Google Developers

DevFest Tokyo 2022
Web standard

今まであ書いてたアレ、これからはこう書けそう

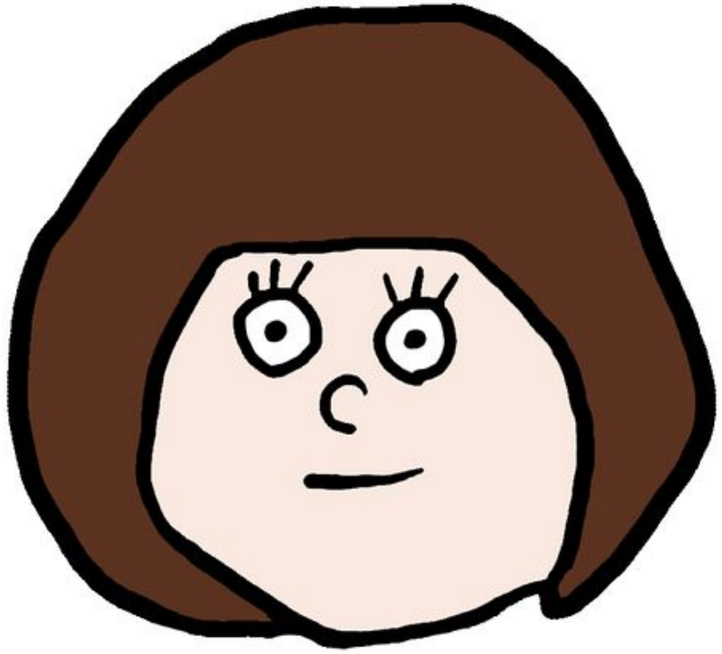
よしこ
Google Developer Experts for Web
@yoshiko_pg

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をしています。
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。

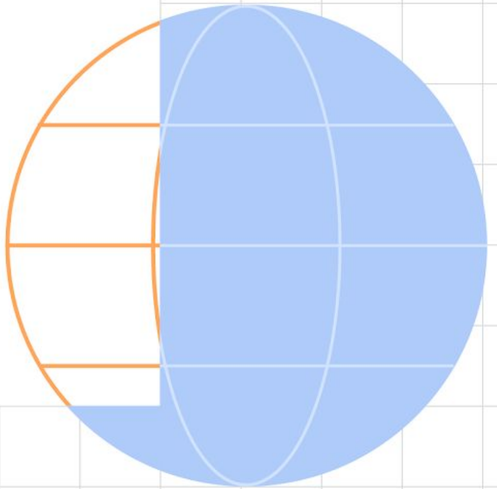
日々新しく提案・実装されていくWebの仕様の中で、モダンブラウザでの実装も進んでおり、
利用したくなる機会も多そうなものについてbefore/afterのコードを軸に紹介していきます。

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も今年の6月にサポートが終了し、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



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

Supported Version	Chrome Edge	Firefox	Safari	Android Chrome	iOS Safari
	92	90	15.4	92	15.4

JavaScript

.findLast()

ECMAScript 2023

配列の末尾から要素を検索したいとき、今までは配列をコピーしてreverseをかけたりしていたが、findLast/findLastIndexで逆からfindとfindIndexをかけられる

```
const array = [1, 2, 3, 4]


// before
[...array].reverse().find(n => n % 2) // 3


// after
array.findLast(n => n % 2) // 3
```

Experts

Supported Version	Chrome Edge	Firefox	Safari	Android Chrome	iOS Safari
	97	104	15.4	97	15.4

Web API

structuredClone()

HTML Living Standard

オブジェクトのDeep Copy（参照ではなく値のコピー）ができる。従来のJSON APIを使う形式だとエラーとなってしまう循環参照もエラーにならず解決できる

```
// before
const myObj = JSON.parse(JSON.stringify(obj));


// after
const myObj = structuredClone(obj);
```

Experts

Supported Version	Chrome Edge	Firefox	Safari	Android Chrome	iOS Safari
	98	94	15.4	98	15.4

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

Safari16で { behavior: 'smooth' } がサポートされ、アニメーションつきの自動スクロールが可能に

```
// 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

Supported Version (smooth)	Chrome Edge	Firefox	Safari	Android Chrome	iOS Safari
	61 79	36	16	61	16

Web API

.showPicker()

HTML Living Standard

ブラウザ標準 input UIのピッカーを
JavaScriptから立ち上げられる
日付ピッカー、カラーピッカー、ファイルピッカーなど
ユーザーのアクション起因の実行でないと権限エラー

```
// before
// pickerを出すクリック領域が限られるタイプのinputでは動かなかったりなどブラウザUIに依存する挙動
inputElement.click()


// after
inputElement.showPicker()
```

Experts

Supported Version	Chrome Edge	Firefox	Safari	Android Chrome	iOS Safari
	99	101	16	99	16

Web API

.requestSubmit()

HTML Living Standard

フォーム内要素のrequired、maxlengthなどの各バリデーションが通っているときだけSubmitしてくれる関数。エラーがあれば画面にブラウザ標準のフィードバックが表示される

```
// before
// 各inputなどのバリデーションは実行されず送信される
formElement.submit()


// after
// 含まれる各inputなどのバリデーションが実行され、
// エラーがあれば送信されない
formElement.requestSubmit()
```

Experts

Supported Version	Chrome Edge	Firefox	Safari	Android Chrome	iOS Safari
	76	75	16	79	16

CSS Selector

:focus-visible

CSS 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

Supported Version	Chrome Edge	Firefox	Safari	Android Chrome	iOS Safari
	86	85	15.4	86	15.4

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

Supported Version	Chrome Edge	Firefox	Safari	Android Chrome	iOS Safari
	63	59	16	63	16

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

Supported Version	Chrome Edge	Firefox	Safari	Android Chrome	iOS Safari
	93	92	15.4	93	15.4

CSS Property

translate, scale, rotate

CSS Transforms Module Level 2

transformプロパティの関数であるこれら3つのアクションがCSS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

```
/* before */
.big-box {
  transform: scale(2);
}
.big-box.downward {
  transform: scale(2) rotate(180deg);
}


/* after */
.big-box {
  scale: 2;
}
.big-box.downward {
  rotate: 180deg;
}
```

Experts

Supported Version	Chrome Edge	Firefox	Safari	Android Chrome	iOS Safari
	104	72	14.1	104	14.5

CSS Unit

L/S/D Viewport Units

CSS Values and Units Module Level 4

vh/vw/vmin/vmaxに対してl/s/dのprefixをつけた単位が追加。URLバーなどで画面領域が変動しうる場合、l(arge)が最大～s(mall)が最小領域を表し、d(ynamic)は動的に現在の領域の値を表す。

```
/* before-1 */
/* 動的に変わるブラウザバーの高さを考慮してくれない */
body {
  height: 100vh;
}

/* before-2 */
/* 画面高さの何%、のような指定はできない */
body {
  height: -webkit-fill-available;
}


/* after */
body {
  height: 100dvh;
}
```

Experts

Supported Version	Chrome Edge	Firefox	Safari	Android Chrome	iOS Safari
	108	101	15.4	108	15.4

CSS At-rule

@layer

CSS Cascading and Inheritance Level 5

CSSのカスケードにおける新しい概念。記述順や詳細度よりも優先されるので、記述の仕方ではなく開発者の意図したレイヤー構造ベースでスタイルの重ねがけルールを定義できる。

```
/* before */
/* 定義順に気をつけてカスタム定義を後勝ちさせる、
   詳細度で負けたら!importantで無理やり勝たせるなど… */
@import url(“ui-library.css”);

.item {
  /* カスタマイズしたい内容 */
}


/* after */
@layer framework, custom;

@layer custom {
  .item {
    /* カスタマイズしたい内容 */
  }
}

@import url(“ui-library.css”) layer(framework);
```

Experts

Supported Version	Chrome Edge	Firefox	Safari	Android Chrome	iOS Safari
	99	97	15.4	99	15.4

HTML

img loading=”lazy”

HTML Living Standard

img要素がブラウザのビューポートに入ってくるまで画像データの読み込みを遅延できる
デフォルトは“eager”で、要素の位置に関わらず即時読み込み

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


<!-- after -->
<img src=”...” alt=”...” loading=”lazy” />
```

Experts

Supported Version	Chrome Edge	Firefox	Safari	Android Chrome	iOS Safari
	77 79	75	15.4	77	15.4

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>
```

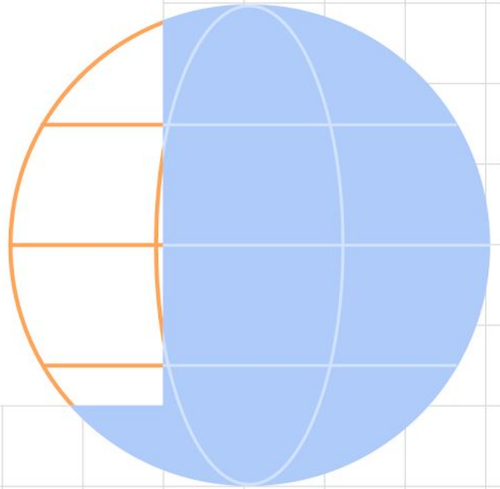
Experts

Supported Version	Chrome Edge	Firefox	Safari	Android Chrome	iOS Safari
	37 79	98	15.4	37	15.4

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 Selector

:has()

CSS Selectors Level 4

ターゲットの状況がパラメータのセレクタと一致する場合にマッチする擬似クラス。子孫要素のみでなく、隣接セレクタを使うことで前後関係を遡ることもできる。

```
/* before */
/* CSSセレクタで子の情報を参照したあと親に遡ることは
   不可能だった。JSを使った何らかの実装が必要 */


/* after */
/* 例えば、すぐあとにh2を持つh1 */
h1:has(+ h2) {
  // …
}
/* 例えば、不正な状態のinputを持つフィールドの
   ラベルやエラーテキストも含めて赤くする */
.formField:has(:invalid) {
  color: red;
}
/* 例えば、focus-visible + focus-withinを
   組み合わせた挙動 */
.selector:has(:focus-visible) {
  outline: 2px solid var(--color-main);
}
```

Experts

Supported Version	Chrome Edge	Firefox	Safari	Android Chrome	iOS Safari
	105	flag	15.4	105	15.4

@media range syntax

CSS Media Queries Level 4

比較演算子を使ってメディアクエリを定義できる。min-width、max-widthだけでは素直に記述できなかった未満・超過の指定も、不等号を使うことで実現できる。

```
/* before */
@media (min-width: 700px) and (max-width: 900px) {
  /* viewport幅が700px以上900px以下なら */
}
@media (max-width: 299px) {
  /* viewport幅が300px未満なら */
}


/* after */
@media (700px <= width <= 900px)  {
  /* viewport幅が700px以上900px以下なら */
}
@media (width < 300px) {
  /* viewport幅が300px未満なら */
}
```

Experts

Supported Version	Chrome Edge	Firefox	Safari	Android Chrome	iOS Safari
	104	63	-	104	-

CSS Syntax

subgrid

CSS Grid Layout Module Level 2

gridが入れ子になる場合に、子のgrid-template-columns および grid-template-rows プロパティの値として subgrid を指定すると、親のグリッド定義を使って整列させることができる。

```
.grid {
  display: grid;
  grid-template-columns: 100px 100px 100px;
  grid-template-rows: 100px 100px;
  grid-template-areas: "nest nest other1"
                       "nest nest other2"
}

/* before */
.subgrid {
  display: grid;
  grid-area: nest;
  grid-template-columns: 100px 100px;
  grid-template-rows: 100px 100px;
}

/* after */
.subgrid {
  display: grid;
  grid-area: nest;
  grid-template-columns: subgrid;
  grid-template-rows: subgrid;
}
```

Experts

Supported Version	Chrome Edge	Firefox	Safari	Android Chrome	iOS Safari
	-	71	16	-	16

CSS At-rule

@container

CSS Containment Module Level 3

media queryのようにviewportのサイズではなく、コンテナーコンテキストとして宣言されたコンテナー要素のサイズを元にクエリで条件を指定してスタイルを分岐させられる。

```
/* before */
@media (max-width: 300px) {
  .cards {
      flex-direction: column;
  }
}


/* after */
.container {
  container-type: inline-size;
}
@container (min-width: 300px) {
  .cards {
      flex-direction: column;
  }
}
```

Experts

Supported Version	Chrome Edge	Firefox	Safari	Android Chrome	iOS Safari
	105	-	16	105	16

Infrastructure

Scroll to Text

Text Fragments

idをふらなくても#:~:text=という
syntaxで特定の文字列に対する
ハッシュリンクを作成できる。URLを
開いたとき、該当の位置までページ
がスクロールされる。

```
<!-- before -->
<a href="#target">jump</div>
<div id="target">come here!</div>


<!-- after -->
<a href="#:~:text=come%20here!">jump</div>
<div>come here!</div>
```

Experts

Supported Version	Chrome Edge	Firefox	Safari	Android Chrome	iOS Safari
	81	-	16.1	81	16.1

Interop 2022

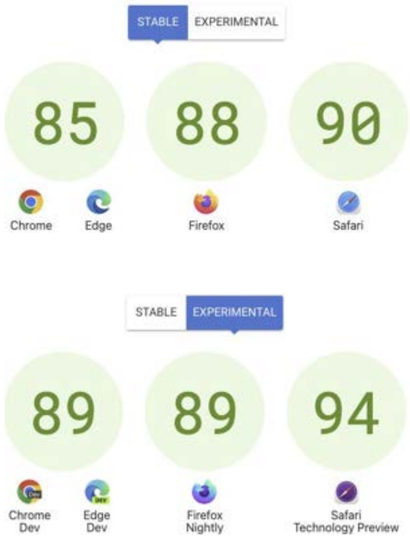
主要ブラウザー間の互換性を上げていく取り組み

ブラウザー間の互換性向上のため、Google Chrome、Safari、Firefoxチームとその周囲の関係者が共同で議論・合意して掲げたベンチマークがInterop2022です。
Focus Areaとして2022年に注力する内容が定義されており、今年を通してより活発に実装が進みました。

この取り組みにおいて、Web開発者の開発体験の向上はとても重視されており、定期的におこなわれる開発者へのサーベイの結果も大変参考にされています。

How are these scores calculated?

2022 Focus Areas (60%)			
Cascade Layers	100%	96%	100%
Color Spaces and Functions	52%	51%	95%
Containment	99%	98%	99%
Dialog Element	99%	94%	95%
Forms	99%	90%	97%
Scrolling	94%	91%	91%
Subgrid	17%	85%	96%
Typography and Encodings	78%	99%	85%
Viewport Units	100%	100%	100%
Web Compat	79%	100%	57%
SUBTOTAL	82%	90%	91%
2021 Focus Areas (30%)			
Aspect Ratio	98%	98%	98%
Flexbox	99%	97%	95%
Grid	99%	90%	94%
Sticky Positioning	100%	89%	100%
Transforms	97%	95%	95%
SUBTOTAL	99%	94%	96%
2022 Investigation (10%)			**Group Progress**
Editing, contenteditable, and execCommand			46%
Pointer and Mouse Events			79%
Viewport Measurement			60%
SUBTOTAL			61%
GRAND TOTAL	**85%**	**88%**	**90%**

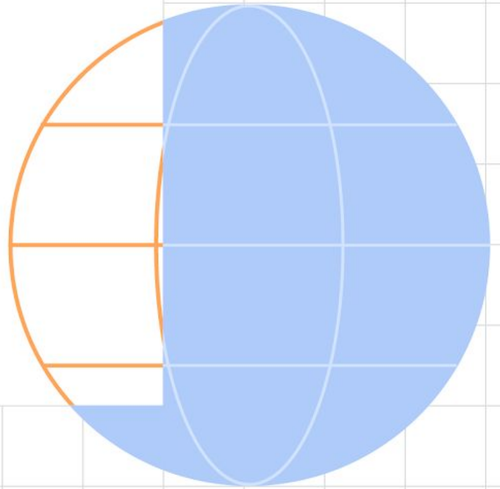


Dashboard: https://wpt.fyi/interop-2022
Survey: https://stateofcss.com/ja-jp/ https://stateofjs.com/ja-jp/ 他

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！

#DevFest2022



Google Developers

Experts

Thank You!

よしこ @yoshiko_pg
Google Developer Experts for Web

Thank you for advice, @kosamari-san!