CSSNITE × KANDIGI

CSS Nite in **Osaka**, Vol.52

# Modern Coding

きちんと理解して活用する SVG と CSS Grid

powered by MEBIC

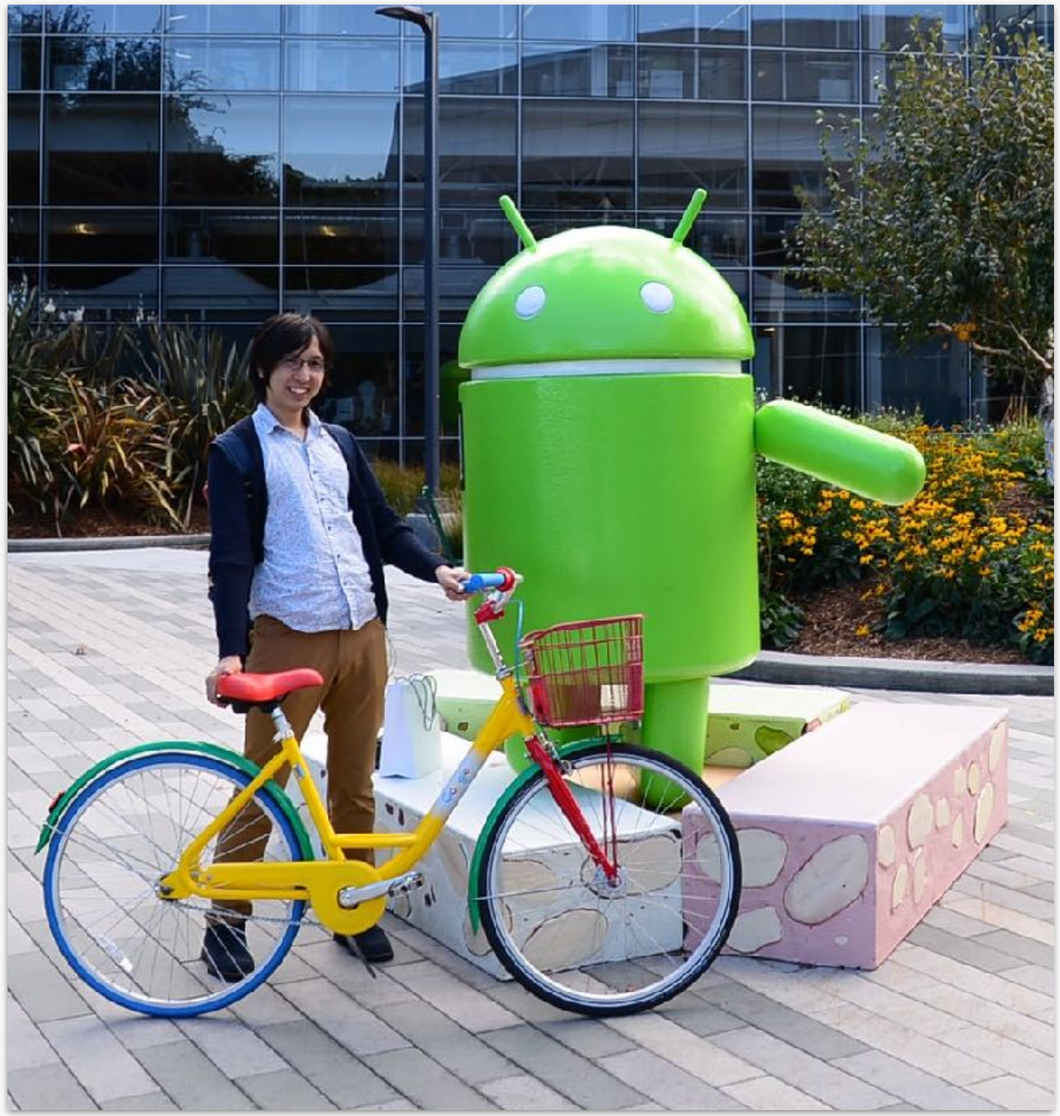
2019.9.7 土

# 現場で働くコーダーのための CSS Grid + モダンコーディング

株式会社ICS 鹿野 壮

2019年9月7日 CSS Nite in Osaka, vol.52「Modern Coding」

# 自己紹介



株式会社ICS

福岡県出身

## 鹿野 壮 (かの たけし)

@tonkotsuboy_com

# オウンドメディア「ICS MEDIA」


ウェブデザイナーと
フロントエンドエンジニアのための
情報メディア

ics.media

# 「JavaScriptコードレシピ集」の著者


- ECMAScript、DOM APIなど
  JavaScriptの構文や使い方を解説
- スグに現場で使える
  サンプルを数多く準備
- 時代が変わっても
  長く使えることを目標としている

CSSには、
便利な機能が次々と追加されている

当たり前だと思っていた手法を見直すことで
より便利にウェブ開発ができる

# 現場で働くコーダーのためのCSS Grid + モダンコーディング

1. CSS Gridとボックスレイアウト
2. 抑えておきたいCSSの新機能
3. 情報のキャッチアップ方法

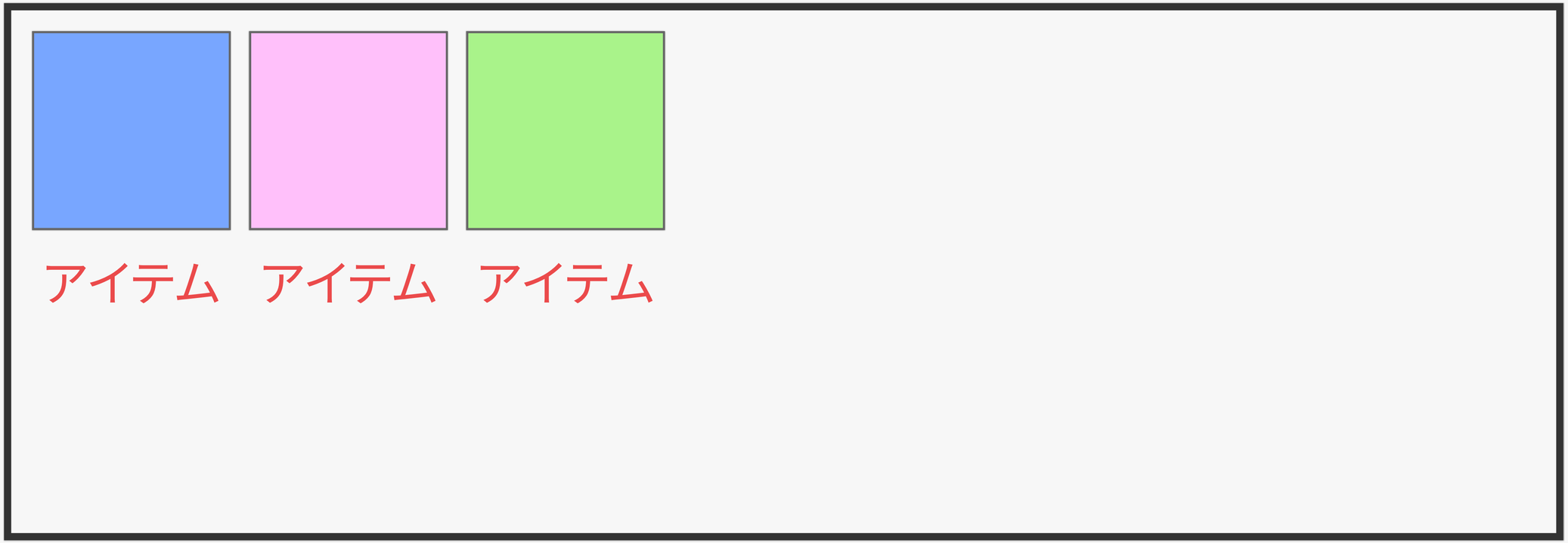
# 現場で働くコーダーのためのCSS Grid + モダンコーディング

1. CSS Grid
   - 従来のボックスレイアウト手法
   - CSS Gridの基本
   - CSS Gridの便利な機能
   - CSS GridのIE11対応方法
2. 抑えておきたいCSSの新機能
3. 情報のキャッチアップ方法

# float

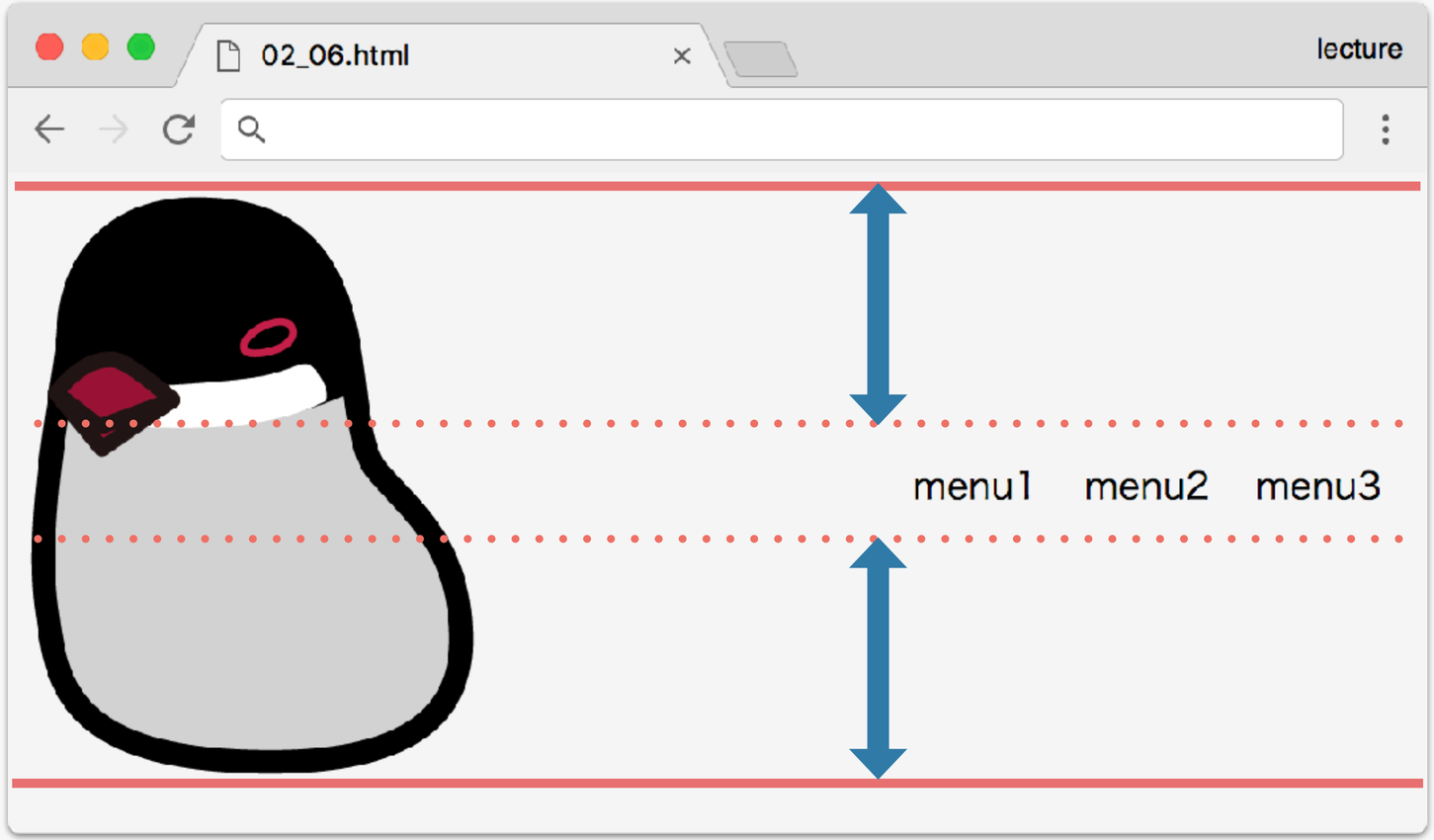
もともとはテキストの回り込みのためのプロパティだった

# Flexbox：1軸方向のレイアウト



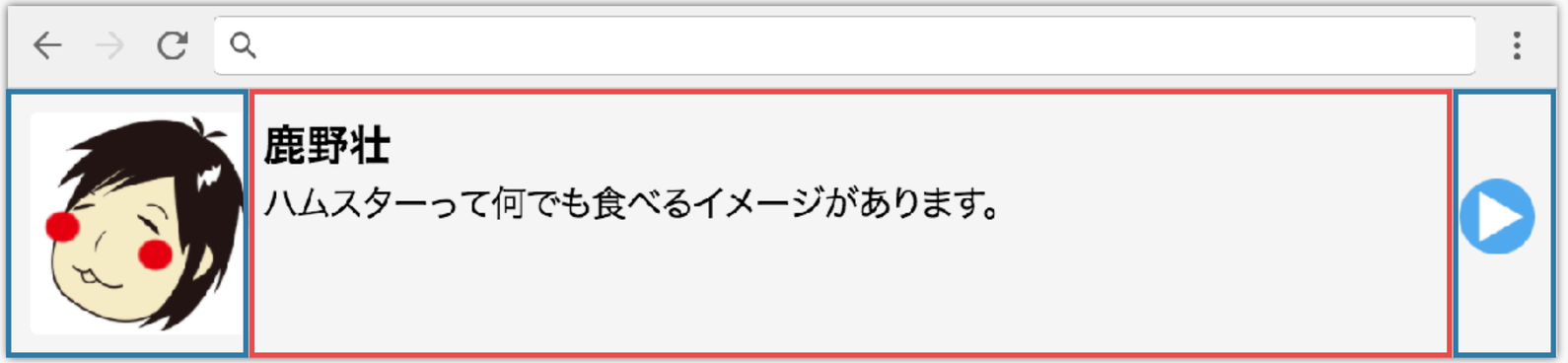
# Flexbox：レイアウト用プロパティが豊富にある

- justify-contentプロパティ: 主軸方向のレイアウト
- align-itemsプロパティ: 交差軸方向のレイアウト

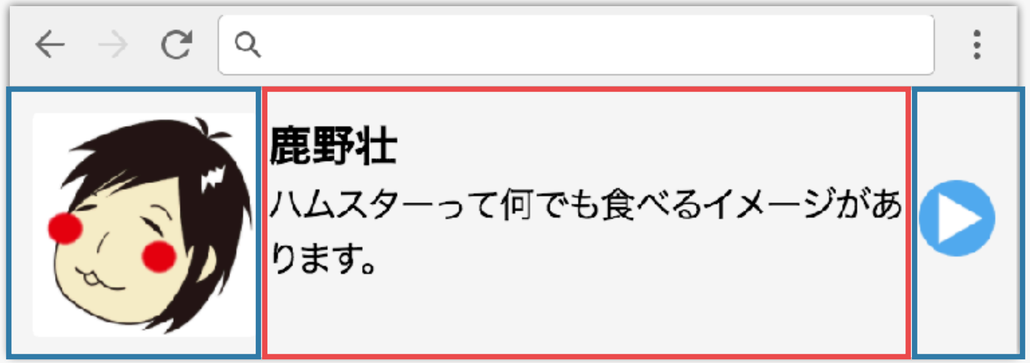


# Flexbox：固定幅・可変幅の組み合わせがラク

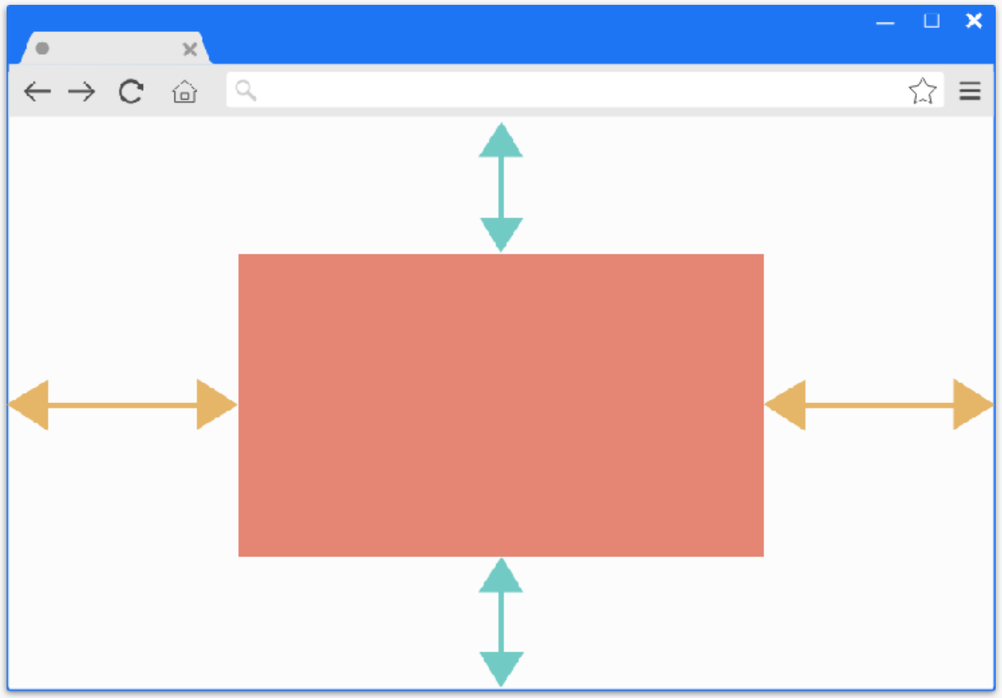
- flex-growプロパティ
- flex-shrinkプロパティ




固定幅　可変幅　固定幅



# Flexbox：上下中央揃えがたった3行でできる



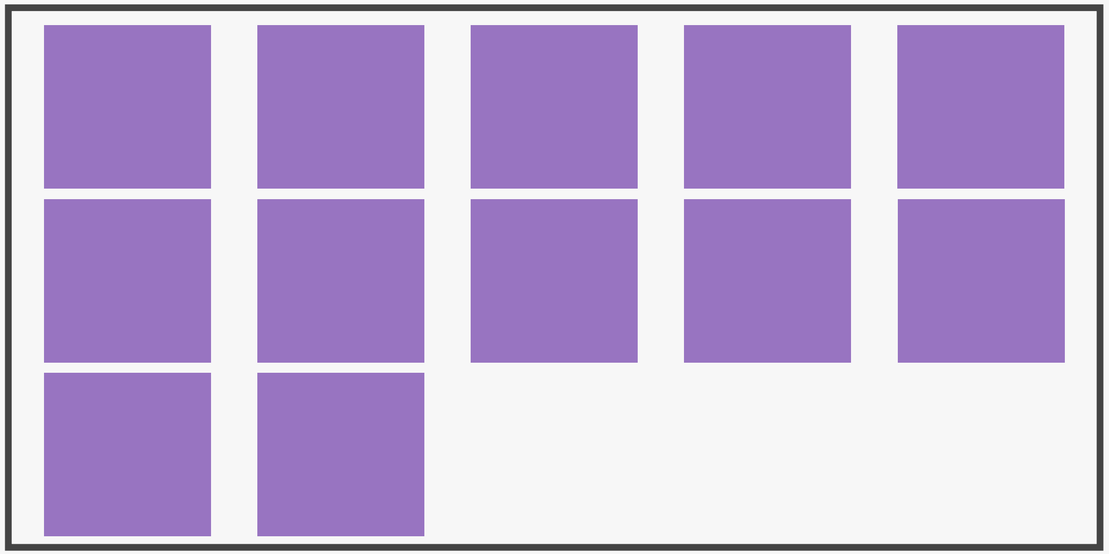
```
.container {
  display: flex;
  justify-content: center;
  align-items: center;
}
```

上下中央揃えのCSSまとめ。Flexboxがたった3行で最も手軽

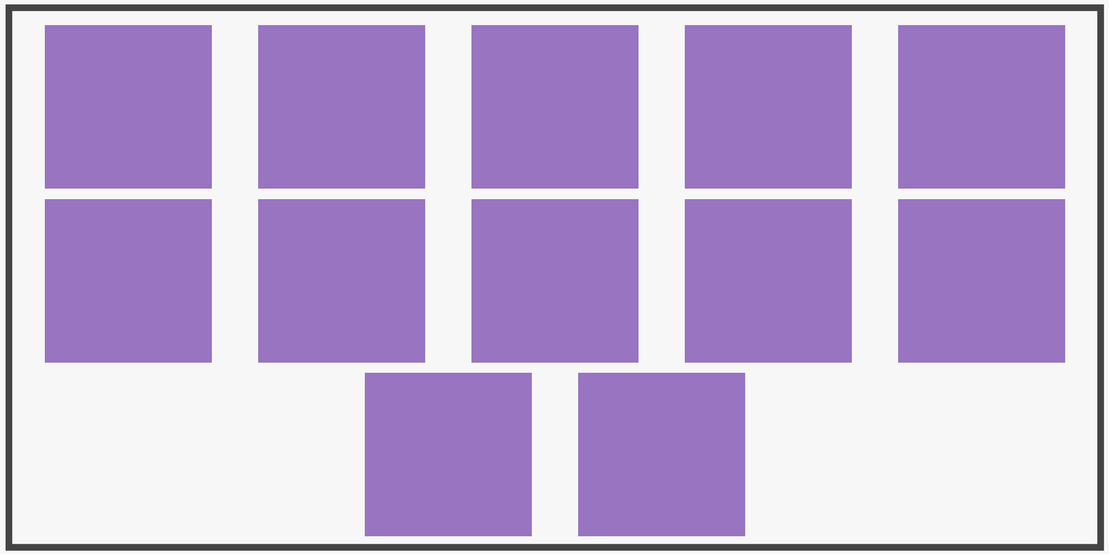
# Flexboxは
# 複数行のレイアウトが苦手

# Flexboxは複数行レイアウトがニガテ

**justify-content: space-aroundで均等配置しても、**
**最終行が揃わない**



理想

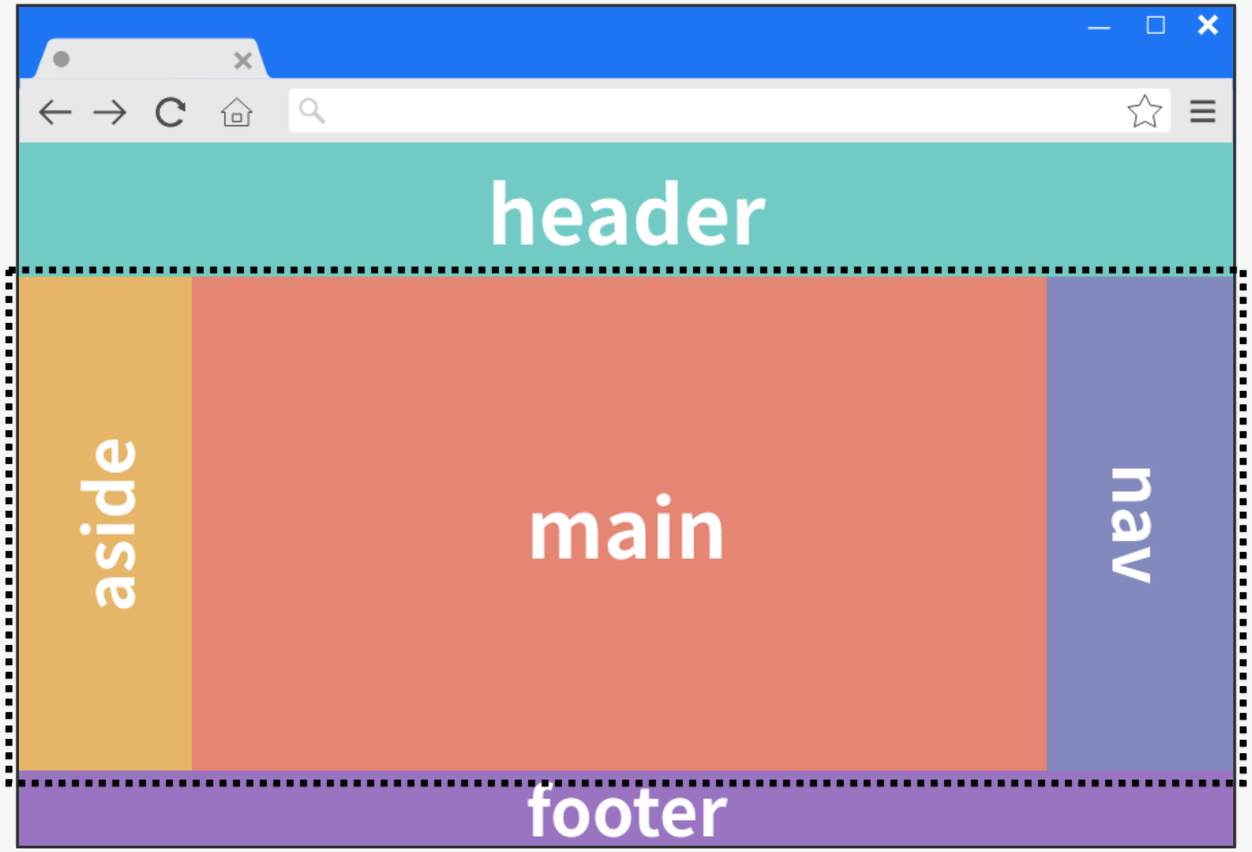


現実

※ 透明なボックスを配置したり、flex-startとcalc()を組み合わせて解決可能だが、煩雑

# Flexboxは複数行レイアウトがニガテ

## 聖杯レイアウトを実現しようとすると要素の入れ子が必要



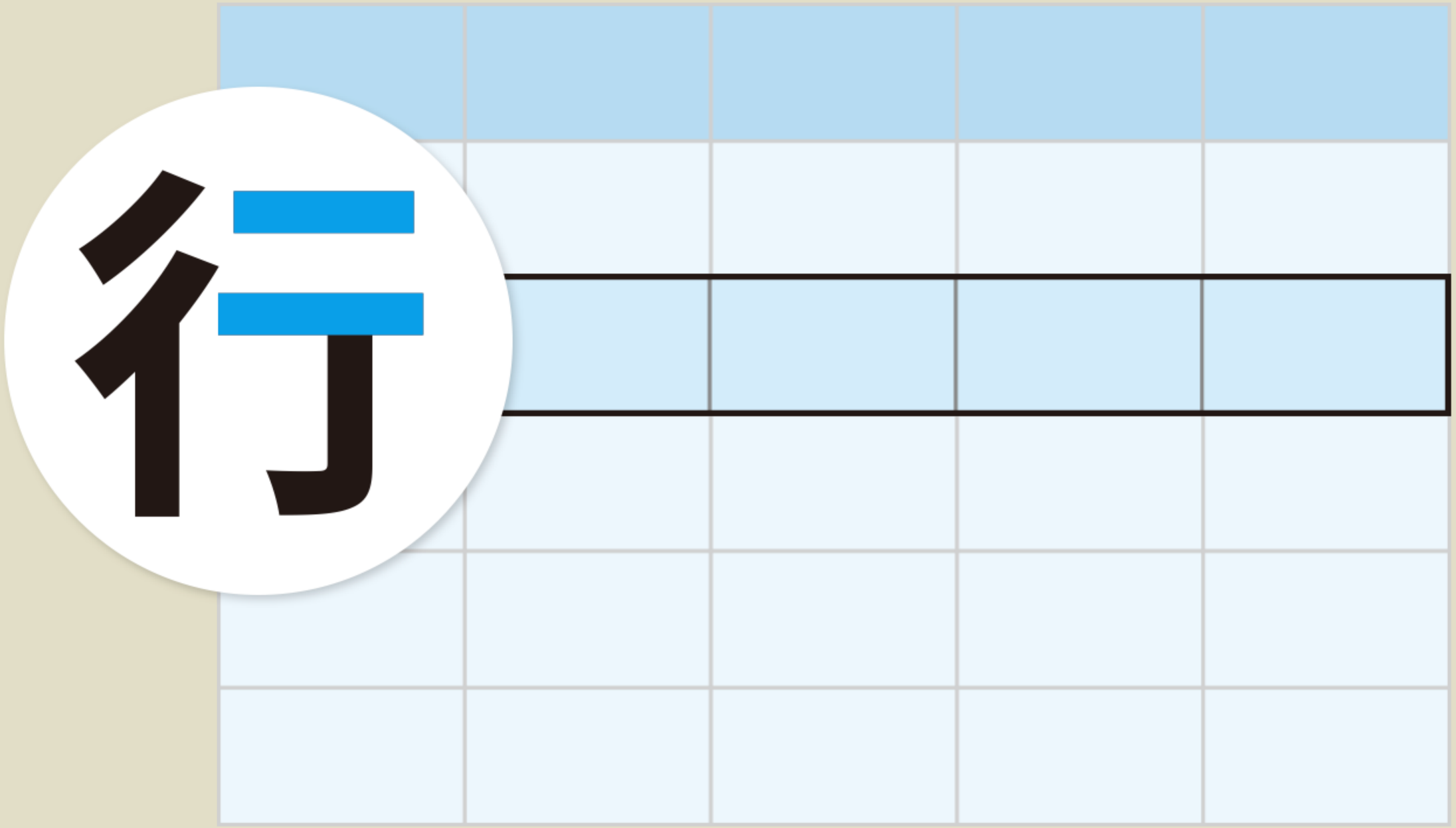

```
<div class="container">
  <header>header</header>
  <div class="middle">
    <aside>aside</aside>
    <main>main</main>
    <nav>nav</nav>
  </div>
  <footer>footer</footer>
</div>
```

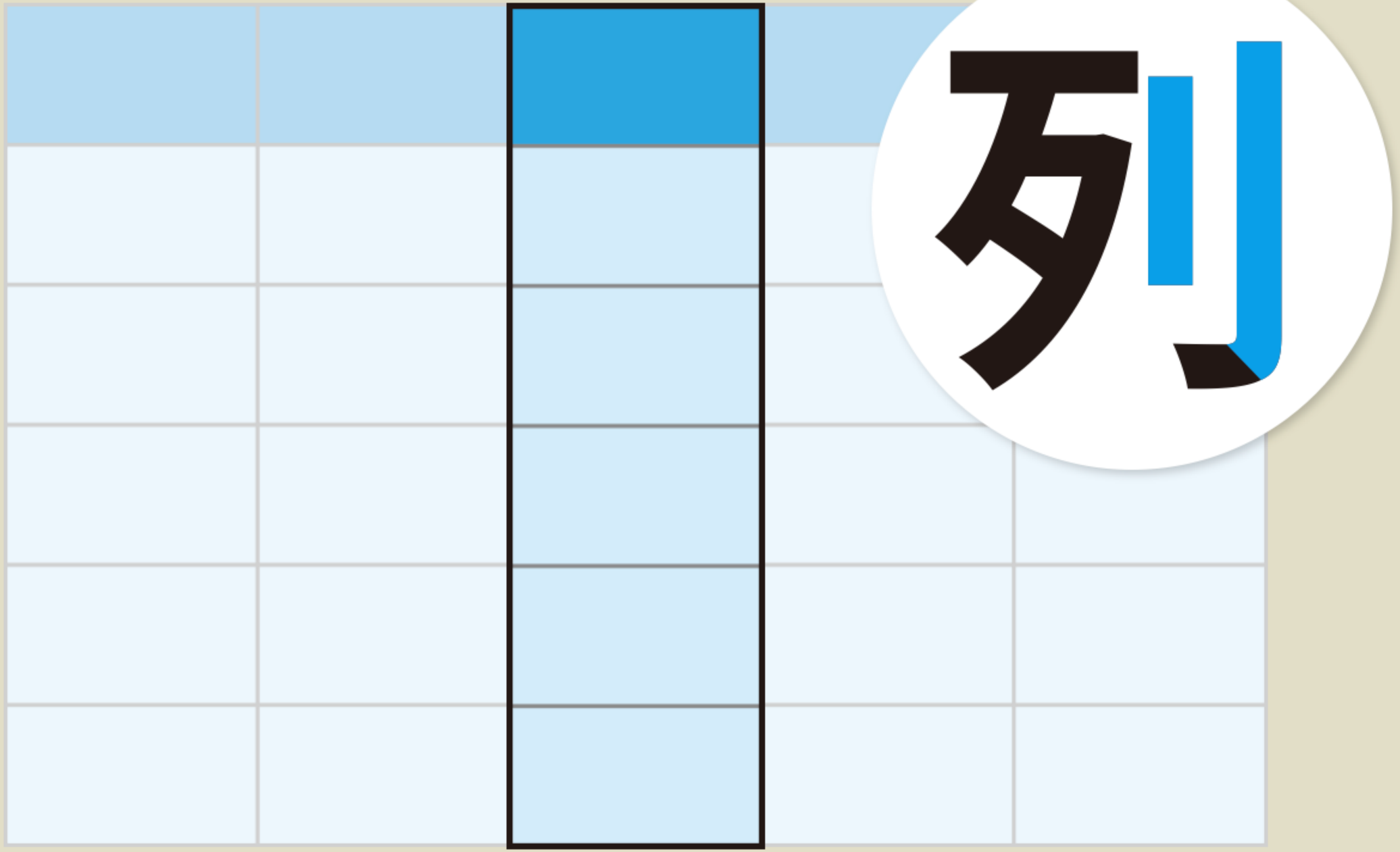
# 現場で働くコーダーのためのCSS Grid + モダンコーディング

1. CSS Grid
   - 従来のボックスレイアウト手法
   - CSS Gridの基本
   - CSS Gridの便利な機能
   - CSS GridのIE11対応方法
2. 抑えておきたいCSSの新機能
3. 情報のキャッチアップ方法

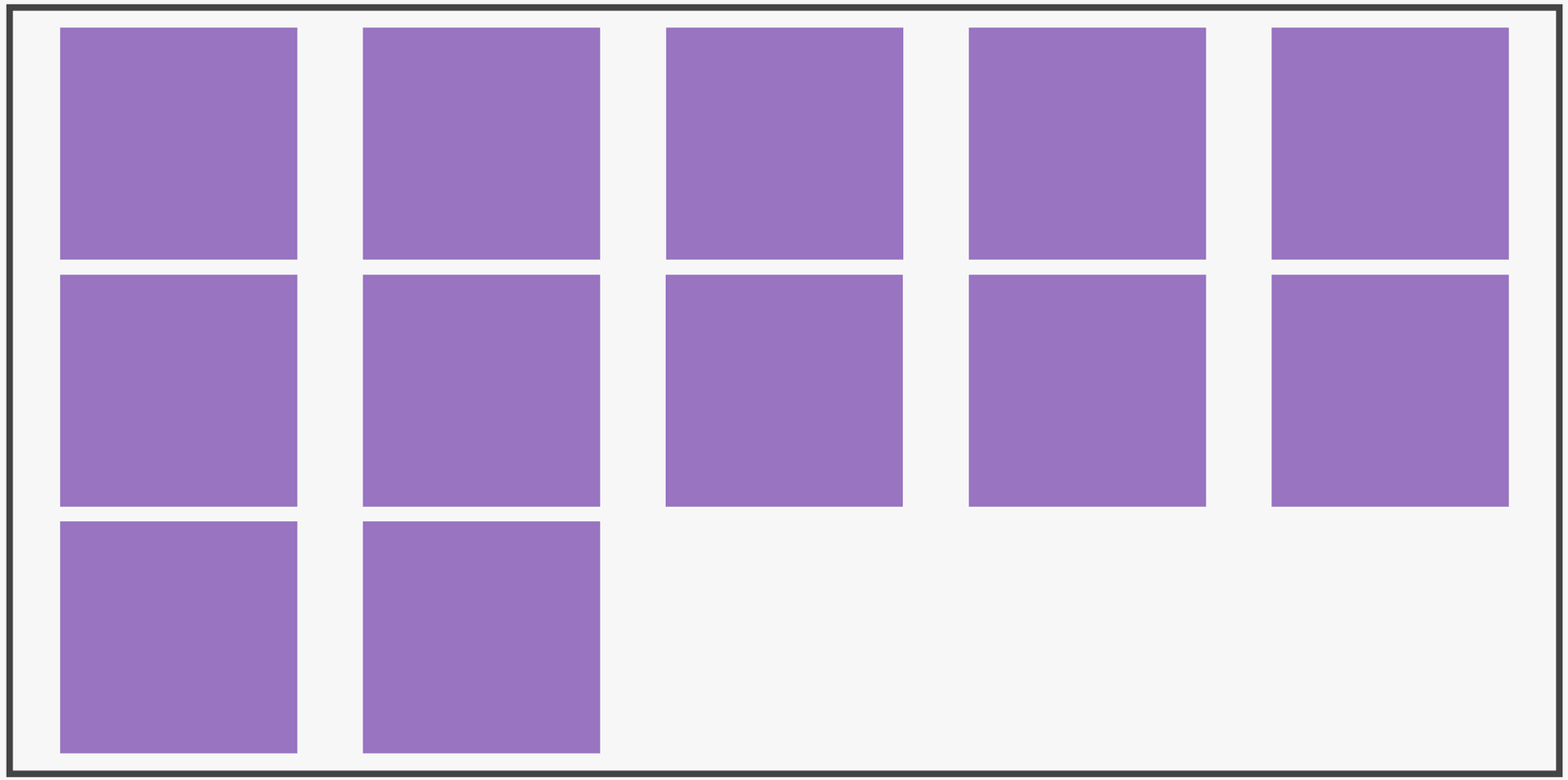
# 複数行のレイアウトには
# Flexboxではなく
# CSS Gridを使う

# CSS Gridとは
# 「行列」を使ったレイアウトのこと

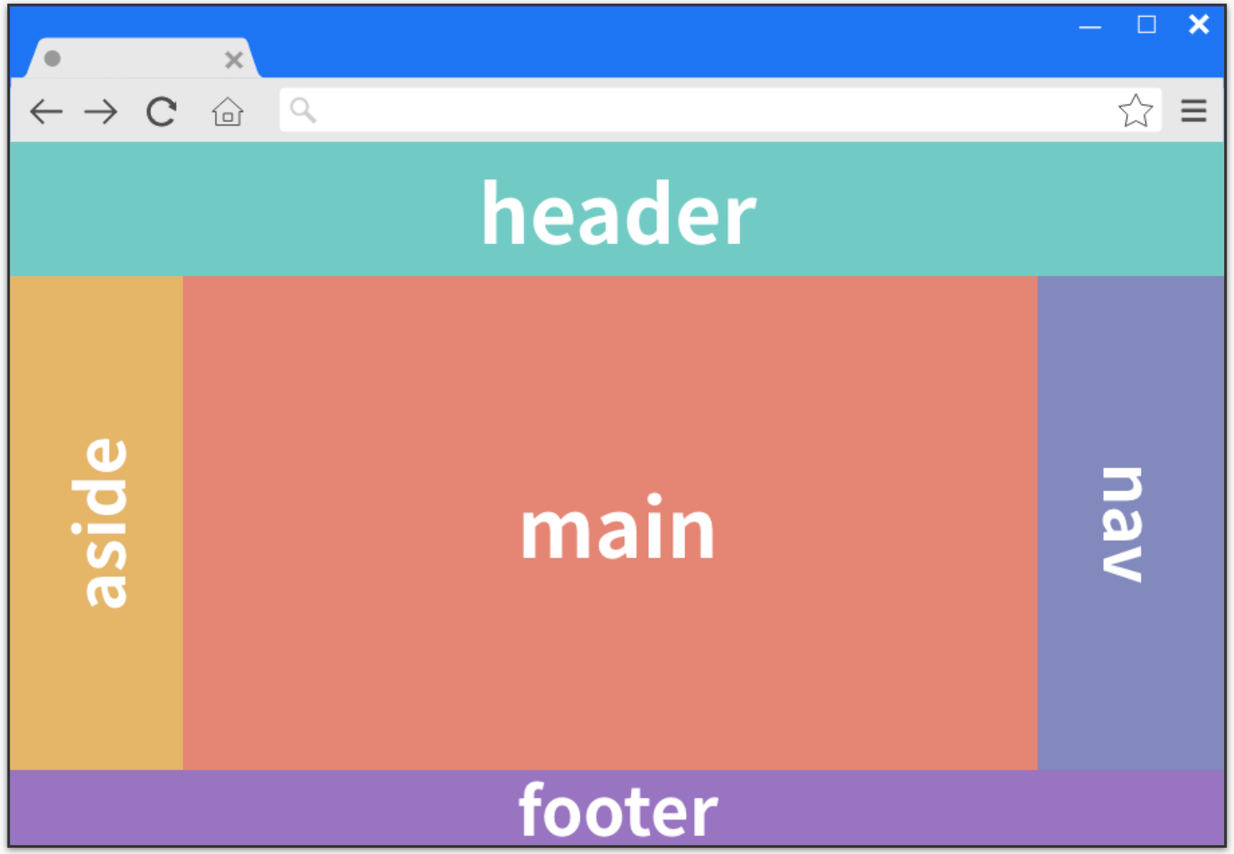

行


列

# CSS Gridでは複数行レイアウトの最終行が揃う



# CSS Gridではレイアウトのための HTMLの入れ子は増えない



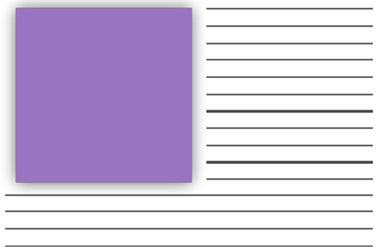
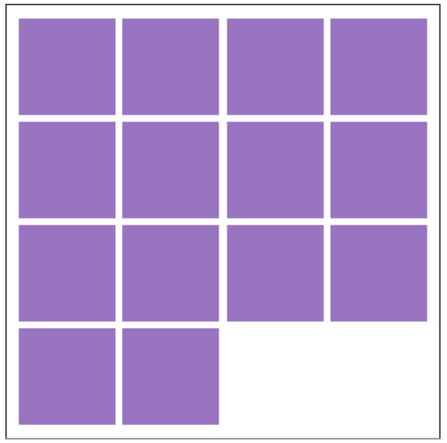

```
<div class="container">
  <header>header</header>
  <aside>aside</aside>
  <main>main</main>
  <nav>nav</nav>
  <footer>footer</footer>
</div>
```

# CSS GridはEdgeを含む全モダンブラウザで対応済み

| |  |  |  |  |  |  |
|---|---|---|---|---|---|---|
| ブラウザ名 | Chrome | Firefox | Safari | Edge | Safari (iOS版) | Chrome (Android版) |
| 最新バージョン | 76 | 68 | 12 | 18 | 12 | 76 |
| Grid対応 | ○ | ○ | ○ | ○ | ○ | ○ |

※ IE 11については、後のスライドで解説します

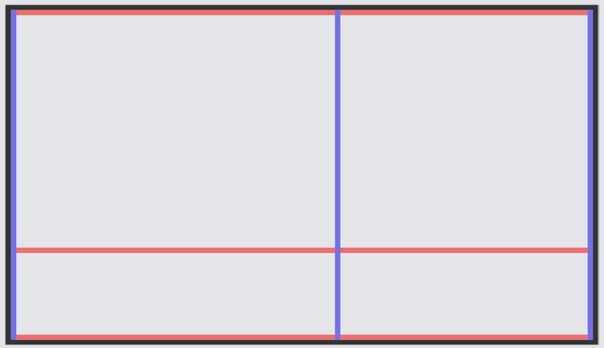
# 用途別レイアウトの使い分け

| | | |
|---|---|---|
| テキストの回り込み | **float** |  |
| 1行の横並び、縦並び | **Flexbox** |  |
| 格子状のレイアウト<br>ページ全体のレイアウト | **CSS Grid** |  |

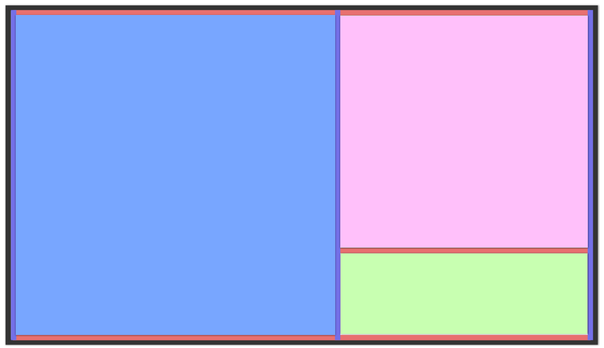
# CSS Gridを書くための3ステップ

1. コンテナを作る

2. 行と列を作る



3. アイテムを配置する



# 1. コンテナを作る


コンテナ

# 2. 行と列を作る（テンプレートの作成）



行(row)

行(row)

列(column)

列(column)

# 2. 行と列を作る（テンプレートの作成）



行(row)

行(row)

列(column)

列(column)

# 行列は
# 「サイズ」と「エリアの名前」で決定する

# 2-1. 行と列のサイズの決定

Uni of the day

UNI OF THE DAY

今日のうにちゃん

70%

30%

50%

50%

# 2-2. 各エリアに名前をつける


Uni of the day

logo

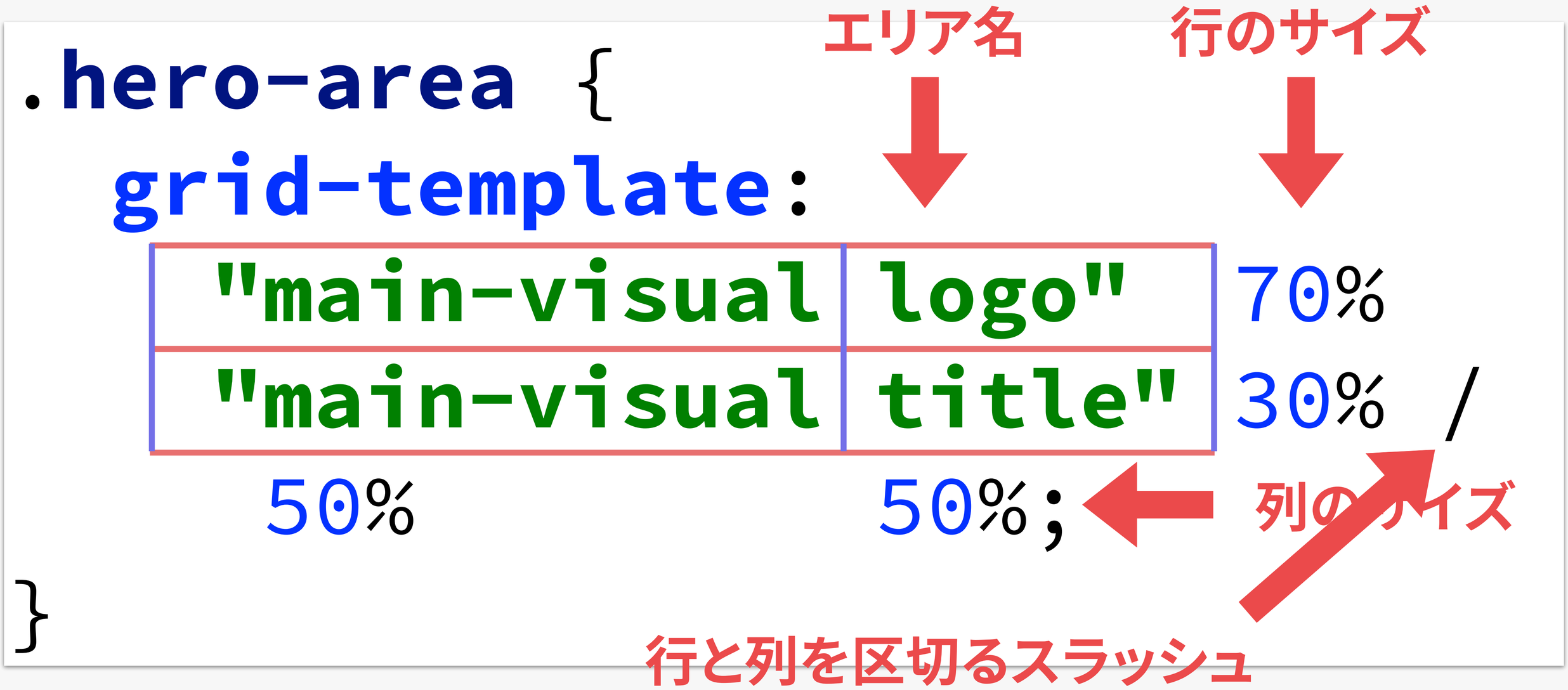

UNI OF THE DAY

title

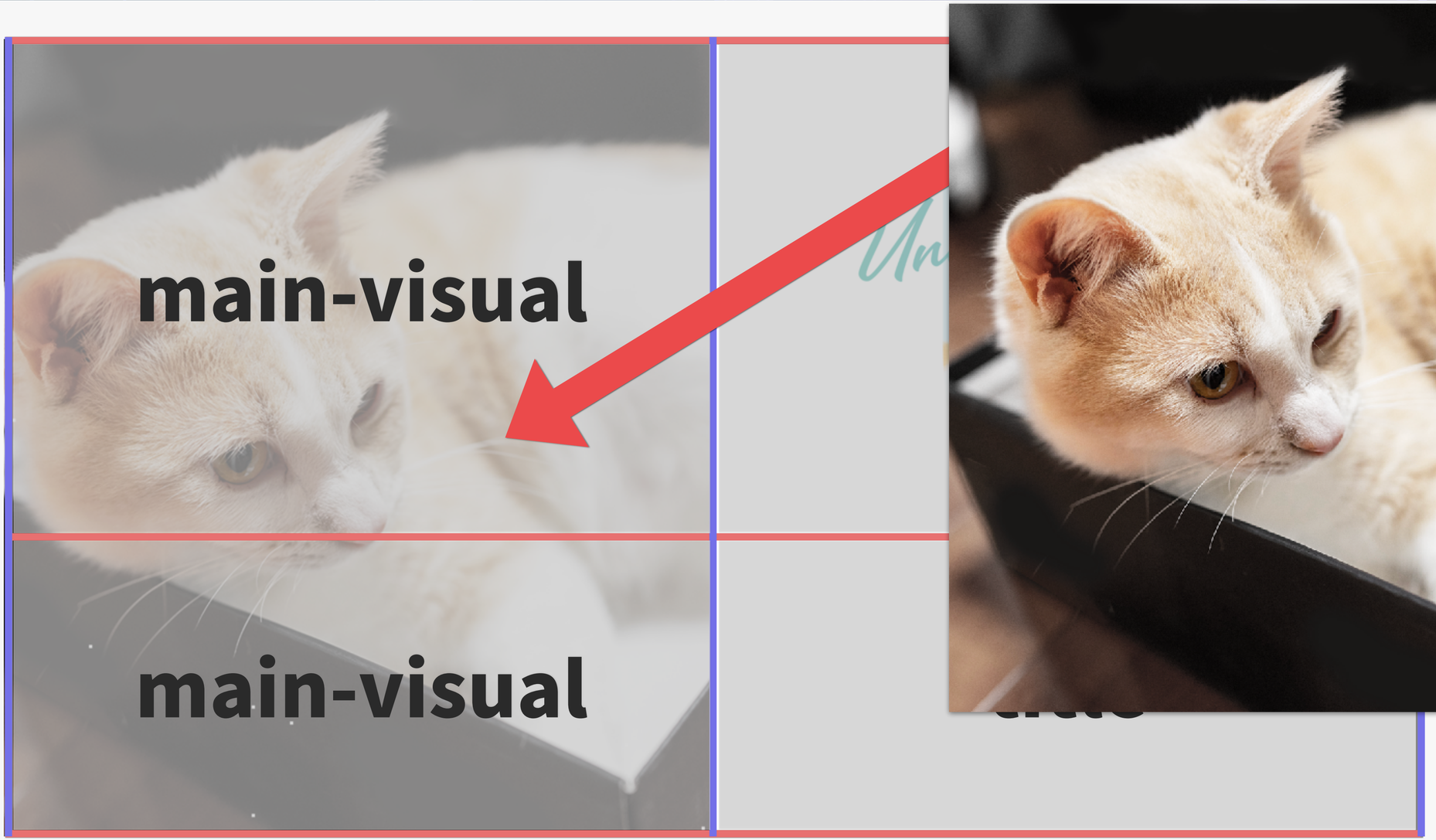
# 行と列を作るためのCSSコード

```
.hero-area {
  grid-template:
    "main-visual logo"   70%
    "main-visual title" 30% /
     50%                50%;
}
```

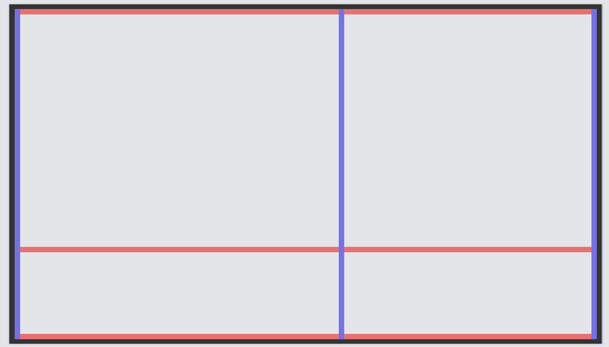
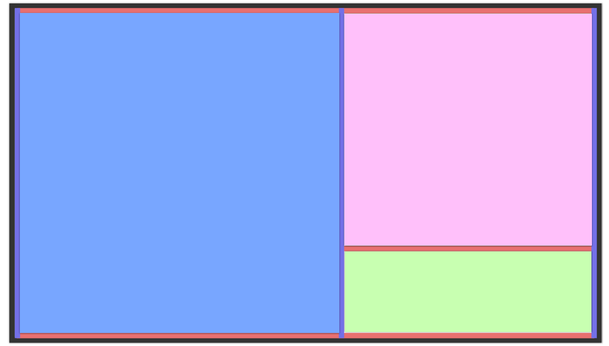
# 行と列を作るためのCSSコード



# 3. アイテムを配置する



# CSS Gridを書くための3ステップ

| | | |
|---|---|---|
| 1. コンテナを作る | display: grid |  |
| 2. 行と列を作る | grid-template<br>プロパティ |  |
| 3. アイテムを配置する | grid-area<br>プロパティ |  |

## ■ サンプルファイル

sample1.html

sample1.css

# 現場で働くコーダーのためのCSS Grid + モダンコーディング

1. CSS Grid
   - 従来のボックスレイアウト手法
   - CSS Gridの基本
   - CSS Gridの便利な機能
   - CSS GridのIE11対応方法
2. 抑えておきたいCSSの新機能
3. 情報のキャッチアップ方法

# レスポンシブ対応

grid-templateを書き換えるだけ

```
@media (max-width: 520px) {
   .hero-area {
     grid-template:
        "main-visual" 50%
        "logo" 35%
        "title" 1fr /
        100%;
   }
}
```

# 行や列同士のスペース

gapプロパティ（旧名grid-gapプロパティ）

- floatやFlexboxにはない、CSS Gridの魅力の一つ

```
.container {
  gap: 10px;
}
```

# 繰り返しの記述

行や列を指定サイズで繰り返すrepeat()メソッド

- repat(繰り返し数, 繰り返しのサイズ)

```
.container {
  grid-template-rows: repeat(5, 50px);
  grid-template-columns: repeat(2, 50%);
}
```

# CSS コードを書きながら解説します

■ サンプルファイル

sample2.html

sample2.css

# 現場で働くコーダーのためのCSS Grid + モダンコーディング

1. CSS Grid
   - 従来のボックスレイアウト手法
   - CSS Gridの基本
   - CSS Gridの便利な機能
   - CSS GridのIE11対応方法
2. 抑えておきたいCSSの新機能
3. 情報のキャッチアップ方法

# Q. CSS Gridを使いたいけど
# IE11では使えないよね？

# A. 使えます

# IE 11はCSS Gridの古い記法に対応している

| プロパティ | IE 11 |
| --- | --- |
| grid-template | × |
| grid-template-areas | × |
| grid-template-columns<br>grid-template-rows | ○※ |
| grid-column<br>grid-row | ○※ |
| grid-row-start<br>grid-column-start | ○※ |
| grid-row-end<br>grid-column-end | × |

| プロパティ | IE 11 |
| --- | --- |
| grid-auto-flow | × |
| grid-auto-columns<br>grid-auto-rows | × |
| gap | × |
| column-gap<br>row-gap※ | × |
| align-self | ○※ |
| justify-self | ○※ |

※ 記法が異なる

# Autoprefixerを使って
# IE 11に対応した記法に変換する

# 多くのCSS Gridプロパティを IE 11で変換できる

| プロパティ | IE 11 | Autoprefixer |
| --- | --- | --- |
| grid-template | × | ○ |
| grid-template-areas | × | ○ |
| grid-template-columns<br>grid-template-rows | ○※ | ○ |
| grid-column<br>grid-row | ○※ | ○ |
| grid-row-start<br>grid-column-start | ○※ | ○ |
| grid-row-end<br>grid-column-end | × | ○ |

| プロパティ | IE 11 | Autoprefixer |
| --- | --- | --- |
| grid-auto-flow | × | × |
| grid-auto-columns<br>grid-auto-rows | × | × |
| gap | × | ○ |
| column-gap<br>row-gap※ | × | ○ |
| align-self | ○※ | ○ |
| justify-self | ○※ | ○ |

※ 記法が異なる

# 各ツールと連携する方法

最新版で学ぶwebpack 4入門 - スタイルシート取り込む方法

https://ics.media/entry/17376

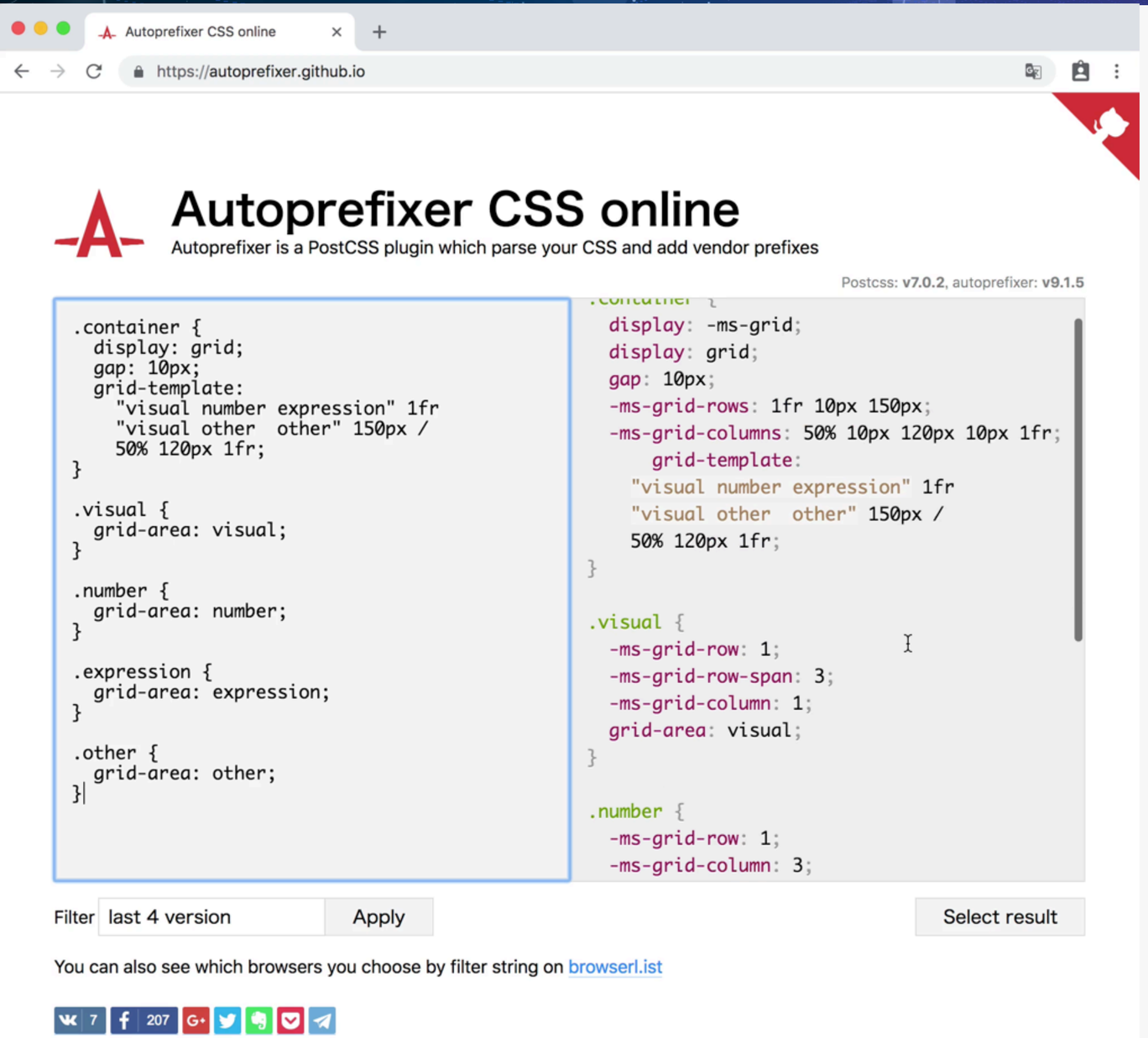
CSSベンダープレフィックスを今この瞬間に辞める為Autoprefixerの導入


https://qiita.com/tonkotsuboy_com/items/377913c51b1ac00deffe

Sassの変換にオススメ! Parcel入門

https://ics.media/entry/19580/

# オンラインツールで自動変換する方法




## Autoprefixer CSS online

- 環境構築不要

https://autoprefixer.github.io/

# Autoprefixerで自動変換

```
.container {
  display: grid;
  grid-template:
    "visual number expression" 1fr
    "visual other  other" 220px /
    40% 120px 1fr;
}
```

```
.container {
  display: -ms-grid;
  display: grid;
  -ms-grid-rows: 1fr 220px;
  -ms-grid-columns: 40% 120px 1fr;
  grid-template:
    "visual number expression" 1fr
    "visual other  other" 220px /
    40% 120px 1fr;
}
```

# Autoprefixerで自動変換

```
.visual {
  grid-area: visual;
}

.number {
  grid-area: number;
}
```

```
.visual {
  -ms-grid-row: 1;
  -ms-grid-row-span: 2;
  -ms-grid-column: 1;
  grid-area: visual;
}

.number {
  -ms-grid-row: 1;
  -ms-grid-column: 2;
  grid-area: number;
}
```

# エリア名の変換

# 昔は、IE 11向けにエリアを指定できなかった

## 昔は番号指定が必要だった

### 行と列の定義

```
.container {
 grid-template-rows: 1fr 220px;
 grid-template-columns: 40% 120px 1fr;
}
```

### アイテムの配置

```
.main-visual {
  grid-row-start: 1;
  grid-row-end: 3;
  grid-column-start: 1;
  grid-column-end: 2;
  -ms-grid-row-span: 2;
}
```

# 今は、AutoprefixerでエリアをIE 11向けに変換可能

before

```
.container {
  display: grid;
  grid-template:
    "visual number expression" 1fr
    "visual other  other" 220px /
    40% 120px 1fr;
}

.visual {
  grid-area: visual;
}
```

# 今は、AutoprefixerでエリアをIE 11向けに変換可能

after

```
.container {
  display: -ms-grid;
  display: grid;
  -ms-grid-rows: 1fr 220px;
  -ms-grid-columns: 40% 120px 1fr;
  grid-template:
    "visual number expression" 1fr
    "visual other  other" 220px /
    40% 120px 1fr;
}

.visual {
  -ms-grid-row: 1;
  -ms-grid-row-span: 2;
  -ms-grid-column: 1;
  grid-area: visual;
}
```

# 行・列間の隙間（gap）の IE11向け変換

# CSS Gridで行と列の隙間を指定する従来コード

IE 11はgap（旧名grid-gap）に未対応なので、
marginやpaddingで指定していた

```
.number,
.expression,
.other {
  padding: 10px;
}
```

# AutoprefixerでgapをIE 11向けに変換

before

```
.container {
  display: grid;
  grid-template:
    "visual number expression" 1fr
    "visual other  other" 220px /
    40% 120px 1fr;
}

.visual {
  grid-area: visual;
}
```

# AutoprefixerでgapをIE 11向けに変換

after

```
.container {
  /* 中略 */

  gap: 10px;
  -ms-grid-rows: 1fr 10px 220px;
  -ms-grid-columns: 40% 10px 120px 10px 1fr;
  grid-template:
    "visual number  expression" 1fr
    "visual other   other"      220px /
    40%     120px   1fr;
}

.visual {
  -ms-grid-row: 1;
  -ms-grid-row-span: 3;
  -ms-grid-column: 1;
  grid-area: visual;
}
```

# CSS コードを書きながら解説します

■ サンプルファイル

sample1.html

sample1.css

# repeatをIE11向けに変換するときは
# ちょっと注意が必要

# CSS Gridで行と列の隙間を指定する従来コード

IE 11はgap（旧名grid-gap）に未対応なので、
marginやpaddingで指定していた

```
.number,
.expression,
.other {
  padding: 10px;
}
```

# AutoprefixerでgapをIE 11向けに変換

before

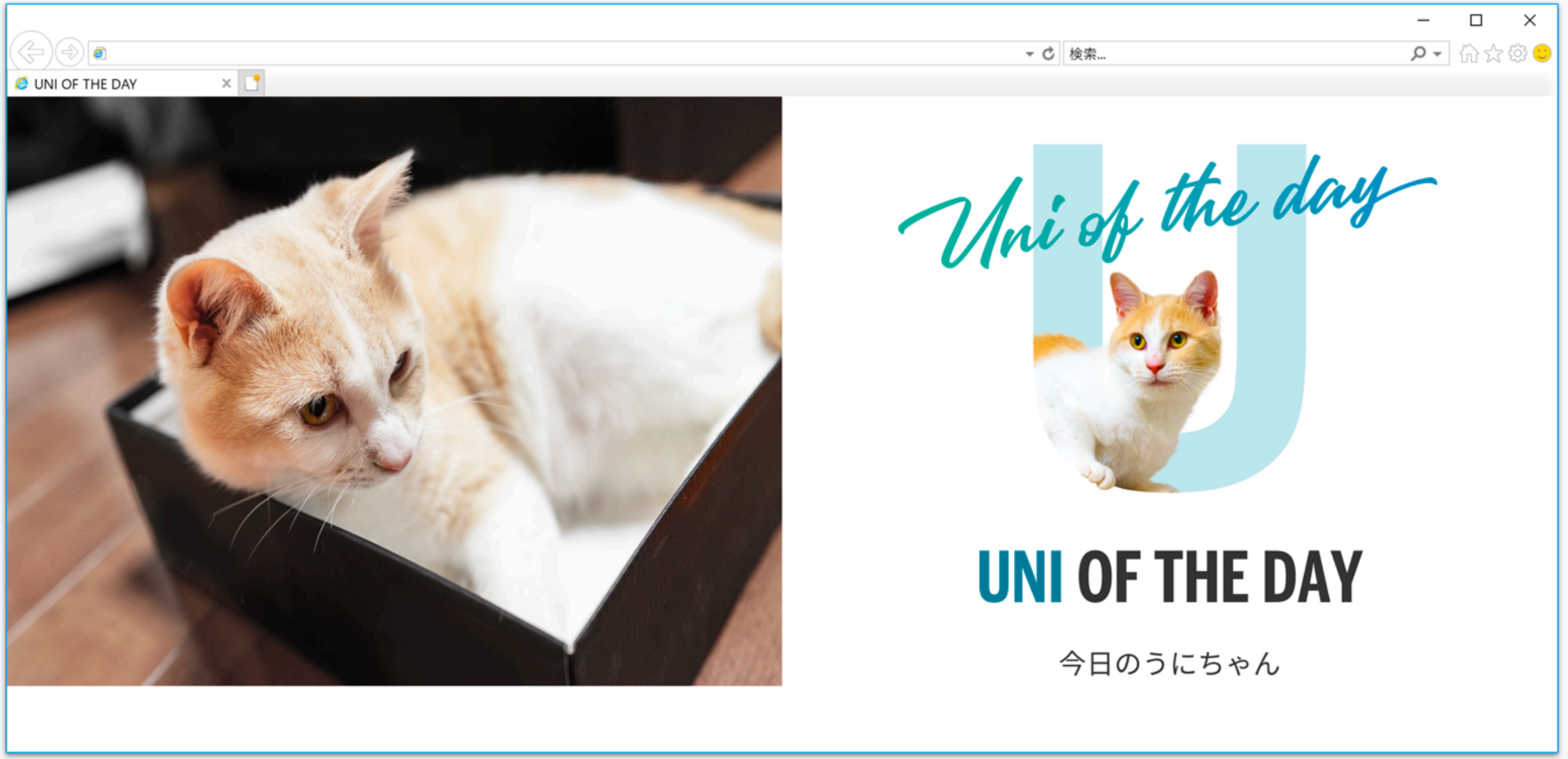
```
.container {
  display: grid;
  grid-template:
    "visual number expression" 1fr
    "visual other  other" 220px /
    40% 120px 1fr;
}

.visual {
  grid-area: visual;
}
```

# AutoprefixerでgapをIE 11向けに変換

after

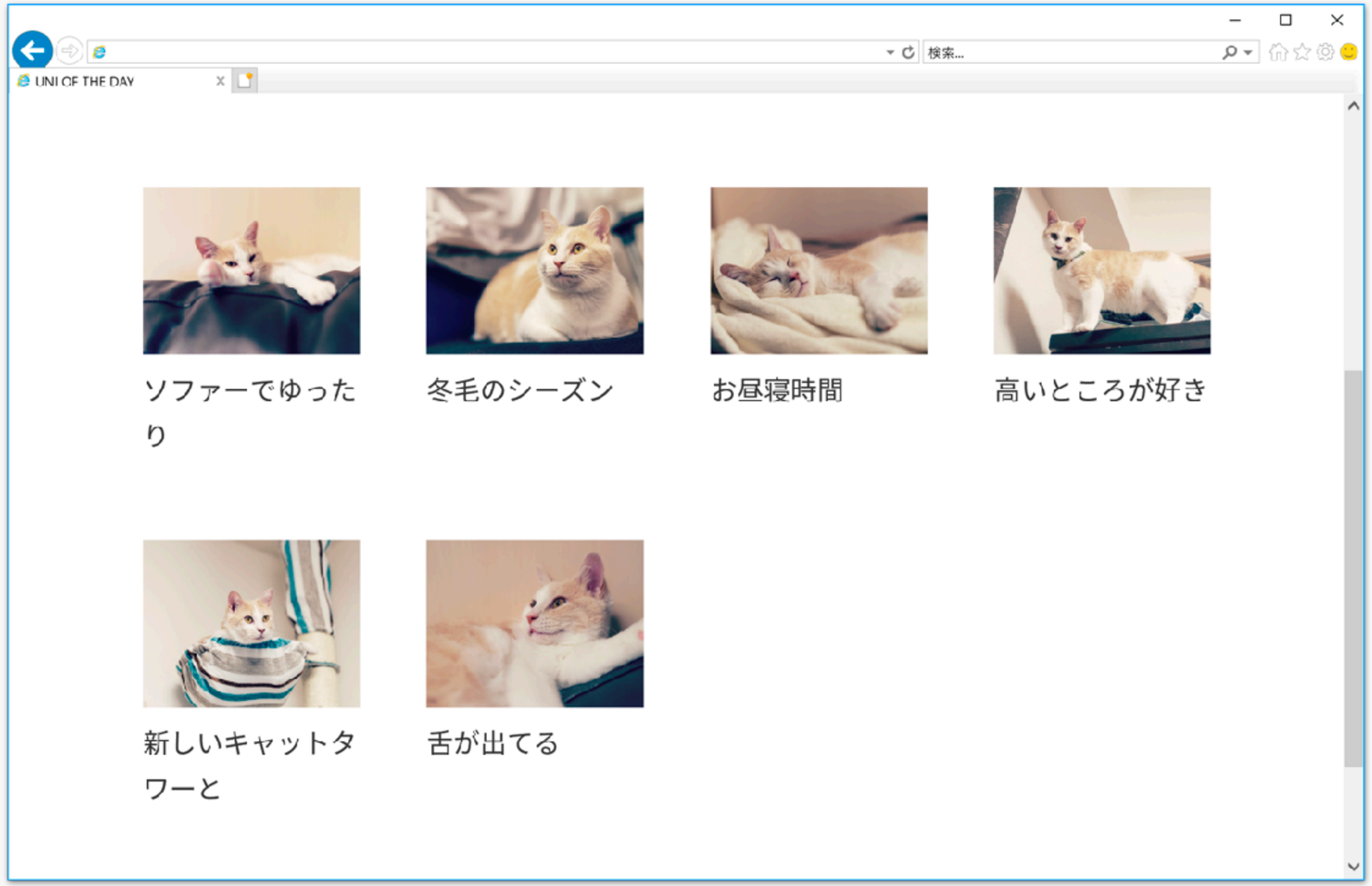
```
.container {
  /* 中略 */

  gap: 10px;
  -ms-grid-rows: 1fr 10px 220px;
  -ms-grid-columns: 40% 10px 120px 10px 1fr;
  grid-template:
    "visual number  expression" 1fr
    "visual other   other"      220px /
    40%     120px   1fr;
}

.visual {
  -ms-grid-row: 1;
  -ms-grid-row-span: 3;
  -ms-grid-column: 1;
  grid-area: visual;
}
```

# sample1をIE 11に対応したもの




sample1_ie11.html / sample1_ie11.css

# sample2をIE 11に対応したもの




sample2_ie11.html / sample2_ie11.css

※ grid-templateの記述に注意が必要（フォローアップ参照）

# 現場で働くコーダーのためのCSS Grid + モダンコーディング

1. CSS Grid
2. 抑えておきたいCSSの新機能
    1. 最近のモダンブラウザーで使えるCSSの機能
    2. 近い将来使えるようになるCSSの機能
3. 情報のキャッチアップ方法

# Microsoft Edgeの対応により
# 全モダンブラウザで使えるようになった
# CSSの機能

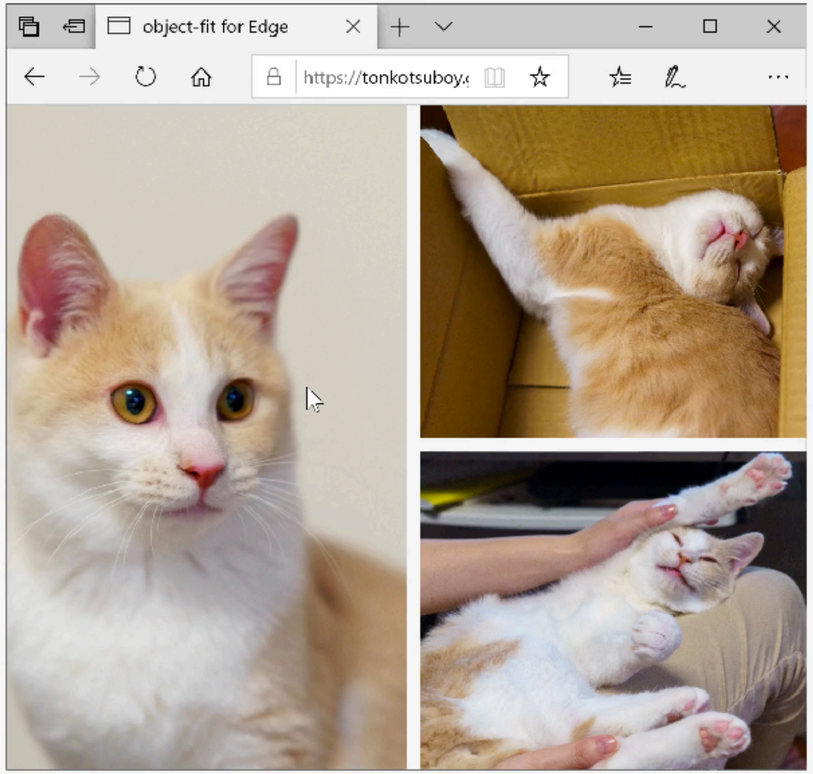
# CSS変数

# CSS変数：CSSで変数が使える

```css
/* 通常時は黒色テキスト・白色背景 */
:root {
  --text-color: #000000;
}

/* ダークモードでは灰色テキスト・黒色背景を指定する */
@media (prefers-color-scheme: dark) {
  :root {
    --text-color: #ffffff;
  }
}

/* 通常時は黒色、ダークモードで白色に変わるテキスト */
p {
  color: var(--text-color);
}
```

# 画像の比率を保ちながら
# 変形できる
# 「object-fit」プロパティ

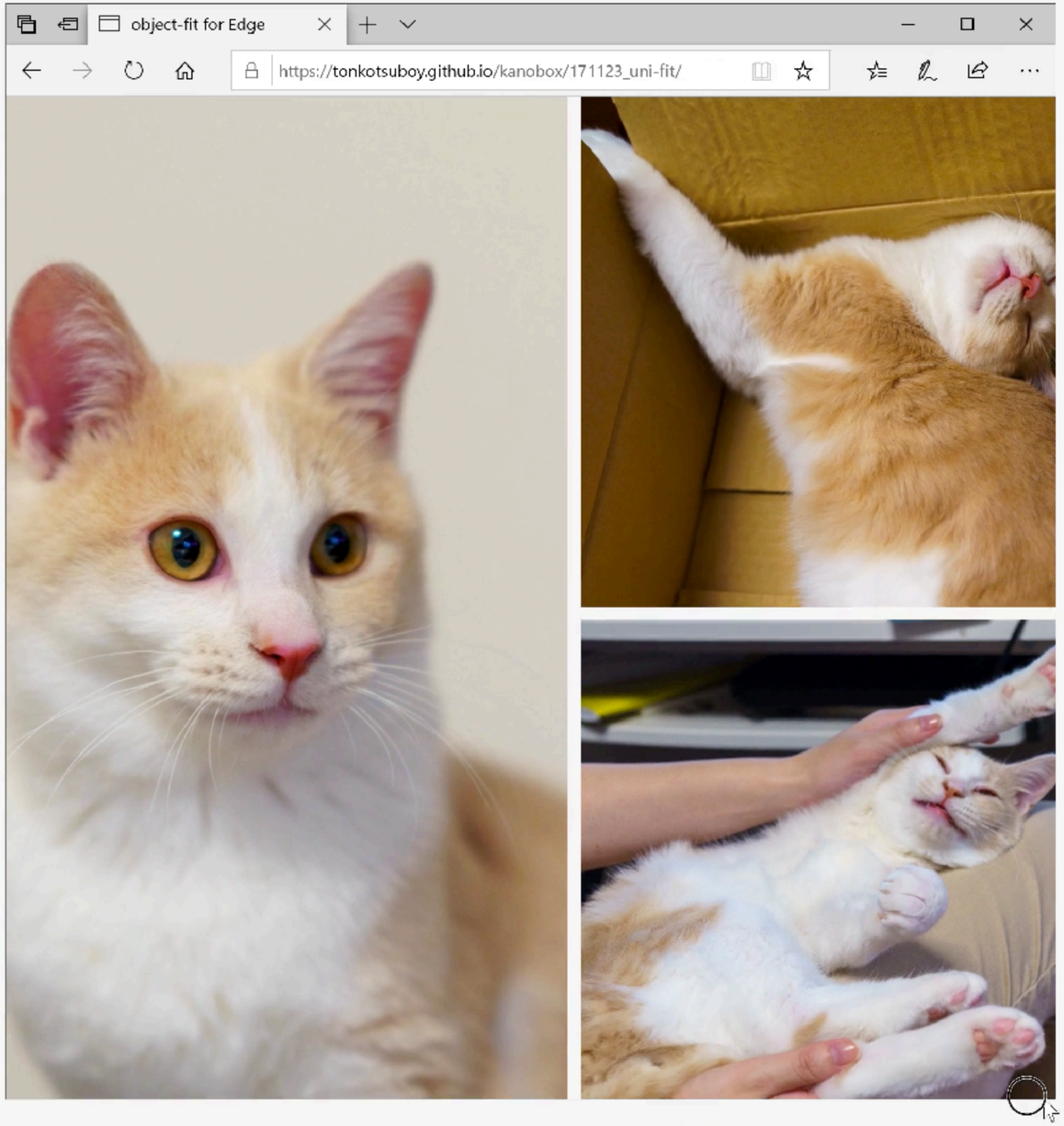
# imgタグで設定した画像の比率を保つ




HTMLコード

```
<img class="image1" src="cat_1.png" alt="">
<img class="image2" src="cat_2.png" alt="">
<img class="image3" src="cat_3.png" alt="">
```

# imgタグで設定した画像の比率を保つ




HTMLコード

```
<img class="image1" src="cat_1.png" alt="">
<img class="image2" src="cat_2.png" alt="">
<img class="image3" src="cat_3.png" alt="">
```

# imgタグで設定した画像の比率を保つ

## object-fitプロパティ

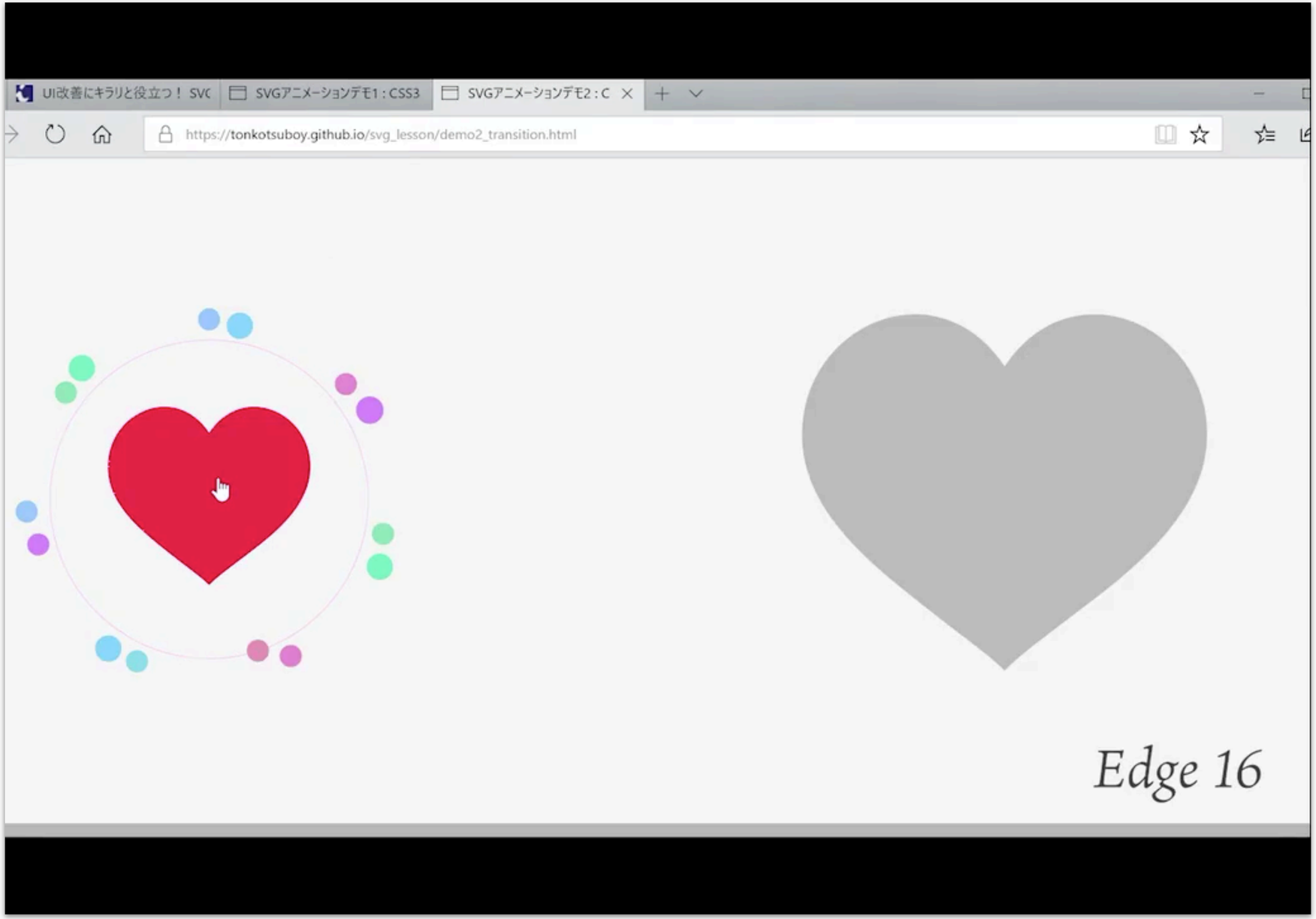
（2017年10月リリースのEdge v16で対応）

```
img {
  object-fit: cover;
}
.image1 {
  object-position: 30% 50%;
}
.image2 {
  object-position: 0 0;
}
```

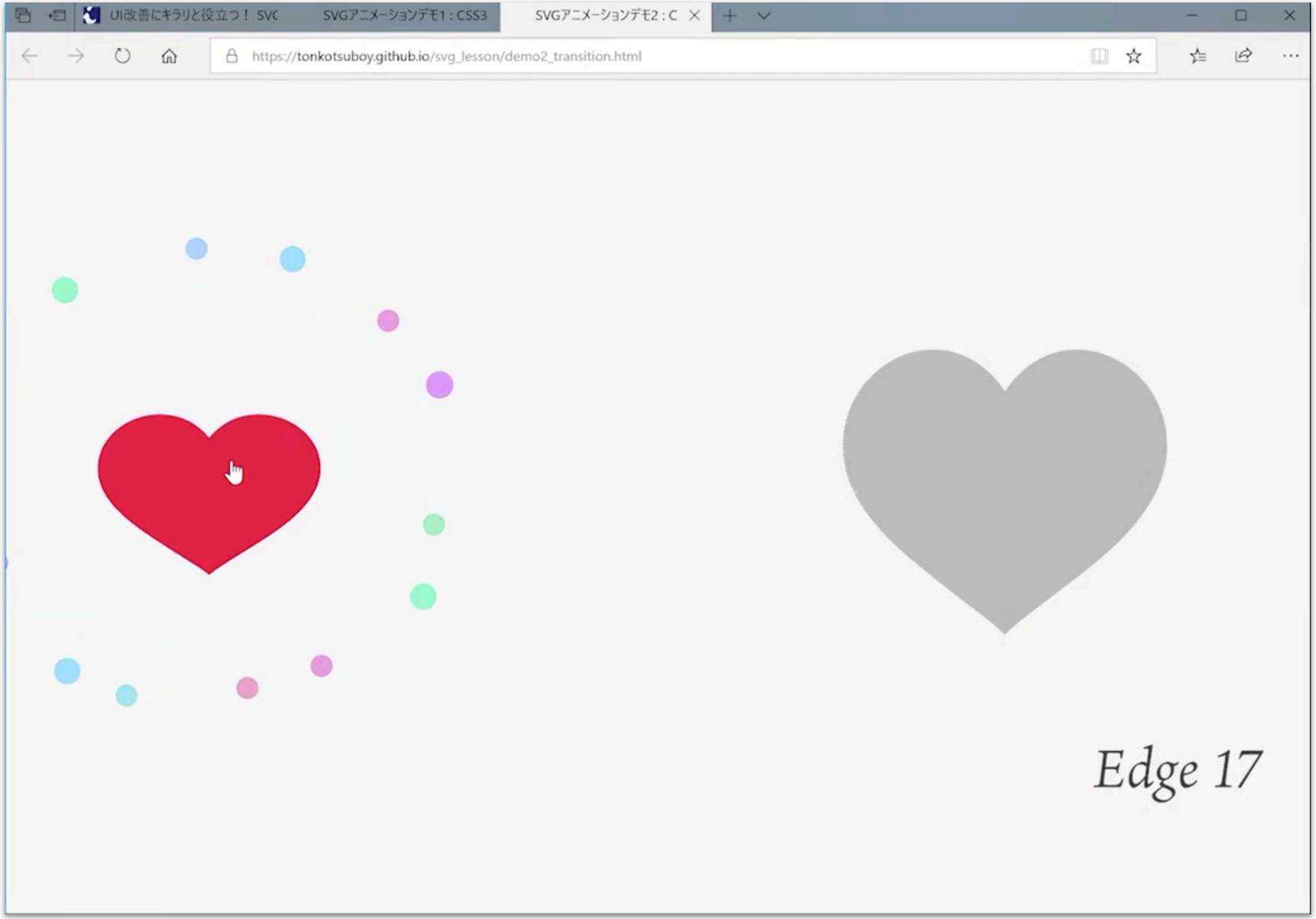
うに fit

# SVG要素を
# CSS Transformsで変形する

# SVG要素をCSS Transformで変形する



# SVG要素をCSS Transformで変形する



# SVG要素をCSS Transformで変形する

## SVG要素をtransformで変形する

### (2018年4月リリースのEdge v17で対応)

```
/* SVGのハート図形のアニメーション（抜粋） */
@keyframes heartAnime {
  0% {
    transform: scale(0);
  }
  60% {
    transform: scale(1.2, 1.2);
  }
  80% {
    transform: scale(0.95, 1.05) translate(0%, -3%);
  }
  100% {
    transform: scale(1.0, 1.0) translate(0%, 0%);
  }
}
```

UI改善にキラリと役立つ！ SVGアニメーションの作り方まとめ - ICS MEDIA

# 画像にマスクをかける
# maskプロパティ

# 画像にマスクをかける


Edge 17
CodePen - #自分のお金
https://codepen.io/tonkotsuboy/full/rroPzz/
#自分のお金で寿司が食べたい
A PEN BY Takeshi Kano PRO
Fork
Change View
#自分のお金で寿司が食べたい!
Edge 18
CodePen - #自分のお金
https://codepen.io/tonkotsuboy/full/rroPzz/
#自分のお金で寿司が食べたい
A PEN BY Takeshi Kano PRO
Fork
Change View
#自分のお金で寿司が食べたい!

# 画像にマスクをかける

## maskプロパティ

（2018年10月リリースのEdge v18で対応）

```
.mask {
  mask-image: url("../images/mymask.png");
  mask-repeat: no-repeat;
  mask-position: center;
  mask-size: contain;
}
```

#自分のお金で寿司が食べたい

# 一つのフォントファイルが
# 複数のフォントのように動作する
# 「variable font」

# variable font：1つのフォントから複数の見栄え

## 複数のフォントファイルを読み込まず、CSSで見栄えを制御可能



Font Playground

# variable font：1つのフォントから複数の見栄え

複数のフォントファイルを読み込まず、CSSで見栄えを制御可能

```
p {
   font-variation-settings:
      "wght" 25.31,
      "XHGT" 353,
      "opsz" 13.03;
}
```

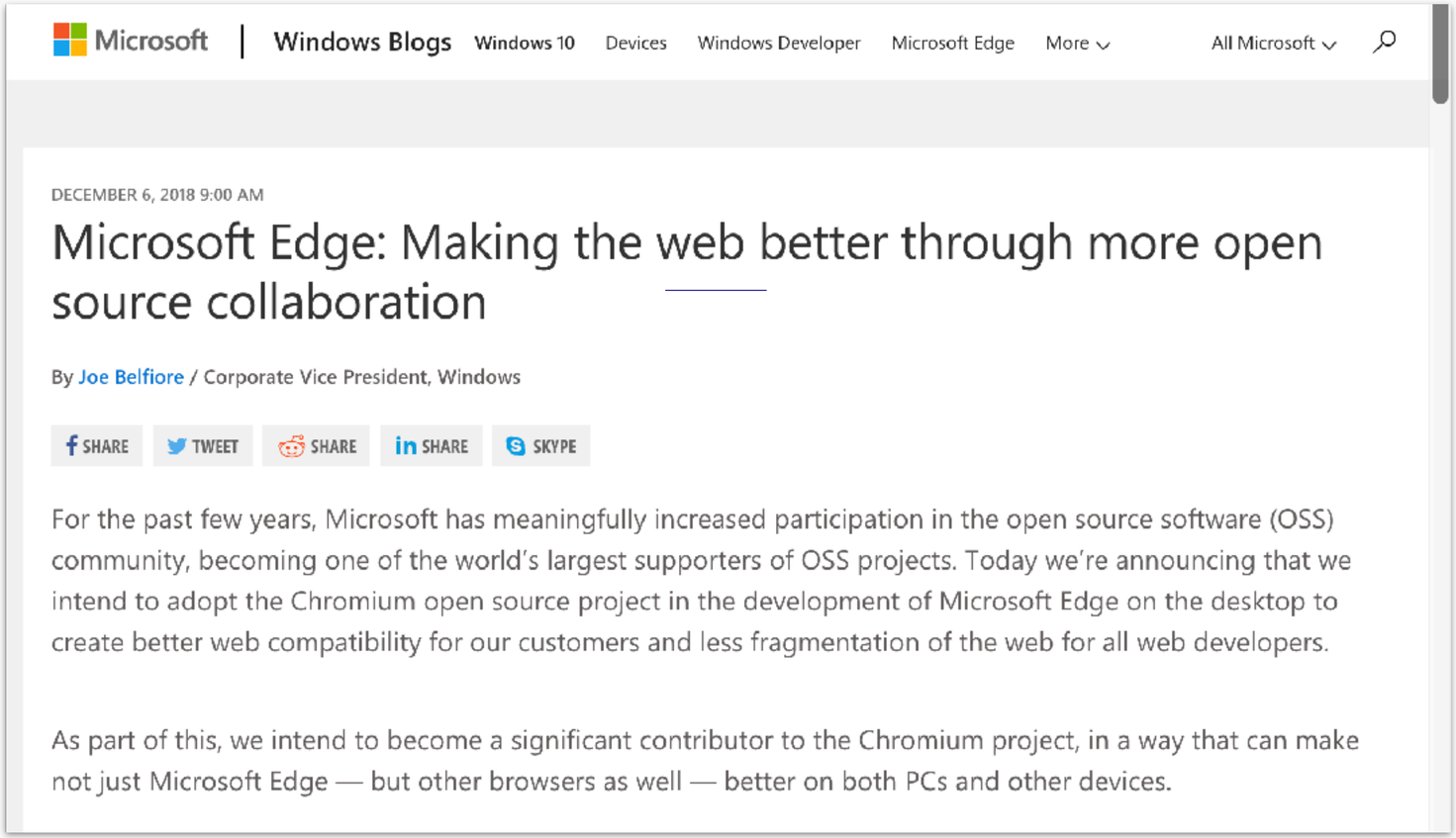
# 現場で働くコーダーのためのCSS Grid + モダンコーディング

1. CSS Grid
2. 抑えておきたいCSSの新機能
    1. 最近のモダンブラウザーで使えるCSSの機能
    2. 近い将来使えるようになるCSSの機能
3. 情報のキャッチアップ方法

# Microsoft Edgeの次バージョンより
# 全モダンブラウザ向けCSSの機能は
# 更に増える

# Microsoft EdgeがChromiumベースになる

## 2018年12月に発表




Windows Experience Blog （2018年12月）

# 「Chromium」とは？

Chromium（クロミウム）

- Google Chromeのもとなっているプログラム
- HTML・CSSのレンダリング：Blink（ブリンク）
- JavaScriptの実行：V8

Chromiumのロゴ

# Edgeの次バージョンからはChromeと同じ構成になる

Edge v18までの構成

- HTML・CSSのレンダリング：EdgeHTML
- JavaScriptの実行：Charka（チャクラ）

次のEdgeからの構成

- HTML・CSSのレンダリング：Blink
- JavaScriptの実行：V8

## ちなみにチャクラとは…

サンスクリットで円、円盤、車輪、ろくろを意味する語。

ヒンドゥー教のタントラやハタ・ヨーガ、仏教の後期密教では、人体の頭部、胸部、腹部などにあるとされる中枢を指す言葉として用いられる。

Wikipedia - チャクラより

# EdgeのChromium化によりCSSの対応が増える

## Edgeの次バージョンで使えるCSS機能の一例

will-change property

prefers-reduced-motion media query

prefers-color-scheme media query

display: flow-root

CSS text-orientation

CSSOM Scroll-behavior

#rrggbbaa hex color notation

:placeholder-shown CSS pseudo-class

CSS Motion Path

CSS font-rendering controls

:focus-within CSS pseudo-class

CSS Environment Variables env()

:default CSS pseudo-class

CSS Conical Gradients

Case-insensitive CSS attribute selectors

CSS background-blend-mode

CSS all property

CSS Font Loading

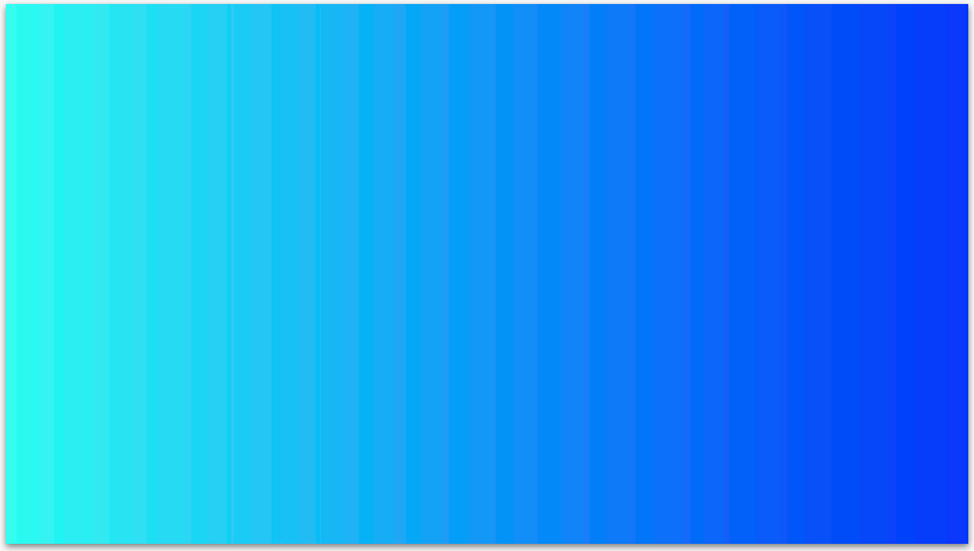
:matches() CSS pseudo-class

CSS display: contents

# 「円錐状」のグラデーション
# Conical Gradients

# 従来のグラデーション

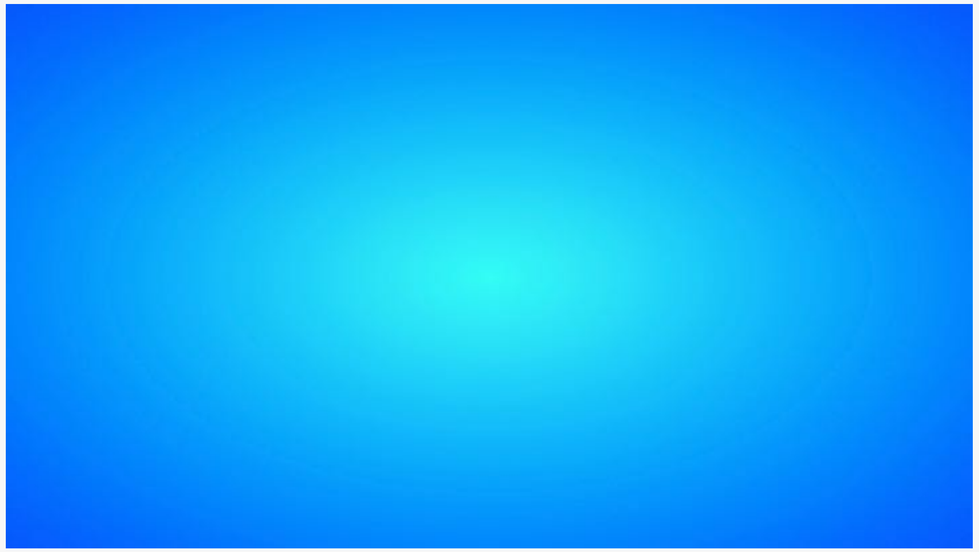
## 線形



```
.box {
  background-image: linear-gradient(
      to right,
      hsl(177, 100%, 50%),
      hsl(232, 100%, 50%)
  );
}
```

# 従来のグラデーション

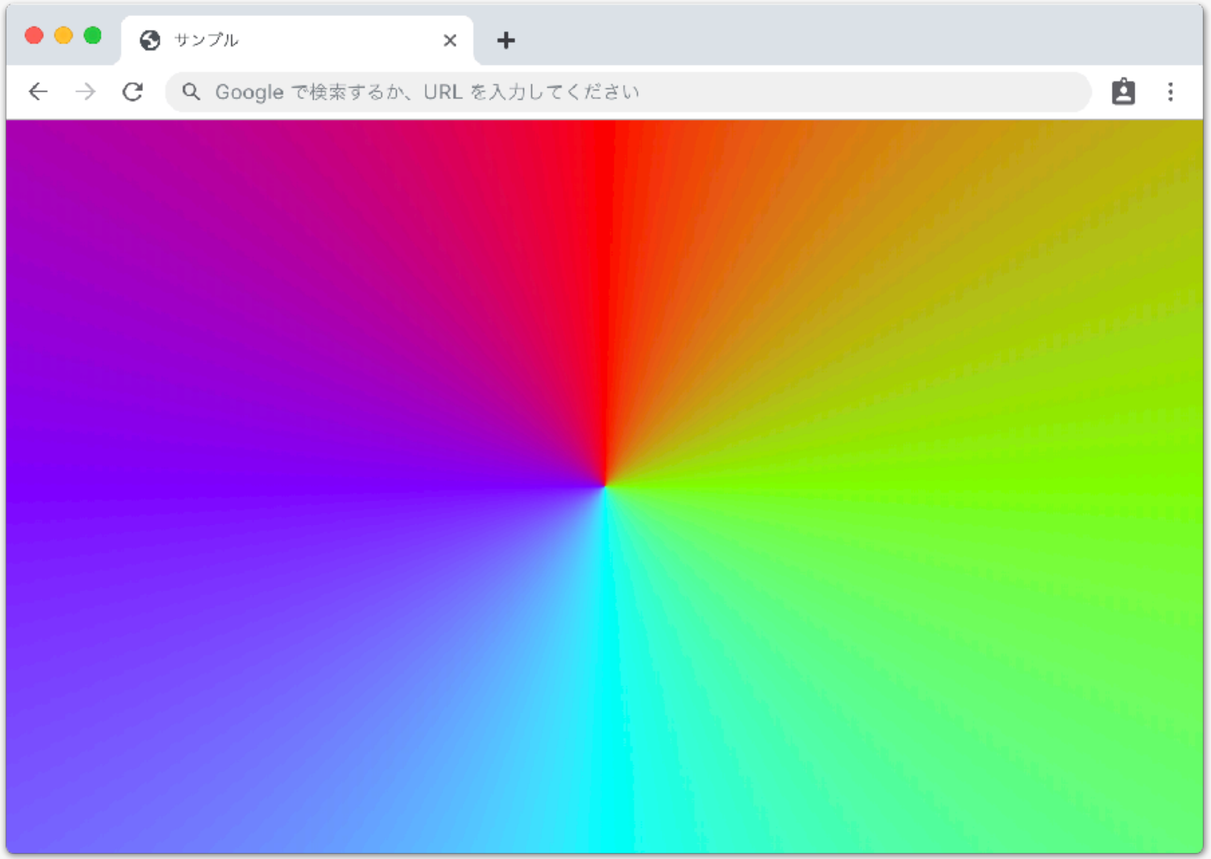
## 円形



```
.box {
  background-image: radial-gradient(
      hsl(177, 100%, 50%),
      hsl(232, 100%, 50%)
  );
}
```

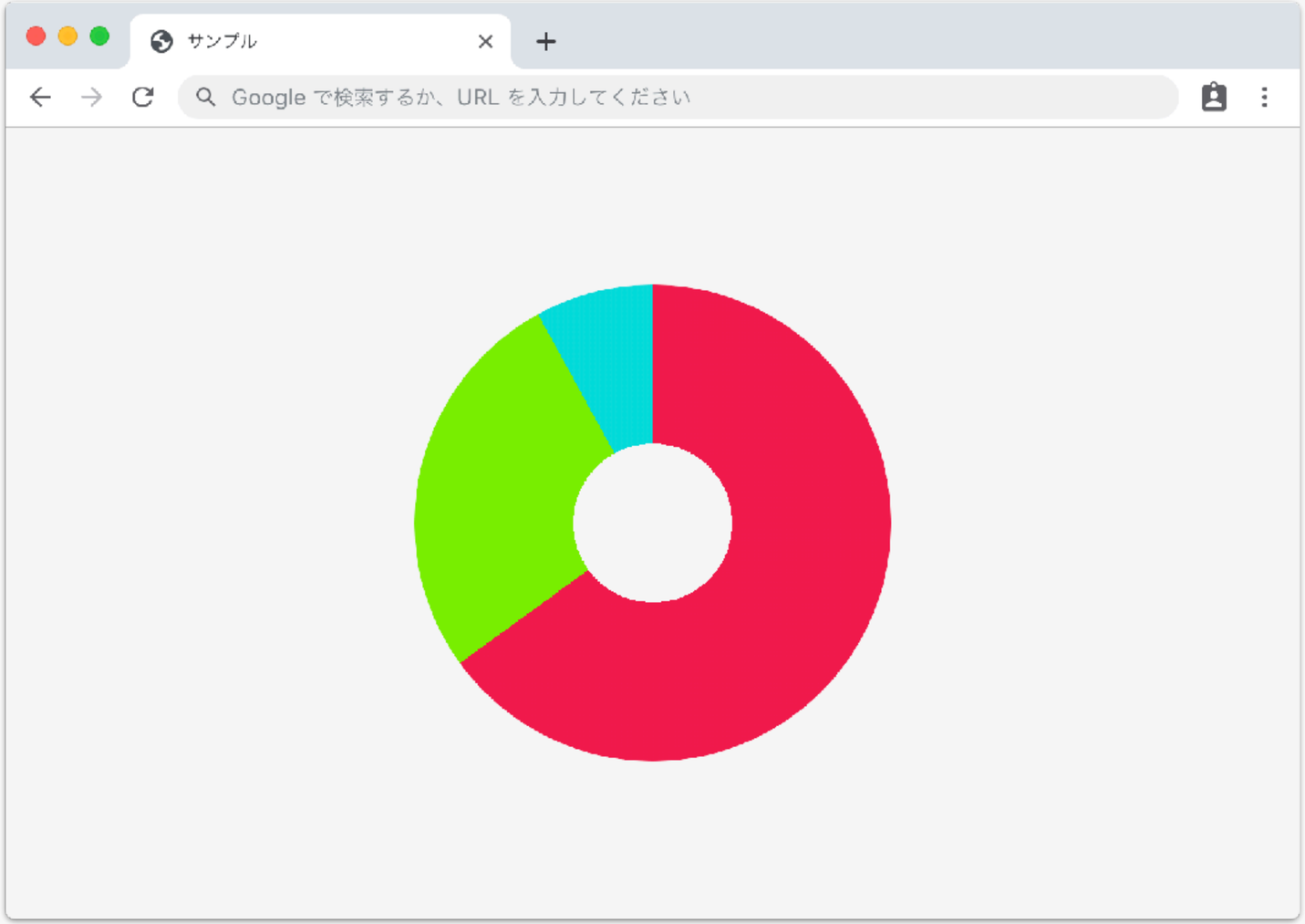
# Conical Gradients

## 「円錐状」のグラデーションを指定する値

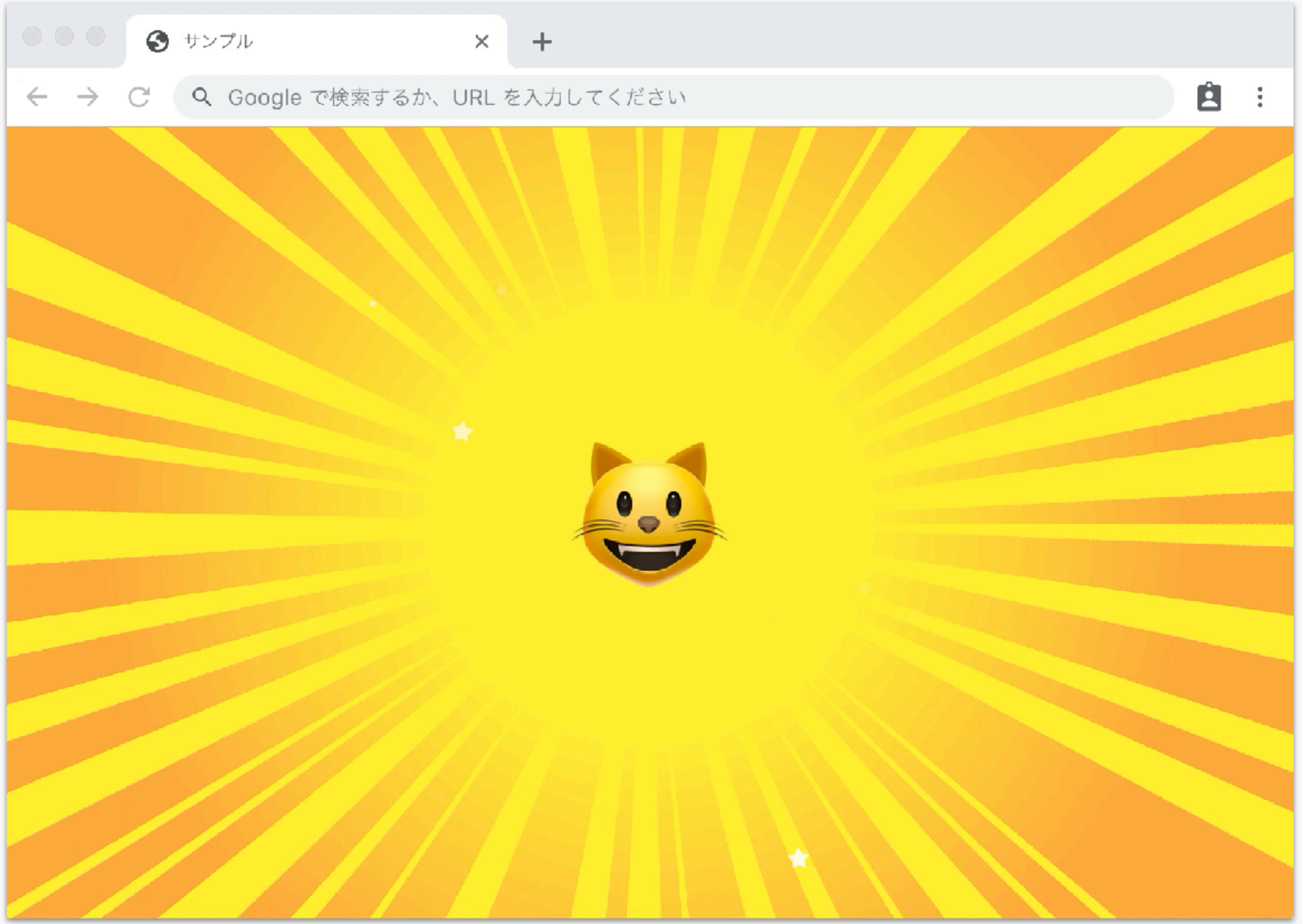



```
.box {
  background-image: conic-gradient(
    hsl(0, 100%, 50%) 0,
    hsl(90, 100%, 50%) 90deg,
    hsl(180, 100%, 50%) 180deg,
    hsl(270, 100%, 50%) 270deg,
    hsl(360, 100%, 50%) 360deg
  );
}
```

# Conical Gradientsで円グラフの表現

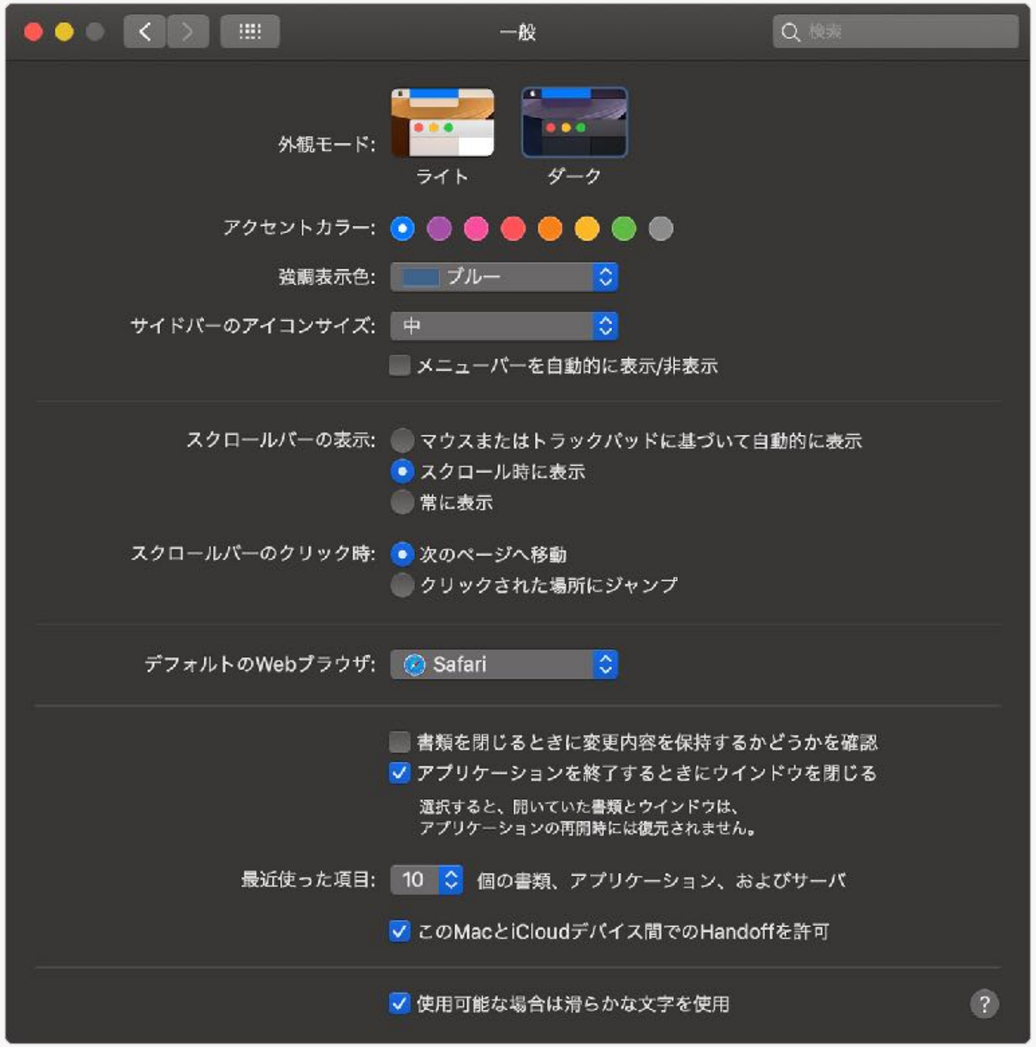


# Conical Gradientsで集中線の表現



# OSのダークモードを判別できる prefers-color-scheme media query

# 各OSにはダークモードがある




- Windows
- macOS
- iOS（13〜）
- Android※

※Androidはダークテーマという名称

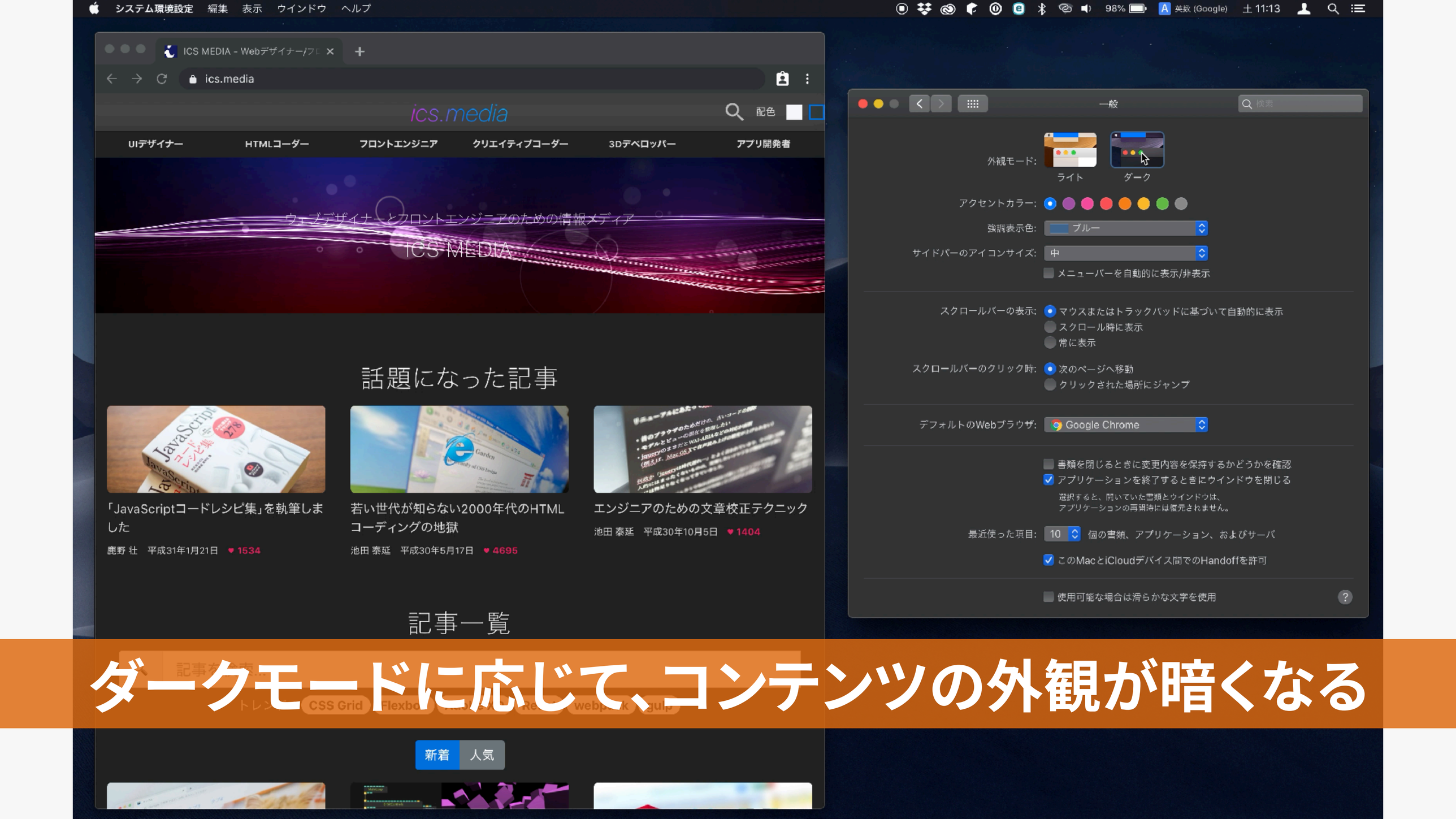

QuickTime Player ファイル 編集 表示 ウインドウ ヘルプ
99% 英数 (Google) 土 11:12
学部・大学院等 | 九州大学 〈KYUS
kyushu-u.ac.jp/ja/faculty
受験生 在学生 企業・研究者 地域・一般 卒業生
日本語 ENGLISH
九州大学
KYUSHU UNIVERSITY
Google カスタム検索
見えにくい場合 標準 黒
教職員/学生限定
アクセス・キャンパスマップ
九州大学について 学部・大学院等 入試・入学 教育・学生支援 研究・産学官民連携 国際交流・留
学部・大学院等
Schools & Centers
附置研究所
国際研究所
情報基盤研究開発センター
アジア・オセアニア研究教
学内共同教育研究センター
先導的研究センター
機構・推進室等
一般
外観モード:
ライト
ダーク
アクセントカラー:
強調表示色:
ブルー
サイドバーのアイコンサイズ:
中
メニューバーを自動的に表示/非表示
スクロールバーの表示:
マウスまたはトラックパッドに基づいて自動的に表示
スクロール時に表示
常に表示
スクロールバーのクリック時:
次のページへ移動
クリックされた場所にジャンプ
デフォルトのWebブラウザ:
Google Chrome
書類を閉じるときに変更内容を保持するかどうかを確認
アプリケーションを終了するときにウインドウを閉じる
選択すると、開いていた書類とウインドウは、
アプリケーションの再開時には復元されません。
最近使った項目:
10
個の書類、アプリケーション、およびサーバ
このMacとiCloudデバイス間でのHandoffを許可
使用可能な場合は滑らかな文字を使用
ダークモードにしても、コンテンツの外観は明るいママ問題


システム環境設定 編集 表示 ウインドウ ヘルプ
99% 英数 (Google) 土 11:12
学部・大学院等 | 九州大学 (KYUS
kyushu-u.ac.jp/ja/faculty
受験生 在学生 企業・研究者 地域・一般 卒業生
日本語 ENGLISH
九州大学
KYUSHU UNIVERSITY
Google カスタム検索
見えにくい場合 標準 黒
教職員/学生限定
アクセス・キャンパスマップ お問い
九州大学について 学部・大学院等 入試・入学 教育・学生支援 研究・産学官民連携 国際交流・留
学部・大学院等
Schools & Centers
附置研究所 国際研究所 情報基盤研究開発センター エネルギー研究教
アジア・オセアニア研究教 学内共同教育研究センター 先導的研究センター 機構・推進室等
一般
外観モード: ライト ダーク
アクセントカラー:
強調表示色: ブルー
サイドバーのアイコンサイズ: 中
メニューバーを自動的に表示/非表示
スクロールバーの表示: マウスまたはトラックパッドに基づいて自動的に表示
スクロール時に表示
常に表示
スクロールバーのクリック時: 次のページへ移動
クリックされた場所にジャンプ
デフォルトのWebブラウザ: Google Chrome
書類を閉じるときに変更内容を保存するかどうかを確認
アプリケーションを終了するときにウインドウを閉じる
選択すると、開いていた書類とウインドウは、
アプリケーションの再開時には復元されません。
最近使った項目: 10 個の書類、アプリケーション、およびサーバ
このMacとiCloudデバイス間でのHandoffを許可
使用可能な場合は滑らかな文字を使用
ダークモードにしても、コンテンツの外観は明るいママ問題

# prefers-color-schemeを使う場合

```
/* 通常は背景を白にする */
body {
  background-color: white;
}

/* ダークモードのときは背景を黒にする */
@media (prefers-color-scheme: dark) {
  body {
    background-color: black;
  }
}
```

# ダークモードに応じて、コンテンツの外観が暗くなる


システム環境設定 編集 表示 ウインドウ ヘルプ
ICS MEDIA - Webデザイナー/フ
ics.media
ics.media
配色
UIデザイナー
HTMLコーダー
フロントエンジニア
クリエイティブコーダー
3Dデベロッパー
アプリ開発者
ウェブデザイナーとフロントエンジニアのための情報メディア
ICS MEDIA
話題になった記事
「JavaScriptコードレシピ集」を執筆しました
鹿野 壮 平成31年1月21日 ♥ 1534
若い世代が知らない2000年代のHTMLコーディングの地獄
池田 泰延 平成30年5月17日 ♥ 4695
エンジニアのための文章校正テクニック
池田 泰延 平成30年10月5日 ♥ 1404
記事一覧
CSS Grid
新着
人気
一般
外観モード:
ライト
ダーク
アクセントカラー:
強調表示色:
ブルー
サイドバーのアイコンサイズ:
中
メニューバーを自動的に表示/非表示
スクロールバーの表示:
マウスまたはトラックパッドに基づいて自動的に表示
スクロール時に表示
常に表示
スクロールバーのクリック時:
次のページへ移動
クリックされた場所にジャンプ
デフォルトのWebブラウザ:
Google Chrome
書類を閉じるときに変更内容を保持するかどうかを確認
アプリケーションを終了するときにウインドウを閉じる
選択すると、開いていた書類とウインドウは、アプリケーションの再開時には復元されません。
最近使った項目:
10
個の書類、アプリケーション、およびサーバ
このMacとiCloudデバイス間でのHandoffを許可
使用可能な場合は滑らかな文字を使用
ダークモードに応じて、コンテンツの外観が暗くなる

# 「CSS変数」と組み合わせて更に便利に

```
/* 通常時は黒色テキスト・白色背景 */
:root {
  --text-color: #000000;
}

/* ダークモードでは灰色テキスト・黒色背景を指定する */
@media (prefers-color-scheme: dark) {
  :root {
    --text-color: #ffffff;
  }
}

/* 通常時は黒色、ダークモードで白色に変わるテキスト */
p {
  color: var(--text-color);
}
```

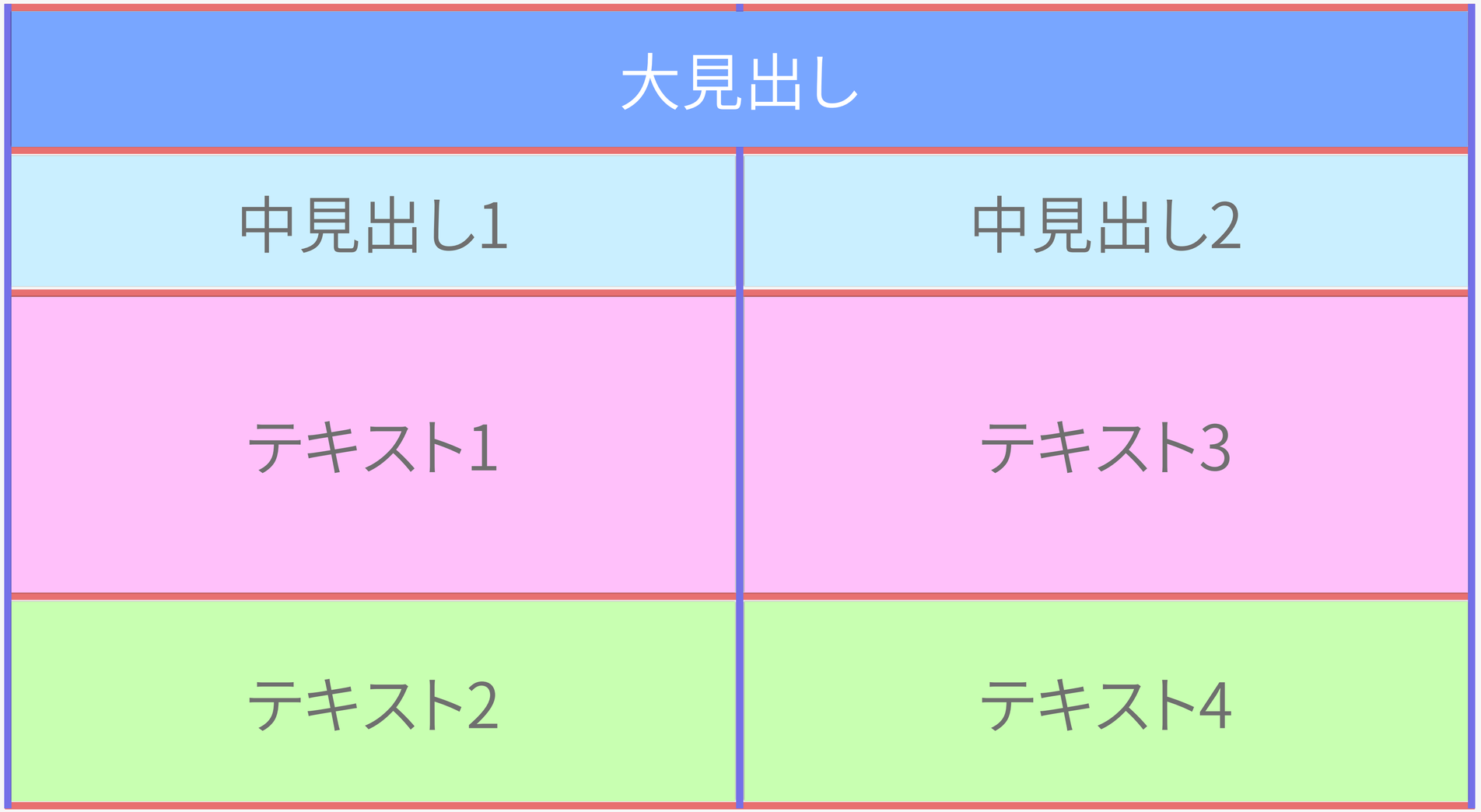
# OSのアニメーションが
# OFFかをどうかを判別できる
# prefers-reduced-motion media query

# prefers-color-scheme

```
/* 通常は各要素にアニメーションを設定する */
.element1,
.element2 {
  animation: 1s myAnimation;
}

/* OSのアニメーションがOFF設定の場合は
   CSSアニメーションもOFFにする */
@media (prefers-reduced-motion: reduce) {
  .element1,
  .element2 {
    animation: none;
  }
}
```

# display: contents

# CSS Gridを使いつつ、情報の意味付けをしたい



# CSS Gridを使いつつ、情報の意味付けをしたい

## 理想のHTML

```
<div class="container">
  <h1>大見出し</h1>
  <section>
    <h2>中見出し</h2>
    <p>テキスト1</p>
    <p>テキスト2</p>
  </section>
  <section>
    <h2>中見出し</h2>
    <p>テキスト3</p>
    <p>テキスト4</p>
  </section>
</div>
```

## CSS

```
.container {
  grid-template:
    "header header" auto
    "title1 title2" 100px
    "text1  text3 " 1fr
    "text2  text4 " 40px /
    50%     50%;
}
```

# アイテムはすべて並列に並べなければいけない

## 現実のHTML

```
<div class="container">
  <h1>大見出し</h1>
  <h2>中見出し</h2>
  <p>テキスト1</p>
  <p>テキスト2</p>
  <h2>中見出し</h2>
  <p>テキスト3</p>
  <p>テキスト4</p>
</div>
```

## CSS

```
.container {
  grid-template:
    "header header" auto
    "title1 title2" 100px
    "text1  text3 " 1fr
    "text2  text4 " 40px /
    50%     50%;
}
```

# display: contentsで解決

子要素（または疑似要素）によって置換されるボックス

```
section {
  display: contents;
}
```

# Gridを使いつつ、セマンティックなコーディングが実現

## HTML

```
<div class="container">
  <h1>大見出し</h1>
  <section>
    <h2>中見出し</h2>
    <p>テキスト1</p>
    <p>テキスト2</p>
  </section>
  <section>
    <h2>中見出し</h2>
    <p>テキスト3</p>
    <p>テキスト4</p>
  </section>
</div>
```

## CSS

```
.container {
  grid-template:
    "header header" auto
    "title1 title2" 100px
    "text1  text3 " 1fr
    "text2  text4 " 40px /
    50%     50%;
}

section {
  display: contents;
}
```

# EdgeのChromium化によりCSSの対応が増える

## Edgeの次バージョンで使えるCSS機能の一例

will-change property

prefers-reduced-motion media query

prefers-color-scheme media query

display: flow-root

CSS text-orientation

CSSOM Scroll-behavior

#rrggbbaa hex color notation

:placeholder-shown CSS pseudo-class

CSS Motion Path

CSS font-rendering controls

:focus-within CSS pseudo-class

CSS Environment Variables env()

:default CSS pseudo-class

CSS Conical Gradients

Case-insensitive CSS attribute selectors

CSS background-blend-mode

CSS all property

CSS Font Loading

:matches() CSS pseudo-class

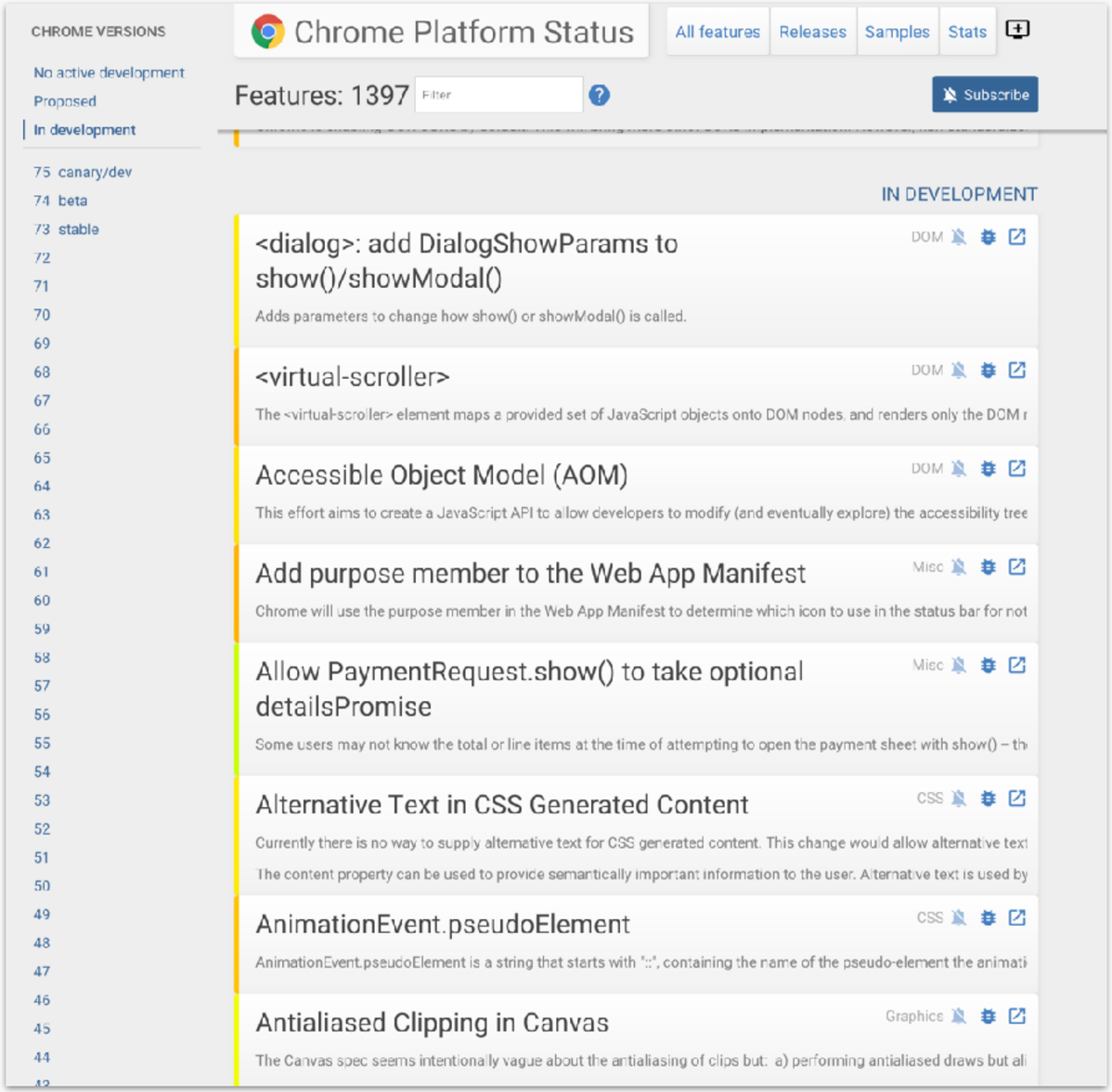
CSS display: contents

# 現場で働くコーダーのためのCSS Grid + モダンコーディング

1. CSS Gridとボックスレイアウト
2. 抑えておきたいCSSの新機能
3. 情報のキャッチアップ方法

# どうやってCSSの
# 最新機能や対応状況を追うか？

# Platform Status



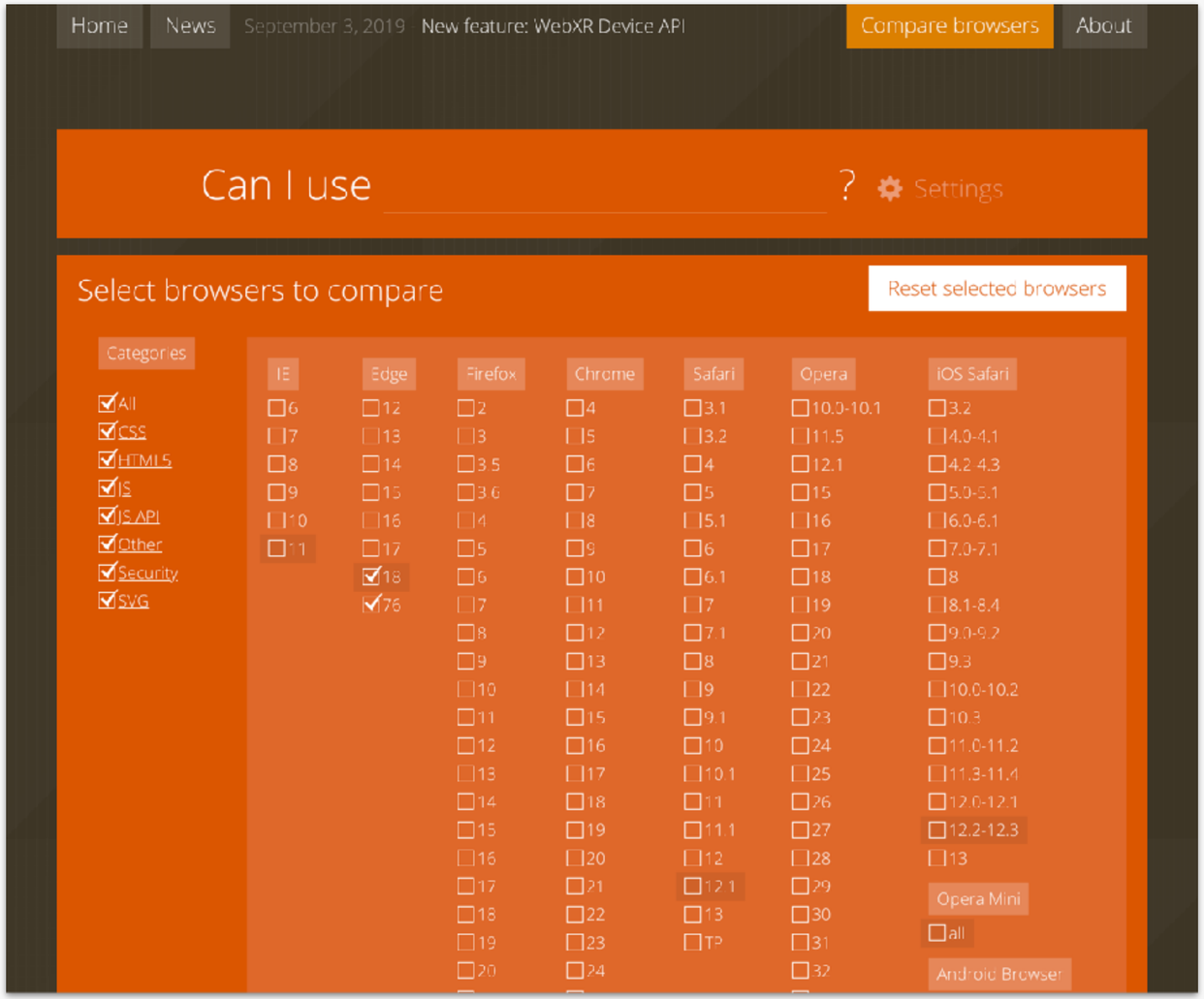

- ブラウザで開発中の機能や実装済みの機能が確認可能
- キャプチャーはChromeのもの

https://www.chromestatus.com/features

# 各ブラウザ向けのPlatform Status

- **Chrome Platform Status**
  https://www.chromestatus.com
- **WebKit Feature Status**
  https://webkit.org/status/
- **Firefox Platform Status**
  https://platform-status.mozilla.org/
- **Microsoft Edge web platform features status**
  https://developer.microsoft.com/microsoft-edge/platform/status/

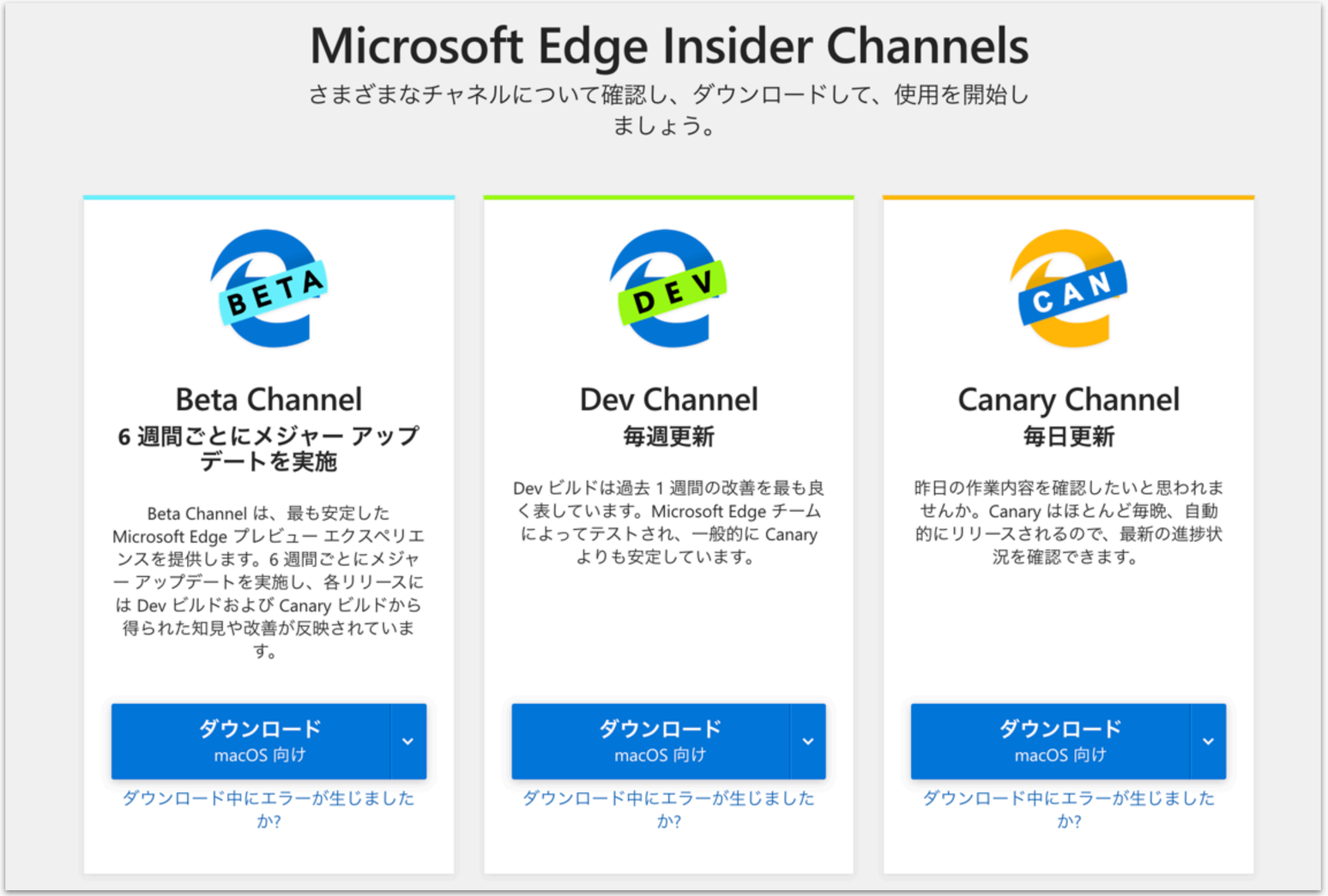
# Can I Use




- 特に、Compare Browsers機能が便利
- ブラウザの新バージョンリリース時はチェックしておく

https://caniuse.com

# 各ブラウザーの次バージョンを使う



# まとめ

# まとめ

- CSSには、
  新しい機能が増えてきている
- 新しい機能とは、
  これまでの開発のスピードを上昇させ、ラクにしてくれる
- 積極的に取り入れ、作業時間を減らし、
  コンテンツのブラッシュアップに時間をかけよう

# 当たり前だと思っていた手法を見直して
# より便利にウェブ開発をしよう!

# ご清聴ありがとうございました

## ICS MEDIAやTwitterでもウェブテクノロジーの情報を発信中

ics.media

鹿野 壮
@tonkotsuboy_com