



[AWS Black Belt Online Seminar]

このコンテンツは公開から3年以上経過しており内容が古い可能性があります
最新情報については[サービス別資料](#)もしくはサービスのドキュメントをご確認ください

AWS Amplify

サービスカットシリーズ

Solutions Architect 水馬 拓也
2020/05/20

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



過去資料
<https://amzn.to/JPArchive>



自己紹介

水馬 拓也 (みずまたくや)

- 所属
アマゾン ウェブ サービス ジャパン 株式会社
Prototyping Specialist Solutions Architect

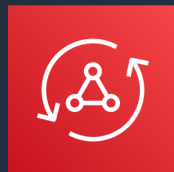


- 好きなサービス

AWS Amplify



AWS AppSync



AWS Lambda



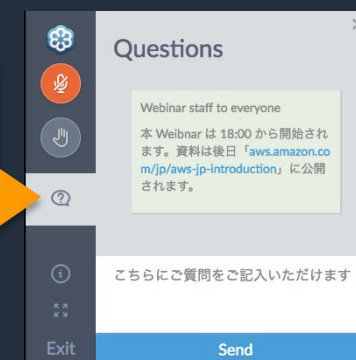
AWS Black Belt Online Seminar とは

「サービス別」「ソリューション別」「業種別」のそれぞれのテーマに分かれて、アマゾンウェブサービスジャパン株式会社が主催するオンラインセミナーシリーズです。

質問を投げることができます！

- 書き込んだ質問は、主催者にしか見えません
- 今後のロードマップに関するご質問は
お答えできませんのでご了承下さい

- ① 吹き出しをクリック
- ② 質問を入力
- ③ Sendをクリック



 Twitter ハッシュタグは以下をご利用ください
#awsblackbelt

内容についての注意点

- 本資料では2020年05月20日時点のサービス内容および価格についてご説明しています。最新の情報はAWS公式ウェブサイト(<http://aws.amazon.com>)にてご確認ください。
- 資料作成には十分注意しておりますが、資料内の価格とAWS公式ウェブサイト記載の価格に相違があった場合、AWS公式ウェブサイトの価格を優先とさせていただきます。
- 価格は税抜表記となっております。日本居住者のお客様には別途消費税をご請求させていただきます。
- AWS does not offer binding price quotes. AWS pricing is publicly available and is subject to change in accordance with the AWS Customer Agreement available at <http://aws.amazon.com/agreement/>. Any pricing information included in this document is provided only as an estimate of usage charges for AWS services based on certain information that you have provided. Monthly charges will be based on your actual use of AWS services, and may vary from the estimates provided.

本セッションの目的

- Amplify の概要をご理解いただく
- Amplify を用いた開発ワークフローをご理解いただく

想定聴講者

- Amplify 全体の概要を知りたい方
- これから Amplify を用いた開発にチャレンジされたい方
- すでにある程度 Amplify をご利用されており、直近のアップデートや機能改めて整理されたい方

本日のアジェンダ

1. Amplify の概要説明
2. Amplify を使ったアプリケーション開発
3. Amplify + GraphQL Deep Dive
4. 直近のアップデート
5. お客様からよくあるご質問
6. まとめ

Amplify の概要説明



AWS Amplify とは？

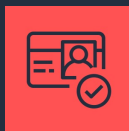


- AWS Amplify は Web フロントエンド、モバイルアプリの開発を加速させるために作られたプラットフォーム
- AWS を用いたサーバーレスなバックエンドの構築するための CLI や、バックエンドと接続するためのクライアントライブラリ、Web サイトのホスティングの仕組みを持つ

なぜ Amplify なのか？

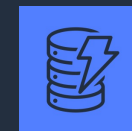
- ✓ アプリケーション開発に必要な一般的な技術要素をゼロから構築するのは重労働
- ✓ 本来のアプリケーションの実装に集中したい

認証機能



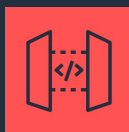
Amazon Cognito

データストア



Amazon DynamoDB

API 基盤



Amazon
API Gateway



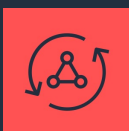
AWS AppSync

ストレージ



Amazon Simple
Storage Service

リアルタイム通知

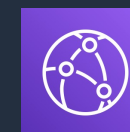


AWS AppSync



AWS IoT Core

Web ホスティング



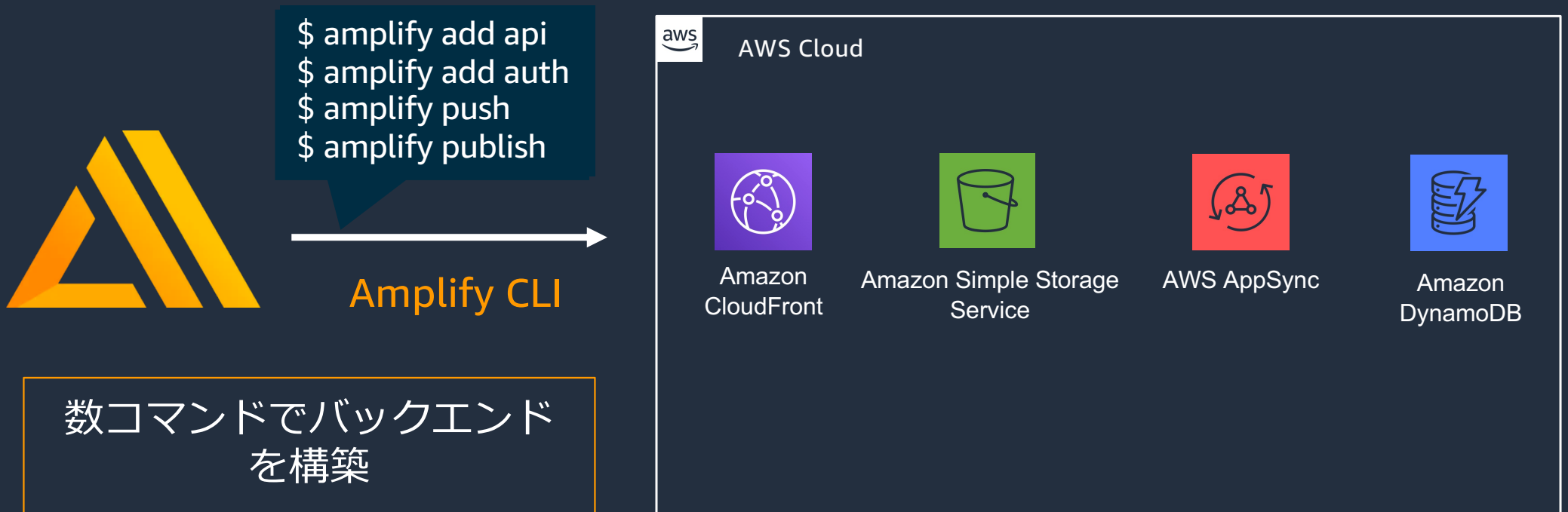
Amazon
CloudFront



Amazon Simple
Storage Service

Amplify で解決されること

- ✓ Amplify CLI を用いて様々なバックエンドをコマンドラインで操作、管理



Amplify で解決されること

- ✓ **Amplify Framework** と呼ばれる、バックエンドに直感的なインターフェースで接続できるライブラリ
- ✓ JavaScript、iOS、Android に対応

例) アクセスを所有者のみに制限したファイルアップロード機能

考えなければいけないこと

- アップロード処理
- ファイルの適切な権限設定
- 不正なユーザが更新しようとしたときの例外処理

```
Storage.put('test.png', file: {  
  contentType: 'image/png'  
  level: 'private',  
})  
});
```



React



Angular



Vue



Ionic



React Native



iOS



Android

本日のアジェンダ

1. Amplify の概要説明
- 2. Amplify を使ったアプリケーション開発**
3. Amplify + GraphQL Deep Dive
4. 直近のアップデート
5. お客様からよくあるご質問
6. まとめ

Amplify を使ったアプリケーション開発のワークフロー

1. Amplify CLI を用いたバックエンドの構築



2. Amplify Framework を用いたアプリケーションの実装



3. アプリケーションのデプロイ

Amplify を使ったアプリケーション開発のワークフロー

1. Amplify CLI を用いたバックエンドの構築



2. Amplify Framework を用いたアプリケーションの実装



3. アプリケーションのデプロイ

Amplify CLI を使ったバックエンドの構築

Amplify CLI
のインストール

プロジェクトに
Amplify を導入

機能(カテゴリ)
の設定

バックエンドの
構築

Amplify CLI を使ったバックエンドの構築

Amplify CLI
のインストール

プロジェクトに
Amplify を導入

機能(カテゴリ)
の設定

バックエンドの
構築

- ローカル端末に Amplify CLI のインストール

```
$ npm install -g @aws-amplify/cli
```

- Amplify CLI の初期セットアップ

```
$ amplify configure
```

- コマンドラインから初期セットアップの選択肢を選択する
- この時指定する情報
 - ✓ 使用するリージョンを
 - ✓ Amplify CLI が使用する クレデンシャル情報

```
Specify the AWS Region
? region: (Use arrow keys)
> us-east-1
  us-east-2
  us-west-2
  eu-west-1
  eu-west-2
  eu-central-1
  ap-northeast-1
(Move up and down to reveal more choices)
```


Amplify CLI を使ったバックエンドの構築

Amplify CLI
のインストール

プロジェクトに
Amplify を導入

機能(カテゴリ)
の設定

バックエンドの
構築

- プロジェクトのルートディレクトリで初期化コマンドを実行

```
$ amplify init
```

- コマンドラインからプロジェクトのセットアップ情報を選択
- この時指定する情報
 - ✓ プロジェクト名
 - ✓ 環境名 (production / staging / develop / test ..etc)
 - ✓ 使用するプラットフォーム (プログラム言語/フレームワーク)
 - ✓ プロジェクトが使用するプロファイル情報
 - ✓ 使用するエディタ
 - ✓ プロジェクトのディレクトリ情報 ..等

Amplify CLI を使ったバックエンドの構築

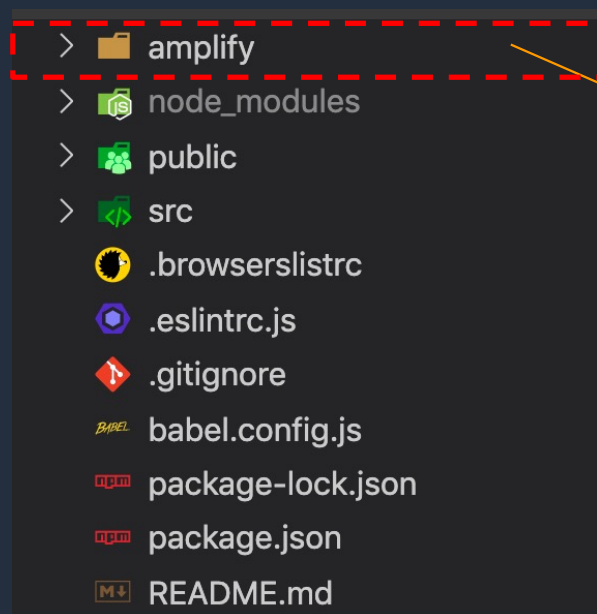
Amplify CLI
のインストール

プロジェクトに
Amplify を導入

機能(カテゴリ)
の設定

バックエンドの
構築

- \$amplify init コマンドが完了すると、プロジェクトのルートディレクトリに amplify フォルダが作成される



このフォルダにこれから作成する
バックエンドの設定が出力される

Amplify CLI を使ったバックエンドの構築

Amplify CLI
のインストール

プロジェクトに
Amplify を導入

機能(カテゴリ)
の設定

バックエンドの
構築

- 「カテゴリ」という単位でバックエンドの設定を追加する

```
$ amplify add <カテゴリ名>
```

- 例) API のバックエンドを構築する場合は

```
$ amplify add api
```

- カテゴリに応じたバックエンドの設定をコマンドラインから**対話的**に選択
- api カテゴリの設定項目の例
 - ✓ REST API or GraphQL ?
 - ✓ API のスキーマ
 - ✓ パブリックに公開する API か? 認証が必要な API か? ..等

Amplify CLI を使ったバックエンドの構築

Amplify CLI
のインストール

プロジェクトに
Amplify を導入

機能(カテゴリ)
の設定

バックエンドの
構築

- 既存カテゴリの更新、削除を行うことも可能
 - カテゴリ設定の更新

```
$ amplify update <カテゴリ名>
```

- カテゴリ設定の削除

```
$ amplify remove <カテゴリ名>
```

Amplify CLI を使ったバックエンドの構築

Amplify CLI
のインストール

プロジェクトに
Amplify を導入

機能(カテゴリ)
の設定

バックエンドの
構築

- \$amplify add/update/remove コマンドを発行しても、設定が作成されたのみでバックエンドへの反映はされていない
- 設定のステータスについては amplify status コマンドで確認する

```
$ amplify status
```

```
Current Environment: dev
```

Category	Resource name	Operation	Provider plugin
Storage	s3fea91542	Create	awscloudformation
Api	blackbelt	Update	awscloudformation
Auth	amplifybbca5f893e	No Change	awscloudformation

新規作成したが、バックエンドに
反映していない

更新したが、バックエンドに反映
していない

クラウド同期済み

Amplify CLI を使ったバックエンドの構築

Amplify CLI
のインストール

プロジェクトに
Amplify を導入

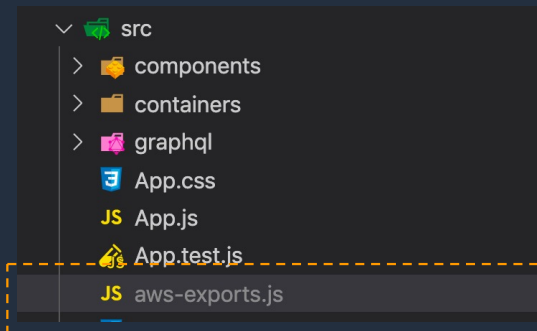
機能(カテゴリ)
の設定

バックエンドの
構築

- バックエンドの構築は `amplify push` コマンドでバックエンドを構築する

```
$ amplify push
```

- バックエンド反映後に、エンドポイントの設定ファイルが出力される
- アプリケーションはこの設定ファイルを読み込んでバックエンドに接続する



```
// WARNING: DO NOT EDIT. This file is automatically generated by  
const awsmobile = {  
  "aws_project_region": "ap-northeast-1",  
  "aws_cognito_identity_pool_id": "ap-northeast-1:4cc52855-1b5f-4481-8000-000000000000",  
  "aws_cognito_region": "ap-northeast-1",  
  "aws_user_pools_id": "ap-northeast-1_fg6nio5bs",  
  "aws_user_pools_web_client_id": "bcbtu31jjon8asfp1rfkuu2fl",  
  "oauth": {},  
  "aws_appsync_graphqlEndpoint": "https://iktyadathnd7po57znrz-awsappsync.ap-northeast-1.amazonaws.com/graphql",  
  "aws_appsync_region": "ap-northeast-1",  
  "aws_appsync_authenticationType": "AMAZON_COGNITO_USER_POOLS"  
};  
  
export default awsmobile;
```

このファイルはAmplify
CLIが設定を上書きするた
め直接編集しないこと

Amplify を使ったアプリケーション開発のワークフロー

1. Amplify CLI を用いたバックエンドの構築



2. Amplify Framework を用いたアプリケーションの実装



3. アプリケーションのデプロイ

Amplify Framework を用いたアプリケーションの実装

- **Amplify Framework** は Amplify カテゴリを使用するためのクライアントライブラリ
- プロジェクトに Amplify Framework をインストールする
 - ✓ 例) JavaScript のプロジェクトに Amplify Framework をインストール

```
$ npm install aws-amplify
```

- アプリケーションで Amplify Framework を用いるには、バックエンド構築時に出
力されたバックエンドの設定ファイルを読み込む
 - ✓ 例) JavaScript のアプリケーション

```
import Amplify from "aws-amplify"; ←----- Amplify Framework の読み込み
import awsExports from "./aws-exports"; ←----- バックエンド設定の読み込み
Amplify.configure(awsExports); ←----- Amplify Frameworkにバックエン  
ドの設定を登録
```


Amplify で構築できる機能(カテゴリ)の一覧

- **Analytics**
 - ユーザーのセッションや属性などを計測
- **API**
 - REST /GraphQL API の利用
- **Authentication**
 - 認証 API と pre-build UI component
- **Storage**
 - Static contents の シンプルな管理
- **Interactions**
 - Deep Learning を利用したBot
- **PubSub**
 - リアルタイムなデータのやりとり
- **Notification**
 - キャンペーンや分析機能をもったプッシュ通知
- **Predictions**
 - AI / ML コンテンツの組み込み
- **XR**
 - AR / VR コンテンツの組み込み

Authentication カテゴリ

- **Amazon Cognito** と統合されたカテゴリ
- アプリケーションに認証・認可、フェデレーション機能を簡単に実装が可能
- 他のカテゴリと組み合わせることで、認証ユーザのみ API を呼び出すといった制御が可能

```
$ amplify add auth
```

- **Amplify Framework** のソースコード例

- ✓ 外部IDプロバイダーを用いたフェデレーションの実装例

```
Auth.federatedSignIn({ provider: 'facebook' })}
```

- ✓ 認証済みユーザの取得

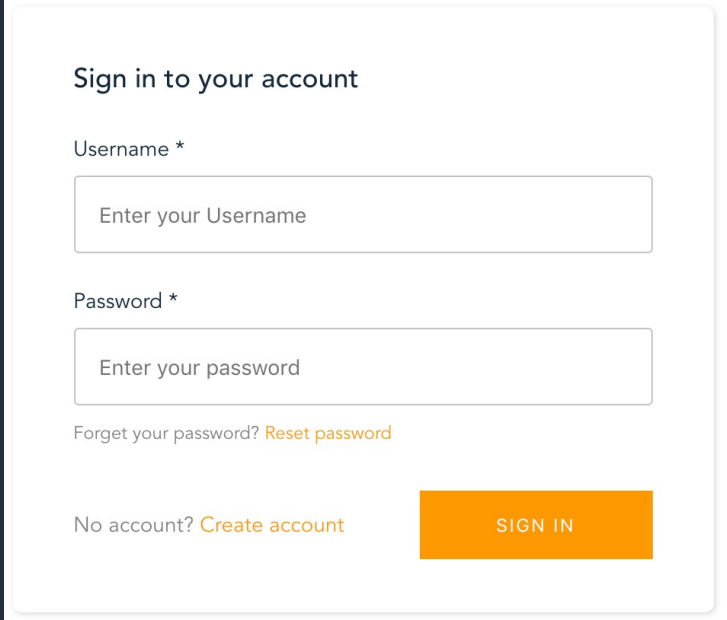
```
let cognitoUser = await Auth.currentAuthenticatedUser();
```

Authentication カテゴリ

- 認証用の UI コンポーネントが提供されており、UI タグを配置するだけで、サインイン、サインアップ、パスワード復旧機能が実装されたコンポーネントを実装可能

例) Vue.js の認証コンポーネント

```
<amplify-authenticator />
```



Sign in to your account

Username *

Password *

Forget your password? [Reset password](#)

No account? [Create account](#)

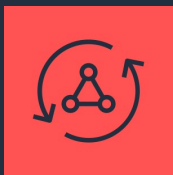
API カテゴリ (GraphQL / REST API)

- API カテゴリには GraphQL と REST の 2 種類のタイプが存在
- GraphQL を選択した場合、**AWS AppSync** と統合された API を構築
- REST を選択した場合、**Amazon API Gateway**、**AWS Lambda** と統合された API を構築

```
$ amplify add api
```

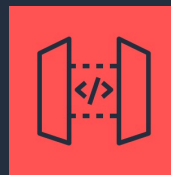
```
? Please select from one of the below mentioned services:  
> GraphQL  
  REST
```

amplify add コマンド発行後に
どちらかを選択



AWS AppSync

- GraphQL のマネージドサービス
- リアルタイム通信を必要とするアプリケーションが構築可能
- AWS が提供する様々なデータソースを指定可能



**Amazon
API Gateway**

- REST、HTTP、WebSocket API の作成、公開、運用を行うためのマネージドサービス
- アプリケーションAPIのIFなどの用途として利用される

Prediction カテゴリ

- Amazon が提供する 各AI/ML サービスと統合されたカテゴリ
- テキスト翻訳、文字読み上げ、Object Detection、文章のネガポジ判定といった機能が簡単に実装できる

\$ amplify add **predictions**

大分類	小分類	対象サービス
画像	画像認識	Amazon Rekognition
音声	Speech-to-Text	Amazon Transcribe
	Text-to-Speech	Amazon Polly
自然言語処理	自然言語解析	Amazon Comprehend
	テキスト翻訳	Amazon Translate
	チャットボット	Amazon Lex

Prediction カテゴリ

Predictions カテゴリ実装の例

例) 画像認識

```
Predictions.identify({
  labels: {
    source: {
      file: files[0]
    },
    type: "ALL"
  }
}).then(
  result => this.labels = result.labels
).catch(
  error => this.error = JSON.stringify(error)
)
```

ファイルを選択 IMG_0334.JPG



約98%の精度で猫(cat)と判定している

Name	Confidence[%]
Cat	98.54254913330078
Mammal	98.54254913330078
Pet	98.54254913330078
Animal	98.54254913330078
Black Cat	94.8594970703125
Manx	55.98295593261719

Amplify を使ったアプリケーション開発のワークフロー

1. Amplify CLI を用いたバックエンドの構築



2. Amplify Framework を用いたアプリケーションの実装



3. アプリケーションのデプロイ

アプリケーションのデプロイ

- デプロイ先となる hosting カテゴリを追加

```
$ amplify add hosting
```

- 2種類のデプロイ方法から一つを選択

```
? Select the plugin module to execute
```

```
> Hosting with Amplify Console (Managed hosting with custom domains, Continuous deployment)  
Amazon CloudFront and S3
```

Amazon CloudFront and S3

- ✓ コマンドラインからデプロイする
- ✓ CI/CD環境、GitHubなどのソースリポジトリとの連携が不要な場合はこちらを選択する

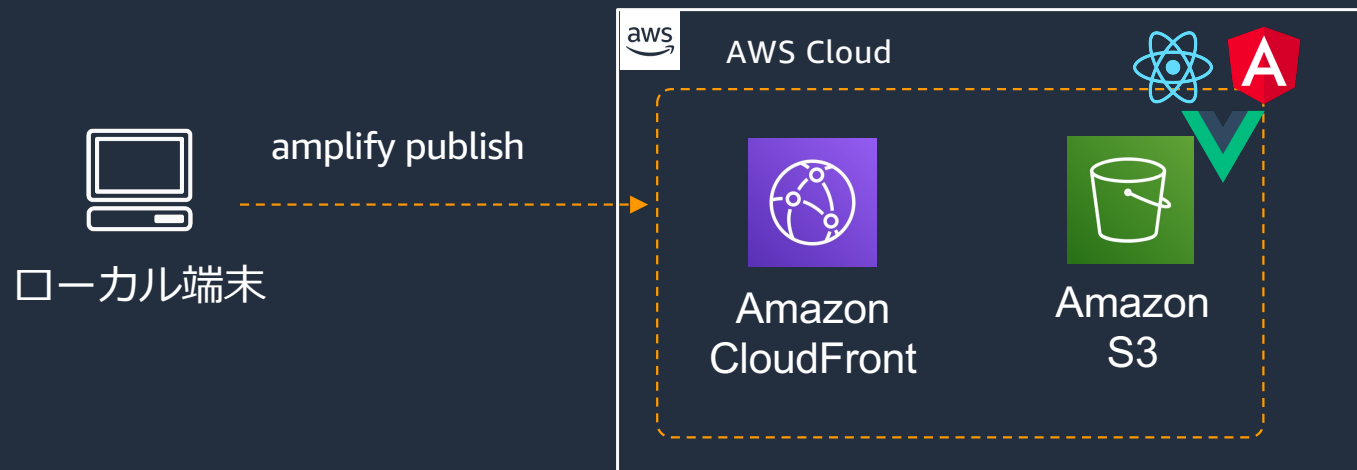
Hosting with Amplify Console

- ✓ **Amplify Console** を用いたデプロイ
- ✓ GitHub などソースリポジトリと連携
- ✓ CI/CD環境を簡単に構築
- ✓ マネジメントコンソールから操作可能

アプリケーションのデプロイ (Amazon CloudFront and S3)

- Amplify CLI のコマンドラインからデプロイを行う
- CI/CD環境や GitHub などのソースリポジトリとの連携が不要なシンプルなデプロイにはこちらを用いる
- ホスティングは S3、もしくは **Amazon CloudFront** + S3 から選択可能

```
$ amplify publish
```



アプリケーションのデプロイ (Amplify Console)

Amplify Console とは何か？

- Git ベースの SPA (Single Page Application) や静的サイトの開発ワークロードを提供するフルスタックなホスティングサービス
- マネジメントコンソールから以下のワークロードの管理が可能
 - ✓ CI/CD パイプラインの構築
 - ✓ SPAや静的ファイルのホスティング
 - ✓ 各 branch とバックエンドの紐付け
 - ✓ CDN、HTTPS のカスタムドメインの構築
 - ✓ Basic認証によるテスト環境の保護
 - ✓ E2E テスト環境を容易に構築 …etc

■ Amplify Console

<https://aws.amazon.com/jp/amplify/console/>

© 2020, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

The screenshot displays the AWS Amplify Console for an application named 'boyaki'. The interface includes a navigation bar with 'aws', 'サービス', 'リソースグループ', and 'サポート'. Below the navigation, there's a section for 'boyaki' with an 'アクション' dropdown. A message box says 'Learn how to get the most out of Amplify Console' with '3 of 5 steps complete'. The main content area is divided into 'Frontend environments' and 'Backend environments'. Under 'Frontend environments', there's a section for 'design-base' with a 'Continuous deploys set up with design backend (Edit)'. A pipeline diagram shows five steps: 'プロビジョン' (Provision), 'ビルド' (Build), 'テスト' (Test), 'デプロイ' (Deploy), and '検証' (Verify). The 'テスト' step is currently active. Below the pipeline, there's a table with deployment details: '前回のデプロイ' (Previous deployment) on 2020/4/12 1:20:21, '最終コミット' (Latest commit) with a message 'これは自動生成されたメッセージです | Auto-build | GitHub - design-base', and 'Previews' which are 'Disabled'. The URL 'https://design-base...amplifyapp.com' is also visible.



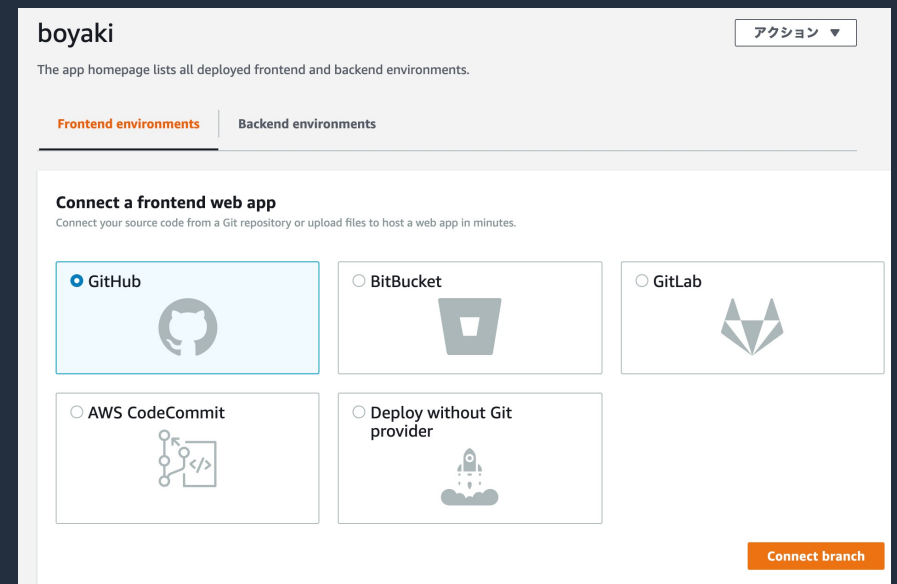
アプリケーションのデプロイ (Amplify Console)

ソースリポジトリとの連携

- Github や **AWS CodeCommit** といったソースリポジトリと連携
- 2020/5月現在で以下のソースリポジトリと連携が可能

- ✓ Github
- ✓ BitBucket
- ✓ GitLab
- ✓ AWS CodeCommit

※ ZIP ファイルを手動アップロードすることでデプロイを実施することも可能



■ Amplify Console Getting Start with Existing Code

<https://docs.aws.amazon.com/amplify/latest/userguide/getting-started.html>

アプリケーションのデプロイ (Amplify Console)

Branch ごとのホスティング

- ソースリポジトリの Branch ごとにホスティングを行うことができる



アプリケーションのデプロイ (Amplify Console)

E2E テスト (Cypress)

- Cypress を用いたE2EテストをCI/CDパイプラインに組み込むことが可能
- amplify.yml にCypressの設定を追加
- テスト結果のArtifactをAmplify Consoleからダウンロード可能

The screenshot shows the AWS Amplify Console interface for a staging environment. At the top, there are buttons for '最新ビルドの表示' and 'ビルド履歴の表示'. Below this, a progress bar shows five steps: 'プロビジョン', 'ビルド', 'テスト', 'デプロイ', and '検証', all marked with green checkmarks. The 'テスト' step is highlighted with a red box. Below the progress bar, there is a 'このバージョンを再デプロイ' button. The main content area displays details for 'ビルド 3', including the domain 'https://staging.d2hgjd3sxyadky.amplifyapp.com', the start time '2020/4/12 1:49:00', and the build duration '6 minutes 52 seconds'. Below this, there are tabs for 'Provision', 'Build', 'Test', 'Deploy', 'Verify', 'ビルド', 'デプロイ', and '検証'. The 'Test' tab is active, showing the message 'All Cypress specs completed! 1 spec(s) passed' and a 'Download artifacts' button, which is also highlighted with a red box. An orange arrow points from the 'テスト' step in the progress bar to the 'Download artifacts' button. At the bottom, there is a table with columns for 'Spec name', 'Number of tests', 'Total duration', and 'Video'. The table contains one entry: 'Authenticator: 1 passed 00:04 Download artifacts to see this video.'

The screenshot shows a video player displaying a Cypress test video. The video title is 'authenticator_spec.js.mp4'. The video content shows a user signing in on a web application. The video player has a progress bar at the bottom showing '00:05' and a play/pause button. On the left side of the video player, there is a code editor showing the Cypress test script. The code includes a 'Sign in' test suite with a 'beforeEach' hook and several test cases for GET and POST requests. The test cases are: 'allows a user to sign in', 'username input', 'password input', 'sign in button', and 'global timeline'.

E2Eテストの様子が動画で確認できる

アプリケーションのデプロイ (Amplify Console)

Amplify Console その他機能

- テスト環境にBasic認証を設定
- CI/CD のデプロイ結果をEmailで通知
- Webhook でCI/CDパイプラインを起動
- amplify.yml を編集してビルド設定のカスタマイズ

例) master ブランチでは Unit テストを実施しない場合の設定

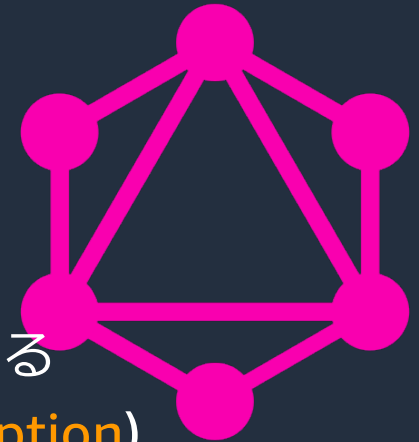
```
frontend:
  phases:
    preBuild:
      commands:
        - yarn install
    build:
      commands:
        - if [ "${AWS_BRANCH}" != "master" ]; then yarn test; fi
        - yarn run build
```

本日のアジェンダ

1. Amplify の概要説明
2. Amplify を使ったアプリケーション開発
- 3. Amplify + GraphQL Deep Dive**
4. 直近のアップデート
5. お客様からよくあるご質問
6. まとめ

GraphQL とは何なのか？

- GraphQL とは？
 - ✓ GraphQL API 用 OSS の **Query 言語**
 - ✓ REST API の課題を解決するAPI仕様として近年注目を集める
 - ✓ 処理形態は、取得(**Query**)、変更(**Mutation**)、購読(**Subscription**)
- GraphQL の特徴
 - ✓ 型指定されたスキーマ
 - ✓ クライアントからレスポンスの形式を指定
 - ✓ サブスクリプションを利用したリアルタイム処理



■ [AWS Black Belt Online Seminar] AWS AppSync

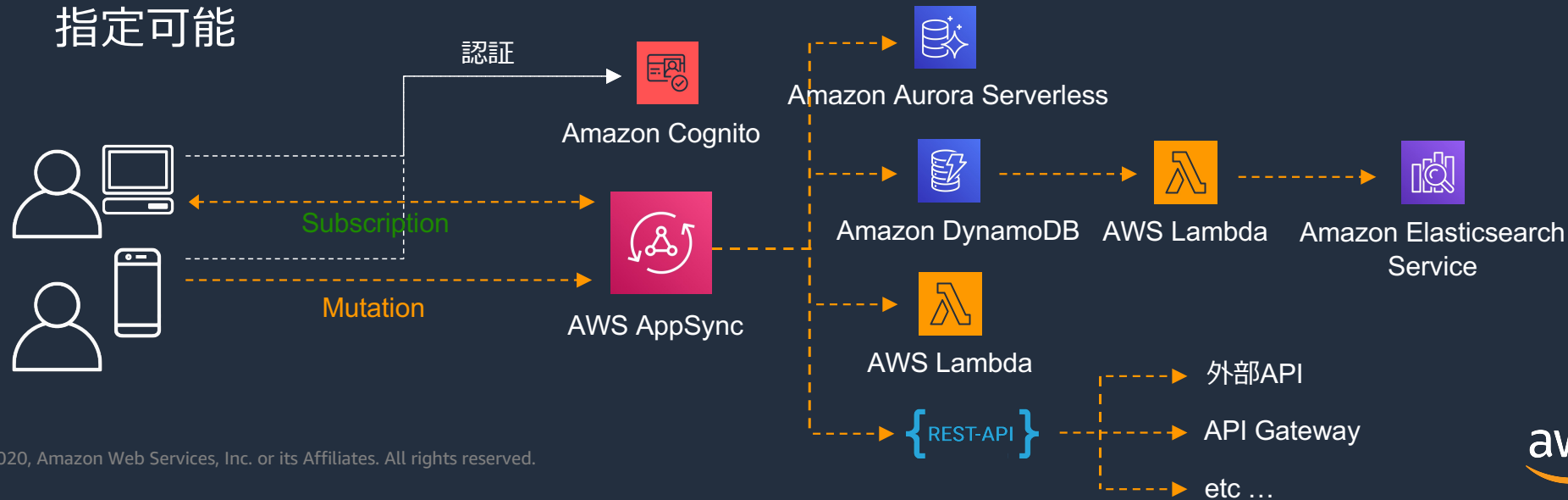
<https://aws.amazon.com/jp/blogs/news/webinar-bb-aws-appsync-2019/>

© 2020, Amazon Web Services, Inc. or its Affiliates. All rights reserved.



API カテゴリ (GraphQL)

- API カテゴリ(GraphQL) は **AWS AppSync** と統合された API を構築
- スキーマ定義を記述するだけで、GraphQL のバックエンドを構築
- **Authentication** カテゴリと併用することで、認証済みユーザのみにアクセスさせるといったきめ細やかな権限制御が可能
- データソースには **Amazon DynamoDB**、**Amazon Elasticsearch Service**、**Aurora Serverless**、**AWS Lambda**、REST API といった様々なデータソースが指定可能

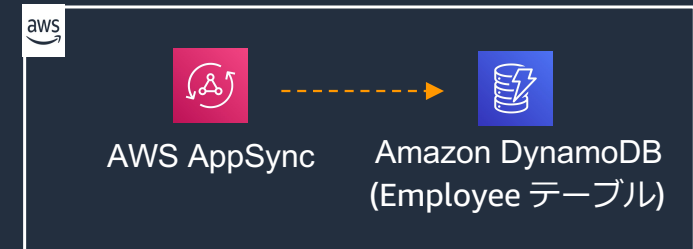



API カテゴリ (GraphQL) スキーマの作成

- スキーマ定義は `amplify/backend/api/<API 名>/schema.graphql` に記述する

```
type Employee @model {  
  id: ID!  
  name: String!  
  department: String!  
  email: String!  
  jobType: String!  
  hireDate: AWSDateTime!  
}
```

`$ amplify push`



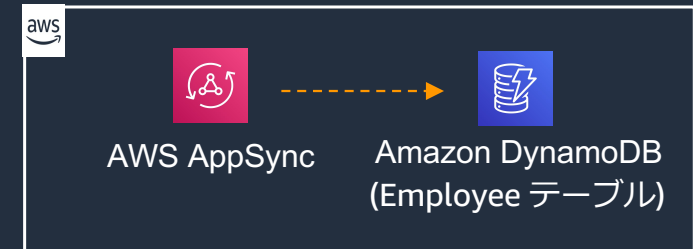

- ディレクティブを用いることで、様々な機能をスキーマに付与できる
 - @model
 - @key
 - @auth
 - @connection
 - @searchable
 - @function
 - @http
 - @versioned
 - @predictions

API カテゴリ (GraphQL) スキーマの作成

- スキーマ定義は `amplify/backend/api/<API 名>/schema.graphql` に記述する

```
type Employee @model {  
  id: ID!  
  name: String!  
  department: String!  
  email: String!  
  jobType: String!  
  hireDate: AWSDateTime!  
}
```

`$ amplify push`



- ディレクティブを用いることで、様々な機能をスキーマに付与できる

- **@model**
- **@key**
- **@auth**
- **@connection**
- **@searchable**
- **@function**
- **@http**
- **@versioned**
- **@predictions**

API カテゴリ (GraphQL) 今から説明するディレクティブの概要

- **@model** すべてのディレクティブの基礎となるディレクトリで、データソースに DynamoDB を指定する
- **@key** データソースである DynamoDB に Index を付与できる
- **@auth** データに対し、きめ細やかなアクセス制御を行う
- **@connection** DynamoDB のデータストアに対しリレーションを定義する
- **@searchable Amazon Elasticsearch Service** を用いた全文検索を可能とする

■ Amplify Docs API (GraphQL) Directives

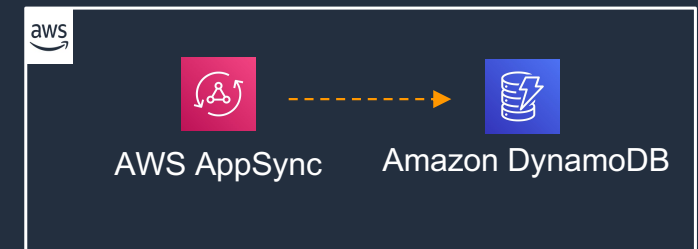
<https://docs.amplify.aws/cli/graphql-transformer/directives>

API カテゴリ (GraphQL) @model (1/5)

- **@model** ディレクティブ 生成されるAPIのトップレベルのディレクトリ
- **@model** ディレクティブをアノテーションされたオブジェクトは **DynamoDB** に格納され、**@key**、**@auth**、**@connection** といった他のオブジェクトとの関連を定義することができる

```
type Employee @model {  
  id: ID!  
  name: String!  
  department: String!  
  email: String!  
  jobType: String!  
  hireDate: AWSDateTime!  
}
```

\$ amplify push



API カテゴリ (GraphQL) @key (2/5)

- @key ディレクティブは DynamoDB の Global Secondary Index を付与する

```
type Employee @model @key(fields["email"], queryField: "employeeByEmail") {  
  id: ID!  
  name: String!  
  department: String!  
  email: String!  
  jobType: String!  
  hireDate: AWSDateTime!  
}
```

email に対し GSI を貼る

emailで検索を行うためのクエリ名

- 複合ソートキー を付与することも可能

```
type Employee @model @key(fields["jobType", "hireDate"], queryField: "employeeByJobType")  
  @key(fields["email"]) {  
  id: ID!  
  name: String!  
  department: String!  
  email: String!  
  jobType: String!  
  hireDate: AWSDateTime!  
}
```

jobTypeとhireDateで複合ソートキーを作成

API カテゴリ (GraphQL) @key (2/5)

- @key ディレクティブは DynamoDB の Global Secondary Index を付与する

```
type Employee @model @key(fields["department", "jobType", "hireDate"]){  
  id: ID!  
  name: String!  
  department: String!  
  email: String!  
  jobType: String!  
  hireDate: AWSDateTime!  
}
```

The diagram shows the fields "department", "jobType", and "hireDate" from the @key directive. A red underline is drawn under "department" and "jobType", with a line pointing to a box labeled "Partition key". Another red underline is drawn under "hireDate", with a line pointing to a box labeled "Sort key".

- 一つのモデルに複数の@key ディレクティブを付与することも可能

```
type Employee @model @key(fields["department", "jobType", "hireDate"]) @key(fields["email"]) {  
  id: ID!  
  name: String!  
  department: String!  
  email: String!  
  jobType: String!  
  hireDate: AWSDateTime!  
}
```

The diagram shows the fields "department", "jobType", and "hireDate" from the first @key directive and the field "email" from the second @key directive. A red underline is drawn under "department", "jobType", and "hireDate", with a line pointing to a box labeled "Partition key". Another red underline is drawn under "email", with a line pointing to a box labeled "一意のデータの場合はSort Keyの指定は不要".

API カテゴリ (GraphQL) @auth (3/5)

- @auth ディレクティブはデータに対し、きめ細やかなアクセス制御を行う
- rules の定義により、誰が、どのデータに対し、何の操作を行えるかを定義

```
type Employee @model @auth(rules[
  {allow: owner , ownerField: "owner" , operation["create", "update", "delete", "read"]}){
  id: ID!
  name: String!
  owner: String!
  group: String!
}
```

データを作成したユーザのみが、
「更新」「削除」の操作が可能

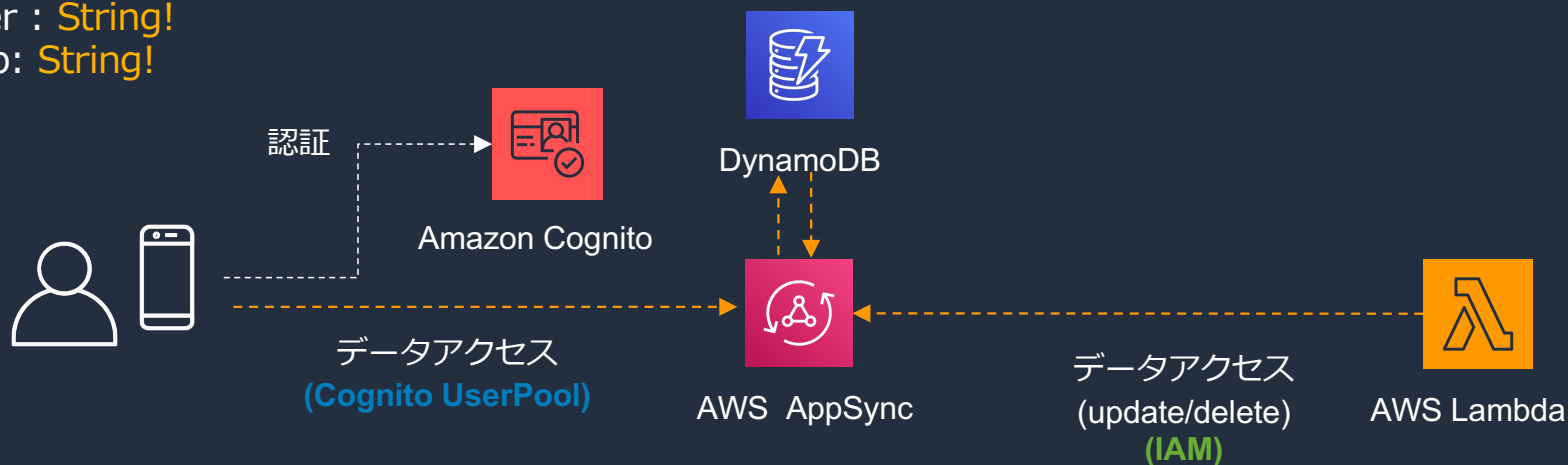
```
type Employee @model @auth(rules[
  {allow: group , groups: ["Admin", "HR"] , operation["create", "update", "delete", "read"]}){
  id: ID!
  name: String!
  owner : String!
  group: String!
}
```

Admin、HRグループに所属するユーザのみが「作成」「更新」「削除」「読み込み」の操作が可能

API カテゴリ (GraphQL) @auth (3/5)

- **@auth** ディレクティブは特定の認証プロバイダでデータアクセスのアクセス設定を上書くことが可能
- 以下の例の場合、Cognito UserPool で認証されたユーザに加え、適切なポリシーを持ったIAMでも、データの更新、削除が可能

```
type Employee @model @auth(rules[{{allow: private , provider: iam, operations, [update, delete]}]}){  
  id: ID!  
  name: String!  
  owner : String!  
  group: String!  
}
```



API カテゴリ (GraphQL) @connection (4/5)

- @connection ディレクティブは テーブル同士のリレーションを定義
- NoSQL のデータベースである DynamoDB でも、リレーション構造を簡単に表現することが可能

1 : 1 のリレーションの例

```
type User @model {  
  id: ID!  
  name: String!  
  empliyeeId: String!  
  employee: Employee @connection(fields: ["empliyeeId"])  
}
```

```
type Employee @model {  
  id: ID!  
  role: String!  
  email: String!  
  hireDate: AWSDateTime!  
}
```

API カテゴリ (GraphQL) @connection (4/5)

- @connection ディレクティブは テーブル同士のリレーションを定義
- NoSQL のデータベースである DynamoDB でも、リレーション構造を簡単に表現することが可能

1 : 多のリレーション の例

```
type Department @model {  
  id: ID!  
  name: String!  
  employees: [Employee] @connection(keyName: "byDepartment", fields: [id])  
}
```

```
type Employee @model {  
  @key(name: "byDepartment", fields: ["departmentId", "name", "email"])  
  id: ID!  
  name: String!  
  departmentId: String!  
  email: String!  
}
```

API カテゴリ (GraphQL) @connection (4/5)

- @connection ディレクティブはテーブル同士のリレーションを定義
- NoSQL のデータベースである DynamoDB でも、リレーション構造を簡単に表現することが可能

多：多のリレーションの例

```
type Project @model {  
  id: ID!  
  name: String!  
  members: [ProjectMember] @connection(keyName: "byProject", fields: ["id"])  
}
```

Project テーブル

```
type ProjectMember @model @key(name: "byProject", fields: ["projectId", "memberId"])  
  @key(name: "byMember", fields: ["memberId", "projectId"]) {  
  id: ID!  
  projectId: String!  
  memberId: String!  
  members: Employee! @connection(fields: ["employeeId"])  
  projects: Project! @connection(fields: ["projectId"])  
}
```

紐付けテーブル

```
type Employee @model {  
  id: ID!  
  name: String!  
  projects: [Project] @connection(keyName: "byMember", fields: ["id"])  
}
```

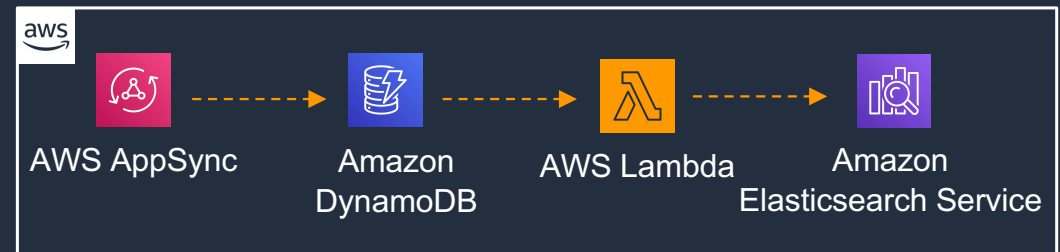
Employee テーブル

API カテゴリ (GraphQL) @searchable (5/5)

- @searchable は Amazon Elasticsearch Service を用いた全文検索が可能
- 登録されたデータは DynamoDB に登録された後、DynamoDB ストリームが起動した AWS Lambda により Amazon Elasticsearch Service に同期される

```
type Book @model @searchable {  
  id: ID!  
  title: String  
  description: String  
  price: Int  
}
```

\$ amplify push



クエリの例)

「Amplify」という文字列を含む、2020年以降に発売された書籍を価格順で表示

API カテゴリ (GraphQL) スキーマの作成 その他

- **@function** は **AWS Lambda** を起動することが可能
- **@http** は **AppSync** のバックエンドとして REST API を呼び出すことが可能
- **@versioned** はオブジェクトのバージョン管理情報を付与し、データのコンフリクトの解決を型に追加する
- **@predictions** は Predictions の機能を GraphQL のインターフェースから利用可能

■ Amplify Docs API (GraphQL) Directives

<https://docs.amplify.aws/cli/graphql-transformer/directives>

本日のアジェンダ

1. Amplify の概要説明
2. Amplify を使ったアプリケーション開発
3. Amplify + GraphQL Deep Dive
- 4. 直近のアップデート**
5. お客様からよくあるご質問
6. まとめ

Amplify DataStore

New!

- AWS re:Invent 2019 で発表されたマルチプラットフォーム (iOS/Android/Web) な **デバイス側ストレージエンジン**
- アプリ実装者が端末の**オンライン/オフラインを意識せず**に実装が行える
- オンライン復旧後のデータ更新時における競合検知 & 自動マージ
- Amplify DataStore では GraphQL のクエリを記述することなく、各開発言語に最適化されたインターフェースで **AWS AppSync** にアクセスが可能



データの競合検知、
自動解決



馴染みのあるローカルファーストプログラミングモデル



デルタ同期と自動マージ



AWS AppSync、GraphQL との連携

3種類の競合検知のオプション

- Auto Merge (default)
 - データの競合を検知すると自動的にマージを試みる
- Optimistic Concurrency
 - データの競合を検知した場合、マージを実施せずクライアントからのリクエストを拒否する
- Custom Lambda
 - データの競合を検知した際に独自に定義した **AWS Lambda** を起動する

```
amplify update api #Select GraphQL
? Do you want to configure advanced settings for the GraphQL API
> Yes, I want to make some additional changes.

? Configure conflict detection? Yes
? Select the default resolution strategy
  Auto Merge
  > Optimistic Concurrency
  Custom Lambda
  Learn More
```

競合オプションは、Amplify CLI で GraphQL API を構築する際に指定 (構築後、変更も可)

Amplify DataStore API

- DataStore API を使った AppSync へのデータアクセスの例

- 更新

```
DataStore.save(new Post("Hello Amplify!", "ACTIVE", 4));
```

- 削除

```
DataStore.delete(Post, c => c.status("eq", "INACTIVE"));
```

- 参照

```
DataStore.query(Post, c => c.rating("gt", 3));
```

- 購読

```
DataStore.observe(Post).subscribe();
```

Amplify iOS / Amplify Android

New! (Preview)

- re:Invent 2019 で発表された Amplify の iOS / Android 向けの SDK
- 従来の **AWS Mobile SDK** から利用するときと比べ、より各カテゴリのユースケースに沿った宣言的インターフェースを提供
- 2020/05/20 時点で Preview 版のため、商用環境では Mobile SDK を推奨

Amplify iOS / Android

- ✓ ユースケース中心
- ✓ 宣言的インターフェース
- ✓ Xcode や Android Studio などの IDE と統合
- ✓ Amplify CLI との統合

AWS Mobile SDK

- ✓ AWS サービス中心
- ✓ 低レベル API
- ✓ 個別の CLI 操作とライブラリの導入

※2020/05/20時点ではこちらを推奨

本日のアジェンダ

1. Amplify の概要説明
2. Amplify を使ったアプリケーション開発
3. Amplify API Deep Dive
4. 直近の最新アップデート
- 5. お客様からよくあるご質問**
6. まとめ

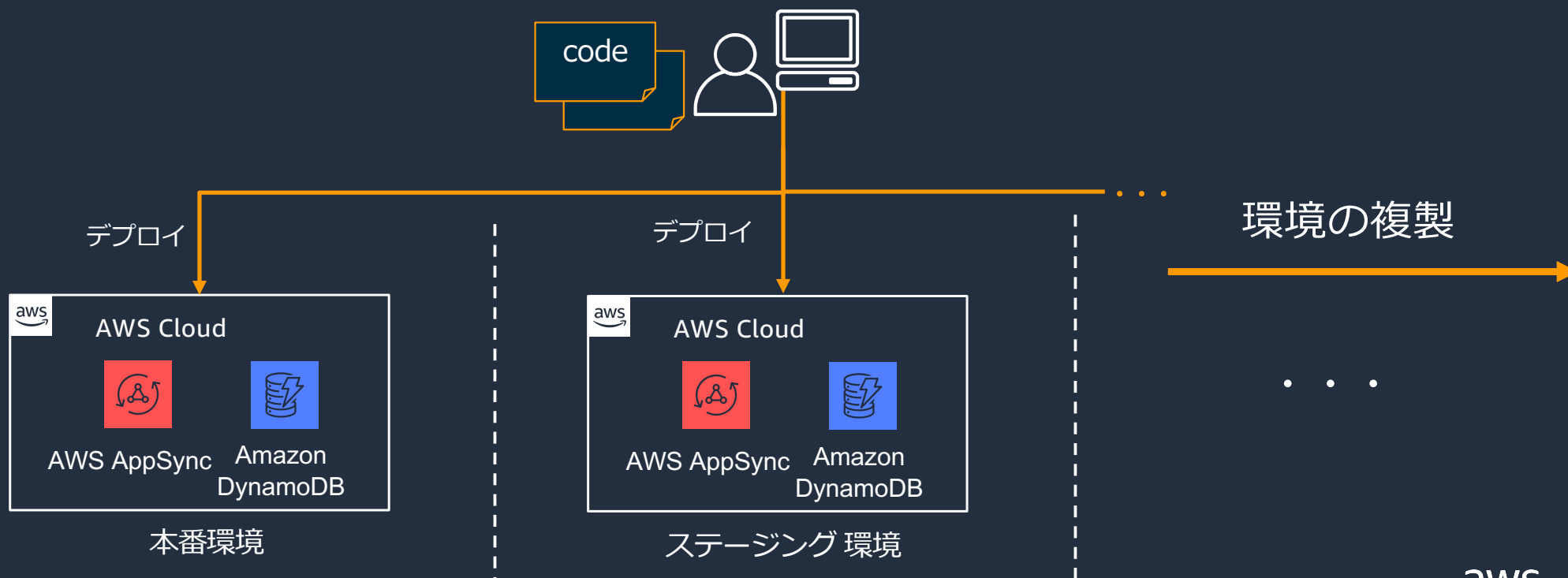
Big Data

よくある質問 1.

開発環境、テスト環境といった複数の環境で開発を行うには どのようなワークフローを設計すればよいでしょうか？



- 本番環境以外にステージング環境や開発環境といった目的に応じた環境を構築することが必要になるケースがあります
- Amplify CLI でバックエンドの環境を複数構築するにはどうすればよいか？
また、デプロイのワークフローはどのように設計すればよいか？



回答.

Amplify CLI の Multi Environment 機能 と Amplify Console を使って複数環境の構築とデプロイフローを整備します。

- amplify env コマンドで環境を複製することが可能

```
$ amplify env add <環境名>  
$ amplify push
```

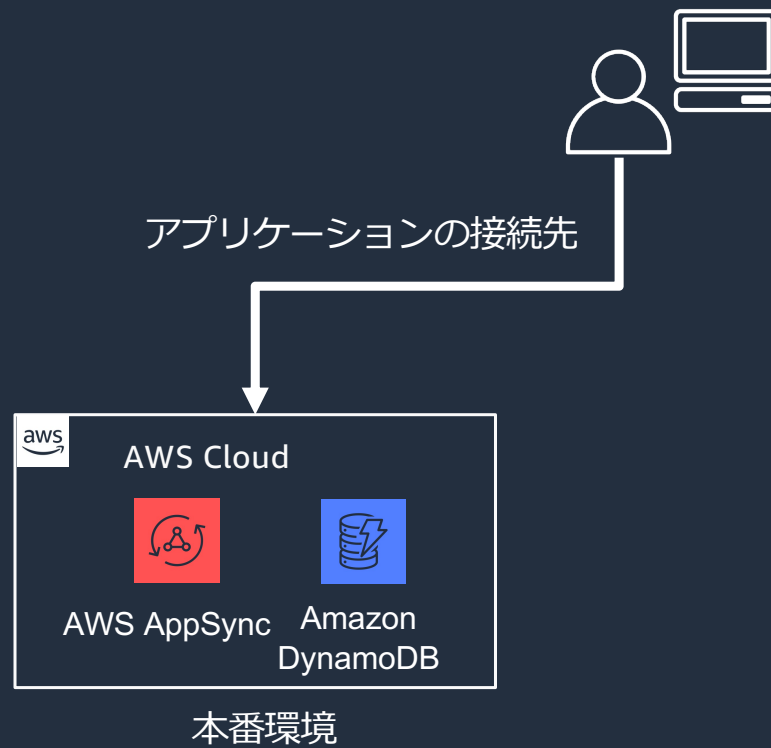
- ✓ amplify push コマンドで複製したバックエンドの設定を反映

- 環境を切り替えも容易に行うことが可能

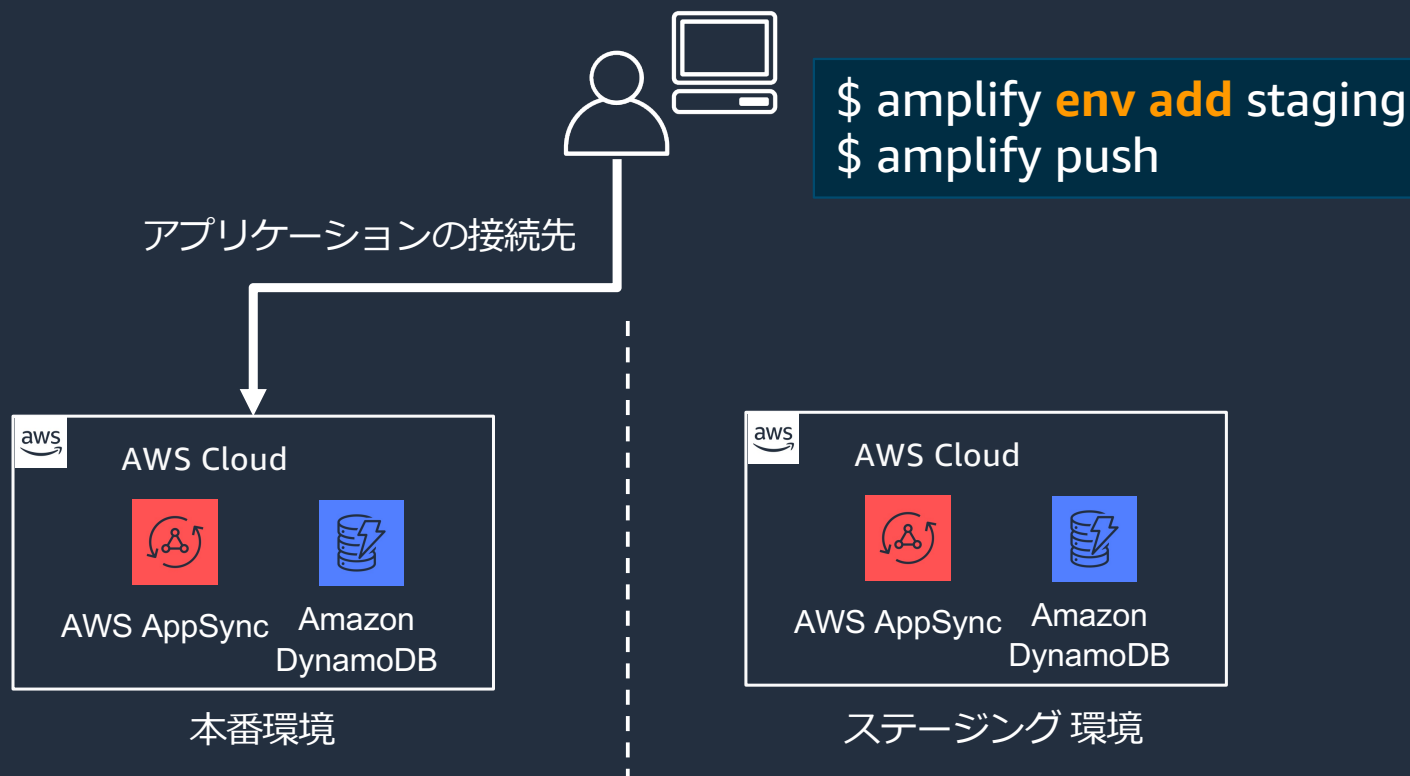
```
$ amplify env checkout <環境名>
```

- ✓ 環境を切り替えるとフロントエンドのバックエンドの設定が指定された環境のものに切り替わる

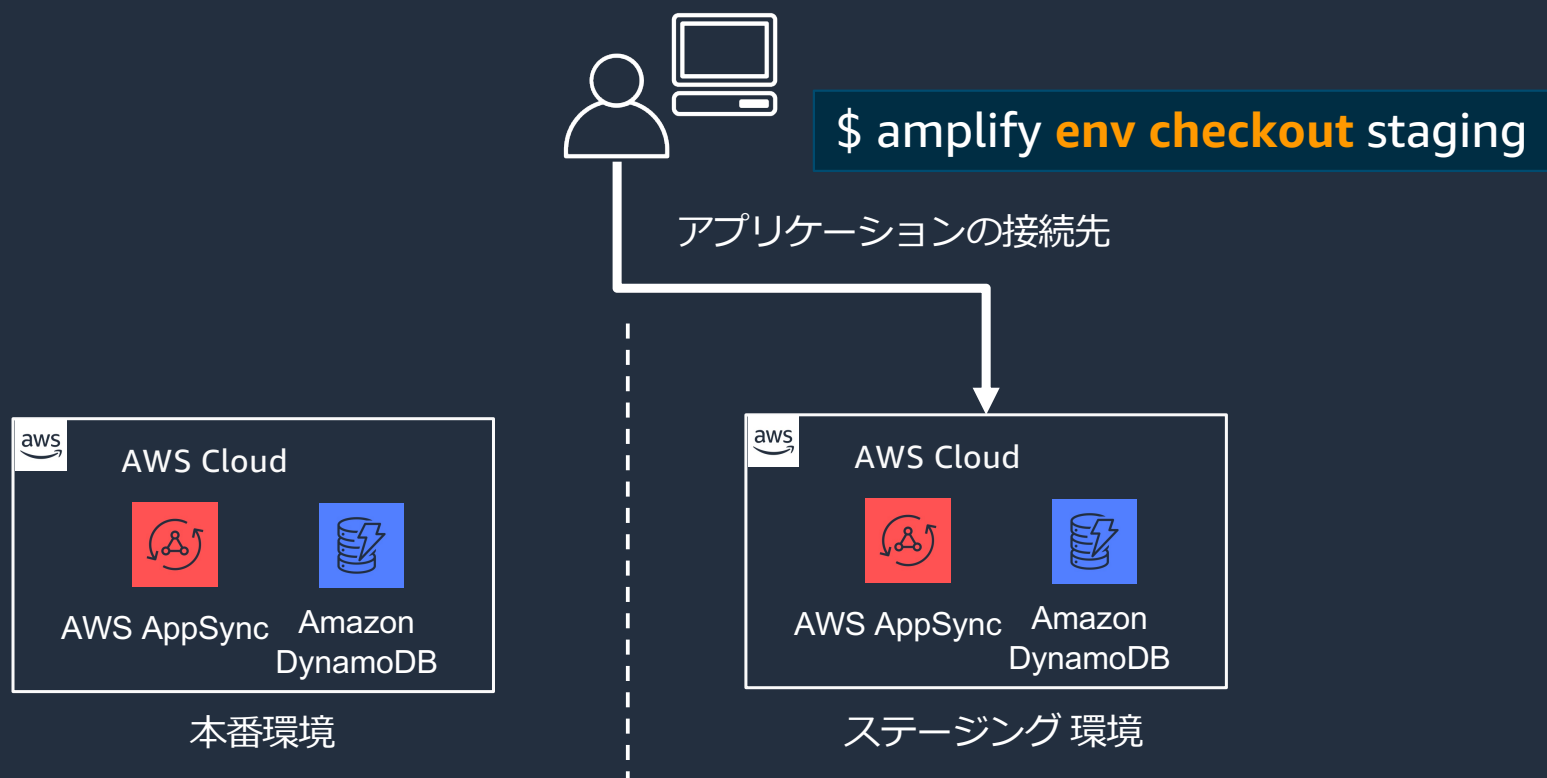
Multiple Environment による環境構築と接続先の切り替え



Multiple Environment による環境構築と接続先の切り替え



Multiple Environment による環境構築と接続先の切り替え



Amplify Console で複数環境のデプロイ

- Amplify Console の画面で、Git の Branch と Multi Environment の機能で追加したバックエンドの環境を紐づけることが可能

すべてのアプリ > amplifyweek2020apr

amplifyweek2020apr アクション

The app homepage lists all deployed frontend and backend environments.

Learn how to get the most out of Amplify Console 1 of 5 steps complete

Frontend environments | Backend environments

This tab lists all connected branches, select a branch to view build details.

master
Continuous deploys set up with production backend (Edit)

プロビジョン | ビルド | デプロイ | 検証

前回のデプロイ
2020/3/24 20:50:56

最終コミット
add staging env | 63443a1 | GitHub - master

Previews
Disabled

<https://master...amplifyapp.com>

ブランチの接続

リポジトリブランチの追加

GitHub

リポジトリサービスプロバイダー

GitHub

ブランチ
選択したリポジトリからブランチを選択します。

develop

Backend environment
Select a backend environment for this branch.

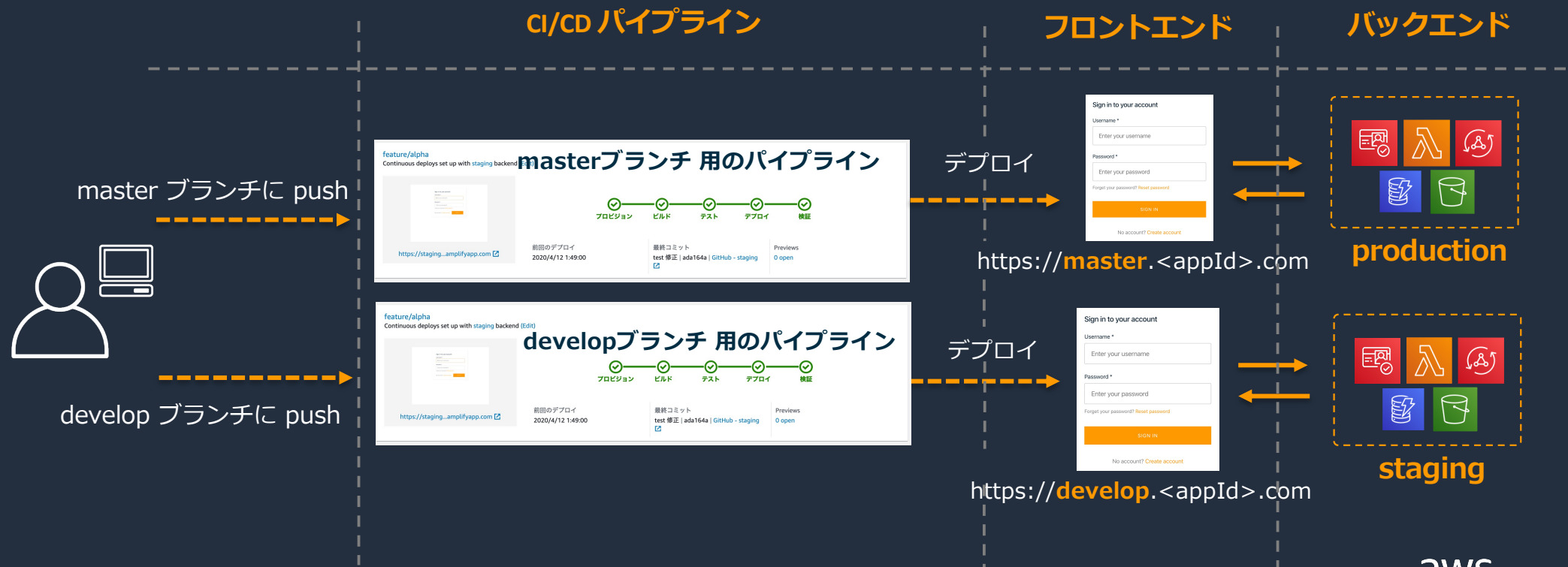
staging

キャンセル 次へ

Git の Branch と Multi Environment の環境を紐づける

Amplify Console で複数環境のデプロイ

- Branch とバックエンドを紐づけることで、環境ごとの CI/CD パイプラインの構築が可能



Big Data

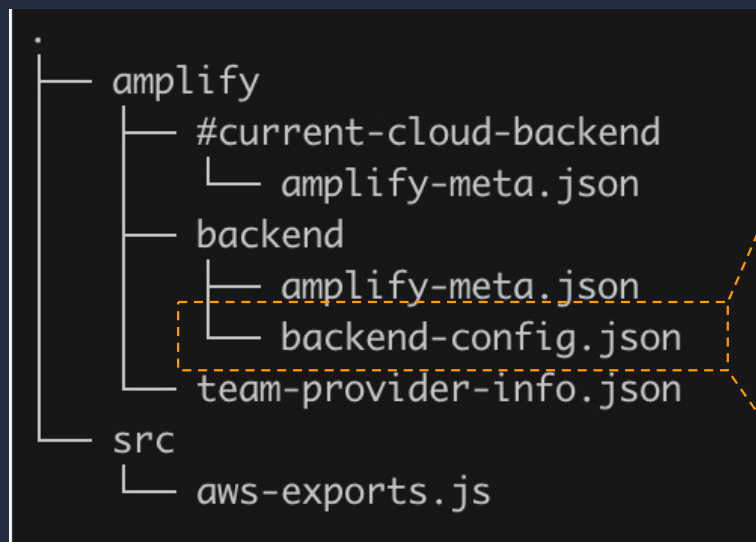
よくある質問 2.

Amplify CLI に対応していないバックエンドを構築するにはどうすれば良いでしょうか？



回答. 任意の CloudFormation テンプレートをカスタムカテゴリとして 定義できます。

- プロジェクトのトップディレクトリにある amplify フォルダ配下にカテゴリを管理する「backend-config.json」というファイルを修正する



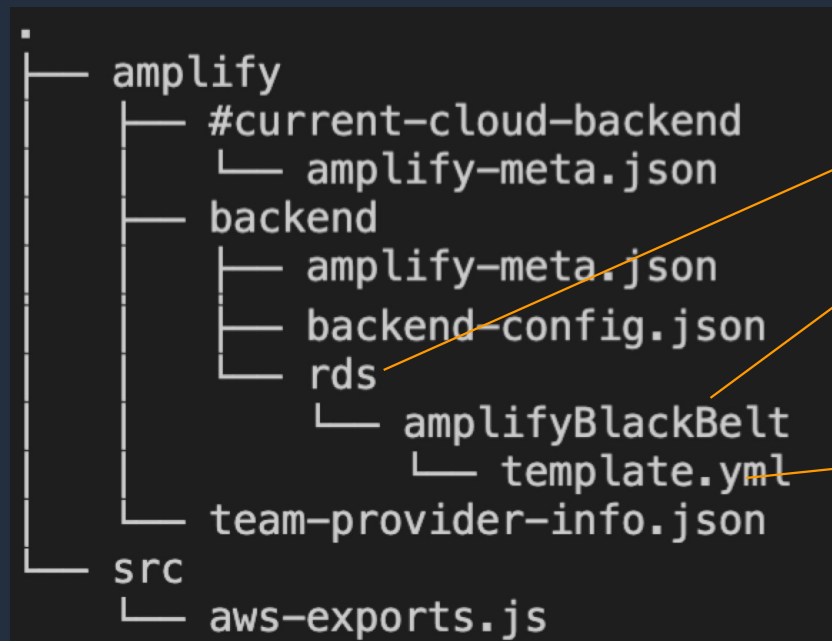
```
{
  "rds": {
    "amplifyBlackBelt": {
      "service": "RDS",
      "providerPlugin": "awscloudformation"
    }
  }
}
```

カテゴリ名

リソース名

追加するAWSのサービス名

- amplify/backend/<カテゴリ名>/<リソース名>/template.yml のディレクトリ構成でリソースのCloudformationテンプレートファイルを配置



カテゴリ名

リソース名

RDSを作成する
Cloud formation
テンプレートファイル

- amplify env コマンドで環境名を指定

```
$ amplify env checkout <現在の環境名>
```

- amplify status コマンドでカスタムカテゴリが追加されていることを確認

```
$ amplify status
```

```
Current Environment: local
```

Category	Resource name	Operation	Provider plugin
Rds	amplifyBlackBelt	Create	awscloudformation

今回作成したカテゴリが追加されている

- amplify push コマンドで作成したリソースを反映

```
$ amplify push
```

- リソースが作成されていることを確認

The screenshot shows the AWS Management Console interface for RDS databases. At the top, there's a breadcrumb 'RDS > データベース'. Below that, the 'データベース' (Databases) section is active, with a toggle for 'グループリソース' (Group Resources) and buttons for '変更' (Change), 'アクション' (Actions), 'S3 から復元' (Restore from S3), and 'データベースの作成' (Create Database). A search bar contains 'データベースのフィルタリング'. Below the search bar is a table with columns: DB 識別子, ロール, エンジン, リージョンと AZ, サイズ, ステータス, and CPU. The table contains one entry: 'amplify-blackbelt-rds' (Instance), 'MySQL Community', 'ap-northeast-1a', 'db.m4.large', '利用可能' (Available), and '0.85%'. An orange box highlights the 'amplify-blackbelt-rds' entry, and an arrow points from this box to a callout box below.

DB 識別子	ロール	エンジン	リージョンと AZ	サイズ	ステータス	CPU
amplify-blackbelt-rds	インスタンス	MySQL Community	ap-northeast-1a	db.m4.large	利用可能	0.85%

amplify-blackbelt-rds のリソースが作成されている

本日のアジェンダ

1. Amplify の概要説明
2. Amplify を使ったアプリケーション開発
3. Amplify API Deep Dive
4. 直近の最新アップデート
5. お客様からよくあるご質問
- 6. まとめ**

まとめ

AWS Amplify は Web フロントエンド、モバイルアプリの開発を加速させるために作られたプラットフォーム

- ✓ Amplify CLI でバックエンドを自動構築
- ✓ Amplify Framework を用いて直感的なにバックエンドと連携
- ✓ Amplify Console で簡単にCI/CD環境を構築

本質的な作業に集中し、最速でサービスをリリース

Q&A

お答えできなかったご質問については

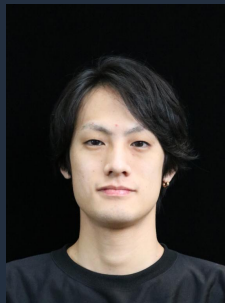
AWS Japan Blog 「<https://aws.amazon.com/jp/blogs/news/>」にて
後日掲載します。

5月の Black Belt Online Seminar 配信予定

<https://amzn.to/JPWebinar>

5/26 (火) 12:00-13:00 AWS X-Ray

5/27 (水) 18:00-19:00 VMware Cloud on AWS



6月のBlack Belt Online Seminar 配信予定

<https://amzn.to/JPWebinar>

- 6/2 (火) 12:00-13:00 AWS AI Language Services
- 6/9 (火) 12:00-13:00 Amazon Quantum Ledger Database (QLDB)
- 6/10 (水) 18:00-19:00 Event Driven Serverless Application
- 6/16 (火) 12:00-13:00 AWS サポート技術支援サービス紹介
- 6/17 (水) 18:00-19:00 Amazon Athena
- 6/23 (火) 12:00-13:00 Amazon Elasticsearch Service
- 6/24 (水) 18:00-19:00 Container Service Update
- 6/30 (火) 12:00-13:00 Amazon Cognito



AWS Amplify ハンズオンのご案内

5月29日（金）18:00 - 21:00

AWS Amplify ～Vue と React えられるオンラインハンズオン～

AWS Amplify でサクッと Web フロントエンドアプリケーションを作ってみましょう。このハンズオンでは、Vue.js を使ったチャットアプリや AI 機能の実装や、React を使った SNS "BOYAKI" の実装が学べます。また、Amplify Console を使って Web サイトホスティング、公開する手順も学ぶことができ、開発したアプリケーションを手軽に公開することができるようになります。フロントエンドエンジニアの方、Web アプリケーションを素早く開発したい方、簡単なサイトホスティングの方法を探している方はぜひご参加ください！

このハンズオンでは、オンラインで各教材の内容を説明した後、各自で教材を見てハンズオンを進めていただき、随時質問を受け付けて返答したり解説したりしていく予定です。ローカルで作業を行いたくない、ローカルでの開発環境構築が難しい方は、AWS Cloud9 を使って作業を進めていただくことができます（Cloud9 を使った場合、React ハンズオンの一部 Amplify mocking の手順が実施できません。それ以外のステップは問題なく取り組んでいただけます）。当日は、インターネットに接続可能な PC、Admin 権限をもつハンズオン用 AWS アカウントをご用意の上ご参加ください。

開催日時 2020年5月29日（金）18:00 - 21:00 (18:00セッション開始)

開催形式 オンラインハンズオン

■申し込みはこちらの URL から

https://pages.awscloud.com/WEBINAR_awsamplify-handson_20200529LandingPage.html

AWS の日本語資料の場所「AWS 資料」で検索



日本担当チームへお問い合わせ サポート 日本語 ▼ アカウント ▼

コンソールにサインイン

製品 ソリューション 料金 ドキュメント 学習 パートナー AWS Marketplace その他 🔍

AWS クラウドサービス活用資料集トップ

アマゾン ウェブ サービス (AWS) は安全なクラウドサービスプラットフォームで、ビジネスのスケールと成長をサポートする処理能力、データベースストレージ、およびその他多種多様な機能を提供します。お客様は必要なサービスを選択し、必要な分だけご利用いただけます。それらを活用するために役立つ日本語資料、動画コンテンツを多数ご提供しております。(本サイトは主に、AWS Webinar で使用した資料およびオンデマンドセミナー情報を掲載しています。)

[AWS Webinar お申込 »](#)

[AWS 初心者向け »](#)

[業種・ソリューション別資料 »](#)

[サービス別資料 »](#)

<https://amzn.to/JPArchive>



AWS Well-Architected 個別技術相談会

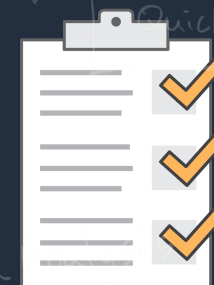
毎週“W-A個別技術相談会”を実施中

- AWSのソリューションアーキテクト(SA)に
対策などを相談することも可能

- **申込みはイベント告知サイトから**
(<https://aws.amazon.com/jp/about-aws/events/>)

AWS イベント で[検索]

AWS Well-Architected



aws

ご視聴ありがとうございました

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



過去資料
<https://amzn.to/JPArchive>

