EXA CORPORATION

React Nativeによるクロスプラットフォームなモバイルネイティブアプリ開発

Exa Value Forum 2020

2020年 11月26日

株式会社エクサ

デジタルビジネス推進部　林田太陽

背景

ネイティブアプリの技術検証に至った経緯

製造業のお客様のプロジェクトにおいて、モバイル活用のニーズが存在したため、ネイティブアプリのサンプル開発を通して、技術検証を行うこととなった。その際、お客様の標準端末が iPhoneであったためiOS端末をメインのターゲットとし、Android端末のニーズを見据えて、iOSとAndroidの両方のOSに対応したクロスプラットフォームフレームワークによる開発を行うこととなった。

担当者の経歴

2017年	3月	佐賀大学 理工学部 知能情報システム学科 卒業
	4月	株式会社エクサ 入社
	9月	製鉄所システムリフレッシュ推進部 配属 (現在のデジタルビジネス推進部 AP基盤開発室)
2020年	11月	現在に至る

普段の業務はJava EEを用いたFW開発やアーキテクチャ設計

EXA CORPORATION

目次

- 技術の紹介
  - モバイルアプリの種類
  - ネイティブアプリ開発技術の比較
  - 開発ツール Expo
  - 開発PCの調達におけるExpoの利点
- 検証内容の紹介
  - 技術検証内容
  - 技術要素紹介
- まとめ

EXA CORPORATION

技術の紹介

モバイルアプリの種類

ネイティブアプリ

各OS向けの標準プログラミング言語やネイティブAPIで実装したアプリ。

モバイルWebアプリ

ネイティブアプリのようなレイアウト・操作性で設計されたWebサイト。Webブラウザ上で動作する。

ハイブリッドアプリ

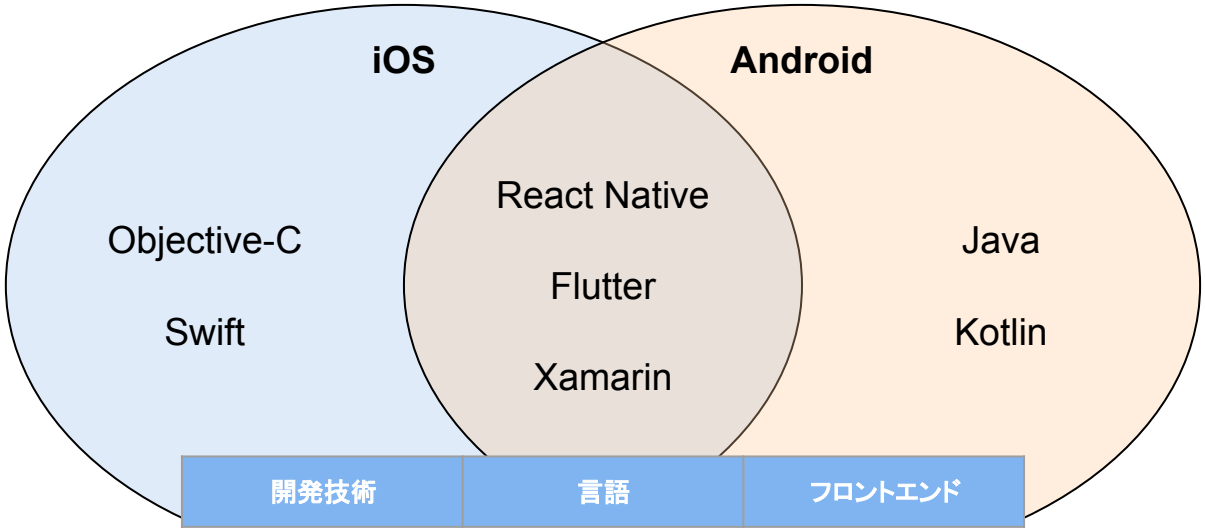
iOSやAndroidが提供するWebViewにHTML・CSS・JavaScriptで作成した画面を表示し、ネイティブ機能を利用できるようにしたアプリ。

項目	ネイティブアプリ	モバイルWebアプリ	ハイブリッドアプリ
動作速度	○	△	△
ネイティブ機能	○	△	○
学習コスト	× または △	○	△
ホーム画面への配置	○	△	○

EXA CORPORATION

ネイティブアプリ開発技術の比較

クロスプラットフォーム開発において、React Nativeは開発言語、フロントエンド技術の両面で既存技術を利用しており、より低い学習コストで開発スタートが可能

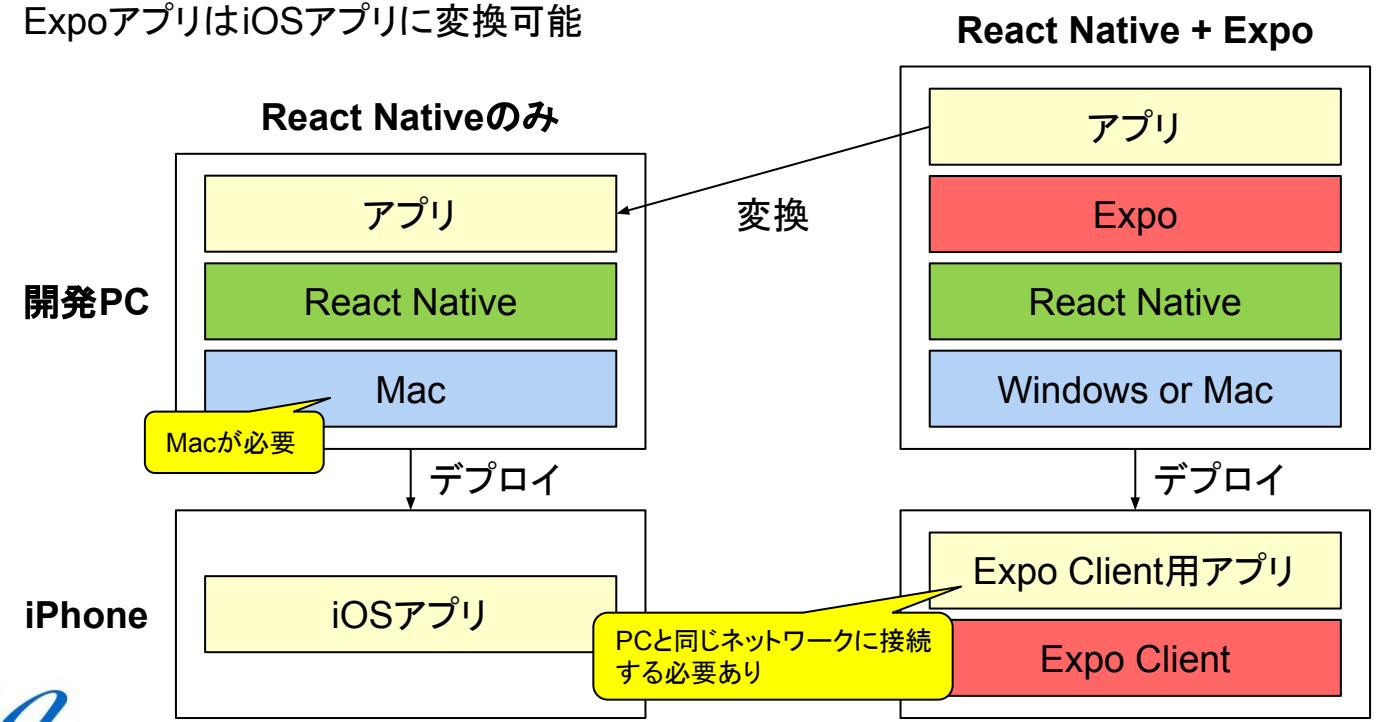


開発技術	言語	フロントエンド
React Native	JavaScript	React（Web）
Flutter	Dart	独自
Xamarin	C#	Xamarin.Forms

開発ツール Expo

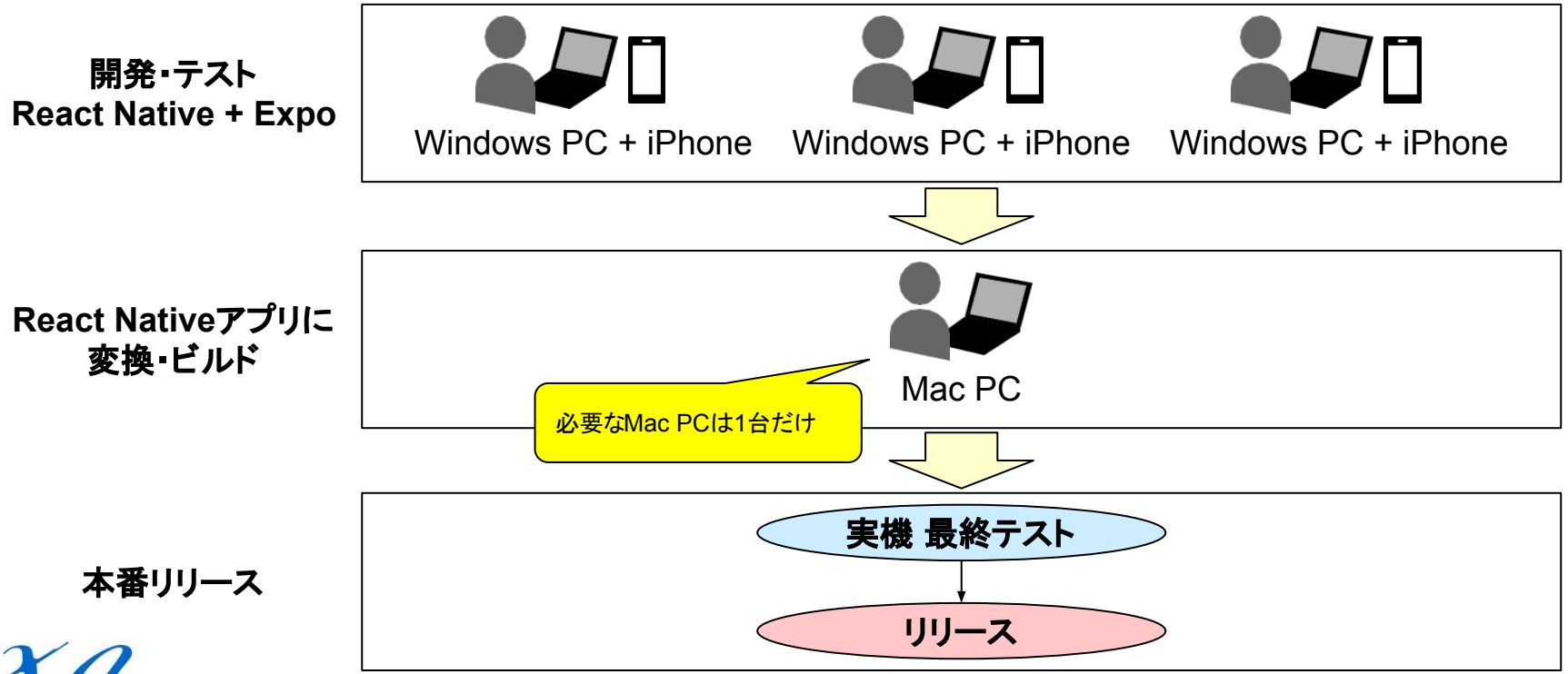
開発における利便性と生産性の向上が可能な開発ツール

- Windows + iPhone実機で開発・テストが可能
- 開発環境構築が容易
- ExpoアプリはiOSアプリに変換可能



開発PCの調達におけるExpoの利点

開発者はWindows PC + iPhoneで開発を行えるため、必要な Mac PCの台数を本番リリースビルド用の 1台のみにできる



検証内容の紹介

技術検証内容

社内の物品を写真付きで管理するサンプルアプリの開発を通じて技術検証を行った

技術検証項目

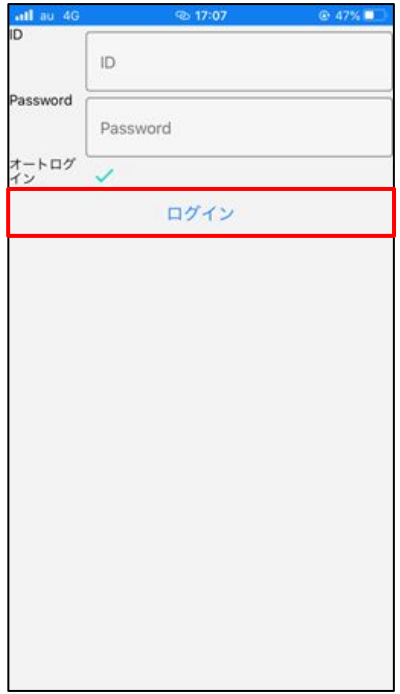
- ❑ 画面表示
- ❑ 画面遷移・状態管理
- ❑ サーバとのデータ送受信
- ❑ 写真撮影・アップロード
- ❑ 自動ログイン認証
- ❑ 端末上のデータ暗号化
- ❑ TypeScript導入

EXA CORPORATION

画面表示

React NativeではJSXと呼ばれるタグでコンポーネントを配置して画面を組み立てる
※ JSXとはReactJSとReact Nativeで使えるJavaScript構文の拡張

```
export default class LogInScreen extends React.Component {
  render() {
    return (
      <SafeAreaView>

        (略)

        <Button title='ログイン'
          onPress={ () => {
            [ボタンがタップされた際の処理
          }}
        />
      </SafeAreaView>
    );
  }
}
```



画面表示

コンポーネントのスタイルを CSSのような書式で指定可能。その際、直接指定することも別ファイルに定義することも可能。

直接指定した場合

```
<TextInput title='ログイン'
  style={{ height: 100 }}
  value={ [TextInputの入力値を保持するオブジェクト] }
/>
```

別ファイルに定義した場合

```
<TextInput title='ログイン'
  style={ StyleSheet.inputStyle }
  value={ [TextInputの入力値を保持するオブジェクト] }
/>
```

StyleSheet.js

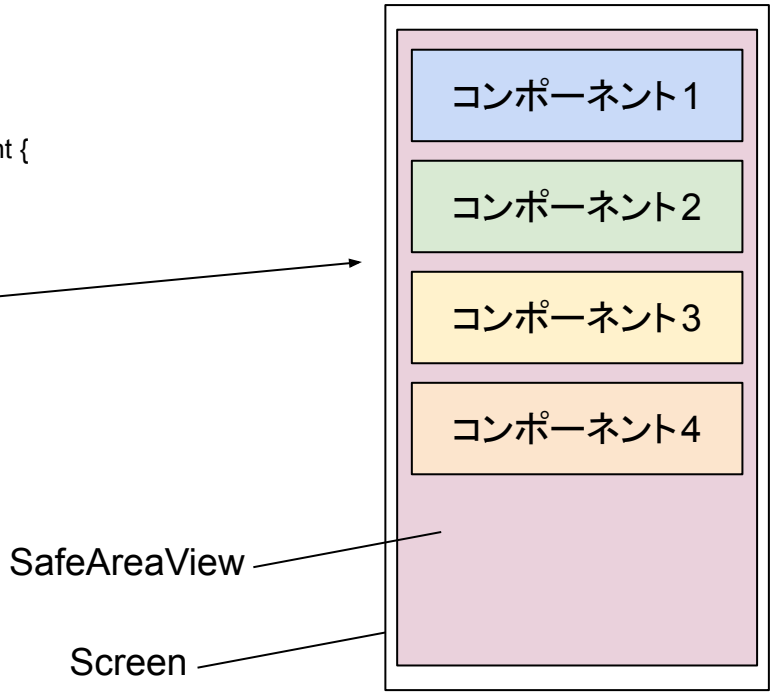
```
import { StyleSheet } from 'react-native';

export default StyleSheet.create({
    inputStyle: {
        height: 100,
    },
    ...
});
```

EXA CORPORATION

画面表示

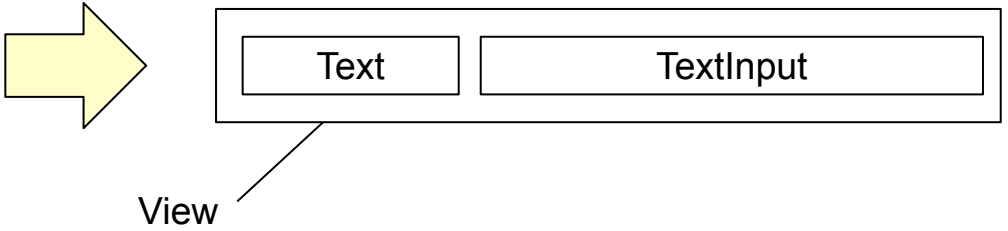
コンポーネントはタグを記載した順番に上から配置される

```
export default class Screen extends React.Component {
  render() {
    return (
      <SafeAreaView>
        <コンポーネント1 />
        <コンポーネント2 />
        <コンポーネント3 />
        <コンポーネント4 />
      </SafeAreaView>
    );
  }
}
```



画面表示

複数のコンポーネントを組み合わせて独自のコンポーネントが作成可能



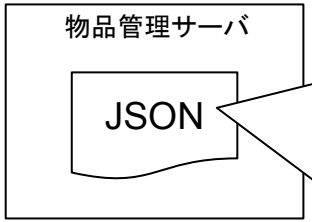

```
<LabeledTextInput
  label='品名'
  onChangeText={ text => {
    [入力値が変更された際の処理]
  }}
  value={ [TextInputの入力値を保持するオブジェクト] }
/>
```

```
export default class LabeledTextImput extends React.Component {
  render() {
    return(
      <View style={{ flexDirection: 'row' }}>
        <Text style={{ flex: 1 }}>{ this.props.label }</Text>
        <View style={{ flex: 4 }}>
          <TextInput { ...this.props } />
        </View>
      </View>
    );
  }
}
```

サーバとのデータ送受信

テスト用のHTTPサーバとして、node.js上で動作するJSON Serverを利用した。
JSON Serverは、データとなるJSONファイルを作成するだけで、参照・更新の REST操作が可能。

Fetch API
REST




```
{
  "items": [
    {
      "id": 1,
      "name": "えんぴつ",
      "type": "文具",
      ...
    },
    {
      "id": 2,
      "name": "消しゴム",
      "type": "文具",
      ...
    }
  ]
}
```

JSON Serverの起動

```
json-server --host [PCのIPアドレス] --watch db.json
```

全件検索

http://[PCのIPアドレス]:3000/items

条件指定検索

http://[PCのIPアドレス]:3000/items?name=消しゴム

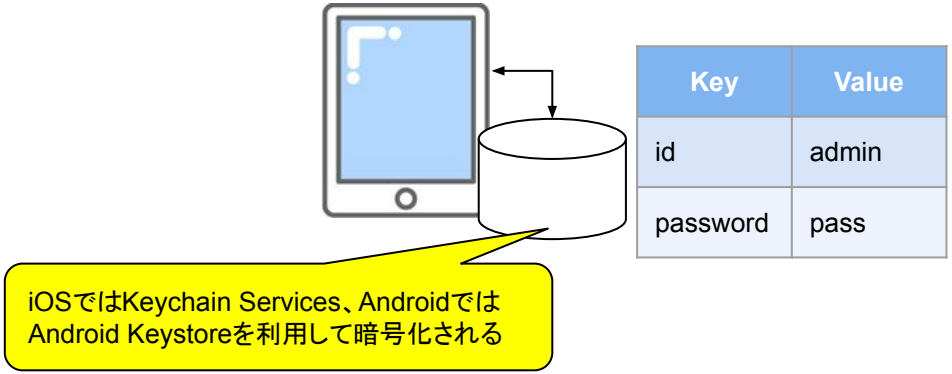
1件追加(POST)

http://[PCのIPアドレス]:3000/items

```
{
  "id": 3,
  "name": "シャーペン",
  "type": "文具",
  ...
}
```

端末上のデータ暗号化

端末上に暗号化したデータを保管するために Expo SecureStoreを利用した。
Expo SecureStoreを利用するとKey-Value形式の値を暗号化して、デバイス上に安全に保管可能。



Key	Value
id	admin
password	pass

値の格納
await SecureStore.setItemAsync([Key], [Value]);

値の参照
await SecureStore.getItemAsync([Key]);

値の削除
await SecureStore.deleteItemAsync([Key]);

各OS標準の暗号化方式を利用するため
暗号化方式の指定は不要

EXA CORPORATION

まとめ

React Native + Expo開発の利点

- 実機によるアプリ実行が簡単に行える
- Mac不要でWindowsのみで開発できるので環境構築が容易
- iPhoneにWi-Fi経由でアプリデプロイができ、コード修正結果の即時確認が可能

React Native、Expoで苦労した点

- 最新のJavaScriptの理解に苦戦
  - 同じ処理を書くのに複数の方法、省略表記がある
- React NativeとExpoのバージョンアップが速い
  - ネットの情報は古いものもあり選別が必要
- サードパーティーのライブラリの活用は必須だが慎重に行う必要あり
  - ライブラリ間の相性によるエラーの調査に時間がかかる
  - 今後のメンテナンスが継続するものを見極める必要がある

実案件適用に向けた課題

- モバイルに最適化した画面設計の検討
- サーバ・クライアントのロジック配置の検討
- アプリの配布方式の検討

EXA CORPORATION

EOF

- 本資料に記載の製品名及び社名は各社の商標もしくは登録商標です。
- 本資料に記載の製品名及び社名には、必ずしも商標表示（ ®,™）を付記していません。

EXA CORPORATION