HTML/CSS実習

10. CSSレイアウト

株式会社ジードライブ

今回学ぶこと

- CSSの初期化
- CSSによるレイアウトの指定方法
  - ボックスを中央に配置する
  - ボックスを横に並べて配置する(フレックスボックス)

ブラウザとレンダリング

- 同じWebサイトでも閲覧環境ごとに表示の違いがある
  - 同じ**HTML**や**CSS**であっても、ブラウザごとに**レンダリング**が異なる

Webサイトを構築するための言語

解釈／表現のこと

ブラウザA

ブラウザB

Windows、Mac、LinuxなどのOS、スマートフォンやタブレット、PCなどのデバイス、そして各ブラウザの仕様ごとに表示が異なる

CSSの初期化とは

- ブラウザにあらかじめ組み込まれているデフォルトのCSS(ユーザーエージェントCSS)をフラットな状態にすること
  - デフォルトのCSSはブラウザによって異なる場合がある
  - デフォルトのCSSをフラットな状態にしてからレイアウト作業に取り組むことで、作業の効率化やブラウザ間の統一を図ることができる

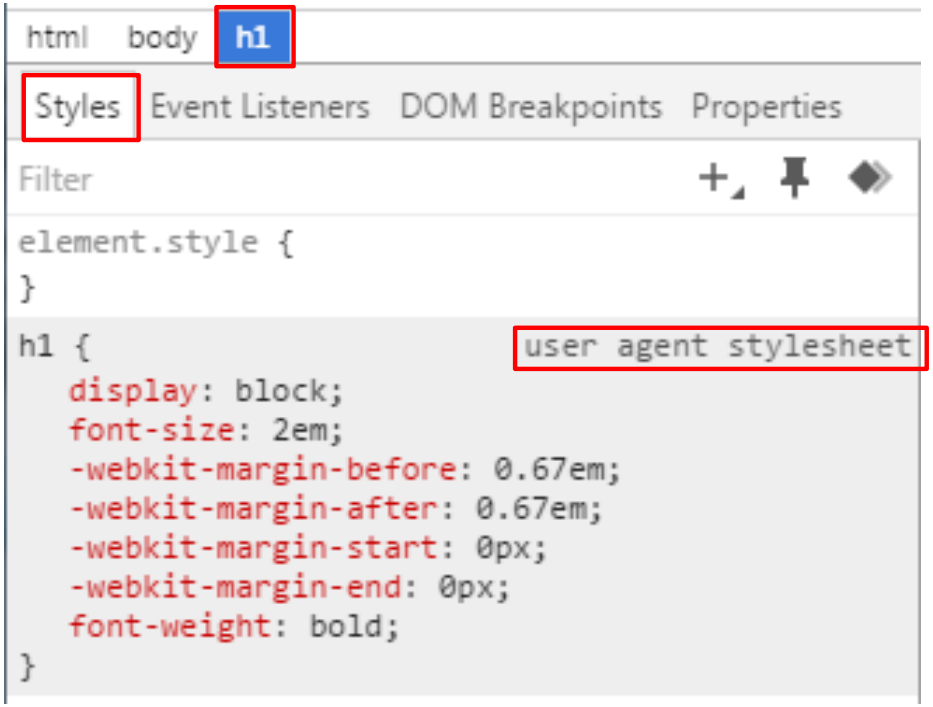
ユーザーエージェントCSSで余白や文字の大きさなどが設定されている

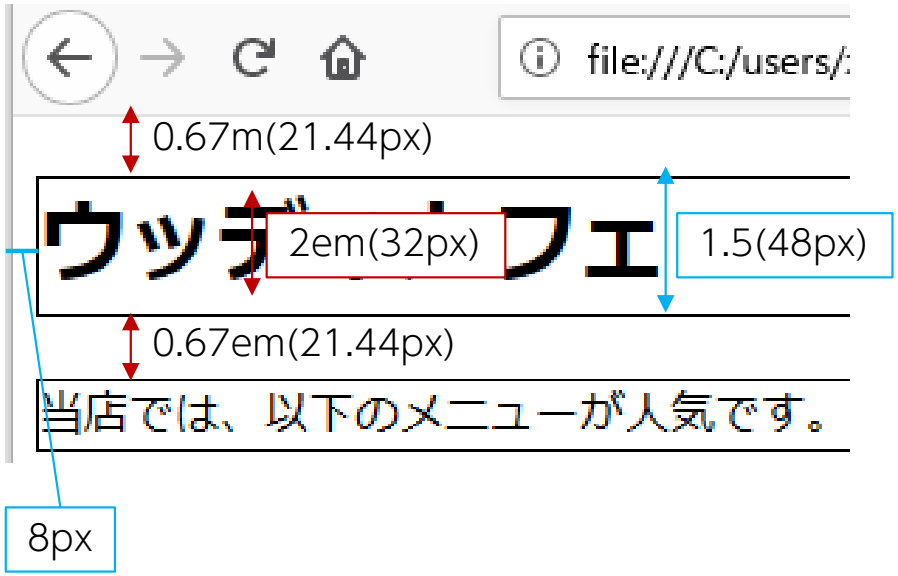
参考(各ブラウザのCSS)：https://coliss.com/articles/build-websites/operation/css/user-agent-stylesheets.html

ブラウザのもつCSSの確認

- ユーザーエージェントCSSは、開発者用ツールで確認することができる

例：h1のCSS



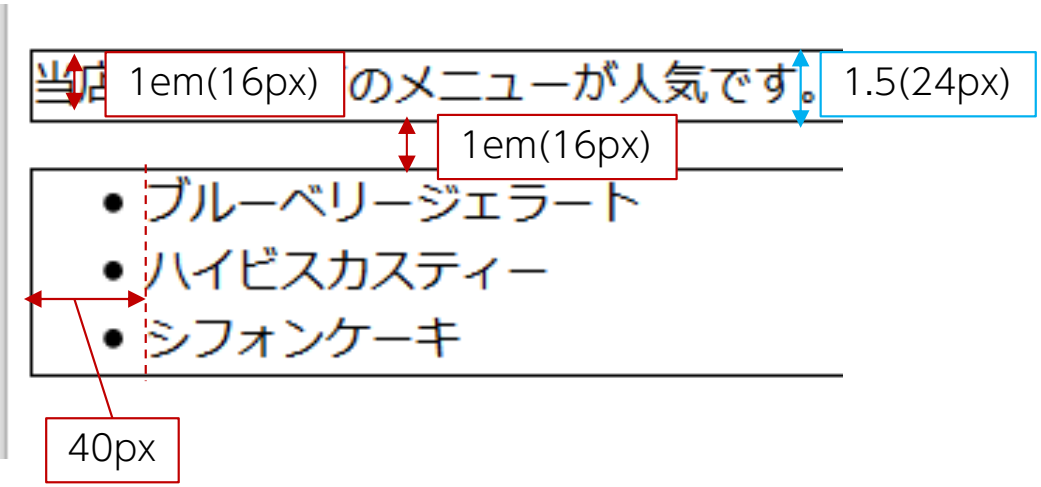


ブラウザのもつCSSの確認

例：p, ul のCSS

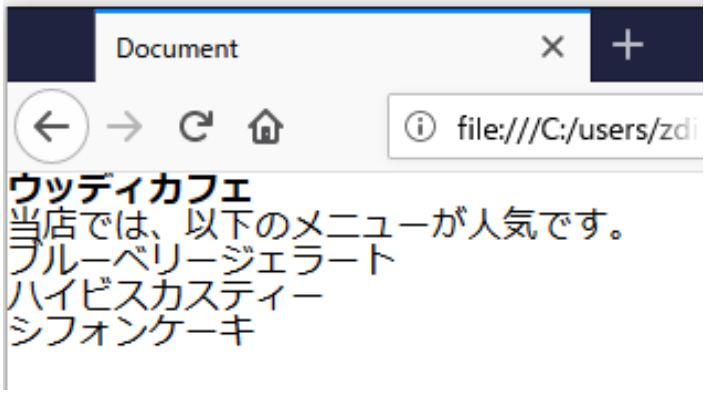
```
p {                                    user ag
    display: block;
    -webkit-margin-before: 1em;
    -webkit-margin-after: 1em;
    -webkit-margin-start: 0px;
    -webkit-margin-end: 0px;
}
```

```
ul, menu, dir {                        user ag
    display: block;
    list-style-type: disc;
    -webkit-margin-before: 1em;
    -webkit-margin-after: 1em;
    -webkit-margin-start: 0px;
    -webkit-margin-end: 0px;
    -webkit-padding-start: 40px;
}
```



初期化の考え方

① リセット：デフォルトのCSSをすべて打ち消す考え方
② ノーマライズ：デフォルトのCSSは活かし、ブラウザ間の差異に対応する考え方



リセットに基づく初期化

ノーマライズに基づく初期化

初期化用のCSS

- 初期化用のCSSは自身で作成することもできるが、無償で公開されているものを使用することもできる
  - これらのCSSは、自身で作成するCSSよりも先に読み込む

CSS名	説明
Reset CSS 2.0	リセットCSSを考案したEric Meyer氏によるコード https://meyerweb.com/eric/tools/css/reset/
destyle.css	見出しの文字の太さやa要素の文字色など、ほぼすべてのスタイルをリセット https://github.com/nicolas-cusan/destyle.css/blob/master/destyle.css
Normalize CSS	ノーマライズに基づく初期化用CSS https://github.com/necolas/normalize.css/blob/master/normalize.css
A Modern CSS Reset	Normalize CSSに比べて、ファイルサイズが小さい(行数が少ないシンプルな実装) https://piccalil.li/blog/a-more-modern-css-reset/
ress.css	基本的にはノーマライズの考えに基づいているが、マージンやパディングのリセットが行なわれている https://github.com/filipelinhares/ress/blob/master/ress.css

練習

- 練習10-1

CSSレイアウト

- ページのレイアウトでは以下のような配置をよく行う

① 中央に配置する
  - 幅を決めて、左右のマージンをautoにする
  - text-align: center;

② 横に並べて配置する
  - displayプロパティ：inline-block, table, flex, grid
  - floatプロパティ

③ 位置を微調整する
  - marginプロパティとpaddingプロパティ
  - positionプロパティ
  - transformプロパティ

本講義で扱う
レイアウト方法

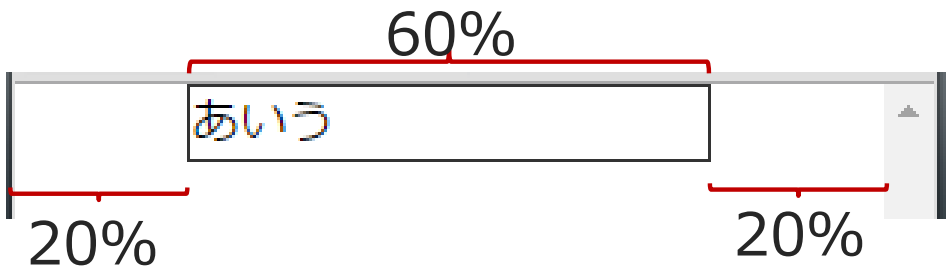
ボックスの中央配置

- widthの設定(px, em, %)を行い、左右のmarginの値をautoにする

```
<div>あいう</div>
```

```
div {
  border: 1px solid #333;
  width: 60%;
  margin: 0 auto;
}
```

左右のmarginを自動で算出



(100-60)÷2=20%＝片側の余白分

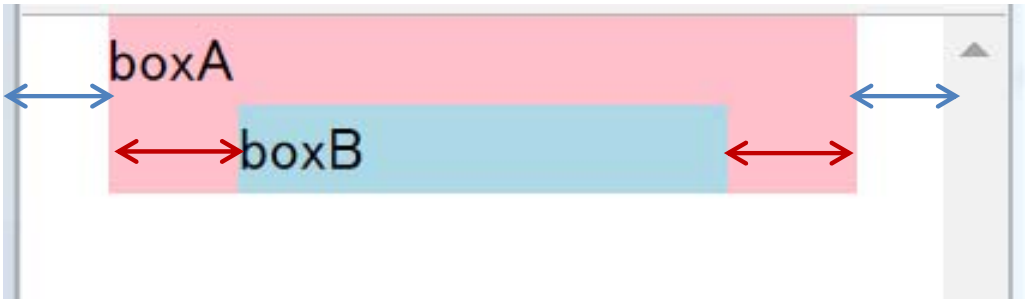
上下のmarginをautoにしても上下の中央に配置されないので注意する

ボックス内での中央配置

- ボックス内のボックスを中央配置する場合も同様に、幅を決めて、左右のmarginをautoにする

```
<div class="boxA">boxA
   <div class="boxB">boxB</div>
</div>
```

```
.boxA {
  background: pink;
  width: 200px;
  margin: 0 auto;
}
.boxB {
  background: lightblue;
  width: 130px;
  margin: 0 auto;
}
```



ボックス内での中央配置

- ボックス内の文字や画像を中央に配置する場合は`text-align: center;` を指定する
  - `inline, inline-block` の要素の場合、その親として存在する `block` の要素に `text-align: center;` を設定する

```
<div>あいう<br>
<img src="pic.jpg">
</div>
```

```
div {
  border: 1px solid #333;
  width: 100px;
  margin: 0 auto;
  text-align: center;
}
```

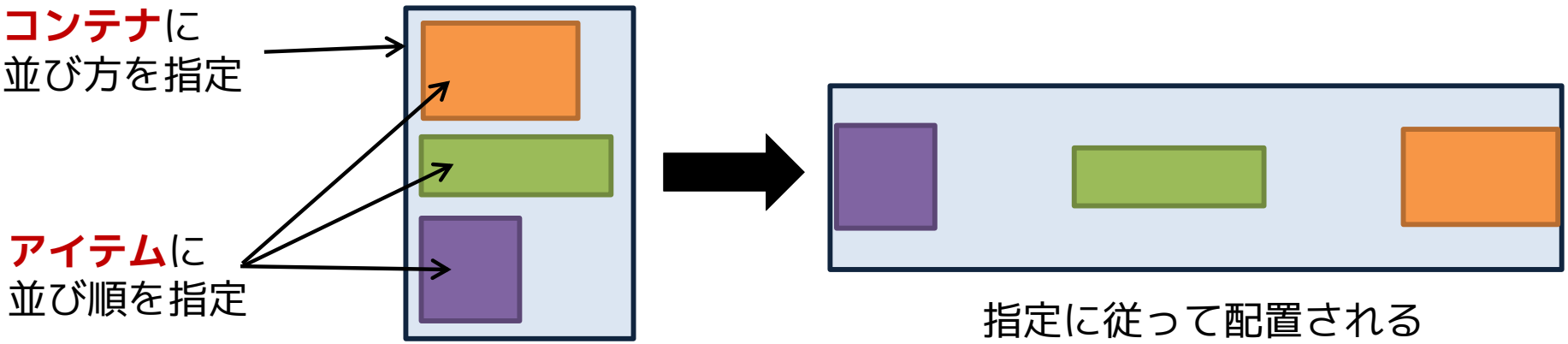
img要素そのものではなく、
その**親要素に対して設定する**
という点に注意する

練習

- 練習10-2

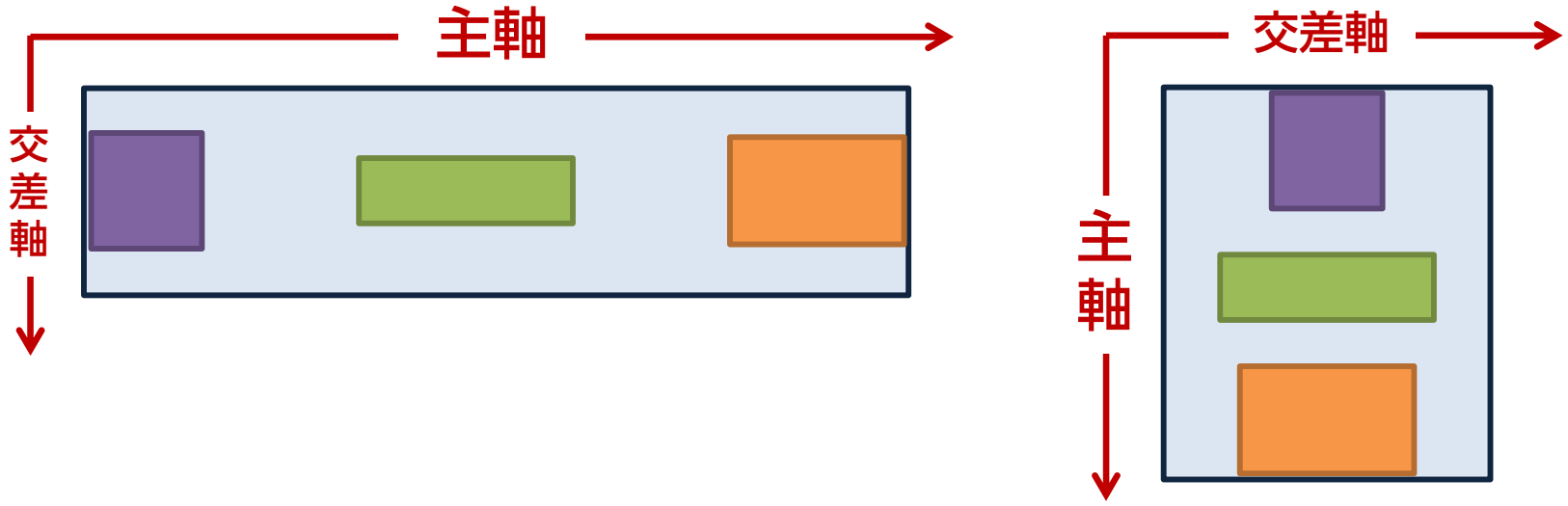
フレックスボックス

- ボックスを縦や横に並べて配置するためのスタイル指定
- **フレックスコンテナ**と**フレックスアイテム**で構成される
  - コンテナには配置のフォーメーションを指定
    ⇒ 左から右に均等に並べる / 下から上に並べる などの
    並び方を指定
  - アイテムには、並ぶ順番を指定



主軸と交差軸

- フレックスアイテムが並ぶ方向を**主軸**と呼ぶ
  - フレックスコンテナの設定によって、横方向(水平方向)または縦方向(垂直方向)が主軸となる
- 主軸に対して垂直に交わる軸を**交差軸**と呼ぶ
  - 主軸が横方向の場合は、交差軸は縦方向になる
  - 主軸が縦方向の場合は、交差軸は横方向になる



フレックスボックスの作成

- レイアウトしたいアイテム群に親となる要素を設け、`display: flex;` を設定する

  ⇒ 親要素がフレックスコンテナとして認識され、子要素の並びを指定することができる

① 親となる要素を作成する

親要素を設置

```
<div class="container">
  <div class="itemA">アイテムA</div>
  <div class="itemB">アイテムB</div>
  <div class="itemC">アイテムC</div>
  <div class="itemD">アイテムD</div>
</div>
```

レイアウトしたい
アイテム群

② 親となる要素に `display: flex;` を設定する

```
.container {
  display: flex;
}
```

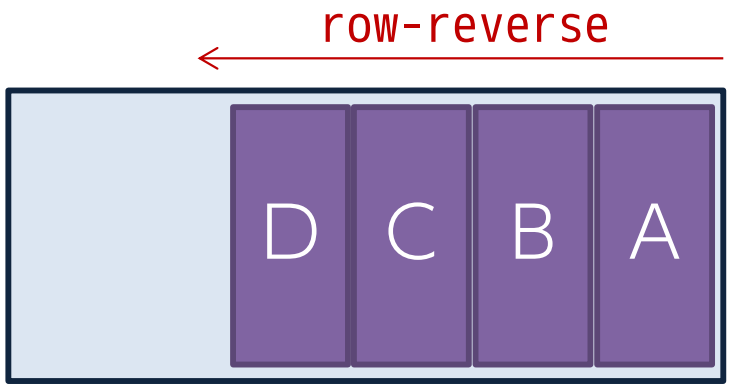
➡ 親要素がフレックスコンテナとして認識され、
アイテム群がフレックスアイテムとして認識される

コンテナの設定：配置の方向

- flex-directionプロパティでアイテムを配置する方向を指定することができる

値	説明
row(初期値)	左から右に横に配置（**主軸**を横方向に、**交差軸**を縦方向に設定する）
row-reverse	右から左に横に配置（主軸を横方向に、交差軸を縦方向に設定する）
column	上から下に縦に配置（主軸を縦方向に、交差軸を横方向に設定する）
column-reverse	下から上に縦に配置（主軸を縦方向に、交差軸を横方向に設定する）

```
.container {
  display: flex;
  flex-direction: row-reverse;
}
```

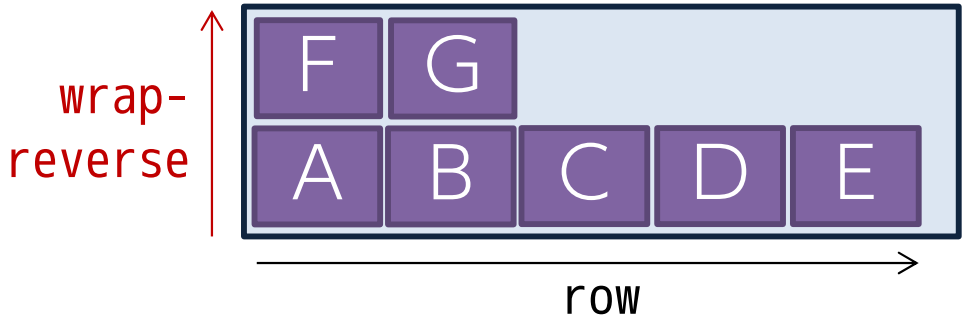


コンテナの設定：折り返し

- flex-wrapプロパティで折り返しの設定ができる

値	説明
nowrap(初期値)	コンテナにアイテムが入りきらない場合、アイテム幅を狭める
wrap	コンテナにアイテムが入りきらない場合、折り返す
wrap-reverse	交差軸が縦方向の場合、下から上に向かって折り返す

```
.container {
  display: flex;
  flex-direction: row;
  flex-wrap: wrap-reverse;
}
```



※ flex-flowプロパティで、direction と wrap をまとめて設定できる

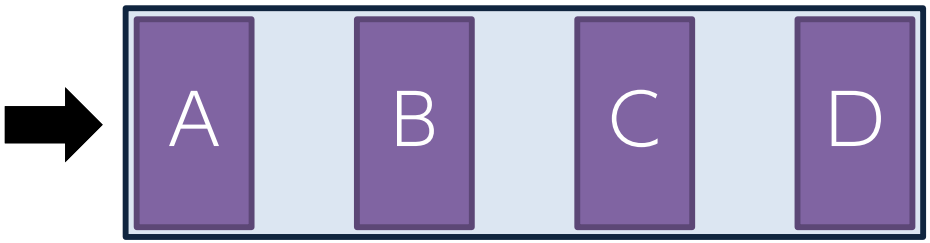
```
flex-flow: row wrap-reverse;
```

コンテナの設定：主軸方向の配置

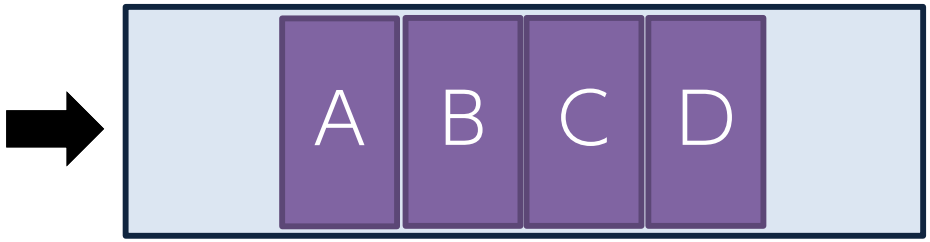
- justify-contentでアイテムの**主軸方向**の配置を設定できる

値	説明
flex-start(初期値)	先頭のアイテム側に詰めて配置
flex-end	末尾のアイテム側に詰めて配置
center	中央にまとめて配置
space-between	コンテナの範囲内で均等に配置
space-around	コンテナの両端に余白をもたせて均等に配置

```
.container {
  display: flex;
  justify-content: space-between;
}
```




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



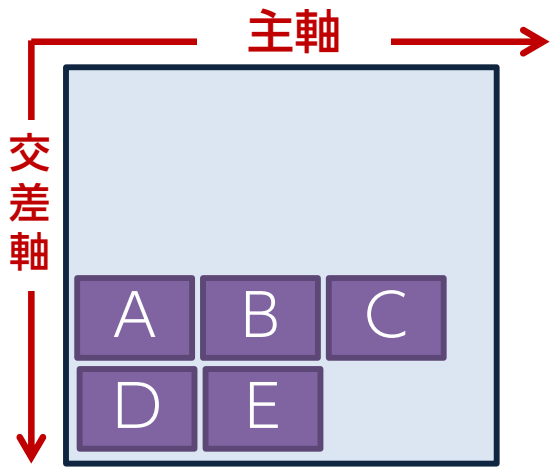
コンテナの設定：交差軸方向の配置

- align-contentでアイテムの**交差軸方向**の配置を設定できる

値	説明
flex-start	主軸が横方向の場合、上に寄せて配置
flex-end	主軸が横方向の場合、下に寄せて配置
center	中央に寄せて配置
space-between	コンテナの範囲内で均等に配置
space-around	両端に余白をもたせて均等に配置
stretch（初期値）	主軸が横方向の場合、下に等分の余白を取りながら配置

```
.container {
  display: flex;
  flex-wrap: wrap;
  align-content: flex-end;
  height: 300px;
}
```

※ 折り返しや高さを指定している

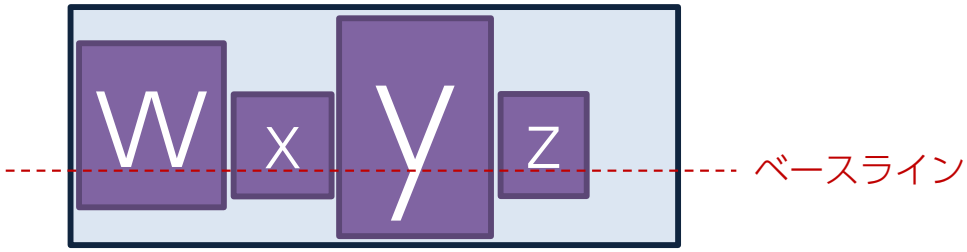


コンテナの設定：位置揃え

- align-itemsでアイテムの位置揃えを設定できる

値	説明
flex-start	主軸が横方向の場合は上揃え、縦方向の場合は左揃え
flex-end	主軸が横方向の場合は下揃え、縦方向の場合は右揃え
center	中央に揃える
baseline	ベースライン(英字の基準線)で揃える
stretch(初期値)	一番大きなアイテムにサイズを合わせて拡大

```
.container {
  display: flex;
  align-items: baseline;
}
```



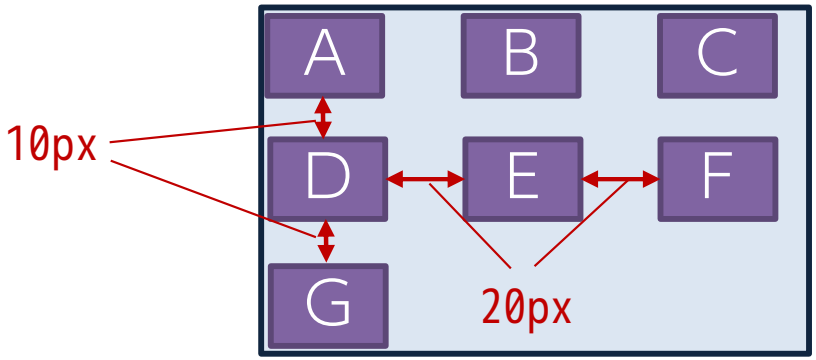
※ アイテムの高さに差を付けた場合の表示

コンテナの設定：アイテムの間隔

- gap, row-gap, column-gapで全体的なアイテム同士の間隔を設定できる
  - 個別に設定したい場合は、各アイテムにマージンを設定する

プロパティ	説明
row-gap	縦方向の間隔を設定する
column-gap	横方向の間隔を設定する
gap	縦方向、横方向の間隔を設定する **gap: 10px;** ⇒ row-gap: 10px; column-gap: 10px; に相当 **gap: 10px 20px;** ⇒ row-gap: 10px; column-gap: 20px; に相当

```
.container {
  display: flex;
  flex-rap: wrap;
  gap: 10px 20px;
}
```



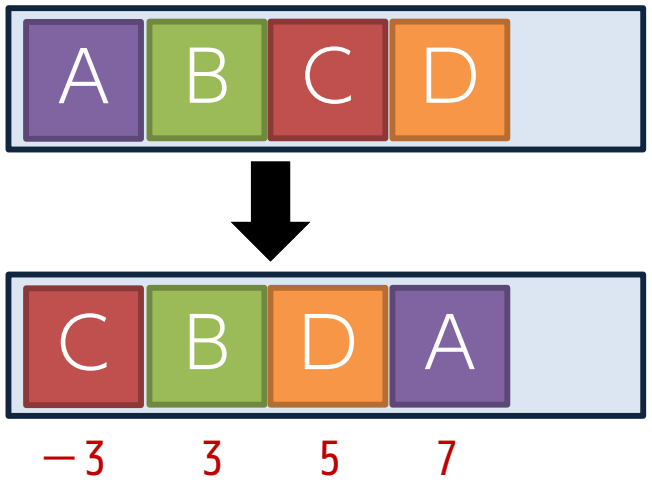
アイテムの設定：並び順

- フレックスアイテムにorderを設定することで、並び順の入れ替えが可能

プロパティ名	説明
order	指定された整数に従い、昇順（小さい順）に配置される。 負の数を記述することも可能

```
.container {
  display: flex;
}

.itemA {order: 7;}
.itemB {order: 3;}
.itemC {order: -3;}
.itemD {order: 5;}
```



アイテムの設定：幅/高さ

- 主軸に対するフレックスアイテムの大きさは、flex-basisで設定できる
  - 幅や高さは width, height で指定することも可能

プロパティ名	説明
flex-basis	フレックスアイテムの大きさを設定できる。 主軸が横方向の場合は幅として、縦方向の場合には高さとして設定される

```
.container {
  display: flex;
  flex-direction: row;
}

.itemA {
  flex-basis: 200px;
}
```

アイテムの設定：幅/高さ

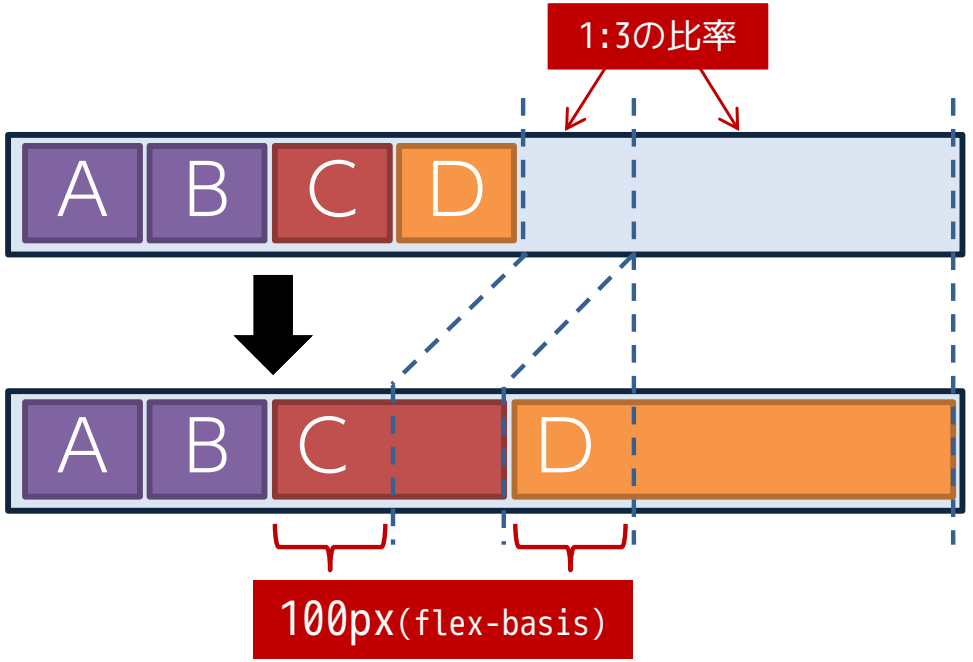
- フレックスコンテナの余りを埋める場合、flex-growプロパティで係数を指定する

プロパティ名	説明
flex-grow	フレックスコンテナの幅や高さに余りがある場合、その余りを埋めるため比率を設定できる。反対にコンテナの大きさが足りない場合の調整比率を指定するプロパティとして、flex-shrinkが存在する

```
.itemA, .itemB, .itemC, .itemD {
  flex-basis: 100px;
}

.itemC {
  flex-grow: 1;
}

.itemD {
  flex-grow: 3;
}
```



練習

- 練習10-3
- 練習10-4
- 練習10-5
- 練習10-6