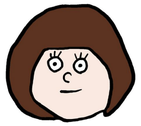
Google Developers

DevFest Tokyo 2021 Web Session

今まであ書いてたアレ、これからはこう書けそう



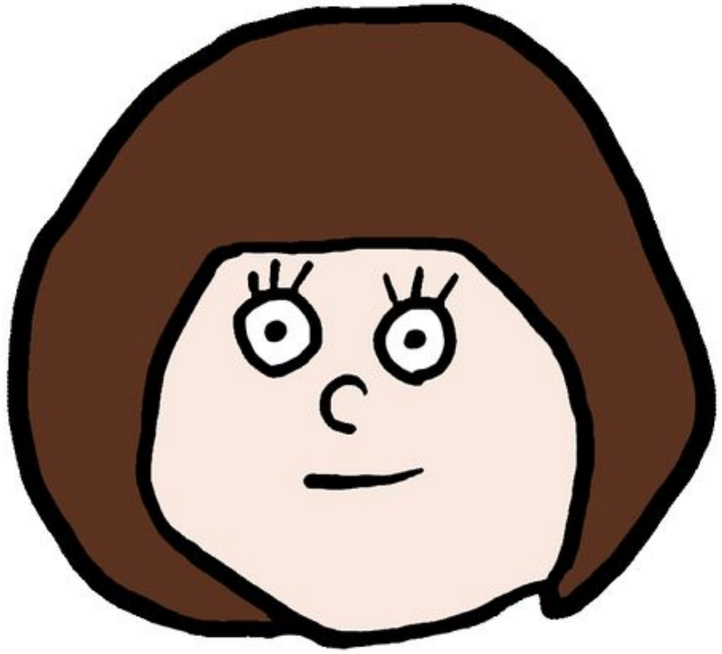
よしこ
Google Developer Experts for Web
@yoshiko_pg

About Speaker

よしこ - @yoshiko_pg

株式会社ナレッジワーク
フロントエンドエンジニア

2021年10月からWebのGDEになりました。
SPAでGUIツールを作るのが好きです。



About Session

このセッションについて

Webフロントエンド開発で頻出するパターンの中で、専用の仕様がなかったために
既存の仕様を使って工夫して実現していたようなものがありました。

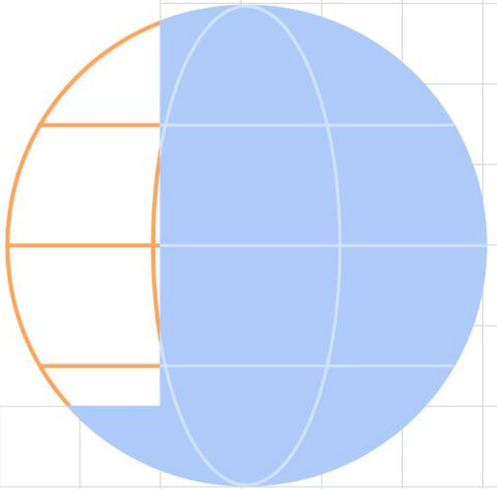
日々新しく提案・実装されていくWebの仕様の中で、モダンブラウザでの実装も進んでおり、
利用したくなる機会も多そうなものについてbefore/afterのコードを軸に紹介していきます。

実際に使えるかどうかは各アプリケーションのサポート環境次第ですが、
IE11のサポート終了も近付いており、一般的なサポート環境の水準にも変化が予想されるので
状況がマッチしたときに引き出せる、知識の引き出しのひとつになれば幸いです。

Experts

サポート環境次第で今から使えそうなもの

最新モダンブラウザで実装済み



CSS Property

gap

CSS Box Alignment Module Level 3

特定のレイアウト内で隣接するボックス間の間隔を指定できる
（間隔なので外側にはみ出ない）
flex, grid, multi-column で利用可能
row-gap column-gapのshorthand

```
/* before */
/* childにmarginをつけ、外側の不要なmarginを
   parentのnegative marginで相殺している */
.parent {
  display: flex;
  flex-wrap: wrap;
  margin: -10px;
}
.child {
  flex: 0 0 100px;
  margin: 10px;
}

/* after */
.parent {
  display: flex;
  flex-wrap: wrap;
  gap: 10px;
}
.child {
  flex: 0 0 100px;
}
```

Experts

Chrome	Firefox	Safari	Edge	Android Chrome	iOS Safari
96	94	15	96	96	15

CSS Property

aspect-ratio

CSS Box Sizing Module Level 4

要素のアスペクト比率を指定できる。値は「幅 / 高さ」で記述 (ratio)

（余談）画像などのLayout Shiftに対するブラウザでの改善のための提案にも登場していたりしました。
標準化の経緯が面白いです。詳しい日本語記事：
https://www.mizdra.net/entry/2020/05/31/192613

Experts

Chrome	Firefox	Safari	Edge	Android Chrome	iOS Safari
96	94	15	96	96	15

```
/* before */
/* paddingの100%が親の横幅と等しいことを利用して
   疑似要素のpadding-topで4:3に比率固定している。
   高さベースの比率固定はできない。
   子要素を置きたい場合はabsoluteなどで配置する */
.box {
}
.box::before {
  content: '';
  display: block;
  padding-top: 75%;
}

/* after */
/* 高さベースの比率固定も可能（heightを指定） */
.box {
  aspect-ratio: 4 / 3;
}
```

CSS Property

Scroll Snap

CSS Scroll Snap Module Level 1

カルーセルやLPなどで求められることのある、子要素の始点や中央にスクロールのポイントを合わせる実装がCSSだけで可能に
親要素に scroll-snap-type, 子要素に scroll-snap-alignを指定

```
/* before */
/* 何らかのJS実装を利用していたのではないでしょうか */

/* after */
.parent {
  overflow: auto;
  scroll-snap-type: y;
  /* y mandatory と指定するとさらに厳密にsnapする */
}
.child-scrollable-item {
  scroll-snap-align: start;
}
```

Experts

Chrome	Firefox	Safari	Edge	Android Chrome	iOS Safari
96	94	15	96	96	15

Web API

.scrollIntoView()

CSSOM View Module

所属するスクロールコンテナを知らなくても、子要素側からスクロール位置を操作できる
{ behavior: 'smooth' } を渡すことでアニメーションも可能だが、こちらは Safari 15には未実装

```
// before
const targetElement =
  window.document.getElementById('target')
const containerElement =
  window.document.getElementById('container')

containerElement.scrollTo({
  top: targetElement.offsetTop
});


// after
const targetElement =
  window.document.getElementById('target')

targetElement.scrollIntoView()
```

Experts

Chrome	Firefox	Safari	Edge	Android Chrome	iOS Safari
96	94	15	96	96	15

JavaScript

Optional Chaining

ECMAScript 2020

nullやundefinedに . を繋げてしまうとエラーになるが、?. を繋げることで「nullかundefinedならundefinedが返り、そうでなければ値が返る」挙動にできる

```
// before
const name = user ? user.name : undefined

// after
const name = user?.name
```

Experts

Chrome	Firefox	Safari	Edge	Android Chrome	iOS Safari
96	94	15	96	96	15

JavaScript

Nullish Coalescing

ECMAScript 2020

|| は左側がfalsyの場合に右側を返すが、??は左側がnullish（nullまたはundefined）の場合に右側を返す。0や空文字やfalseを選り分けたい場合に便利

```
// before
// 0が入っていてもempty判定されてしまうバグがある
const value = nullableNum || 'num is empty'

// after
// nullishのときだけ正しくempty判定される
const value = nullableNum ?? 'num is empty'
```

Experts

Chrome	Firefox	Safari	Edge	Android Chrome	iOS Safari
96	94	15	96	96	15

JavaScript

.replaceAll()

ECMAScript 2021

今までString.prototype.replaceでマッチするすべての対象を置換したい場合には正規表現にした上でgオプションを付けなければならなかったが、replaceAllを使うことで文字列のまま渡せる

```
const str = 'BANANA'

// before
str.replace('A', 'B')  // BBNANA
str.replace(/A/g, 'B')  // BBNBNB

// after
str.replaceAll('A', 'B')  // BBNBNB
```

Experts

Chrome	Firefox	Safari	Edge	Android Chrome	iOS Safari
96	94	15	96	96	15

番外: TypeScript

type Modifiers

TypeScript 4.5

個々のnamed importに個別にtype修飾子を付けられるようになったので、型のimportと値のimportをひとつのimport文で書けるようになった

```
// before
import type { BaseType } from "./some-module";
import { someFunc } from "./some-module";


// after
import {
  type BaseType,
  someFunc,
} from "./some-module"
```

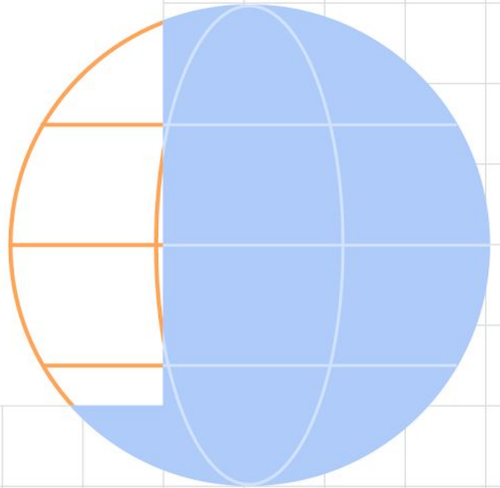
Experts

Web標準じゃないけど、利用頻度が高そうなので
Web Frontend繋がりでご紹介

Experts

今後使えるようになりそうなもの

各モダンブラウザで実装中



CSS Property

translate, scale, rotate

CSS Transforms Module Level 2

これら3つのアクションがプロパティとして指定できるようになるので、個別に上書きやアニメーションができるようになる
Chromeにもフラグ付きでは実装されているので有効化に期待

```
/* before */
.big-box {
  transform: scale(2);
}

.bigBox.downward {
  transform: scale(2) rotate(180deg);
}


/* after */
.big-box {
  scale: 2;
}

.big-box.downward {
  rotate: 180deg;
}
```

Experts

Chrome	Firefox	Safari	Edge	Android Chrome	iOS Safari
96	94	15	96	96	15

CSS Property

overscroll-behavior

CSS Overscroll Behavior Module Level 1

スクロール可能な領域同士が重なっている場合、スクロール操作中の領域のスクロール終点まで辿り着いたときに後ろのスクロール領域がスクロールされはじめてしまう問題をCSSで解消できる

```
/* before */
/* スクロールを止めたい領域のoverflowに一時的に
   hiddenを指定することでスクロールできなくしている。
   スクロール位置がリセットされてしまう問題がある */
.overwrap-scroll-area {
  overflow-y: auto;
}
body {
  overflow: hidden;
}


/* after */
.overwrap-scroll-area {
  overflow-y: auto;
  overscroll-behavior-y: contain;
}
```

Experts

Chrome	Firefox	Safari	Edge	Android Chrome	iOS Safari
96	94	15	96	96	15

CSS Property

accent-color

CSS Basic User Interface Module Level 4

一部のform control要素のアクセントカラーを変更できる
アクセントカラーの上に重なる要素（チェックボックスのチェックなど）の色はブラウザがコントラストを保ってくれる

```
/* before */
/* デフォルトのcheckboxは非表示にし、自分でスタイリングしたcheckboxの見た目の要素を表示していた */
.checkbox input[type=checkbox] {
  visibility: hidden;
}
.checkbox span {
  /* 好きな色のcheckboxな見た目の実装 */
}


/* after */
input[type=checkbox] {
  accent-color: pink;
}
```

Experts

Chrome	Firefox	Safari	Edge	Android Chrome	iOS Safari
96	94	15	96	96	15

CSS Selector

:focus-visible

Selectors Level 4

フォーカスの場所を伝えたほうがよいとブラウザが判断した場合に適用される。たとえば、クリックしたときはoutlineが出ないがタブキーで移動したときはoutlineが出る、という振る舞いを期待できる

```
/* before */
/* クリックしたときにも線が出てしまい、うるさい。
   が、消すとタブキーでの移動が困難になってしまう */
:focus {
  outline: 2px solid blue;
}


/* after */
/* クリックしたときには線が表示されず、
   タブキーでフォーカス移動しているときだけ表示される */
:focus-visible {
  outline: 2px solid blue;
}
```

Experts

Chrome	Firefox	Safari	Edge	Android Chrome	iOS Safari
96	94	15	96	96	15

JavaScript

.at()

ECMAScript 2022

配列の末尾の要素を取得したいとき、今まではlength から1を引いた添字で取得していたが、at()に-1を渡すことで末尾の要素を取得できるようになる

```
// before
array[array.length - 1]

// after
array.at(-1)
```

Experts

Chrome	Firefox	Safari	Edge	Android Chrome	iOS Safari
96	94	15	96	96	15

Web API

userAgentData

User-Agent Client Hints

UserAgentをobject形式で取得可能
.getHighEntropyValues()でさらに詳細な情報も取得可能
文字列のUserAgentは様々な問題により情報が減らされていく方針なので、どこかで置き換える必要がある

```
// before
const uaString = navigator.userAgent
const isMobile = /* uaStringのparse等 */

// after
const uaData = navigator.userAgentData
const isMobile = uaData.mobile // false
```

Experts

Chrome	Firefox	Safari	Edge	Android Chrome	iOS Safari
96	94	15	96	96	15

HTML

dialog

HTML Living Standard

ダイアログ用の要素がHTMLに追加される。フォーカス可能位置が適切にハンドリングされる、アクセシブルなモーダルを楽に作れるようになることが期待できる

```
<!-- before -->
<!-- CSSとJSを用いた何らかの実装 -->


<!-- after -->
<dialog>
  開く前のダイアログ
</dialog>

<dialog open>
  開いているダイアログ
</dialog>
```

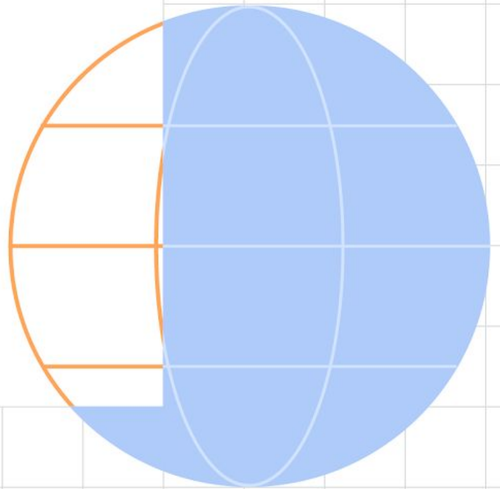
Experts

Chrome	Firefox	Safari	Edge	Android Chrome	iOS Safari
96	94	15	96	96	15

Experts

他にこういうのもあるよ！この仕様便利だよ！
というのがあれば、
ぜひTwitterでつぶやいて教えてください！

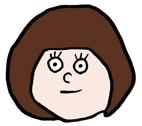
#DevFest2021



Google Developers

Experts

Thank You!



よしこ
Google Developer Experts for Web
@yoshiko_pg