Lightning Web Componets について

スライド:https://powerninja.github.io/SSCLWC/ja/index.html

リポジトリ:https://github.com/powerninja/SSCLWC

はじめに

今回の質問内容

> ・Lightning Web コンポーネント（＝ LWC？aura と web は何が違う？）

- Lightning Web コンポーネント = LWC です
- Lightning Web Components の簡単な説明と Aura との比較を行います
- 今回 Lightning Web Components の開発を行うための環境構築等の詳細な説明はしません

トピックス

1. Salesforce における画面開発の歴史
2. Lightning Web Components と Aura の違いは？
3. なぜ Lightning Web Components を選択するのか
4. LWC を使用した案件紹介
5. 作成した Lightning Web Components の紹介
6. おまけ

Salesforce における画面開発の歴史

- Visualforce
  - Summer '08 くらい？
- Lightning Aura Components
  - Aura と記載される
  - 2014 くらいに発表？
- Lightning Web Components
  - LWC と記載される
  - Summmer '19

Lightning Web Components と Aura の違いは？

- 共通点
  - Saleforce 上での見た目はほぼ同じ(LDSを標準で使用)
  - classic 未対応
  - HTML と JavaScript を用いた開発
  - 外部 JavaScript ライブラリの使用可能(静的リソースで読み込ませる)
  - Tailwind CSSのようにクラス名でスタイルを与えることができる
    - そのため、LDS でよければ CSS ファイルは不要

Lightning Web Components と Aura の違いは？

- 相違点
  - Aura は開発者コンソールで作成可能だが、LWC は Visual Studio Code が必要(chrome の拡張機能で開発は可能)
  - LWC はユニットテストJestに対応している
  - LWC で対応していない機能がまだある、その場合は Aura を使用する必要あり(一部モバイル対応など)
  - Aura は開発がアーカイブ化されている(サポートはしている)
    Aura 開発リポジトリ
    LWC 開発リポジトリ

Lightning Web Components と Aura の違いは？

- Visualforce との比較
  - 共通点
    - あまりない
  - 相違点
    - コントローラーが LWC は JavaScript(ブラウザ動作),Visualforce との比較は Apex(サーバ動作)
      そのため、LWC のパフォーマンスが良い

Lightning Web Components と Aura の違いは？

- 画面フローとの比較
  - 画面フローで実装可能な場合画面フローを使用することが望ましい
  - ただ、ソースレビューを行いたい場合や、マージリクエストベース開発を行いたい場合は LWC の方がスムーズに開発を行える
  - ブラウザの機能(localStorage など)を使用したい場合は LWC を使用することになる

なぜ Lightning Web Components を選択するのか

- Aura
  - アーカイブ化されている(そのうちプロセスビルダーのように廃止されるかも？)
- Visualforce
  - web 標準の HTML ではない
  - コントローラーが Apex のためパフォーマンスが良くない
- Lightning Web Components
  - 今から学習するのであれば、消去法で LWC が良い

なぜ Lightning Web Components を選択するのか

- 開発コミュニティが活発なため、新機能などに期待できる
- 標準的な JavaScript を使用することができるため、JavaScript の開発経験がある方は開発しやすい
    - そのため、学習コストが低い & Web 開発を行う際に役立つかも？
- LWC 開発時に必要なファイルの数が少なく、初期段階の理解が早い(個人差あり)
- (Aura,Visualforce と比べると)パフォーマンスが良い

なぜ Lightning Web Components を選択するのか

- LWC で作成されるファイル数

```
プロジェクト名(任意で設定可能)
 ├ HTML
 ├ JavaScript
 ├ xml
 ├ css(任意)
 └ Jest フォルダ
 └ プロジェクト名.test.js
```

なぜ Lightning Web Components を選択するのか

- Aura で作成されるファイル数
  - 全部が必要なわけではないが。。。

```
プロジェクト名(任意で設定可能)
 ├ auradoc
 ├ cmp(HTML)
 ├ cmp-meta.xml
 ├ css
 ├ design
 ├ svg
 ├ Controller.js
 ├ Helper.js
 └ Render.js
```

なぜ Lightning Web Components を選択するのか

```
<template>
  <div slds-p-left_xx-large>{hello}</div>
</template>
```

```
import { LightningElement } from 'lwc';

export default class Test extends LightningElement {
  hello = 'Hello,World!'
}
```

```
<?xml version="1.0" encoding="UTF-8"?>
<LightningComponentBundle xmlns="http://soap.sforce.com/2006/04/metadata">
    <apiVersion>56.0</apiVersion>
    <isExposed>true</isExposed>
</LightningComponentBundle>
```

LWC を使用した案件紹介

1. Experience Cloud に B to C と B to B 向けの Web ページを開発

- B to C のページは独自のデザインだったため、CSS を使用していた
- B to B の方は LDS だったが、ソースレビューや GitHub で管理を行いたかったため LWC で開発をおこなった

2. ルックアップ検索条件に表示されるレコードの条件を変更したい

- 画面フローでは実装不可だったため LWC を使用した
- その後、保存ボタンを動的に動かしたり、項目全て入力されたら保存ボタンの色を変えたりと色々した

作成した Lightning Web Components の紹介

- 勉強会の環境に一部デプロイあり

おまけ

- LWC は Salesforce の外でも使用することが可能
- lwc.devという web ページがあり、こちらに詳細が記載されている。
- Heroku や web サーバーにデプロイすることで使用可能
- メインの HTML と JavaScript の書き方はほぼ同じ
- 興味のある方は「OSS LWC」などで検索してみてください

まとめ

- Aura と LWC の違いはさまざまあるが、基本 LWC を使用しよう
- 画面フローで実装可能な場合は、画面フローで実装しよう