モバイル開発における

データストアの選び方

Takayuki Shimizu
Solutions Architect, Amazon Data Services Japan, K.K.
AWS Summit Tokyo 2015 - Developer Conference - 2015. 6. 3

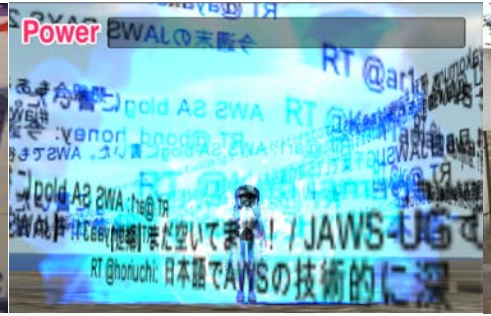
amazon web services

amazon web services

自己紹介

清水 崇之

- アマゾン データ サービス ジャパン 株式会社
  - ソリューション アーキテクト
- 西日本担当（大阪のお客様にもプライム対応！だけどプライスレス）
- ゲーム、モバイル、Web サービス全般
- AWS 芸人 (詳しくは http://www.slideshare.net/shimy_net)



amazon web services

アジェンダ

Agenda

- AWS Mobile サービスの紹介
  - サービスの概要
  - AWS Mobile SDK
  - モバイルサービスと 2-Tier アーキテクチャ
- データストアの選び方
  - Amazon Cognito Sync
  - Amazon DynamoDB
  - Amazon S3
  - Amazon Kinesis
- まとめ

アジェンダ

Agenda

- AWS Mobile サービスの紹介
  - **サービスの概要**
  - AWS Mobile SDK
  - モバイルサービスと 2-Tier アーキテクチャ
- データストアの選び方
  - Amazon Cognito Sync
  - Amazon DynamoDB
  - Amazon S3
  - Amazon Kinesis
- まとめ

AWS Mobile サービス

あなたのモバイルアプリ


ゲームアプリ


ユーティリティアプリ


家計簿アプリ


クーポンアプリ

AWS Mobile SDK


AWS SDK for Android


AWS SDK for iOS


AWS SDK for Unity

モバイルに最適化されたサービス


Amazon Cognito


Amazon Mobile Analytics


Amazon SNS Mobile Push


Lambda Functions

モバイルに最適化されたコネクタ


Kinesis


DynamoDB


S3


SQS


SES


Lambda

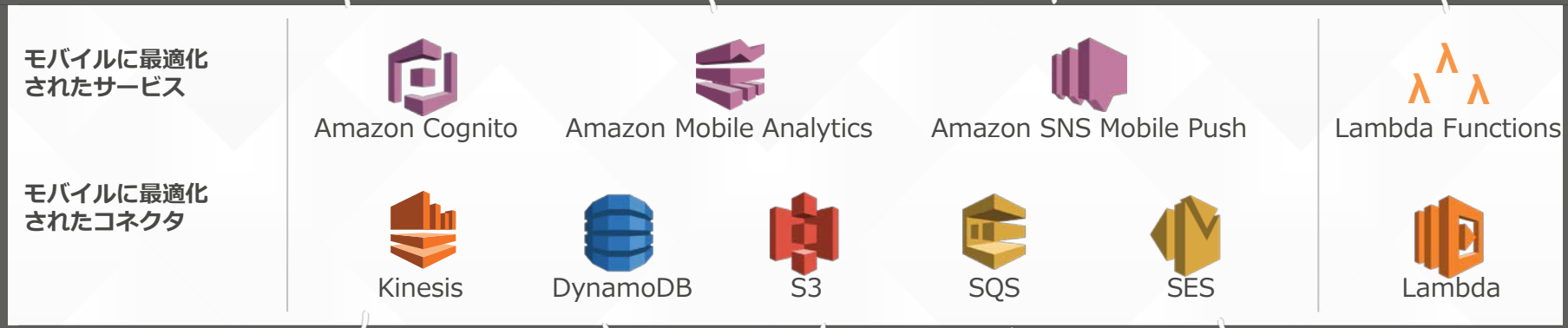
コアとなるサービス群

ストレージ | ネットワーク | データベース | 分析 | コンピュート

グローバルインフラ

amazon web services

AWS Mobile サービス



AWS Mobile サービス

あなたのモバイルアプリ


ゲームアプリ


ユーティリティアプリ


家計簿アプリ


クーポンアプリ

AWS Mobile SDK


AWS SDK for Android


AWS SDK for iOS


AWS SDK for Unity

モバイルに最適化されたサービス


Amazon Cognito


Amazon Mobile Analytics


Amazon SNS Mobile Push


Lambda Functions

モバイルプラットフォームに合わせて AWS SDK を利用

モバイルに最適化されたコネクタ


Kinesis


DynamoDB


S3


SQS


SES

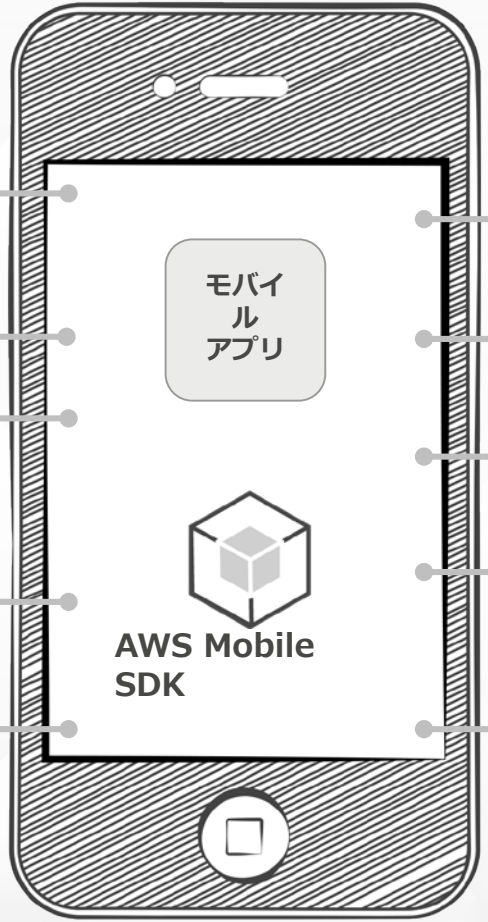

Lambda

コアとなるサービス群

ストレージ　ネットワーク　データベース　分析　コンピュート

グローバルインフラ

モバイルアプリの課題を AWS サービスで解決



ユーザ認証
Amazon Cognito
(Identity)

アクセスの認可
AWS Identity and
Access Management

データの同期
Amazon Cognito
(Sync)

ユーザ行動分析
Amazon Mobile
Analytics

ビジネスロジックの実行
AWS Lambda

メディアの管理
Amazon S3
Transfer Manager

メディアの配信
Amazon CloudFront
(Device Detection)

プッシュ通知の送信
Amazon SNS
Mobile Push

共有データの保存
Amazon DynamoDB
(Object Mapper)

ストリームリアルタイムデータ
Amazon Kinesis
(Recorder)

amazon web services

モバイルアプリの課題を AWS サービスで解決

ユーザ認証
Amazon Cognito (Identity)

アクセスの認可
AWS Identity and Access Management

データの同期
Amazon Cognito (Sync)

ユーザ行動分析
Amazon Mobile Analytics

ビジネスロジ
AWS

メディアの管理
Amazon S3 Transfer Manager

メディアの配信
Amazon CloudFront (Device Detection)

プッシュ通知の送信
Amazon SNS Mobile Push

共有データの保存
Amazon DynamoDB

ルタイムデータ
sis

本来の業務に集中

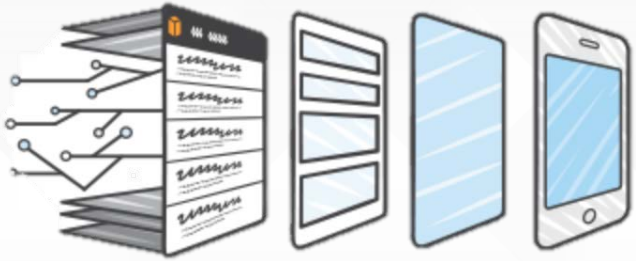
amazon web services

アジェンダ

Agenda

- AWS Mobile サービスの紹介
  - サービスの概要
  - **AWS Mobile SDK**
  - モバイルサービスと 2-Tier アーキテクチャ
- データストアの選び方
  - Amazon Cognito Sync
  - Amazon DynamoDB
  - Amazon S3
  - Amazon Kinesis
- まとめ

AWS Mobile SDK



クロスプラットフォーム

- Android, Fire OS, iOS, Unity をサポート

共通の認証

- Amazon Cognito により認証機構を迅速に導入可能

ネットワーク状態を自動でハンドリング

- 例： ローカルキャッシュによりオフラインでも利用可能

メモリフットプリントの削減

- 導入するパッケージをサービス単位で選択可能

アジェンダ

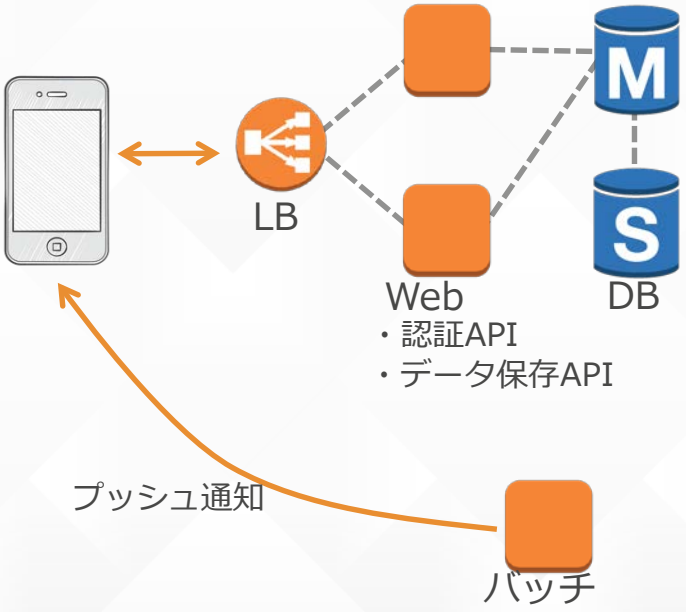
Agenda

- AWS Mobile サービスの紹介
  - サービスの概要
  - AWS Mobile SDK
  - **モバイルサービスと 2-Tier アーキテクチャ**
- データストアの選び方
  - Amazon Cognito Sync
  - Amazon DynamoDB
  - Amazon S3
  - Amazon Kinesis
- まとめ

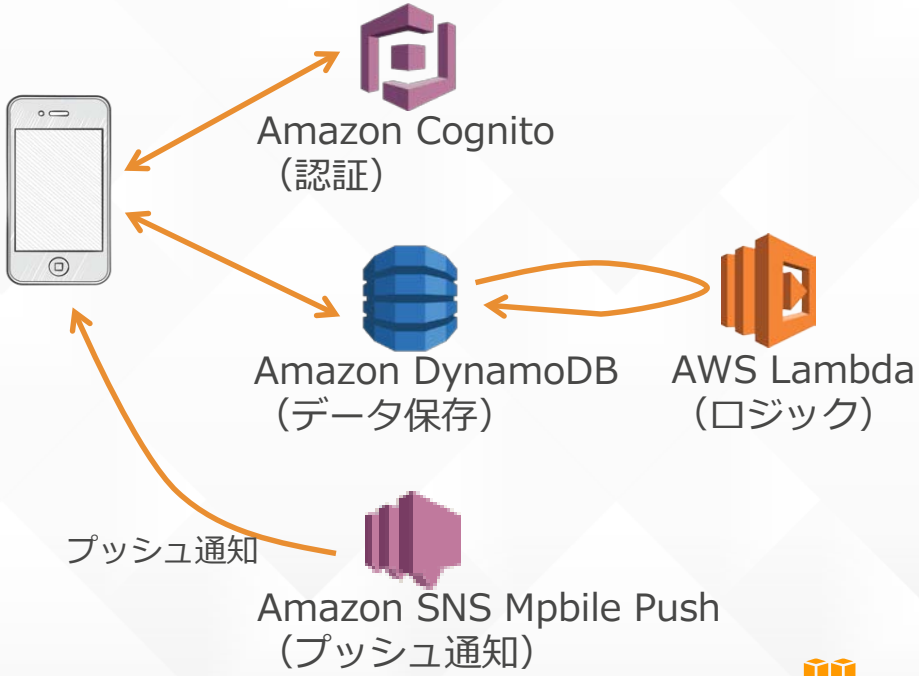
2-Tier アーキテクチャとは？

仮想サーバー(EC2)を利用せずに、クライアントから直接 AWS のサービスを利用する。

従来 アーキテクチャ



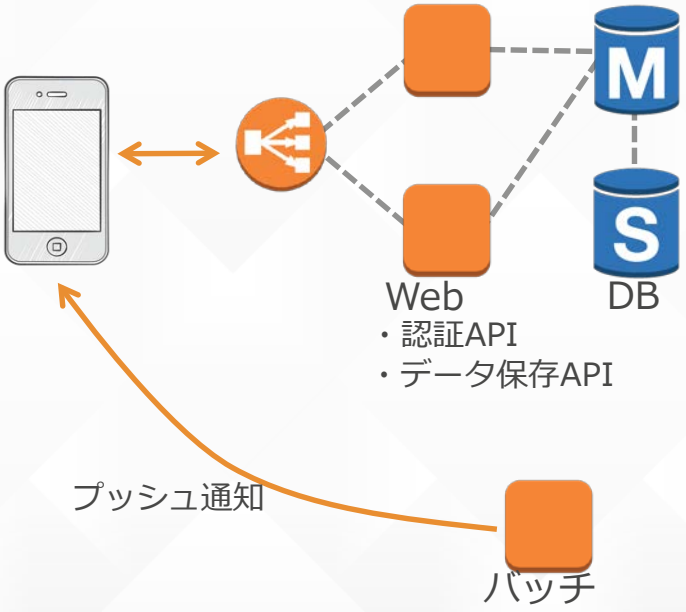
2-Tier アーキテクチャ



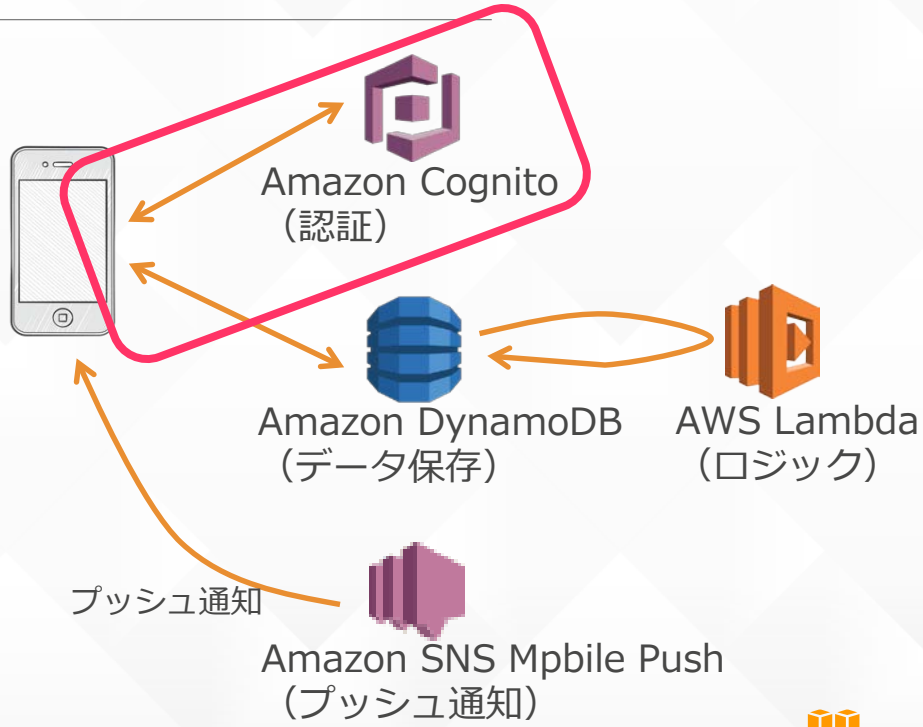
2-Tier アーキテクチャとは？

仮想サーバー(EC2)を利用せずに、クライアントから直接 AWS のサービスを利用する。

従来 アーキテクチャ



2-Tier アーキテクチャ

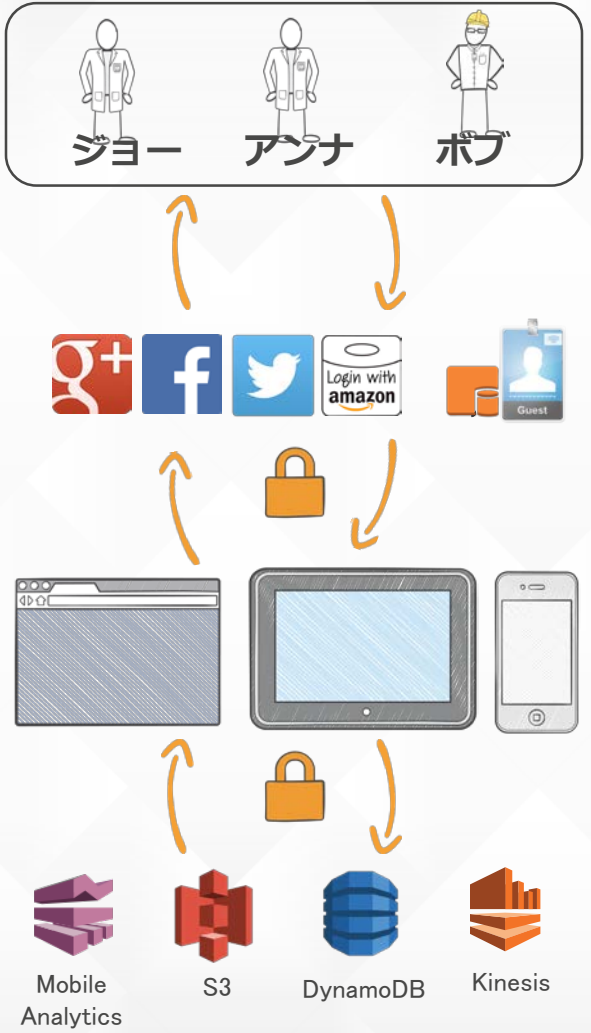


Amazon Cognito Identity

複数の ID プロバイダ
ID プロバイダと簡単に連携して認証
Amazon, Facebook, Twitter, Google,
OpenID Connect などに対応

ゲストユーザアクセス
アカウント作成やログインを必要とせず、セキュアに AWS サービスを利用でき、アプリの特徴を損なわない。

独自認証システム
ID プロバイダによる認証ではなく
独自の Username と Password で認証

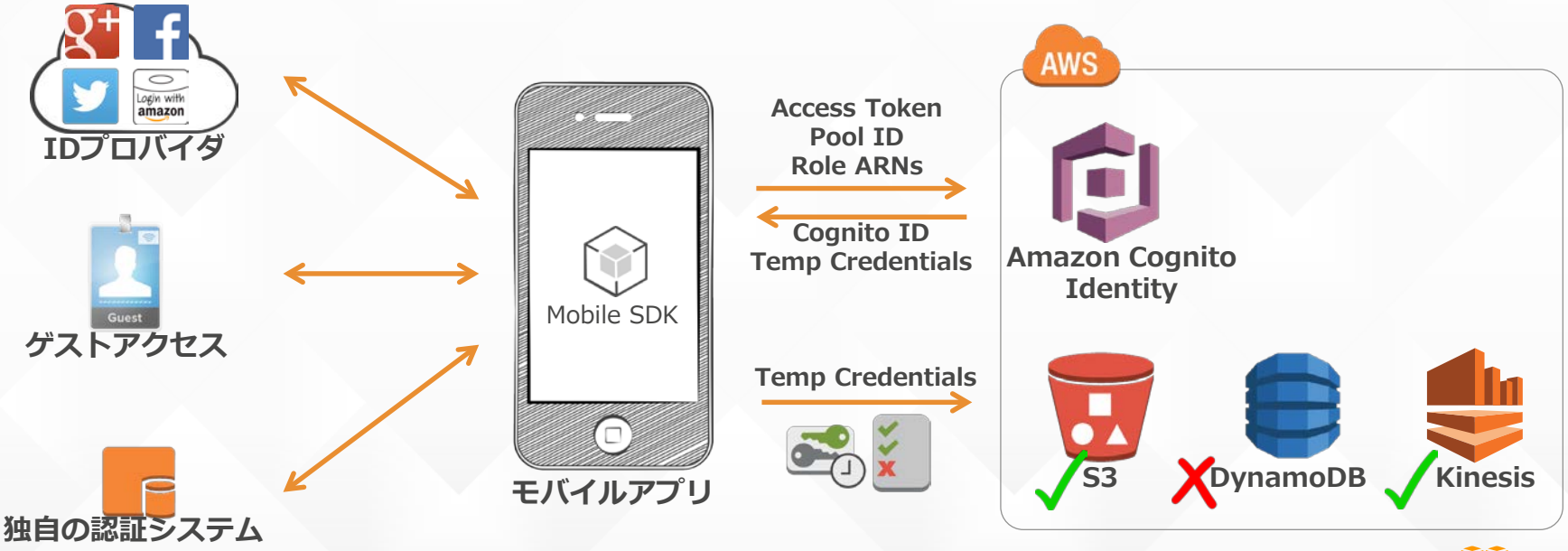


amazon web services

Amazon Cognito Identity

Cognito による認証のフロー

ID プロバイダや独自認証システムと連携して認証することで、任意のポリシーを持った一時的なクレデンシャルが発行され、AWS リソースを利用する。

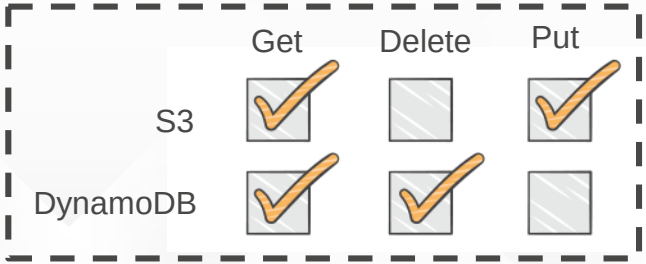


セキュリティ

AWS Credentials の保護
必要な権限だけを付与された一時的なクレデンシャルを取得できるので、アプリにクレデンシャルを埋め込まなくてよい

セキュリティのベストプラクティスを支援
セキュリティトークンサービスと連携したり、トークンベンディングマシンを構築する必要はない
セキュアに AWS サービスを利用

細やかなアクセス制御
IAM との連携で AWS リソースへの細かなアクセス制御を実現

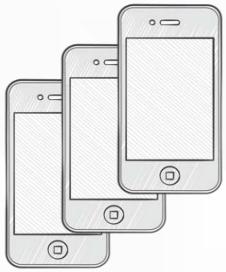


アジェンダ

Agenda

- AWS Mobile サービスの紹介
  - サービスの概要
  - AWS Mobile SDK
  - モバイルサービスと 2-Tier アーキテクチャ
- **データストアの選び方**
  - Amazon Cognito Sync
  - Amazon DynamoDB
  - Amazon S3
  - Amazon Kinesis
- まとめ

4 つの AWS データストア



モバイル

Cognito Sync

Cotnito Identity との密な連携
シンプルなテキストデータの保存

DynamoDB Mapper

複雑な構造、検索が必要なテキスト
データの保存

S3 Transfer Manager

メディア・コンテンツなどのサイズの
大きいデータの保存

Kinesis Recorder

アプリ内で頻繁に発生するユーザー行
動イベントなど、ストリーミングデー
タの保存

amazon web services

アジェンダ

Agenda

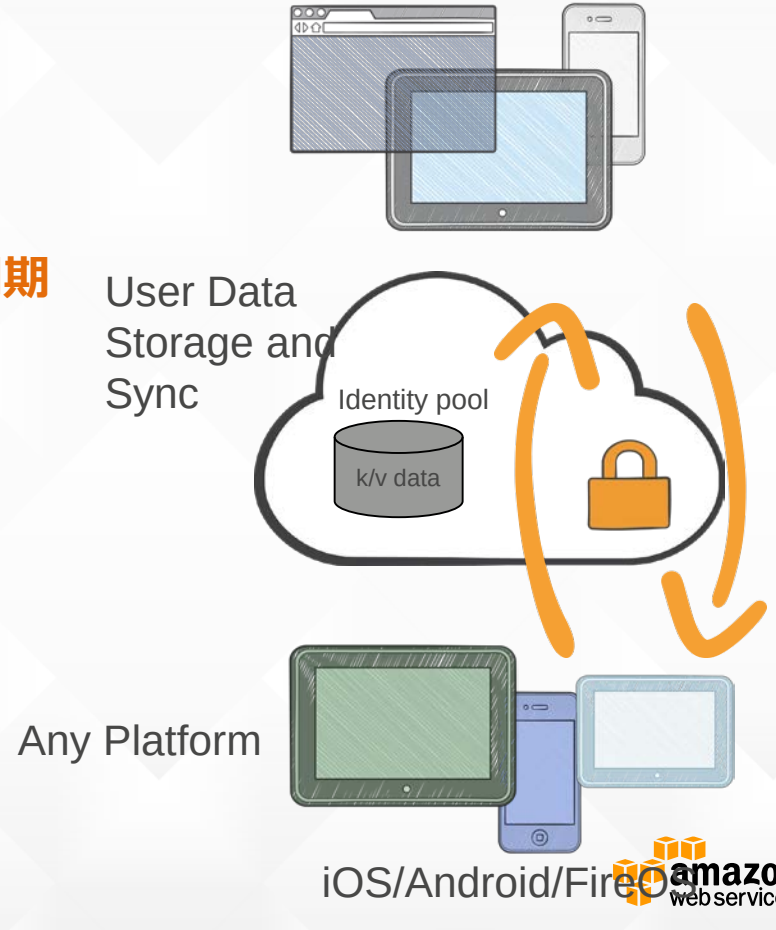
- AWS Mobile サービスの紹介
  - サービスの概要
  - AWS Mobile SDK
  - モバイルサービスと 2-Tier アーキテクチャ
- データストアの選び方
  - **Amazon Cognito Sync**
  - Amazon DynamoDB
  - Amazon S3
  - Amazon Kinesis
- まとめ

Amazon Cognito Sync クラウドへのデータ保存と同期

アプリのデータ、設定、状態などを保存
アプリやデバイスのデータをクラウドに保存でき、ログイン後にマージされる

クロスデバイス、クロスプラットフォームで同期
ユーザデータや設定をデバイスをまたいで同期

ユーザあたり 20MB
各データセットは 1MB までの Key/Value
最大 1024 Key まで保存
バイナリデータは Base64 でエンコードして保存
※2015.6.3 現在



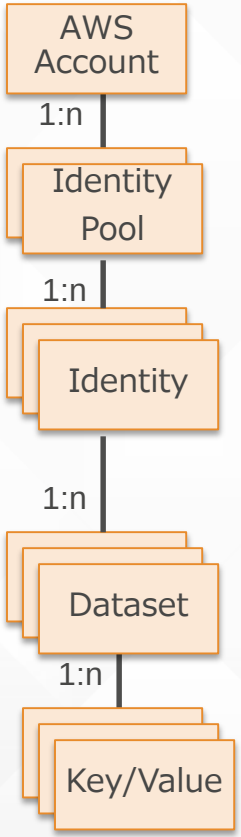
Amazon Cognito Sync クラウドへのデータ保存と同期

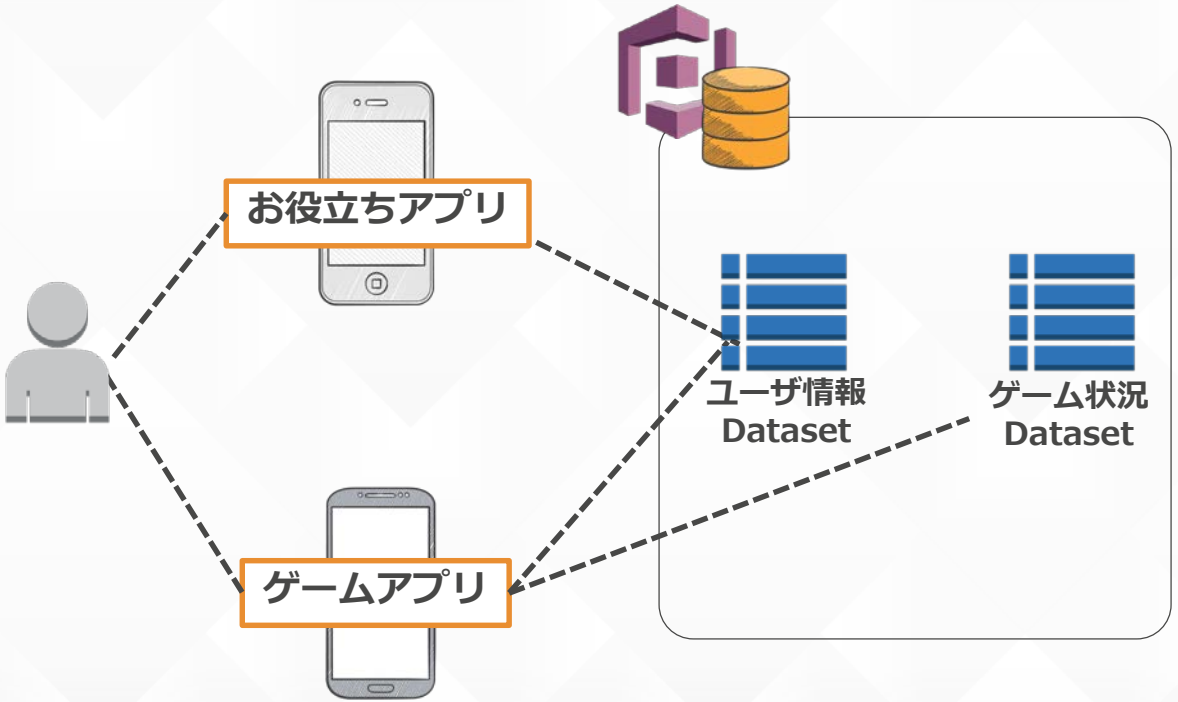
オフライン動作
データはまずローカルのデータストレージに保存されるので、電波が不安定もしくは不通であってもシームレスに動作できる

インテリジェントな同期処理
同期メソッドはローカルとクラウドのデータのバージョンを比較して、データをプッシュ／プル

柔軟なコンフリクトの解決
同期メソッドは、変更を読み取り後、ローカルの変更をクラウドのデータストアへ書き込む。デフォルトでは、最後の書き込みを有効として保存する。開発者はコンフリクト処理を独自に上書きして実装することもできる。

Amazon Cognito Sync データモデル – 例



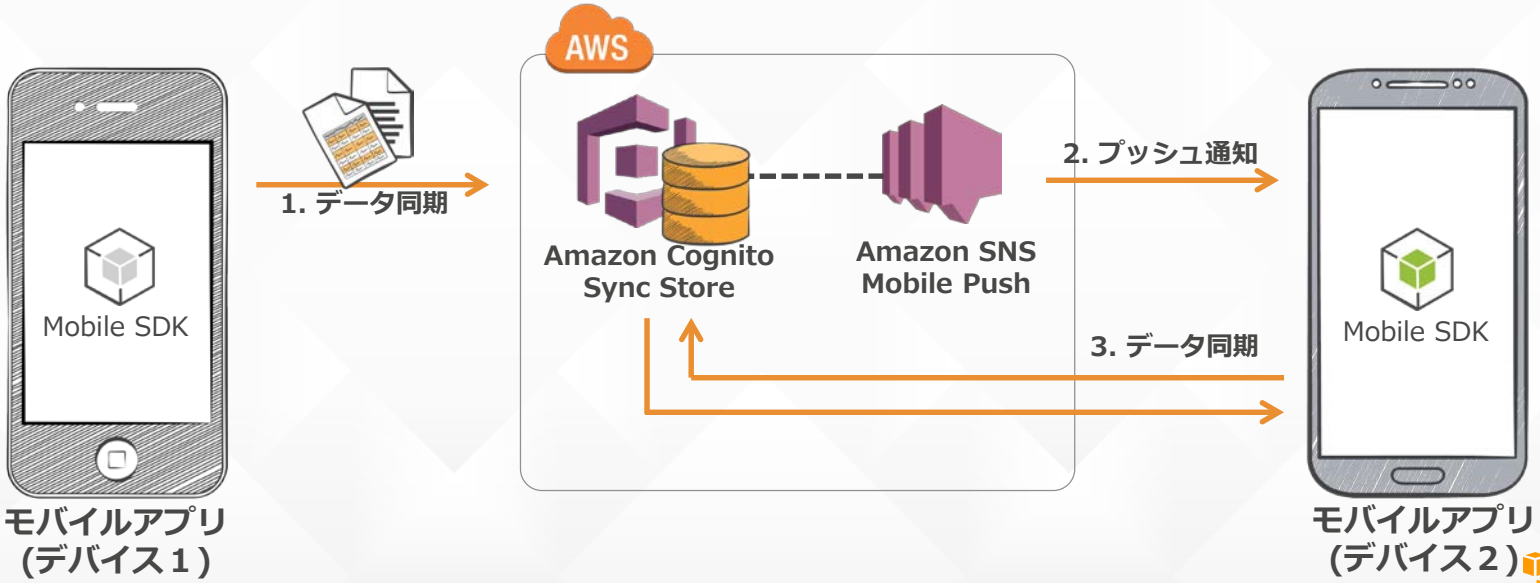


Amazon Cognito Push Sync

Amazon SNS Mobile Push との連携

Amazon Cognito のデータストアが更新されたタイミングで、Amazon SNS Mobile Push と連携して、各デバイスにプッシュ通知を送信できる。

プッシュ通知を受け取ったアプリはデータストアの再同期を行うように実装するなど。



Amazon Cognito Stream

Amazon Kinesis との連携

Amazon Cognito のデータストアが更新されたタイミングで、Amazon Kinesis ストリームで更新や同期のデータを受け取ることができる。

ストリームの内容

```
{
 "identityPoolId" : "Pool Id”
 "identityId" : "Identity Id ”
 "dataSetName" : "Dataset Name”
 "operation" : "(replace|remove)”
 "kinesisSyncRecords" : [
   {
     "key" : "Key",
     "value" : "Value",
     "syncCount" : 1,
     "lastModifiedDate" : 1424801824343,
     "deviceLastModifiedDate" : 1424801824343,
     "op": "(replace|remove)" }, ...
 ],
 "lastModifiedDate": 1424801824343,
 "kinesisSyncRecordsURL": "S3Url",
 "payloadType" : "(S3Url|Inline)",
 "syncCount" : 1
}
```

Amazon Cognito Events

AWS Lambda との連携

元データ

Key	Value
住所	**大阪市□北区□中之島**

修正済みデータ

Key	Value
住所	**大阪市北区中之島**

Lambda Function (Node.js)

空白を削除するプログラム

Amazon Cognito Sync をアプリに組み込む（Android）

CognitoSyncManager の初期化

```
cognito = new CognitoSyncManager (context,
      Regions.US_EAST_1, provider);
```

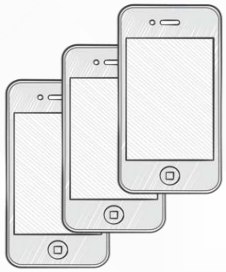
Dataset の作成、 Key Values の追加

```
Dataset dataset =
      cognito.openOrCreateDataset(datasetName);
dataset.put(key, value);
```

同期の実行

```
dataset.synchronize(new SyncCallback(){..});
```

Amazon Cognito Sync の用途



モバイル

Cognito Sync

Cognito Identity との密な連携
シンプルなテキストデータの保存

DynamoDB Mapper

複雑な構造、検索が必要なテキストデータの保存

S3 Transfer Manager

メディア・コンテンツなどのサイズの大きいデータの保存

Kinesis Recorder

アプリ内で頻繁に発生するユーザー行動イベントなど、ストリーミングデータの保存

amazon web services

アジェンダ

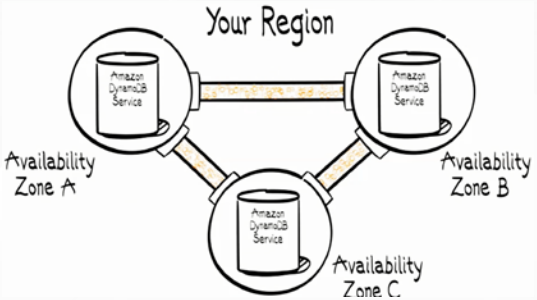
Agenda

- AWS Mobile サービスの紹介
  - サービスの概要
  - AWS Mobile SDK
  - モバイルサービスと 2-Tier アーキテクチャ
- データストアの選び方
  - **Amazon Cognito Sync**
  - **Amazon DynamoDB**
  - Amazon S3
  - Amazon Kinesis
- まとめ

Amazon DynamoDB

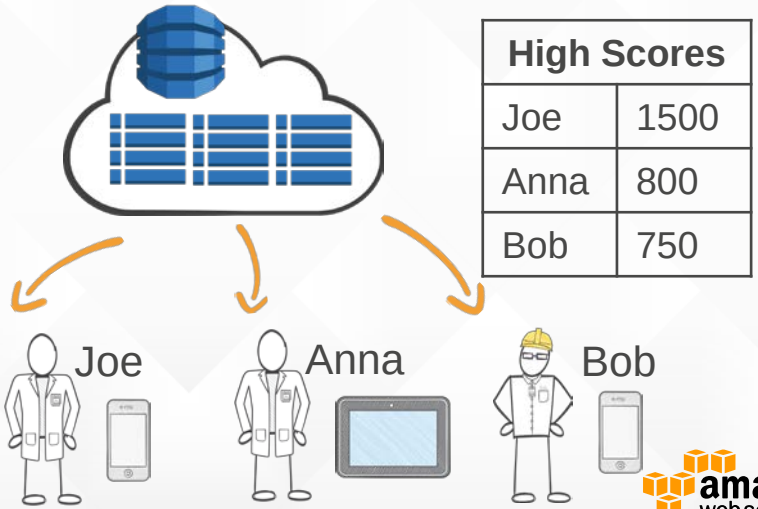
Amazon DynamoDB とは

高い信頼性、スケーラビリティ、低レイテンシで安定した性能を兼ね備えた NoSQL データベースサービス



Amazon DynamoDB Connector : Object Mapper

- クライアント側のクラスが Amazon DynamoDB テーブルにマッピング
- オブジェクトをテーブルに変換することも、その逆も必要ない



amazon web services

DynamoDB テーブル

たとえば、本の検索アプリの DynamoDB テーブルを作成する。

AWS

Amazon DynamoDB

Table: Books

isbn (HashKey)	title	hardCover
22-22222	我輩は猫である	Yes
43-43234	こころ	No
55-12345	三四郎	Yes

amazon web services

マッピングクラスを定義

Table: Books

isbn (HashKey)	title	hardCover
22-22222	我輩は猫である	Yes
43-43234	こころ	No
55-12345	三四郎	Yes

```
@DynamoDBTable(tableName = "Books")
public static class Book {
  private String isbn;
  private String title;
  …
  @DynamoDBHashKey(attributeName = “isbn")
  public String getIsbn() {
    return isbn;
  }

  @DynamoDBAttribute(attributeName="title")
  public String getTitle() {
    return title;
  }
  …
```

データの操作（Android）

DynamoDB Client & Object Mapper の生成

```
AmazonDynamoDBClient client =
  new AmazonDynamoDBClient(cognitoProvider);
DynamoDBMapper mapper = new DynamoDBMapper(client);
```

1件取得

```
Book book = mapper.load(Book.class, "1234567");
```

登録／更新

```
book.setIsbn("1234567");
book.setTitle("Great Expectations");
mapper.save(book);
```

削除

```
mapper.delete(book);
```

全件取得・検索（Android）

全件取得

```
PaginatedScanList<Book> result =
  mapper.scan(Book.class, scanExpression);
```

検索

```
Book bookToFind = new Book();
bookToFind.setAuthor("Charles Dickens");

String queryString = "Great";

Condition rangeKeyCondition = new Condition()
 .withComparisonOperator(ComparisonOperator.BEGINS_WITH.toString())
 .withAttributeValueList(new AttributeValue().withS(queryString.toString()));

DynamoDBQueryExpression queryExpression = new DynamoDBQueryExpression()
 .withHashKeyValues(bookToFind)
 .withRangeKeyCondition("Title", rangeKeyCondition)
 .withConsistentRead(false);

PaginatedQueryList<Book> result = mapper.query(Book.class, queryExpression);
```

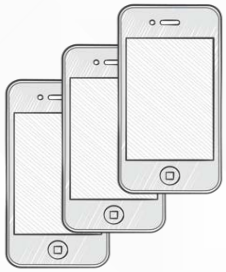
Amazon DynamoDB Stream（プレビュー）

AWS Lambda と連携したり、KCL ワーカーで処理するなど

DynamoDBへの書き込みに応じて値チェックをし別テーブルの更新やプッシュ通知を実行するなど、アイディアはさまざま。

▶プレビュー申し込み
https://aws.amazon.com/jp/dynamodb/preview/

Amazon DynamoDB の用途



モバイル

Cognito Sync

Cognito Identity との密な連携
シンプルなテキストデータの保存

DynamoDB Mapper

複雑な構造、検索が必要なテキストデータの保存

S3 Transfer Manager

メディア・コンテンツなどのサイズの大きいデータの保存

Kinesis Recorder

アプリ内で頻繁に発生するユーザー行動イベントなど、ストリーミングデータの保存

アジェンダ

Agenda

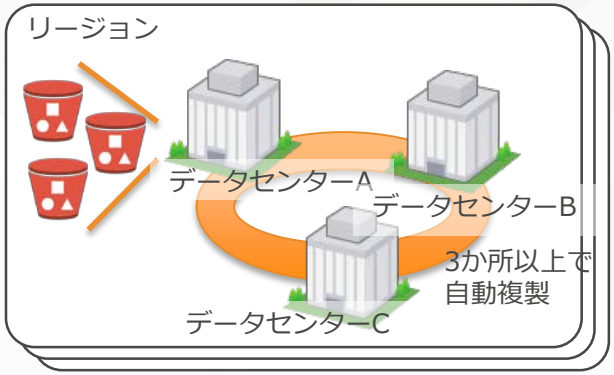
- AWS Mobile サービスの紹介
  - サービスの概要
  - AWS Mobile SDK
  - モバイルサービスと 2-Tier アーキテクチャ
- データストアの選び方
  - Amazon Cognito Sync
  - Amazon DynamoDB
  - **Amazon S3**
  - Amazon Kinesis
- まとめ

Amazon S3

Amazon S3 とは

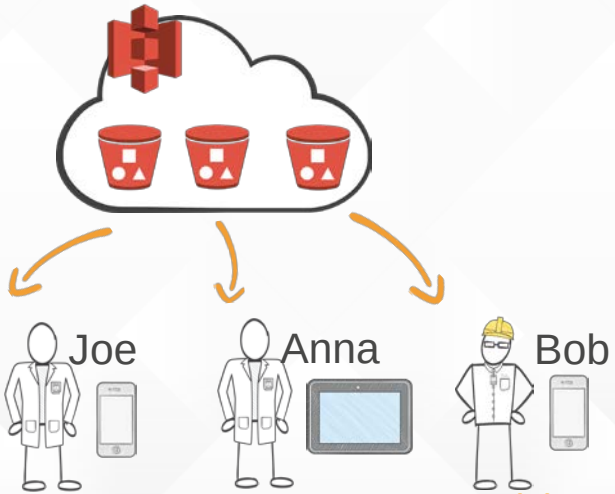
安全で耐久性があり拡張性の高い容量無制限の

マネージドオンラインストレージサービス



Amazon S3 Connector: Transfer Manager

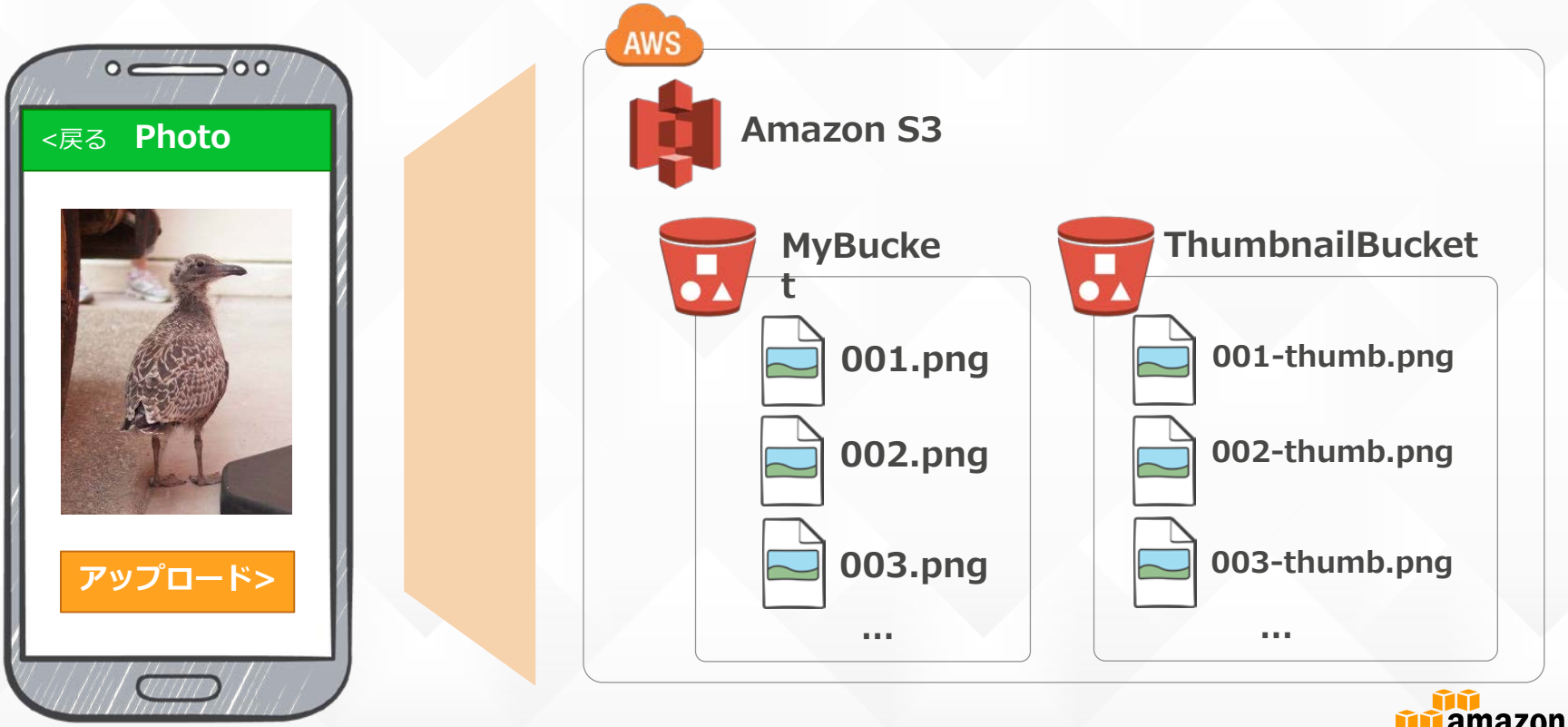
- マルチパートアップロード対応
- フォールトトレラントなダウンロード
- バックエンドサーバー不要
- 自動リトライ
- モバイルアプリ／ネットワークに最適化するための
  ポーズ、レジューム、キャンセル関数



amazon web services

S3 Bucket の作成

たとえば、写真をアップロードするアプリのバケットを作成する。



オブジェクトの操作（Android）

S3 Transfer Manager Client の生成

```
TransferManager transferManager =
  new TransferManager(cognitoProvider);
```

ダウンロード

```
Download download =
  transferManager.download(“MyBucket”, key, file);
```

アップロード

```
Upload upload =
  transferManager.upload(“MyBucket”, key, file);
```

```
while (!upload.isDone()){
  // アップロード中
}
```

オブジェクトの操作（Android）

進捗の取得

```
TransferProgress transferred = upload.getProgress();
transferred.getBytesTransferred();
transferred.getPercentTransferred();
```

ポーズ

```
upload.pause();
```

レジューム

```
transferManager.resumeUpload(upload);
```

キャンセル

```
upload.abort();
download.abort();
```

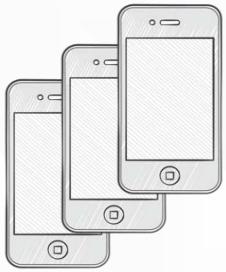
サムネイル画像の変換

AWS Lambda と連携

Amazon S3 にアップロードされた画像をリアルタイムに AWS Lambda で変換し、サムネイルを Amazon S3 の別Bucketへ、メタデータは Amazon DynamoDB へ。

デモ：S3＋DynamoDB＋Lambda 画像アップローダー

Amazon S3 の用途



モバイル

Cognito Sync

Cognito Identity との密な連携
シンプルなテキストデータの保存

DynamoDB Mapper

複雑な構造、検索が必要なテキストデータの保存

S3 Transfer Manager

メディア・コンテンツなどのサイズの大きいデータの保存

Kinesis Recorder

アプリ内で頻繁に発生するユーザー行動イベントなど、ストリーミングデータの保存

amazon web services

アジェンダ

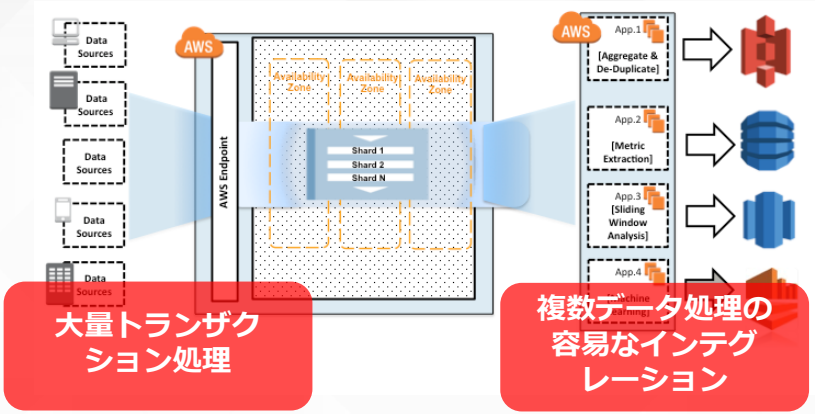
Agenda

- AWS Mobile サービスの紹介
  - サービスの概要
  - AWS Mobile SDK
  - モバイルサービスと 2-Tier アーキテクチャ
- データストアの選び方
  - Amazon Cognito Sync
  - Amazon DynamoDB
  - Amazon S3
  - **Amazon Kinesis**
- まとめ

Amazon Kinesis

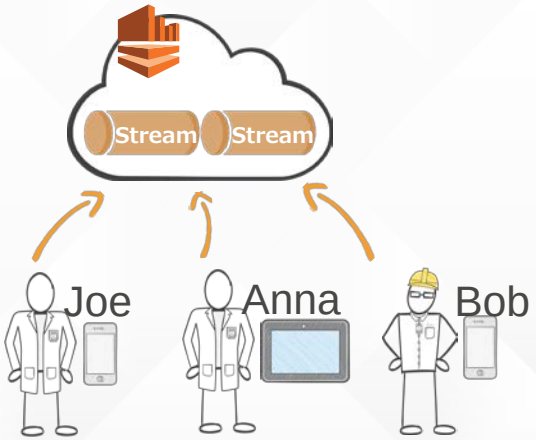
Amazon Kinesis とは

可用性や拡張性に優れたマネージドリアルタイム
大規模ストリーミング処理サービス



Amazon Kinesis Connecter: Kinesis Recorder

- PutRecord リクエストをローカルストレージに蓄積してから一括送信
- 複数リクエストを一括送信することでバッテリー消費を低減
- オフラインでも動作しデータのロストがない。

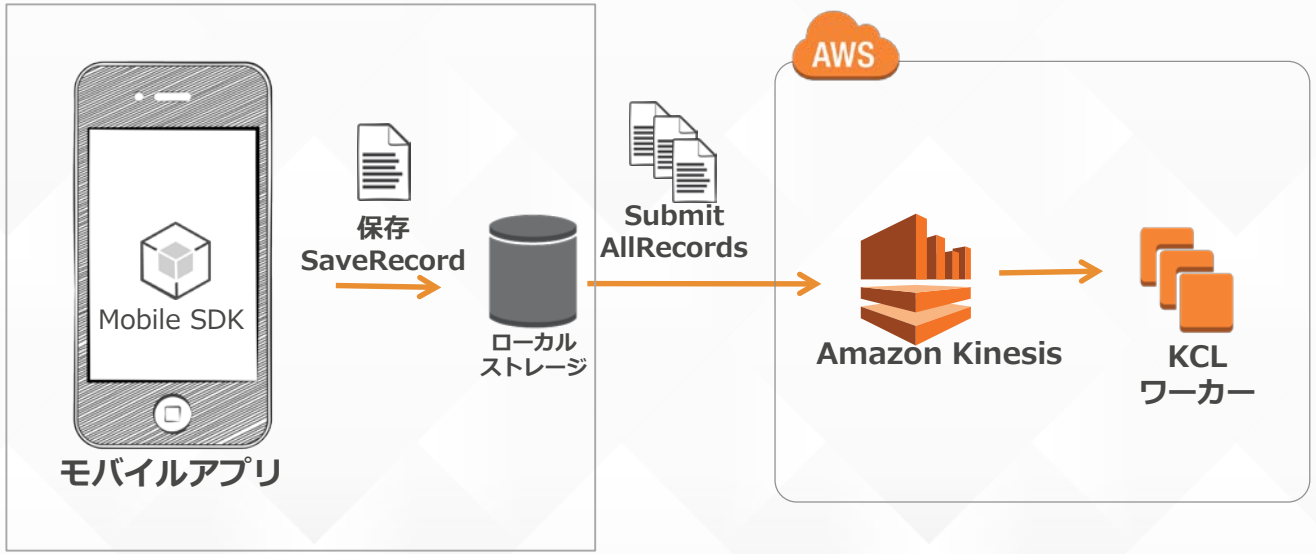


amazon web services

Amazon Kinesis

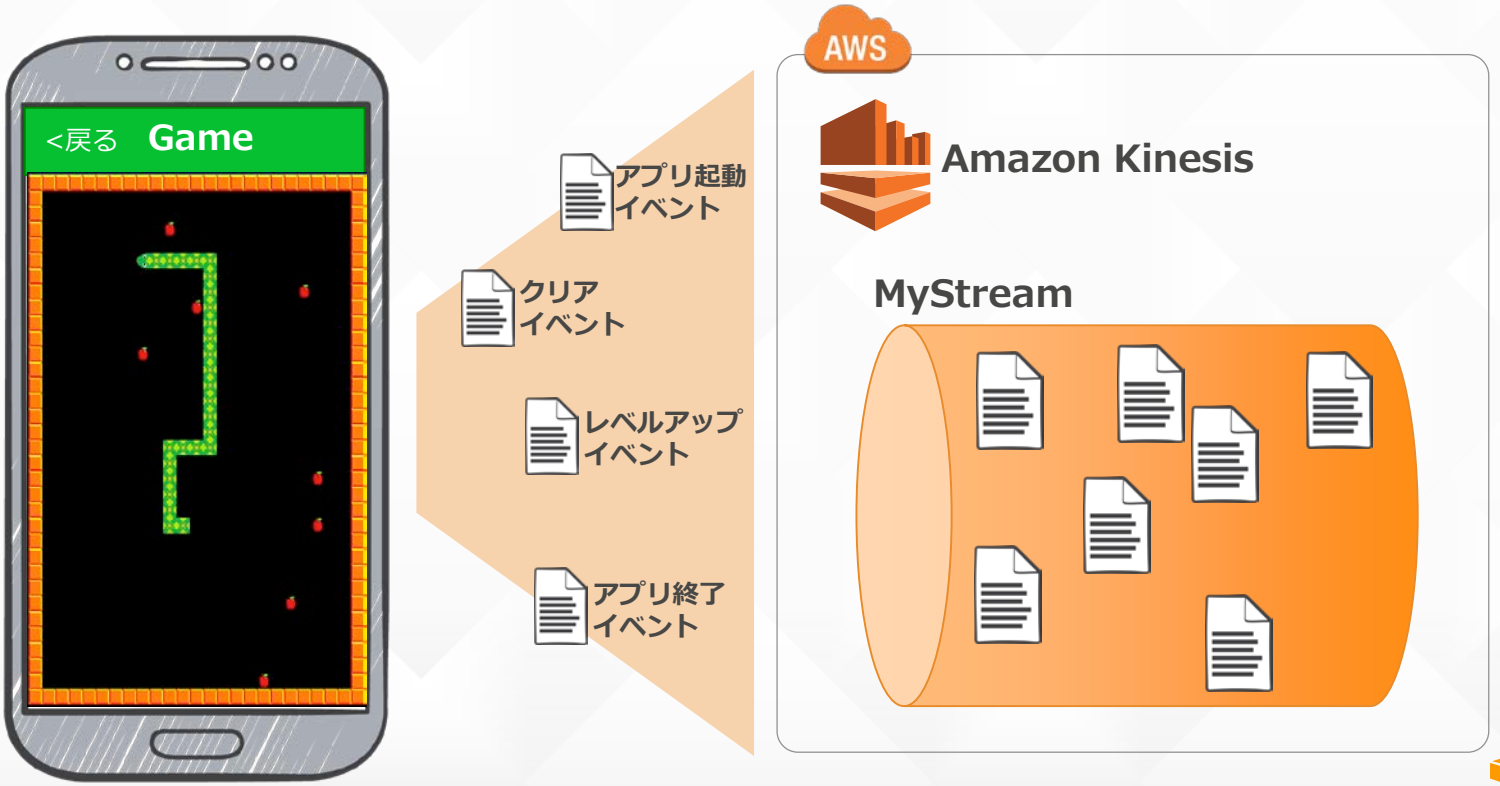
Amazon Kinesis のフロー

データをローカルストレージに蓄積してから一括送信する。登録されたデータは KCL ワーカーなどで処理する。



Kinesis Stream の作成

たとえば、ゲームアプリのユーザー行動イベントを収集するための Stream を作成する。



データの送信（Android）

Kinesis Recorder の生成

```
KinesisRecorder recorder = new KinesisRecorder(
  cognitoProvider,
  Regions.US_WEST_2,
  getDir("YOUR_DIRECTORY", 0));
```

ローカルストレージへ保存

```
recorder.saveRecord("MyData".getBytes(),"MyStream");
```

一括送信

```
recorder.submitAllRecords();
```

注意）Kinesis Recorder は同期的なコールを利用しているので、
メインスレッドで Kinesis Recorder のメソッドをコールすべきではない。

ローカルストレージの状態（Android）

ストレージの残容量

```
recorder.getDiskByteLimit();
```

ストレージの使用量

```
recorder.getDiskBytesUsed();
```

最大ストレージ容量の確認（デフォルト 8MB）

```
KinesisRecorderConfig config =
  recorder.getKinesisRecorderConfig();
config.getMaxStorageSize();
```

最大ストレージ容量の変更

```
config.withMaxStorageSize(1024 * 1024 * 16L);
```

ビッグデータ分析

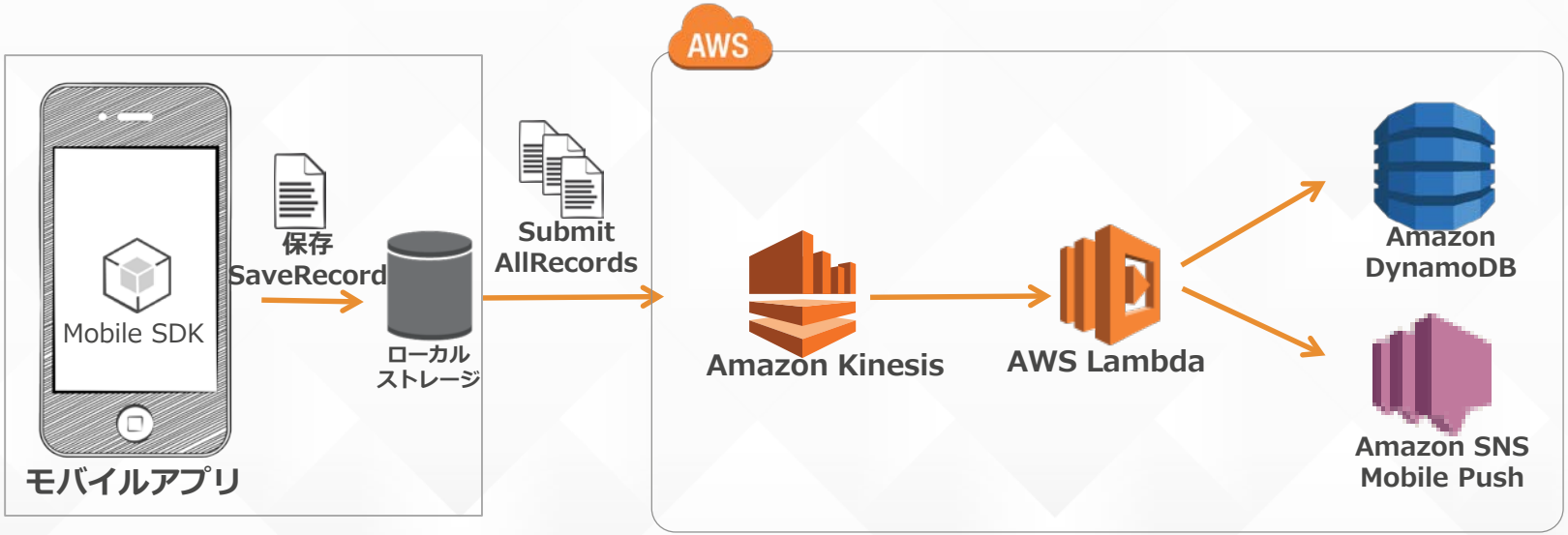
Amazon Redshift と連携

Amazon Kinesis に登録されたアプリのイベントデータを KCL ワーカーなどで処理し、
Amazon Redshift に連携してデータ分析する。

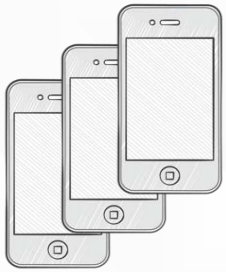
もちろん AWS Lambda とも連携

AWS Lambda と連携

Amazon Kinesis に登録されたアプリのイベントデータをリアルタイムに AWS Lambda で処理して集計やプッシュ通知を実行する。



Amazon Kinesis の用途



モバイル

Cognito Sync

Cognito Identity との密な連携
シンプルなテキストデータの保存

DynamoDB Mapper

複雑な構造、検索が必要なテキストデータの保存

S3 Transfer Manager

メディア・コンテンツなどのサイズの大きいデータの保存

Kinesis Recorder

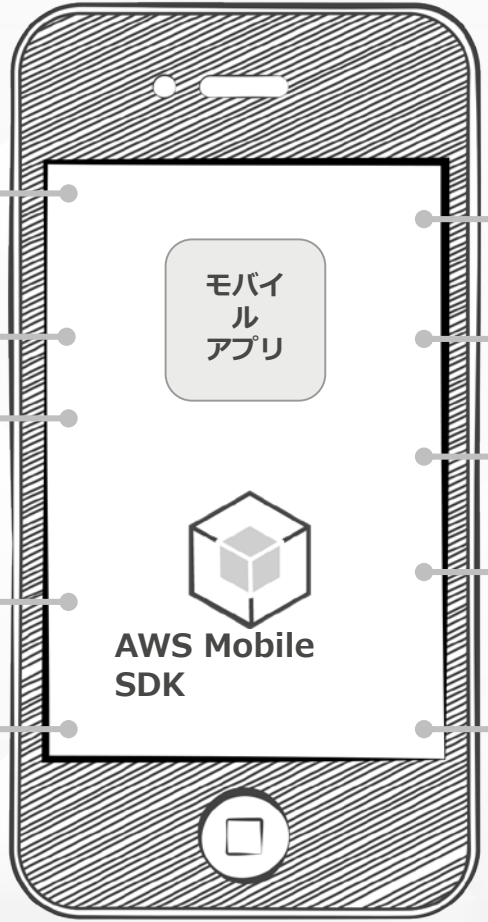
アプリ内で頻繁に発生するユーザー行動イベントなど、ストリーミングデータの保存

アジェンダ

Agenda

- AWS Mobile サービスの紹介
  - サービスの概要
  - AWS Mobile SDK
  - モバイルサービスと 2-Tier アーキテクチャ
- データストアの選び方
  - Amazon Cognito Sync
  - Amazon DynamoDB
  - Amazon S3
  - Amazon Kinesis
- **まとめ**

まとめ：モバイルアプリの課題を AWS サービスで解決



ユーザ認証
Amazon Cognito
(Identity)

アクセスの認可
AWS Identity and
Access Management

データの同期
Amazon Cognito
(Sync)

ユーザ行動分析
Amazon Mobile
Analytics

ビジネスロジックの実行
AWS Lambda

メディアの管理
Amazon S3
Transfer Manager

メディアの配信
Amazon CloudFront
(Device Detection)

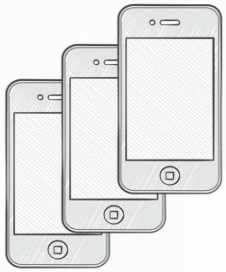
プッシュ通知の送信
Amazon SNS
Mobile Push

共有データの保存
Amazon DynamoDB
(Object Mapper)

ストリームリアルタイムデータ
Amazon Kinesis
(Recorder)

amazon web services

まとめ：用途に合わせてデータストアを利用



モバイル

Cognito Sync

Cognito Identity との密な連携
シンプルなテキストデータの保存

DynamoDB Mapper

複雑な構造、検索が必要なテキスト
データの保存

S3 Transfer Manager

メディア・コンテンツなどのサイズの
大きいデータの保存

Kinesis Recorder

アプリ内で頻繁に発生するイベントな
ど、ストリーミングデータの保存

amazon web services

まとめ：AWS を活用して本来の業務へ集中！！

モバイルアプリのコード
ユーザID管理、認証
ユーザデータの同期処理
非同期通信
アクティブデバイスの分析
ユーザ行動の分析
コンバージョンの分析
プッシュ通知
イベントトリガー
プラットフォームごとのはっきりしない仕様
データチェックと変換
ファイルやメディアのストレージ
共有データベースのストレージ
データコレクション
その他さまざま….
AWS Cloud インフラストラクチャ

開発者が直面する課題

- ✓ 複数のプラットフォーム
- ✓ スケーラビリティの確保
- ✓ 高コストな管理や運用
- ✓ ユーザ体験の阻害
- ✓ ユーザIDの管理

まとめ：AWS を活用して本来の業務へ集中！！

モバイルアプリのコード
ユーザID管理、認証
ユーザデータの同期処理
非同期通信
アクティブデバイスの分析

開発者が直面する課題

ォーム
の確保
運用

プラットフ

共有データベースのストレージ
データコレクション
その他さまざま….
AWS Cloud インフラストラクチャ

AWS を活用して
アプリを差別化