

Web基礎

Webのおさらい

- バックエンド
 - データベースと通信してブラウザに返す
 - 適材適所で Ruby, Python, Node, PHP, Java, Go...
 - サーバーサイド、サーバー側の言語
- フロントエンド
 - 見た目を構成
 - HTML, CSS, Javascriptを返す
 - ブラウザで動く

ハンズオンで主に学ぶもの

- フロントエンド
- 主にJavascript
- フレームワークとして Next.js

フロントエンドって重要ななの？

- HTML,CSSで綺麗な見た目のWebサイトなだけではダメなの？
- なぜフレームワークを使うの？

→少しだけ歴史の話を

静的なHTMLから動的なHTMLへ

見るだけのWebサイトが生きたWebサイトになった

- Ajax (Google Map)
 - javascript側でリクエストを行い、ページ遷移なくレンダリング
- jQuery
 - ボタンクリックやAjax処理など、簡単に書けるようになった
 - 大きくなるにつれ複雑になる為、Angular.jsなどに置き換わっていく

デバイスや技術の進化

さらに豪華なUIに

- **WebGL**
 - <https://webglsamples.org/aquarium/aquarium.html>
- **AMP**
 - キャッシュや非同期処理のみ許容など徹底した速度改善

デバイスや技術の進化

- シングルページアプリケーション
 - 再読み込みがなくページ遷移のない、アプリのようなWebページ
- データバインディング(リアクティブプログラミング)
 - Angular.jsの台頭で簡単、かつルールのあるレンダリングが可能に！
 - 双方向データバインディングの重さやデータフローの明示性が問題になり単方向のReactや両対応のAngular2やVue.jsが出てくる
 - 機能の分離や保守性の為にコンポーネント指向という概念も出てくる

クロスプラットフォーム

- Flutter, React Native, Xamarin
- Unity
- Monaca, PhoneGap

それぞれ違うけど

- Web技術をアプリに生かしたり
- 各プラットフォーム向けにビルド可能だったり
- Webサイト自体をアプリ化したり
- <https://worldflipper.jp/demo/>

Webサイトをアプリ化

- **PWA アプリと同等を目指す機能群**
 - **Push通知**
 - **オフラインモード**
 - **ホーム画面上でのアイコンの表示**
 - **カメラやマイク、センサー類、BluetoothやNFCなどHWアクセス**
- **WebAssembly**
 - **ブラウザ上でバイナリを動かす (Goなど、コンパイルして利用)**
- **WebGPU support**

フロントエンドって重要(まあどれも重要)

- 高機能化、高速化、複雑化
- トレンドを加味して、時代にあった技術選定と利用するのがエンジニアの仕事

Javascript超入門

デバッグを実践

Javascriptを知るためにはHTMLもCSSも学ぶ必要がある

それぞれChrome dev toolsを使って試してみよう

HTMLをブラウザで表示してみよう

1. Chromeで適当なページを開く (<https://www.google.com/>)
2. ctrl + shift + i でChrome dev toolsを開く (macはcmd + option + i)
3. 適当な要素(Googleの画像とか)の上で右クリック、検証
4. Elementsタブに表示されたhtmlの適当な要素の上で右クリック、Edit as HTML

```
<div>
```

```
  <h1>HTMLをブラウザで表示してみよう</h1>
```

```
  <p>これが表示されればOK!</p>
```

```
  <p>アラートボタンも押してみよう</p>
```

```
  <button onclick="alert('hello world')">アラート</button>
```

```
</div>
```

HTMLをブラウザで表示してみよう

これが表示されればOK!

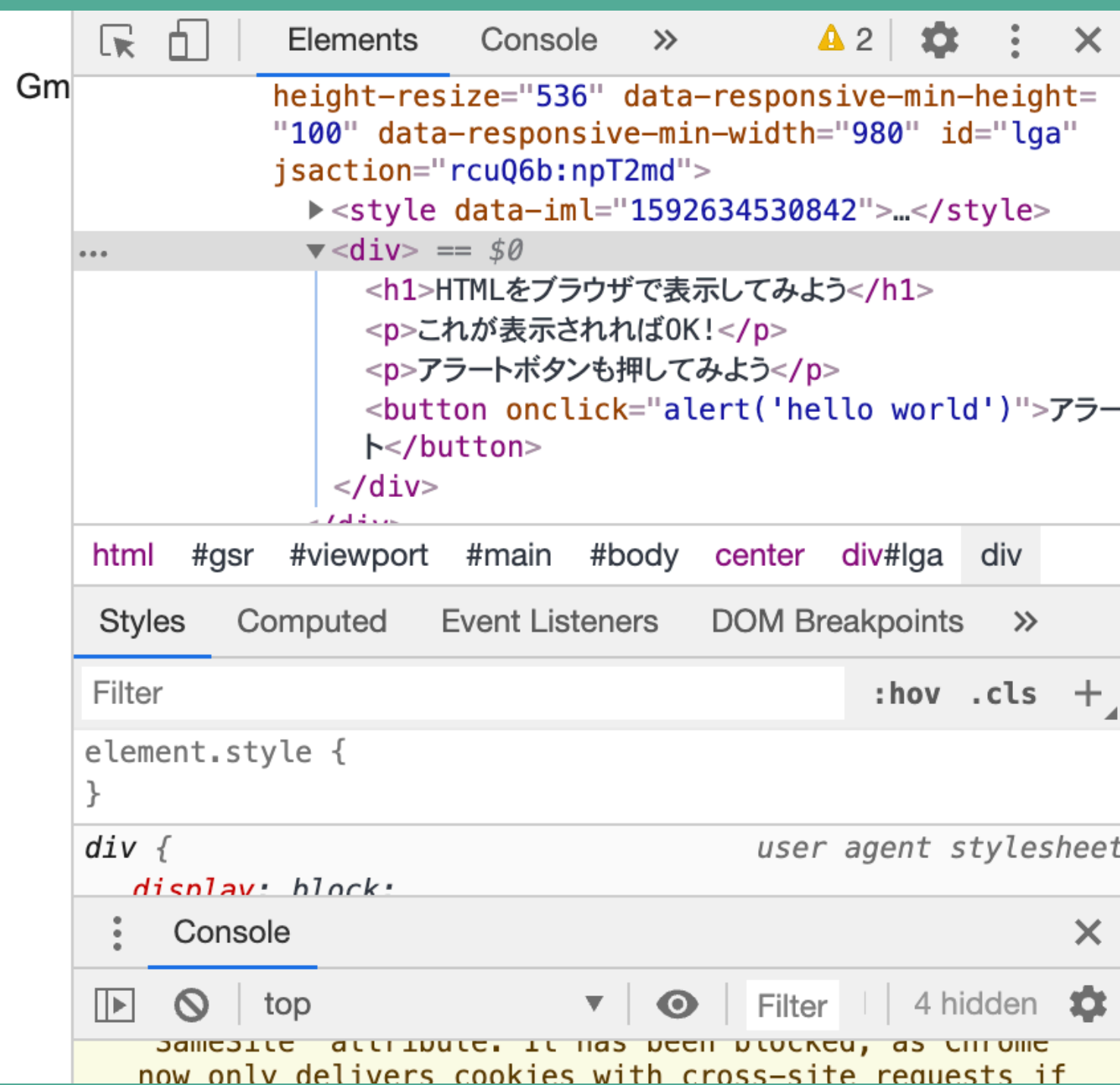
アラートボタンも押してみよう

アラート



Google 検索

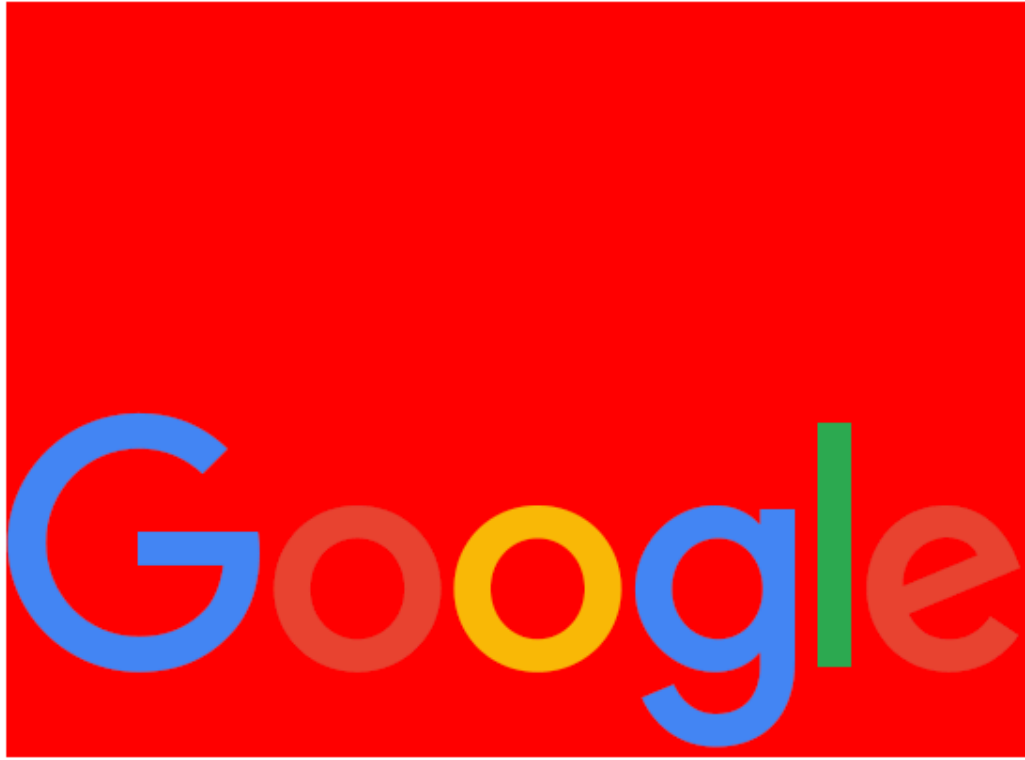
I'm Feeling Lucky



CSSを使ってStyleを変えてみよう

1. Chromeで適当なページを開く (<https://www.google.com/>)
2. ctrl + shift + i でChrome dev toolsを開く (macはcmd + option + i)
3. 適当な要素(Googleの画像とか)の上で右クリック、検証
4. Elementsタブ、下のStyleタブのelement.styleを書き換え

```
element.style {  
  background-color: red;  
}
```



Google 検索

I'm Feeling Lucky

Elements Console Sources >> ⚙️ ⋮ ✕

```
jsaction="rcuQ6b:npT2md">
  > <style data-impl="1592634490834">...</style>
...
   == $0
  </div>
```

... #gsr #viewport #main #body center #lga img#hplogo

Styles Computed Event Listeners DOM Breakpoints >>

Filter :hov .cls +

```
element.style {
  padding-top: 109px;
  background-color: red;
}
```



```
img[Attributes Style] {
  height: 92px;
  width: 272px;
}
```

Inherited from center

Javascriptで計算結果をconsoleに表示しよう

1. Chromeで適当なページを開く (<https://www.google.com/>とか)
2. ctrl + shift + i でChrome dev toolsを開く (macはcmd + option + i)
3. Consoleタブで以下jsを貼り付け

```
var sum = function(a, b) {  
    return a + b;  
};  
var result = sum(1, 2);  
console.log(result);
```





Elements

Console

Sources


»





top

▼



Filter

3 hidden



```
> var sum = function(a, b) {  
    return a + b;  
};  
var result = sum(1, 2);  
console.log(result);
```

3

VM2433:5

← undefined

> |

Javascriptにはいろんな種類がある

- ES5 Javascript
- ES6 Javascript
 - ES5の上位互換、ブラウザによってはサポートされていない
将来の仕様
- TypeScript
 - 型やMicrosoft独自の仕様を含むES6の上位互換、トランス
パイル必須

Javascript ES6で計算結果をconsoleに表示しよう

1. Chromeで適当なページを開く (<https://www.google.com/>)

2. **command + option + i** でChrome dev toolsを開く
(windowsは**ctrl + shift + i**)

3. Consoleタブで以下jsを貼り付け

```
const sum = (a, b) => a + b;  
const result = sum(1, 2);  
console.log(result);
```

ES6検索のコツ

- ノイズ防止のためにES6でググる
- 古い記事は見ない
- jQueryが見えたら、みなかったことにする
- 調査方法がわからなかったら、メンターに聞く！！