KeyPair'
    Properties:
      KeyName: MyKeyPair
  MyInstance:
```


Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

YAML

```
Type: AWS::EC2::KeyPair
Properties:
  KeyFormat: String
  KeyName: String
  KeyType: String
  PublicKeyMaterial: String
  Tags:
    - Tag
```


https://docs.aws.amazon.com/ja_jp/AWSCloudFormation/latest/UserGuide/aws-template-resource-type-ref.html

AWS::EC2::KeyPair

Specifies a key pair for use with an Amazon Elastic Compute Cloud instance as follows:

- To import an existing key pair, include the `PublicKeyMaterial` property.
- To create a new key pair, omit the `PublicKeyMaterial` property.

When you import an existing key pair, you specify the public key material for the key. We assume that you have the private key material for the key. AWS CloudFormation does not create or return the

When you create a new key pair, the private key is saved to AWS Systems Manager Parameter Store, using a parameter with the following name: `/ec2/keypair/{key_pair_id}`. For more informa

When AWS CloudFormation deletes a key pair that was created or imported by a stack, it also deletes the parameter that was used to store the private key material in Parameter Store.

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

YAML

```
Type: AWS::EC2::KeyPair
Properties:
  KeyFormat: String
  KeyName: String
  KeyType: String
  PublicKeyMaterial: String
  Tags:
    - Tag
```

Properties

`KeyFormat`

The format of the key pair.

Default: `pem`

Required: No

Type: String

Allowed values: `pem` | `ppk`

Update requires: Replacement

`KeyName`

A unique name for the key pair.

Constraints: Up to 255 ASCII characters

Required: Yes

Type: String

Update requires: Replacement

`KeyType`

The type of key pair. Note that ED25519 keys are not supported for Windows instances.

If the `PublicKeyMaterial` property is specified, the `KeyType` property is ignored, and the key type is inferred from the `PublicKeyMaterial` value.

Default: `rsa`

Required: No

Type: String

Allowed values: `ed25519` | `rsa`

Update requires: Replacement

`PublicKeyMaterial`

The public key material. The `PublicKeyMaterial` property is used to import a key pair. If this property is not specified, then a new key pair will be created.

Required: No

Type: String

Update requires: Replacement

`Tags`

The tags to apply to the key pair.

Required: No

Type: List of Tag

Update requires: Replacement

Return values

Ref

When you pass the logical ID of this resource to the intrinsic `Ref` function, `Ref` returns the name of the key pair.

For more information about using the `Ref` function, see Ref.

Fn::GetAtt

`KeyFingerprint`

If you created the key pair using Amazon EC2:

- For RSA key pairs, the key fingerprint is the SHA-1 digest of the DER encoded private key.
- For ED25519 key pairs, the key fingerprint is the base64-encoded SHA-256 digest, which is the default for OpenSSH, starting with OpenSSH 6.8 [5].

If you imported the key pair to Amazon EC2:

- For RSA key pairs, the key fingerprint is the MD5 public key fingerprint as specified in section 4 of RFC 4716.
- For ED25519 key pairs, the key fingerprint is the base64-encoded SHA-256 digest, which is the default for OpenSSH, starting with OpenSSH 6.8 [5].

`KeyPairId`

The ID of the key pair.

Examples

Create a new key pair and specify it when launching an instance

The following example omits the `PublicKeyMaterial` property to create a new key pair, and specifies the key pair when launching an instance.

YAML

```
Resources:
  MyNewKeyPair:
    Type: 'AWS::EC2::KeyPair'
    Properties:
      KeyName: MyKeyPair
  MyEC2Instance:
```

Properties

KeyFormat

The format of the key pair.

Default: `pem`

Required: No

Type: String

Allowed values: `pem` | `ppk`

Update requires: Replacement

KeyName

A unique name for the key pair.

Constraints: Up to 255 ASCII characters

Required: Yes

Type: String

Update requires: Replacement

KeyType

AWS::EC2::KeyPair

Specifies a key pair for use with an Amazon Elastic Compute Cloud instance as follows:

- To import an existing key pair, include the `PublicKeyMaterial` property.
- To create a new key pair, omit the `PublicKeyMaterial` property.

When you import an existing key pair, you specify the public key material for the key. We assume that you have the private key material for the key. AWS CloudFormation does not create or return the private key material for the key.

When you create a new key pair, the private key is saved to AWS Systems Manager Parameter Store, using a parameter with the following name: `/ec2/keypair/{key_pair_id}`. For more information about

When AWS CloudFormation deletes a key pair that was created or imported by a stack, it also deletes the parameter that was used to store the private key material in Parameter Store.

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

YAML

```
Type: AWS::EC2::KeyPair
Properties:
  KeyFormat: String
  KeyName: String
  KeyType: String
  PublicKeyMaterial: String
  Tags:
    - Tag
```

Properties

`KeyFormat`

The format of the key pair.

Default: `pem`

Required: No

Type: String

Allowed values: `pem` | `ppk`

Update requires: Replacement

`KeyName`

A unique name for the key pair.

Constraints: Up to 255 ASCII characters

Required: Yes

Type: String

Update requires: Replacement

`KeyType`

The type of key pair. Note that ED25519 keys are not supported for Windows instances.

If the `PublicKeyMaterial` property is specified, the `KeyType` property is ignored, and the key type is inferred from the `PublicKeyMaterial` value.

Default: `rsa`

Required: No

Type: String

Allowed values: `ed25519` | `rsa`

Update requires: Replacement

`PublicKeyMaterial`

The public key material. The `PublicKeyMaterial` property is used to import a key pair. If this property is not specified, then a new key pair will be created.

Required: No

Type: String

Update requires: Replacement

`Tags`

The tags to apply to the key pair.

Required: No

Type: List of Tag

Update requires: Replacement

Return values

Ref

When you pass the logical ID of this resource to the intrinsic `Ref` function, `Ref` returns the name of the key pair.

For more information about using the `Ref` function, see R

Fn::GetAtt

`KeyFingerprint`

If you created the key pair using Amazon EC2:

- For RSA key pairs, the key fingerprint is the SHA-1 digest of the DER encoded private key.
- For ED25519 key pairs, the key fingerprint is the base64-encoded SHA-256 digest, which is the default for OpenSSH, starting with OpenSSH 6.8.

If you imported the key pair to Amazon EC2:

- For RSA key pairs, the key fingerprint is the MD5 public key fingerprint as specified in section 4 of RFC 4716.
- For ED25519 key pairs, the key fingerprint is the base64-encoded SHA-256 digest, which is the default for OpenSSH, starting with OpenSSH 6.8.

`KeyPairId`

The ID of the key pair.

Examples

Create a new key pair and specify it when launching an instance

The following example omits the `PublicKeyMaterial` property to create a new key pair, and specifies the key pair when launching an instance.

YAML

```
Resources:
  NewKeyPair:
    Type: 'AWS::EC2::KeyPair'
    Properties:
      KeyName: MyKeyPair
  Ec2Instance:
    Type: 'AWS::EC2::Instance'
    Properties:
      ImageId: ami-02b92c281a4d3dc79
      KeyName: !Ref NewKeyPair
```


Examples

Create a new key pair and specify it when launching an instance

The following example omits the `PublicKeyMaterial` property to create a new key pair, and specifies the key pair when launching an instance.

YAML

```
Resources:
  NewKeyPair:
    Type: 'AWS::EC2::KeyPair'
    Properties:
      KeyName: MyKeyPair
  Ec2Instance:
    Type: 'AWS::EC2::Instance'
    Properties:
      ImageId: ami-02b92c281a4d3dc79
      KeyName: !Ref NewKeyPair
```

テンプレートの記述例

- スタック作成時、ユーザに設定値を入力させたい場合は **Parameters** セクションを使いましょう
  - データ型、デフォルト値、最大最小値などのプロパティを設定可能
  - 入力された値は、テンプレート中で”!Ref”という組み込み関数を使って参照

順序	セクション名	説明
1	AWSTemplateFormatVersion	テンプレートのバージョン
2	Description	テンプレートの説明文
3	Metadata	テンプレートに関する追加情報
4	Parameters	実行時にユーザ入力を求めるパラメータ
5	Rules	スタックの作成または更新前にパラメータを検証
6	Mappings	条件パラメータ値の指定に使用
7	Conditions	リソースが作成または設定される条件を登録
8	Transform	変換および拡張処理の呼出しに使用
9	Resources	スタックを構成するリソースを定義
10	Outputs	スタック構築後に出力させる値

```yaml
AWSTemplateFormatVersion: "2010-09-09"
Description:  "AWS CloudFormation #1 Sample Template"
Parameters:
  EnvironmentName:
    Default: Development
    Type: String
  InstanceType:
    Default: t2.micro
    Type: String



Resources:
  NewKeyPair:
    Type: 'AWS::EC2::KeyPair'
    Properties:
      KeyName: !Sub ${EnvironmentName} key
  MyEC2Instance:
    Type: AWS::EC2::Instance
    Properties:
      InstanceType: !Ref 'InstanceType'
      ImageId: ami-0ae49954dfb447966
      KeyName: !Ref NewKeyPair
```


https://docs.aws.amazon.com/ja_jp/AWSCloudFormation/latest/UserGuide/parameters-section-structure.html

テンプレートの記述例

- リージョンや環境別に AMI を使いわけたいなど、条件によって設定値を変えたい場合は、**Mappings** セクションを使いましょう
  - キーと値のマッピングテーブルによってテンプレートの再利用性が向上
  - 組み込み関数”Find::InMap”を使って値を取得

順序	セクション名	説明
1	AWSTemplateFormatVersion	テンプレートのバージョン
2	Description	テンプレートの説明文
3	Metadata	テンプレートに関する追加情報
4	Parameters	実行時にユーザ入力を求めるパラメータ
5	Rules	スタックの作成または更新前にパラメータを検証
6	Mappings	条件パラメータ値の指定に使用
7	Conditions	リソースが作成または設定される条件を登録
8	Transform	変換および拡張処理の呼出しに使用
9	Resources	スタックを構成するリソースを定義
10	Outputs	スタック構築後に出力させる値

```
AWSTemplateFormatVersion: "2010-09-09"
Description:  "AWS CloudFormation #1 Sample Template"
Parameters:
  EnvironmentName:
    Default: Development
    Type: String
  InstanceType:
    Default: t2.micro
    Type: String
Mappings:
  RegionMap:
    us-west-1:
      "AMI": "ami-0fd61683ae1a27a64"
    us-west-2:
      "AMI": "ami-0ae49954dfb447966"
Resources:
  NewKeyPair:
    Type: 'AWS::EC2::KeyPair'
    Properties:
      KeyName: !Sub ${EnvironmentName} key
  MyEC2Instance:
    Type: AWS::EC2::Instance
    Properties:
      InstanceType: !Ref 'InstanceType'
      ImageId:
        Fn::FindInMap:
          - "RegionMap"
          - Ref: "AWS::Region"
          - "AMI"
      KeyName: !Ref NewKeyPair
```


https://docs.aws.amazon.com/ja_jp/AWSCloudFormation/latest/UserGuide/mappings-section-structure.html

テンプレートの記述例

- インスタンス ID や IP アドレスなど、スタック構築後に確認、使用したい情報がある場合は **Outputs** セクションを使いましょう
  - Outputs（出力）のエクスポート名を介して別のスタックからリソースを参照する用途にも用いる（クロススタック参照）

順序	セクション名	説明
1	AWSTemplateFormatVersion	テンプレートのバージョン
2	Description	テンプレートの説明文
3	Metadata	テンプレートに関する追加情報
4	Parameters	実行時にユーザ入力を求めるパラメータ
5	Rules	スタックの作成または更新前にパラメータを検証
6	Mappings	条件パラメータ値の指定に使用
7	Conditions	リソースが作成または設定される条件を登録
8	Transform	変換および拡張処理の呼出しに使用
9	Resources	スタックを構成するリソースを定義
10	Outputs	スタック構築後に出力させる値

```yaml
AWSTemplateFormatVersion: "2010-09-09"
Description:  "AWS CloudFormation #1 Sample Template"
Parameters:
  EnvironmentName:
    Default: Development
    Type: String
  InstanceType:
    Default: t2.micro
    Type: String
Mappings:
  RegionMap:
    us-west-1:
      "AMI": "ami-0fd61683ae1a27a64"
    us-west-2:
      "AMI": "ami-0ae49954dfb447966"
Resources:
  NewKeyPair:
    Type: 'AWS::EC2::KeyPair'
    Properties:
      KeyName: !Sub ${EnvironmentName} key
  MyEC2Instance:
    Type: AWS::EC2::Instance
    Properties:
      InstanceType: !Ref 'InstanceType'
      ImageId:
        Fn::FindInMap:
          - "RegionMap"
          - Ref: "AWS::Region"
          - "AMI"
      KeyName: !Ref NewKeyPair
Outputs:
  MyEC2InstanceId:
    Value: !Ref MyEC2Instance
  MyEC2InstancePrivateIp:
    Value: !GetAtt MyEC2Instance.PrivateIp
```


https://docs.aws.amazon.com/ja_jp/AWSCloudFormation/latest/UserGuide/outputs-section-structure.html

テンプレートの要素と構成管理の流れ

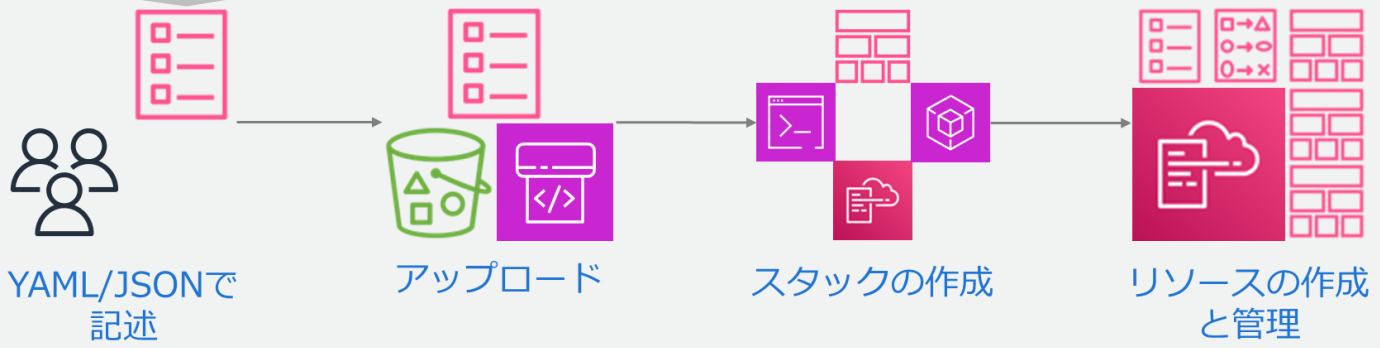
順序	セクション名	説明
1	AWSTemplateFormatVersion	テンプレートのバージョン
2	Description	テンプレートの説明文
3	**Metadata**	**テンプレートに関する追加情報**
4	Parameters	実行時にユーザ入力を求めるパラメータ
5	**Rules**	**スタックの作成または更新前にパラメータを検証**
6	Mappings	条件パラメータ値の指定に使用
7	**Conditions**	**リソースが作成または設定される条件を登録**
8	**Transform**	**変換および拡張処理の呼出しに使用**
9	Resources	スタックを構成するリソースを定義
10	Outputs	スタック構築後に出力させる値

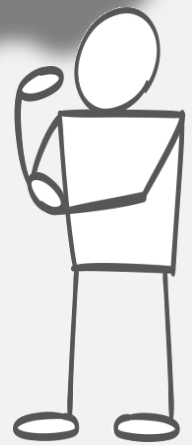
スタックにメタデータを
付加したい…
→ **Metadata セクション**

ユーザが入力する
パラメータを検証したい…
→ **Rules セクション**

条件が真の時にだけ、
作成または設定したい…
→ **Condition セクション**

テンプレートを
変換、拡張したい…
→ **Transform セクション**





まとめ

まとめ

- AWS CloudFormation とは
  - **AWS CloudFormation** とは、あるべき状態を記載した設定ファイル (テンプレート) を元にAWS の構築を自動化できるサービス
  - **テンプレート**とは、構築したいリソースの“設計図“。どんなリソースを構築するのか(状態)を記述したテキストファイル
  - **スタック**とは、テンプレートからプロビジョニングされるリソースの集合
  - **変更セット**を作成すると稼働中のリソースに与える影響をスタックの変更前に確認できる
- テンプレートの構成
  - テンプレートを構成する**セクション**について記述例を通して紹介しました

参考資料

- ドキュメント
  - https://docs.aws.amazon.com/ja_jp/cloudformation/index.html#lang/ja_jp
- よくある質問
  - https://aws.amazon.com/jp/cloudformation/faqs/
- aws re:Post 情報センター
  - https://repost.aws/ja/tags/knowledge-center/TAm3R3LNU3RfSX9L23YIpo3w/aws-cloudformation
- AWS CloudFormation Workshop
  - https://catalog.workshops.aws/cfn101/en-US
- ロードマップ
  - https://github.com/aws-cloudformation/cloudformation-coverage-roadmap
- AWS 初心者向けハンズオン（AWS Hands-on for Beginners）
  - https://aws.amazon.com/jp/events/aws-event-resource/hands-on/
- AWS クラウドサービス活用資料集（AWS Black Belt Online Seminar）
  - https://aws.amazon.com/jp/events/aws-event-resource/archive/

Thank you!

Appendix

AWSTemplateFormatVersion と Description

- AWSTemplateFormatVersion
  - 最新のフォーマットバージョンは 2010-09-09 であり、現時点で唯一の有効な値
  - 値を指定しない場合、CloudFormation は最新のフォーマットバージョンを使用する
- Description
  - Description セクションにテンプレートに関する説明やコメントをつけることができる
  - Description セクションだけを修正してスタックを更新することはできない
  - 最大1024バイトのリテラル文字列にする必要があり関数やパラメータを使うことはできない

```
AWSTemplateFormatVersion: "2010-09-09"
Description:  >
  Here are some
  details about
  the template.
```

https://docs.aws.amazon.com/ja_jp/AWSCloudFormation/latest/UserGuide/format-version-structure.html
https://docs.aws.amazon.com/ja_jp/AWSCloudFormation/latest/UserGuide/template-description-structure.html

Metadata

- テンプレートに関する追加情報として JSON または YAML オブジェクトを含めることができる
  - スタックの一部として作成する EC2 インスタンスから、この Metadata を取得して初期設定に活用したりできる
  - Metadata セクションにパスワードなどの機密情報を書かないこと
  - Metadata セクションだけを修正してスタックを更新することはできない

```yaml
Metadata:
  Instances:
    Description: "Information about the instances"
  Databases:
    Description: "Information about the databases"
```

https://docs.aws.amazon.com/ja_jp/AWSCloudFormation/latest/UserGuide/metadata-section-structure.html

Metadata Key

CloudFormation の一部の機能(パラメータや CloudFormation デザイナーなど)から、Metadata セクションで定義した内容を利用できる。
例えば、AWS::CloudFormation::Interface というキーを用いてパラメーターをグループ化し、順序を指定できる（マネジメントコンソールの画面上でパラメータをわかりやすく表示できる）

AWS::CloudFormation::Interface メタデータキーが**ない**とき

```yaml
Parameters:
  VPCId:
    Type: String
  SubnetId:
    Type: String
  KeyName:
    Type: String
```

マネジメントコンソールの表示

AWS::CloudFormation::Interface メタデータキーが**ある**とき

```yaml
Metadata:
  AWS::CloudFormation::Interface:
    ParameterGroups:
      - Label:
          default: "Network Configuration"
        Parameters:
          - VPCId
          - SubnetId
      - Label:
          default: "EC2 Configuration"
        Parameters:
          - KeyName
Parameters:
  VPCId:
    Type: String
  SubnetId:
    Type: String
  KeyName:
    Type: String
```

マネジメントコンソールの表示

https://docs.aws.amazon.com/ja_jp/AWSCloudFormation/latest/UserGuide/metadata-section-structure.html

Parameters

- パラメーターを使用すると、スタックを作成または更新するたびにユーザに指定させる値を定義できる
  - データ型、デフォルト値、最大最小値などのプロパティを設定可能
  - 入力されたパラメータの値は、テンプレート中で”!Ref”という組み込み関数を使って参照
  - テンプレートあたり最大200 個のパラメーターを指定可能

```yaml
Parameters:
  S3NameParam:
    Type: String
    Default: mybucket
    Description: Name for your AWS S3 bucket
    MinLength: 5
    MaxLength: 30
Resources:
  S3Bucket:
    Type: AWS::S3::Bucket
    Properties:
      AccessControl: PublicRead
      BucketName: !Ref S3NameParam
    DeletionPolicy: Retain
```

https://docs.aws.amazon.com/ja_jp/AWSCloudFormation/latest/UserGuide/parameters-section-structure.html

Parameters のプロパティ

プロパティ	内容
Type	データ型。String, Number, CommaDelimitedList, AWS 固有のパラメータタイプ, SSM パラメータータイプ をサポート
Default	デフォルト値。スタックの作成時に値を指定しなかった場合に使用される
NoEcho	入力されたパラメータ値をアスタリスク(*)でマスクする
AllowedValues	入力可能値の一覧指定 （例：[“true”,”false”] ）
AllowedPattern	正規表現で入力可能パターンを指定（例：[a-zA-Z]*）
MaxLength	最大文字数
MinLength	最小文字数
MaxValue	最大値
MinValue	最小値
Description	パラメータの詳細説明。最大4000文字
ConstraintDescription	入力した値がAllowedPatternやMaxLengthなどの制約に引っかかった場合に表示する説明（どのような制約があるかの説明を記述）

https://docs.aws.amazon.com/ja_jp/AWSCloudFormation/latest/UserGuide/parameters-section-structure.html

疑似パラメータ（Pseudo Parameter）

CloudFormation によって事前定義されていて、テンプレートでは宣言不要のパラメータ群。“Ref” で参照できる

擬似パラメータ名	説明
AWS::AccountId	AWSアカウントIDを取得
AWS::NotificationARNs	notificationAmazonResourceNames(ARNs)を取得
AWS::NoValue	指定されたプロパティを無視するようCloudFormationに伝える
AWS::Region	リージョン名を取得
AWS::StackId	スタックIDを取得
AWS::StackName	スタック名を取得

```
MyDB:
  Type: AWS::RDS::DBInstance
  Properties:
    DBSnapshotIdentifier:
      Fn::If:
        - UseDBSnapshot
        - Ref: DBSnapshotName # UseDBSnapshotがTrueのとき「DBSnapshotIdentifier」としてDBSnapshotNameの値を使う
        - Ref: AWS::NoValue # Falseのときプロパティ「DBSnapshotIdentifier」が定義されていないものとしてDBSnapshotIdentifierを無視する
    Tags:
      - key:
        Value: !Ref AWS::Region
```

http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/pseudo-parameter-reference.html

組み込み関数

- 組み込み関数は、パラメータの参照や値の加工などに利用する
  - 記法は「Fn::<関数名>」。YAMLの場合は、短縮形「!<関数名>」も利用可

完全関数名	短縮形(YAMLの場合)	機能概要
Fn::Base64	!Base64	文字列をBase64エンコードする
Fn::FindInMap	!FindInMap	Mappingsのキーに対応する値を取得する
Fn::ForEach	*N/A*	ループ処理によって繰り返し定義する
Fn::GetAtt	!GetAtt	リソースの属性値を取得する　例）"Fn::GetAtt" : ["MyELB" , "DNSName"]
Fn::GetAZs	!GetAZs	指定したリージョンのアベイラビリティゾーンのリストを取得する
Fn::ImportValue	!ImportValue	別のスタックにてエクスポートされた出力の値を取得する（クロススタック参照）
Fn::Join	!Join	文字列を結合する 例）“Fn::Join” : [“:”, [“a”, “b”]]　は　「a:b」を返す
Fn::Select	!Select	Index値に応じた値をListから選択する 例）{ “Fn::Select” : [“1”, [“Jan”, “Feb”, “Mar”, “Apr”, “Jun”]] } は”Feb”を返す
Fn::Sub	!Sub	文字列内の変数を指定した値で置き換える
Ref	!Ref	指定したパラメータのキーから値、またはリソースの論理IDから物理IDを参照する

https://docs.aws.amazon.com/ja_jp/AWSCloudFormation/latest/UserGuide/intrinsic-function-reference.html

Resources

- 唯一、必須のセクション。作成、更新する Amazon EC2 インスタンスや Amazon S3 バケットなどのリソースを定義する
  - リソース毎に定められているプロパティを記述する
  - 利用可能なリソースタイプとプロパティはリファレンスを参照
  - プロパティの値には、リテラル文字列、文字列のリスト、ブール値、パラメーター、組み込み関数によって返される値が使用可能

```yaml
Resources:
  MyEC2Instance: #論理ID
    Type: "AWS::EC2::Instance" #リソースタイプ
    Properties: #リソースごとのプロパティ
      AvailabilityZone: "us-east-1a"
      ImageId: "ami-0ff8a91507f77f867"
      InstanceType: !Ref InstanceType
      KeyName: !Ref KeyName
```

https://docs.aws.amazon.com/ja_jp/AWSCloudFormation/latest/UserGuide/resources-section-structure.html

論理IDと物理ID

- 論理ID
  - テンプレート内で一意
  - テンプレートの他の部分のリソースを参照するために使用
  - !Refや!GetAttで使用
- 物理ID
  - リソースに実際に割り当てられている名前（EC2のインスタンスID、S3バケット名など）
  - AWS CloudFormationテンプレート外のリソースを識別する場合に使用

```yaml
Resources:
  MyEC2Instance: #論理ID
    Type: "AWS::EC2::Instance”
    Properties:
      SubnetId: "subnet-xxxxxxxxxxxxxxxx" #ここで指定しているのは物理ID
Outputs:
  MyEC2PhysicalID:
    Value: !Ref MyEC2Instance #！Ref を使って論理IDから物理IDを取得
```

出力

キー	値	説明
MyEC2PhysicalID	i-0d6c1d5278786087a	

https://docs.aws.amazon.com/ja_jp/AWSCloudFormation/latest/UserGuide/resources-section-structure.html#resources-section-structure-resource-fields

Mappings

- キーと値のマッピングテーブルを管理できる
  - リージョンやユーザ入力パラメータによって、値が変わるものに利用
  - Mappingsを利用することでテンプレートの再利用性が向上
  - 組み込み関数”Find::InMap”を使って値を取得
  - 例）"Fn::FindInMap" : ["MapName", "Key", "Value"]
  - 引数のMapName,Key,Valueには”!Ref”が利用可能

```yaml
Mappings:
  RegionMap: #RegionMap という名称でMappings を定義
    ap-northeast-1:
      AMIID: ami-06cd52961ce9f0d85
      KeyPair: My-Key-Tokyo
    ap-northeast-3:
      AMIID: ami-06cd52961ce9f0d85
      KeyPair: My-Key-Osaka
Resources:
  myEC2Instance:
    Type: "AWS::EC2::Instance"
    Properties: #FindInMap 関数によってRegion に合致するAMIIDの値を取得する
      ImageId: !FindInMap [RegionMap, !Ref "AWS::Region", AMIID]
      InstanceType: m1.small
```

https://docs.aws.amazon.com/ja_jp/AWSCloudFormation/latest/UserGuide/mappings-section-structure.html

Conditions

- Resourcesセクションなどで、“ある条件が成立しているときのみリソースを作成“ といった条件ベースの制御が可能
  - Conditions セクションに条件名と成立条件を列挙
  - Fn:: If、Fn:: Equals、Fn:: Not などの組み込み関数を使用可
  - 条件が true となるリソースが作成され、false のリソースは無視される
  - スタックの更新時、リソースが更新される前にこれらの条件が再評価される
  - Conditions セクションだけを修正してスタックを更新することはできない

```
Parameters:
  EnvType:
    Description: "Environment type."
    Default: "development"
    Type: String
    AllowedValues: ["production", "staging", "development"]
    ConstraintDescription: "must specify."
Conditions:
  # EnvTypeの値が”production”であれば、CreateProdResources条件が成立
  CreateProdResources: {"Fn::Equals" : [{"Ref" : "EnvType"}, “production"]}
Resources:
  Ec2Instance:
    Type: "AWS::EC2::Instance"
    # CreateProdResources条件が成立した場合、EC2リソースを作成
    Condition: "CreateProdResources"
```

https://docs.aws.amazon.com/ja_jp/AWSCloudFormation/latest/UserGuide/conditions-section-structure.html

Transform

- スタックや変更セットの作成時に、テンプレートを変換するための事前定義済の CloudFormation マクロを呼び出す
  - CloudFormation マクロには、CloudFormation によって管理されているマクロ（ AWS::Serverless や AWS::Include など）とユーザが定義するマクロがある

AWS::Serverless の例：サーバーレスアプリケーションの場合、使用するAWS SAMのバージョンを指定

```
Transform: AWS::Serverless-2016-10-31
Resources:
  MyServerlessFunctionLogicalID:
```

AWS::Include の例：メインのテンプレートとは別のS3に保存されたテンプレートスニペットを指定

```
Transform:
  Name: 'AWS::Include'
  Parameters:
    Location: 's3://MyAmazonS3BucketName/MyFileName.yaml'
```

https://docs.aws.amazon.com/ja_jp/AWSCloudFormation/latest/UserGuide/transform-section-structure.html

Outputs

- スタック構築後に取得・表示したい情報を定義できる
  - アクセスURLや、DBの通信先情報、作成したIAMユーザー名など、あとで使用したい情報がある場合に便利
  - 出力のエクスポート名を介して別のスタックからリソースを参照できるので、易い粒度でスタックを分割できる。（クロススタック参照）
  - Outputs セクションにはパスワードなどの機密情報を書かないこと

```yaml
Resources:
  EC2WebServer01:
    Type: AWS::EC2::Instance
    Properties:
      ImageId: !Ref EC2AMI
      InstanceType: t2.micro
Outputs:
  EC2WebServer01: #出力データの名称。キー。
    Value: !Ref EC2WebServer01 #出力するデータ。値。
    Export:
      Name: !Sub ${AWS::StackName}-EC2WebServer01 #組み込み関数を使って文字列を加工
```

キー	値	説明	エクスポート名
EC2WebServer01	i-0bad9055e937004e5	-	handson-cfn-ec2-EC2WebServer01

https://docs.aws.amazon.com/ja_jp/AWSCloudFormation/latest/UserGuide/outputs-section-structure.html

Rules

- スタックの作成または更新時にテンプレートに渡されるパラメータまたはパラメータの組み合わせを検証できる
  - ルールは、RuleCondition と、Assertions の2つのプロパティから成り、RulesCondition が true に評価された場合に、Assertions を評価してパラメータ値が有効かどうか確認する
  - Rules セクション固有の組み込み関数が使用できる

    Fn::And
    Fn::Contains
    Fn::EachMemberEquals
    Fn::EachMemberIn
    Fn::Equals
    Fn::If
    Fn::Not
    Fn::Or
    Fn::RefAll
    Fn::ValueOf
    Fn::ValueOfAll

```
Rules:
  testInstanceType:
    RuleCondition: !Equals
      - !Ref Environment
      - test
    Assertions:
      - Assert:
          'Fn::Contains':
            - - a1.medium
            - !Ref InstanceType
        AssertDescription:
          'For a test environment, the instance type must be a1.medium'
```

https://docs.aws.amazon.com/ja_jp/AWSCloudFormation/latest/UserGuide/rules-section-structure.html