

多分きっと今からでもいける PWA超入門！

エムスリーキャリア株式会社

堀 正斉

自己紹介



名前：堀 正斉

所属：エムスリーキャリア（株） フロントエンドエンジニア

言語：CSS(sass), Javascript, Ruby on Rails

夢：5年後に海外のサッカークラブでエンジニアとして働く

5,000,000,000

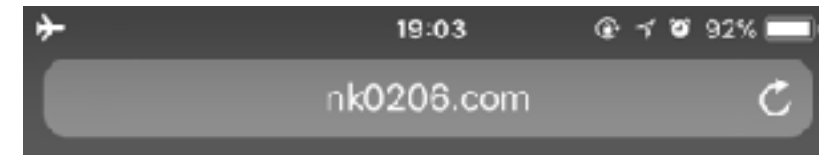
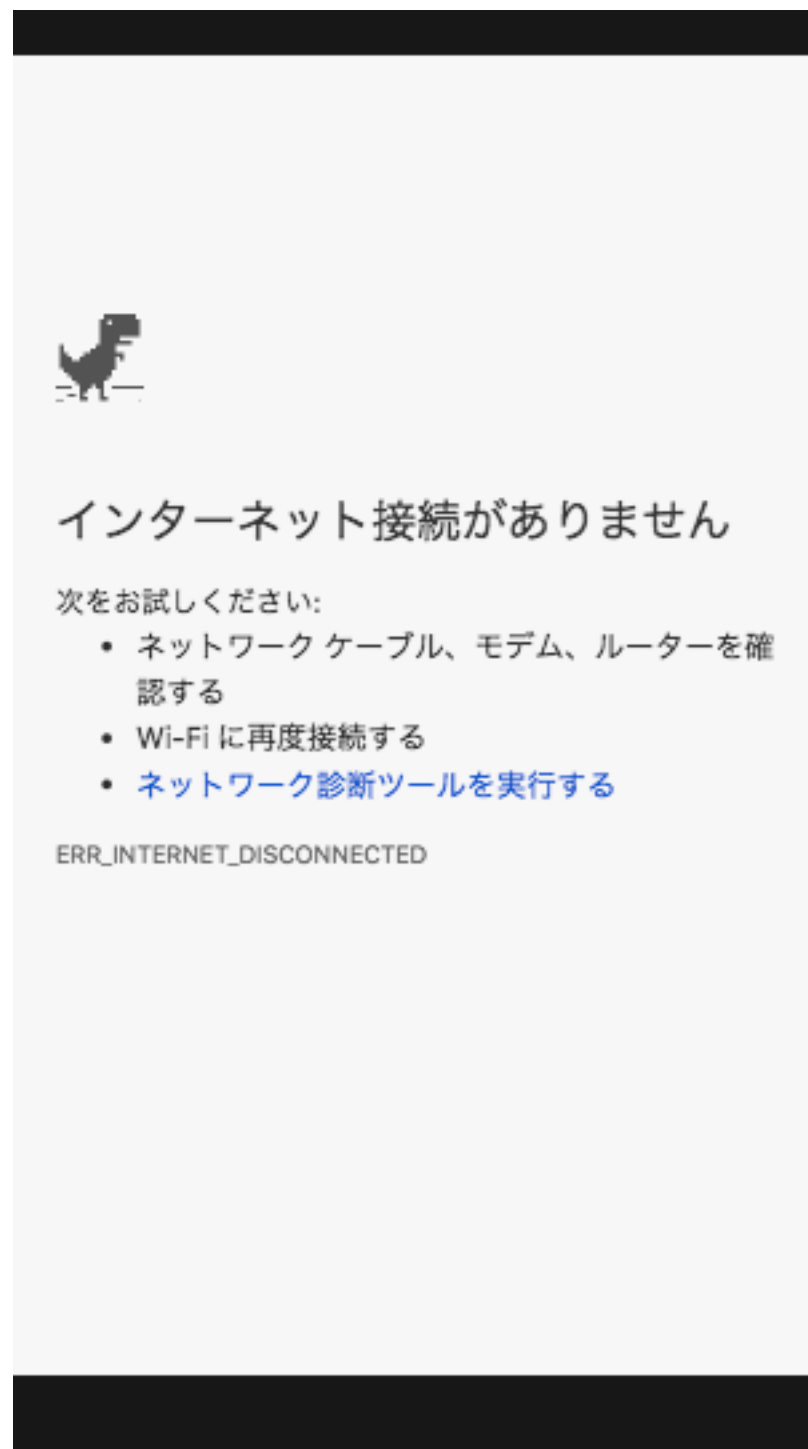
答え：webと繋がっているデバイスの数

多分こんなエコシステムはありません。

But

今までのwebのUIって？

混んでる電車の中、 急に切れる電波。



ページを開けません。iPhoneがインターネットに接続していません。



Oh... Shit.

ネイティブアプリ VS webアプリ

Monthly Unique Visitor

webアプリの勝ち

Average Time Per User

ネイティブアプリの圧勝

ネイティブアプリって
帯域パツパツでも動いてくれるし
プッシュ通知もあるし
ホーム画面に追加されるし
なんだかんだで便利。

webの課題はスピードとエンゲージメント

ここでPWAの登場

主にエンゲージメント面で。スピード面はAMP

What's PWA?

著作権無視ですいません

Googleのおじさん

Progressive web apps, are just websites
that took all the right vitamins.

What's PWA?

個人的には
Webでできることを増やしていこうよ！
みたいな考え方だと捉えています。

PWA化の影響

BEYONDTHERACK

プッシュ通知を導入。

滞在時間 **+72%**

平均訪問時間 **+26%**

再訪問率 **+50%**

appのみの戦略を変更し、webappを開発。

2G回線への対応を行う。

Flipkart



滞在時間 **3倍**

データ量 **1/3**

ホーム画面に追加ボタンを購入画面に進んだユーザーにのみ出すようにしている。

What's service worker?

バックグラウンドで起動するJavascript環境

【主に出来ること】

- ・ オフラインでのアプリケーションへのアクセス
- ・ ネイティブアプリUIをwebアプリに
- ・ プッシュ通知

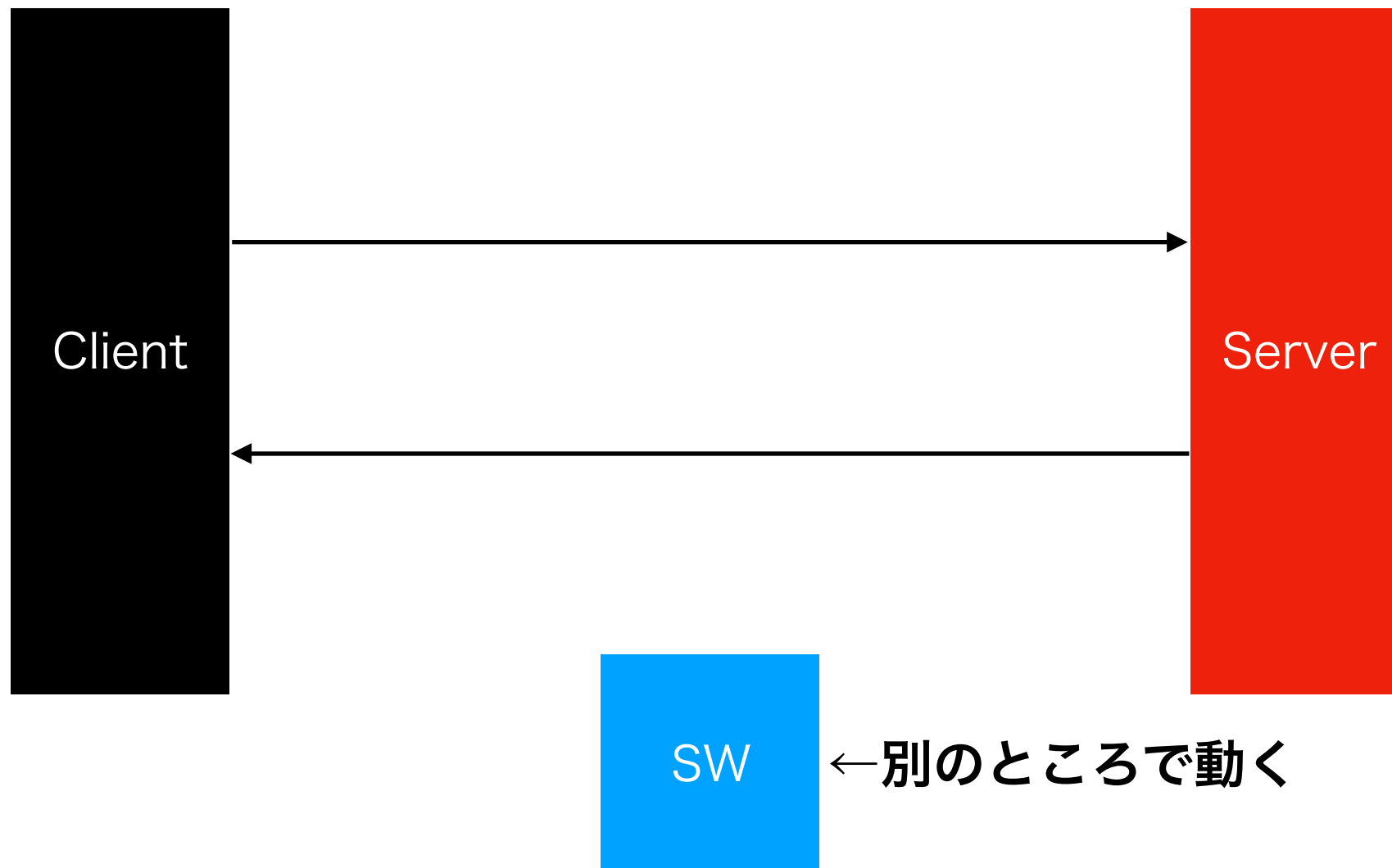
※例：日経電子版TOPページ

<https://r.nikkei.com/>

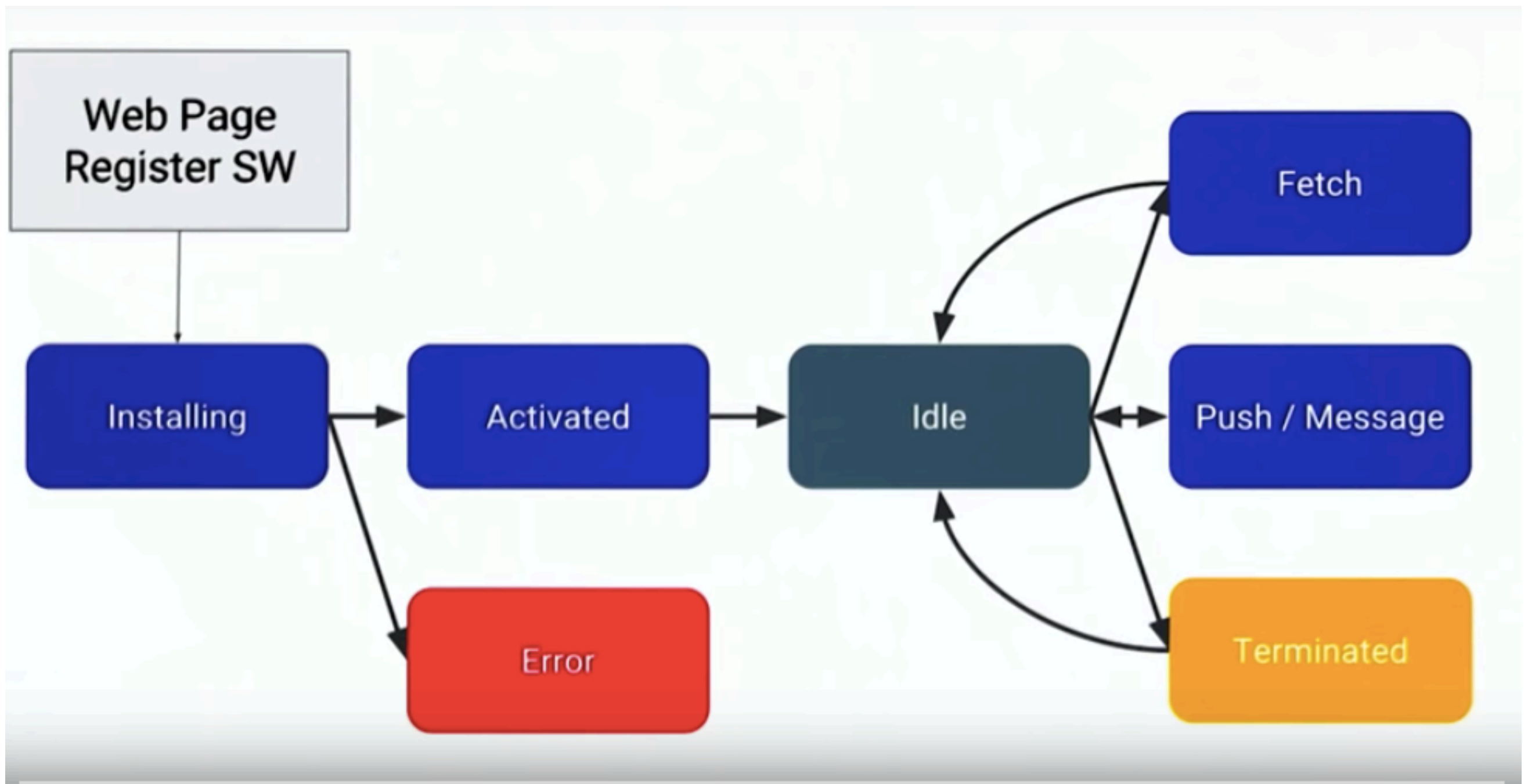
service workerの特徴

- ①webのライフサイクルとは全く別。
→ブラウザを閉じていても動いている
- ②DOMに直接アクセスできない
- ③ネットワークプロキシのようなもので、
ネットワークリクエストをコントロール出来る

service workerのライフサイクル (1)



service workerのライフサイクル (2)



service workerの注意点



①対応ブラウザはChromeとFirefox
SafariとEdgeは開発中。

※Safariについてはv11.3で対応、
Edgeも次のversionで対応するぽい
参考

<https://caniuse.com/#search=service>

<https://jakearchibald.github.io/isserviceworkerready/>

localhost



②dev環境とhttps通信の環境下でないと動作しない。
→接続へのハイジャック、改ざんなどが行われてしまう
可能性があるため、これを防ぐため。

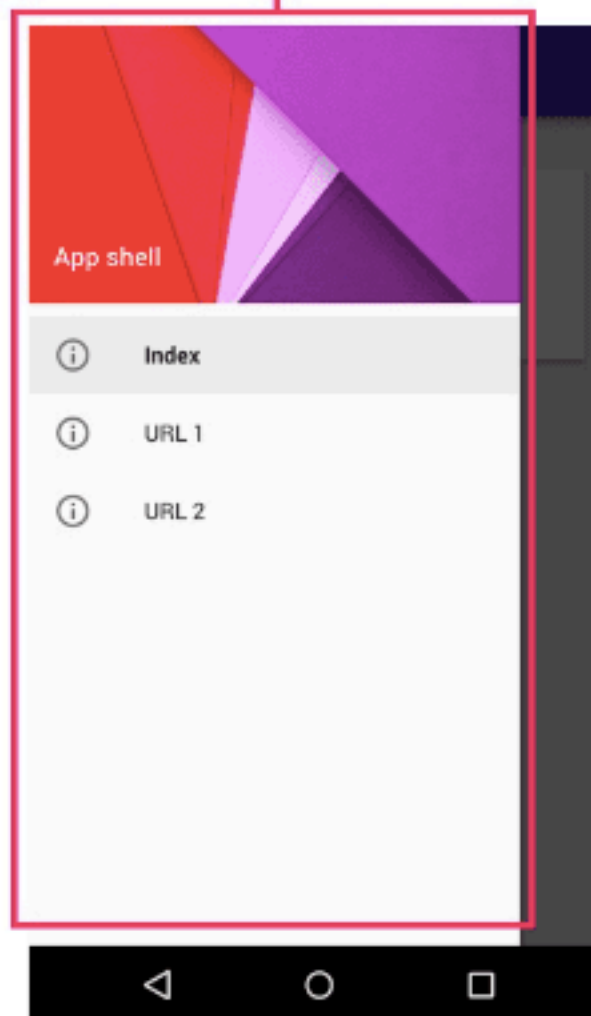
AppShellモデルとは？

application shell



Cached shell loads **instantly** on repeat visits.

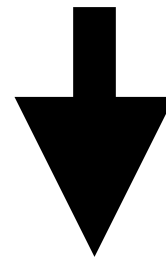
content



Dynamic content then populates the view

AppShellモデルとは？

アプリケーション シェル (App Shell) アーキテクチャは、ネイティブ アプリのように瞬時に、そして確実にユーザーの画面に読み込める Progressive Web App を構築する方法の 1 つ。



Shell→UIが最低限機能するためのHTML/CSS/Javascriptとかオフラインで使えるようにキャッシュしておくことで、ユーザーが同じページに再度アクセスした際に高いパフォーマンスを出せる。ユーザーがアクセスするたびにネットワークから全て読み込まれる訳ではなく、必要なコンテンツだけが読み込まれる。

デモ（普通のページ）

The screenshot shows a web browser window displaying the R-STORE website. The browser's address bar shows the URL <https://www.r-store.jp>. The website's header includes the R-STORE logo, navigation links, and contact information (TEL 03-6421-7595). A sidebar on the left contains search filters for RENT and BUY properties, with options for language (English), category, location, and area. The main content area features a large banner for the Kanakura branch opening, followed by a 'PICK UP' section displaying four property images. A Wi-Fi network selection menu is open on the right side of the browser window, showing various available networks.

QuickTime Player ファイル 編集 表示 ウィンドウ ヘルプ

100% 22:36

Wi-Fi ネットワークを検索中...
Wi-Fiを切にする

- CCAF7B23F53F
- auhome_acx3QA
- auhome_acx3QA-A
- auhome_acx3QA-W
- Buffalo-A-D7E0
- Buffalo-G-D7E0
- corega 3
- HP-Print-61-ENVY 4500 serl...
- planexuser
- pr500k-8393aa-1
- pr500k-8393aa-2
- pr500k-8393aa-3

ほかのネットワークに接続...
ネットワークを作成...
"ネットワーク"環境設定を開く...

R-STORE
REAL ESTATE BOUTIQUE

物件オーナー様へ | 管理会社様へ

TEL 03-6421-7595 9:00-18:00 (土休日・年末)

R-STOREとは? | 借りたい | 買いたい | その他いろいろ | 採用

SEARCH 物件検索

賃貸物件を探す RENT | 売買物件を探す BUY

ENGLISH

カテゴリ (複数選択可)
指定しない >>

沿線 (複数選択可)
選択済み
地下鉄から探す >>
私鉄から探す >>

エリア (複数選択可)
指定しない >>

家賃
指定しない >> 8.0万円以内

面積
20㎡以上 | 指定しない >>

駅徒歩分
指定しない >>

海辺の街に暮らす。鎌倉支店OPEN!

R-STORE Kanakura

PICK UP 新着物件一覧を見る

デモ（オフラインアプリ）


Chrome ファイル 編集 表示 履歴 ブックマーク ユーザー ウィンドウ ヘルプ

chrome x かんたん x Intro to x SW 開発 x Service x ナンズ x フォーム x はじめ x git.com x https:// x develop x Weame x kyuku








← → ↻ 📍 local host:8887 ☆ ⋮

Weather PWA


New York, NY
Fri Apr 01 2016 03:17:44 GMT+0900 (JST)
Mostly Cloudy

 **69°F**








Feels like: 69°F
Precipitation: 0%
Humidity: 51%
Wind: 11 mph 188°

Fri	Sat	Sun	Mon	Tue	Wed	Thu
						
70° 45°	72° 60°	62° 48°	51° 34°	59° 34°	48° 29°	47° 26°

Chicago, IL
Fri Apr 01 2016 03:17:36 GMT+0900 (JST)
Drizzle

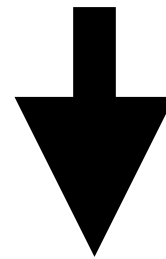
 **64°F**

Feels like: 64°F
Precipitation: 68%
Humidity: 73%
Wind: 11 mph 201°

Fri	Sat	Sun	Mon	Tue	Wed	Thu
						
65° 47°	48° 40°	44° 32°	53° 29°	52° 33°	44° 31°	56° 40°

なぜオフラインでもページが表示された？

Service Workerを通じてcache APIを利用
リソースを端末にダウンロードしておく
キャッシュをアプリ側にレスポンスとして返す



オフラインでも、キャッシュしたデータを用いて画面表示ができた！

ソースコードはこんな感じ。

①register

```
//serviceworkerを登録するためのコード
if ('serviceWorker' in navigator) {
  navigator.serviceWorker
    //serviceworkerのスコープはrootに
    .register('/service-worker.js')
    .then(function() {
      //consoleにメッセージを出す
      console.log('Service Worker Registered');
    });
}
```

Service Workerのコントロール下に
置くページを定義する。

②install

```
self.addEventListener('install', function(e) {  
  console.log('[ServiceWorker] Install');  
  e.waitUntil(  
    // まず、<cache.open>でキャッシュを開き、キャッシュ名を指定する。  
    // キャッシュ名を指定すると、ファイルをバージョンアップしたり、  
    // アプリケーションシェルからデータを分離したりすることができる。  
    caches.open(cacheName).then(function(cache) {  
      console.log('[ServiceWorker] Caching app shell');  
      // <cache.addAll>は、URLのリストを取得し、キャッシュに追加する。  
      return cache.addAll(filesToCache);  
    })  
  );  
});
```

主に静的ファイルをcacheに追加。
ここでフレームワークとかも読み込んでも良さそう。

③activate

```
// 新規インストール時には特に関係ない。cachenameが更新されて、  
// SWが更新されたと判断されてinstallイベントが発火して、新しいcachenameで保存される。  
  
self.addEventListener('activate', function(e) {  
  console.log('[ServiceWorker] Activate');  
  e.waitUntil(  
    // cache.keys()によって保存されているキャッシュの名前が列挙されて、  
    // ホワイトリストの中にない古いcacheが削除される  
    caches.keys().then(function(keyList) {  
      return Promise.all(keyList.map(function(key) {  
        // cacheName と keyが不一致、dataCacheNameとkeyが不一致の  
        // 時には古いキャッシュを削除する。  
        if (key !== cacheName && key !== dataCacheName) {  
          console.log('[ServiceWorker] Removing old cache', key);  
          return caches.delete(key);  
        }  
      }));  
    }));  
  });  
});
```

Service Workerを更新するタイミングで発火。
古いキャッシュは削除して、差分を更新する。

④fetch -1

```
// リクエストを傍受し、URL の先頭が Weather API のアドレスかどうかを確認。  
self.addEventListener('fetch', function(e) {  
  if (e.request.url.startsWith(weatherAPIUrlBase)) {  
    e.respondWith(  
      // URL の先頭が Weather API のアドレスであれば、fetch を使用してリクエストを行う。  
      // 応答が返されたらキャッシュを開き、応答をコピーして格納した後、リクエストの送信元に応答を返す。  
      fetch(e.request)  
        .then(function(response) {  
          return caches.open(dataCacheName).then(function(cache) {  
            cache.put(e.request.url, response.clone());  
            console.log('[ServiceWorker] Fetched & Cached', e.request.url);  
            return response;  
          });  
        })  
    );  
  } else {
```

Service Worker がブラウザをコントロールしている時に
リソースのリクエストが発生すると fetch イベントが発火。

④fetch -2

```
    } else {  
      e.respondWith(  
        // caches.match() を使用して、fetch イベントを呼び出したウェブリクエストを評価し、  
        // キャッシュからのデータが利用可能かどうかを確認。  
        // 次に、キャッシュ データで応答するか、fetch を使用して ネットワークからコピーを取得  
        // そして、e.respondWith() を使用して ウェブページに response を返す。  
        caches.match(e.request).then(function(response) {  
          console.log('[ServiceWorker] Fetch Only', e.request.url);  
          return response || fetch(e.request);  
        })  
      );  
    }  
  });
```

fetch イベントで何もしなければ、普通にネットワーク経由で
リクエストが処理される。

おまけ

manifest jsonの紹介

What's manifest json?

ネイティブアプリのUIをwebアプリにも提供する

【主に出来ること】

- ・ ネイティブアプリUIをwebアプリに
- ・ ホーム画面に追加

例1 : twitter lite

<https://lite.twitter.com/>

例2 : wego

<https://www.wegotravel.jp/>

manifest jsonのサンプル

HTML

```
<link rel="manifest" href="manifest.json">
```

JSON 一応、service workerも必要

```
{
  "name": "name",
  "short_name": "name",
  "icons": [
    {"src": "img/name_36.png", "sizes": "36x36", "type": "image/png"},
    {"src": "img/name_48.png", "sizes": "48x48", "type": "image/png"},
    {"src": "img/name_72.png", "sizes": "72x72", "type": "image/png"},
    {"src": "img/name_96.png", "sizes": "96x96", "type": "image/png"},
    {"src": "img/name_144.png", "sizes": "144x144", "type": "image/png"},
    {"src": "img/name_192.png", "sizes": "192x192", "type": "image/png"},
    {"src": "img/name_256.png", "sizes": "256x256", "type": "image/png"},
    {"src": "img/name_512.png", "sizes": "512x512", "type": "image/png"}],
  "display": "standalone",
  "start_url": "/?homescreen",
  "display": "standalone"
}
```

こんな感じのファイルを追加するだけで

以下のようなことができる

プロンプトが出る

ホーム画面に追加される

ネイティブアプリっぽいUI



まとめ

- ①AMPとの併用で阿部寛みたいなページロード速度を目指してみたい。
- ②ES6のPromiseについてはちゃんと理解しないとダメ。。
(頑張ります・・・)
- ③アプリがオワコンだとは思わないけど、webで出来ることどんどん増えてく。
(なので境界線は曖昧になっていく気もする)

参考サイト

- ①
- ②
- ③
- ④
- ⑤
- ⑥

#PWA深掘りしてる人と繋がりたい

おしまい