

Slide URL

<https://vu5.sfc.keio.ac.jp/slide/>

Web情報システム構成法 第6回 CSS入門（続き）

萩野 達也 (hagino@sfc.keio.ac.jp)

CSSの役割

▶ HTMLに表現を与える

▶ 背景

- ▶ 色, 画像, 画像の繰り返し

▶ 文字

- ▶ 色, 種類, 太さ, 傾き, 大きさ

▶ 文書

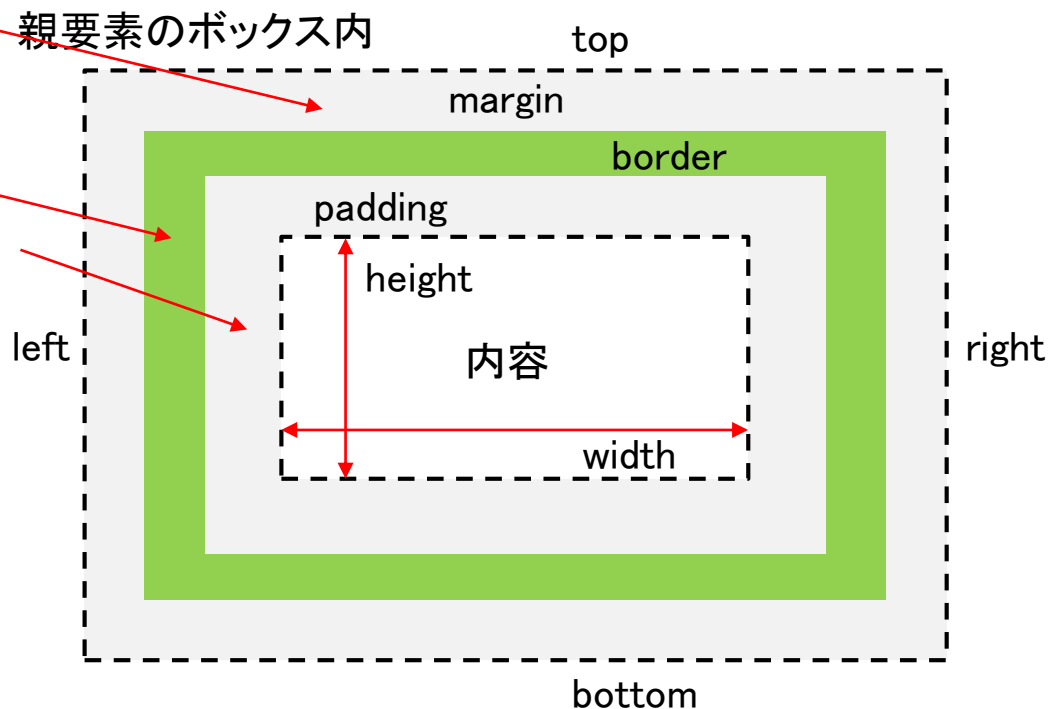
- ▶ 整列(左揃え, 中央揃え, 右揃え, 均等割り付け)
- ▶ 飾り(下線, 取り消し)
- ▶ インデント

▶ 配置

- ▶ ボックスモデル(上下左右の隙間, 境界線)
- ▶ float
- ▶ position

ボックスモデル

- ▶ margin
 - ▶ 親要素とborderまでの隙間
- ▶ border
 - ▶ 境界線
- ▶ padding
 - ▶ 境界線から自分の中身までの隙間



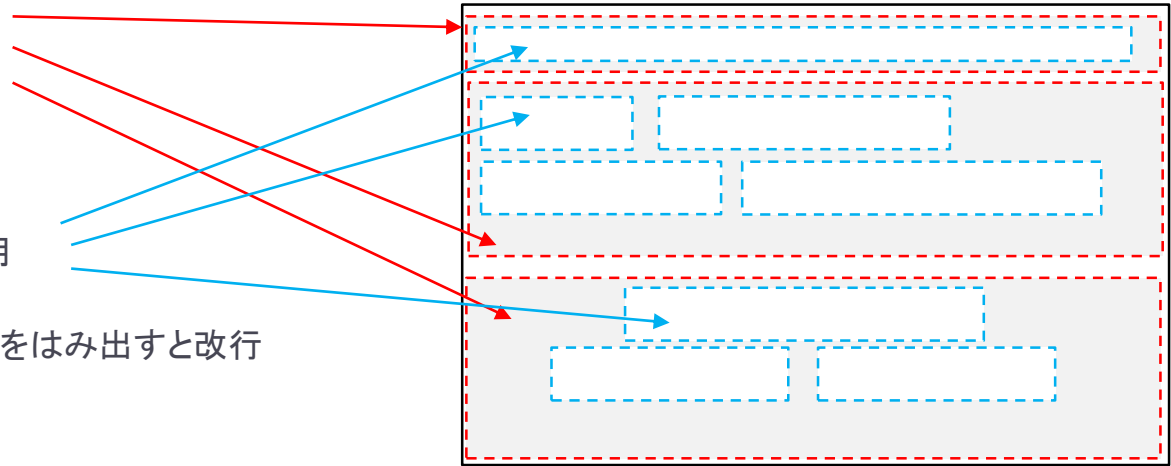
例

```
h1 {  
  margin: 25px;  
  border: 1px solid navy;  
  padding: 10px;  
}  
  
div.main {  
  margin-left: 200px;  
  padding: 10px 5px 15px 2px;  
}
```

top right bottom left

ブロックボックスとインラインボックス

- ▶ **ブロックボックス**
 - ▶ 段落用
 - ▶ 垂直につながる
- ▶ **インラインボックス**
 - ▶ 段落内の文書要素用
 - ▶ 横につながる
 - ▶ 親のブロックボックスをはみ出すと改行
- ▶ **display プロパティ**
 - ▶ デフォルトのボックスタイプを変更
 - ▶ **display: block;**
 - ▶ ブロックボックス
 - ▶ **display: inline;**
 - ▶ インラインボックス
 - ▶ width や height は指定できない
 - ▶ **display: inline-block;**
 - ▶ インラインボックス
 - ▶ width や height を指定可能
 - ▶ IE7.0以前ではインライン要素のみ指定可能
 - ▶ **display: none;**
 - ▶ 非表示



例

```
li {  
  display: inline;  
}  
  
em.sfc {  
  display: block;  
}  
  
div.hidden {  
  display: none;  
}
```

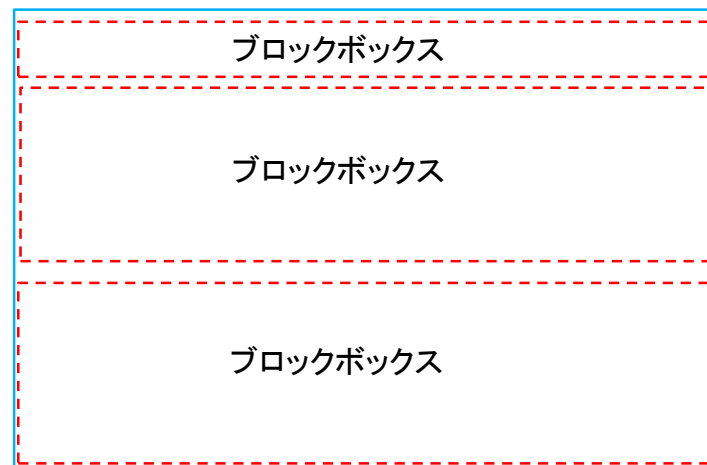
水平方向に
箇条書き

表示させない

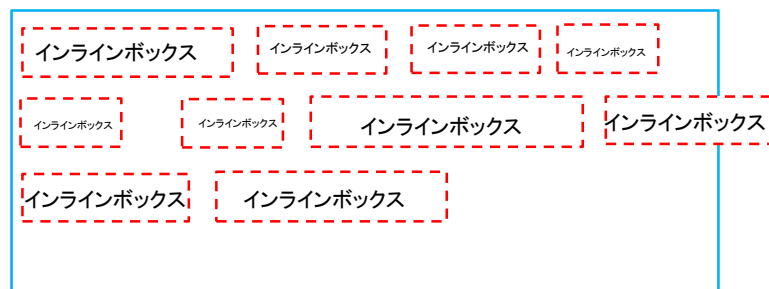
CSSにおけるボックスの配置

- ▶ コンテナブロック (container block)
 - ▶ 子要素を描画する箱
 - ▶ 子要素は親要素のコンテナブロック内におかれる
 - ▶ はみ出しても構わない
- ▶ 初期コンテナブロック
 - ▶ ルート要素のコンテナブロック
 - ▶ `width` と `height` 属性で大きさを指定
 - ▶ `width` が `auto` のときは `viewport` の幅を使う
 - ▶ `height` が `auto` の場合は自動的に伸びる
- ▶ 配置
 - ▶ ブロックボックスはコンテナブロック内に縦に配置される
 - ▶ インラインボックスはコンテナブロック内に横に配置される

コンテナブロック



コンテナブロック



float プロパティ

▶ 画像などを文章の横に配置したい

コンテナブロック

SFCの南門から入ったところでは、中高の校舎と運動場の間の桜が、毎年春になるときれいに咲きます。
たくさんの方が記念写真を撮ったりしています。日本の桜は多くはソメイヨシノで、まったく同じ遺伝子を持った桜の木であるため、毎年、同じ時期に同時に咲きます。ソメイヨシノの特徴は、葉が出る前に花が咲くことです。そのため花の色が強調されます。



文章

画像

▶ float プロパティ

▶ `float: left;`

- ▶ 指定された箱をコンテナブロックの左端に移動させる

▶ `float: right;`

- ▶ 指定された箱をコンテナブロックの右端に移動させる

▶ ブロックボックスは float と重なるように配置

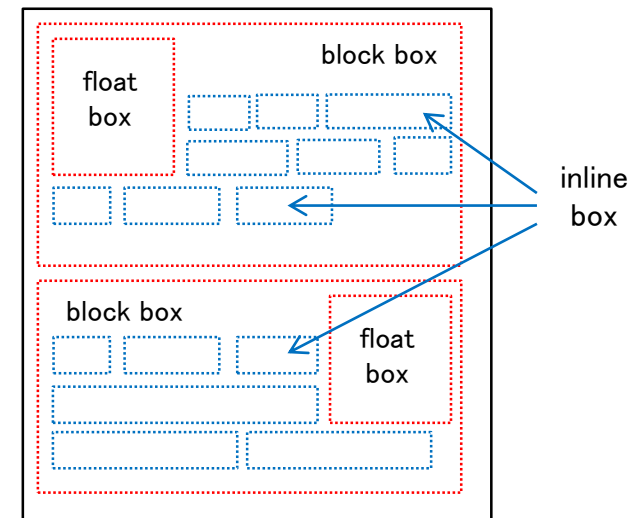
- ▶ `clear` 属性でブロックボックスを float を重ならないように指定可能

▶ `clear: left;`

▶ `clear: right;`

▶ `clear: both;`

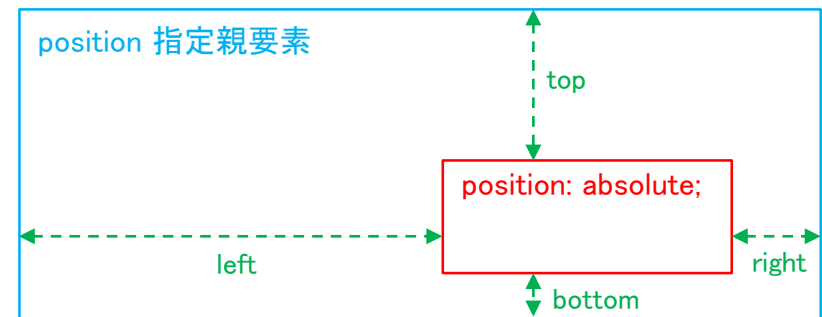
▶ インラインボックスは float と重ならないように配置



position プロパティ

▶ ボックスの配置場所を変更したい

- ▶ `position: static;`
 - ▶ 通常の位置配置
 - ▶ ブロックは縦, インラインは横に続けて並べられる
- ▶ `position: relative;`
 - ▶ 通常は配置の位置から相対的にずらす
 - ▶ 空いた隙間はそのままで, 別の箱が詰められるわけではない
 - ▶ ずらす大きさの指定は
 - ▶ `top, left, right, bottom`
- ▶ `position: fixed;`
 - ▶ viewport に相対的に配置
 - ▶ 画面上で位置が固定されるので, スクロールで動かない
 - ▶ 空いた場所は隙間とならず, 次の箱が詰められる
- ▶ `position: absolute;`
 - ▶ `position` 指定された親要素 (static を除く) に相対的に配置



レイアウト例

HTML

```
<body>
  <header>
    <h1>Header</h1>
  </header>
  <nav>
    Menu
  </nav>
  <article>
    Main content
  </article>
  <footer>
    Footer
  </footer>
</body>
```

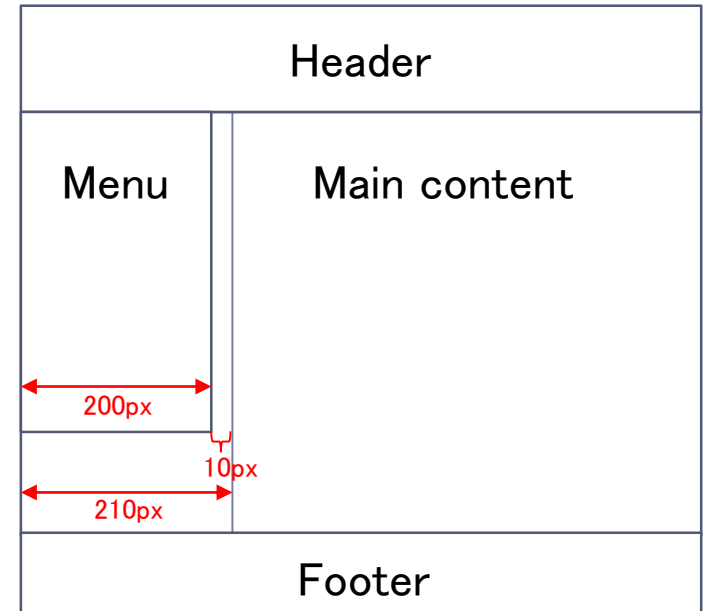
CSS

```
header h1 {
  text-align: center;
}

nav {
  float: left;
  width: 200px;
}

article {
  left-margin: 210px;
}

footer {
  clear: left;
  text-style: italic;
}
```

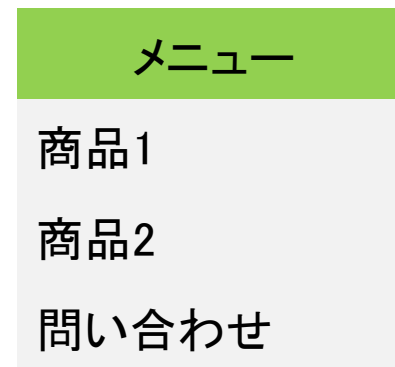


- ▶ float でメニューをメインの左に配置
- ▶ left-margin を指定してメインがメニューの下に回り込まないように
- ▶ footer はメニューに重ならないように clear を指定

縦メニューの作り方

HTML

```
<nav class="tate">
  <h2メニュー</h2>
  <ul>
    <li><a href="goods1.html">商品1</a></li>
    <li><a href="goods2.html">商品2</a></li>
    <li><a href="help.html">問い合わせ</a></li>
  </ul>
</nav>
```



CSS

```
.tate ul {
  list-style-type: none;
  margin: 0;
  padding: 0;
  width: 190px;
  background-color: #f0f0f0;
}

.tate h2 {
  text-align: center;
  margin: 0;
  padding: 8px 16px;
  background-color: #92d050;
}
```

```
.tate a {
  display: block;
  color: #000;
  padding: 8px 16px;
  text-decoration: none;
}

.tate a:hover {
  background-color: #555;
  color: #fff;
}
```

横メニューの作り方

HTML

```
<nav class="yoko">
  <h2メニュー</h2>
  <ul>
    <li><a href="goods1.html">商品1</a></li>
    <li><a href="goods2.html">商品2</a></li>
    <li><a href="help.html">問い合わせ</a></li>
  </ul>
</nav>
```

メニュー

商品1

商品2

問い合わせ



CSS

```
.yoko h2 {
  float: left;
  margin: 0;
  padding: 14px 16px;
  text-align: center;
  font-size: 100%;
}

.yoko ul {
  list-style-type: none;
  margin: 0;
  padding: 0;
  overflow: hidden;
  background-color: #f0f0f0;
}
```

```
.yoko li {
  float: left;
}

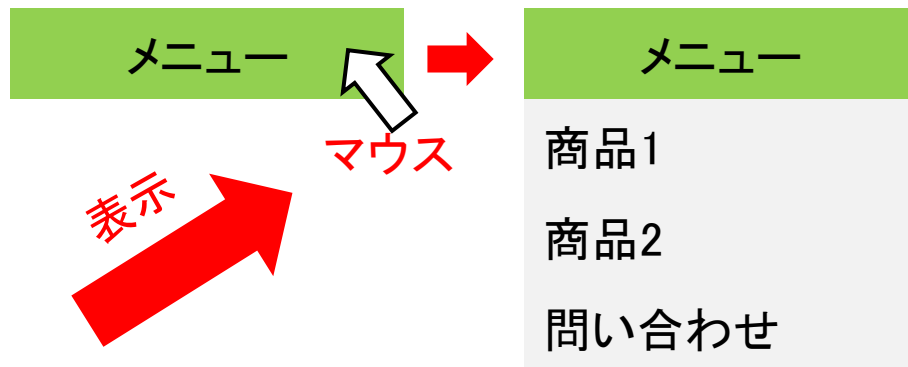
.yoko a {
  display: block;
  color: #000;
  padding: 14px 16px;
  text-decoration: none;
  min-width: 100px;
}

.yoko a:hover {
  background-color: #555;
  color: #fff;
}
```

ドロップダウンメニューの作り方

HTML

```
<nav class="dropdown">
  <h2メニュー</h2>
  <ul>
    <li><a href="goods1.html">商品1</a></li>
    <li><a href="goods2.html">商品2</a></li>
    <li><a href="help.html">問い合わせ</a></li>
  </ul>
</nav>
```



CSS

```
.dropdown {
  position: relative;
  width: 190px;
}

.dropdown h2 {
  text-align: center;
  background-color: #92d050;
  margin: 0;
  padding: 12px 16px;
}

.dropdown ul {
  display: none;
  position: absolute;
  background-color: #f0f0f0;
  box-shadow: 0px 8px 16px rgba(0,0,0,0.2);
  list-style-type: none;
  margin: 0;
  padding: 0;
  z-index: 1;
}
```

```
.dropdown a {
  display: block;
  color: #000;
  padding: 12px 16px;
  text-decoration: none;
  width: 158px;
}

.dropdown a:hover {
  background-color: #555;
  color: #fff;
}

.dropdown:hover h2 {
  background-color: #82e040;
}

.dropdown:hover ul {
  display: block;
}
```

media 問い合わせ

- ▶ 表示するデバイスごとにスタイルを変更する

```
@media not|only メディアタイプ and ( メディア機能 and|or|not メディア機能 ) {  
    CSS  
}
```

- ▶ プリンタではメニューを印刷しない

- ▶ 幅600px以下のディスプレイではメニューを表示しない。

```
@media print {  
    nav {  
        display: none;  
    }  
}
```

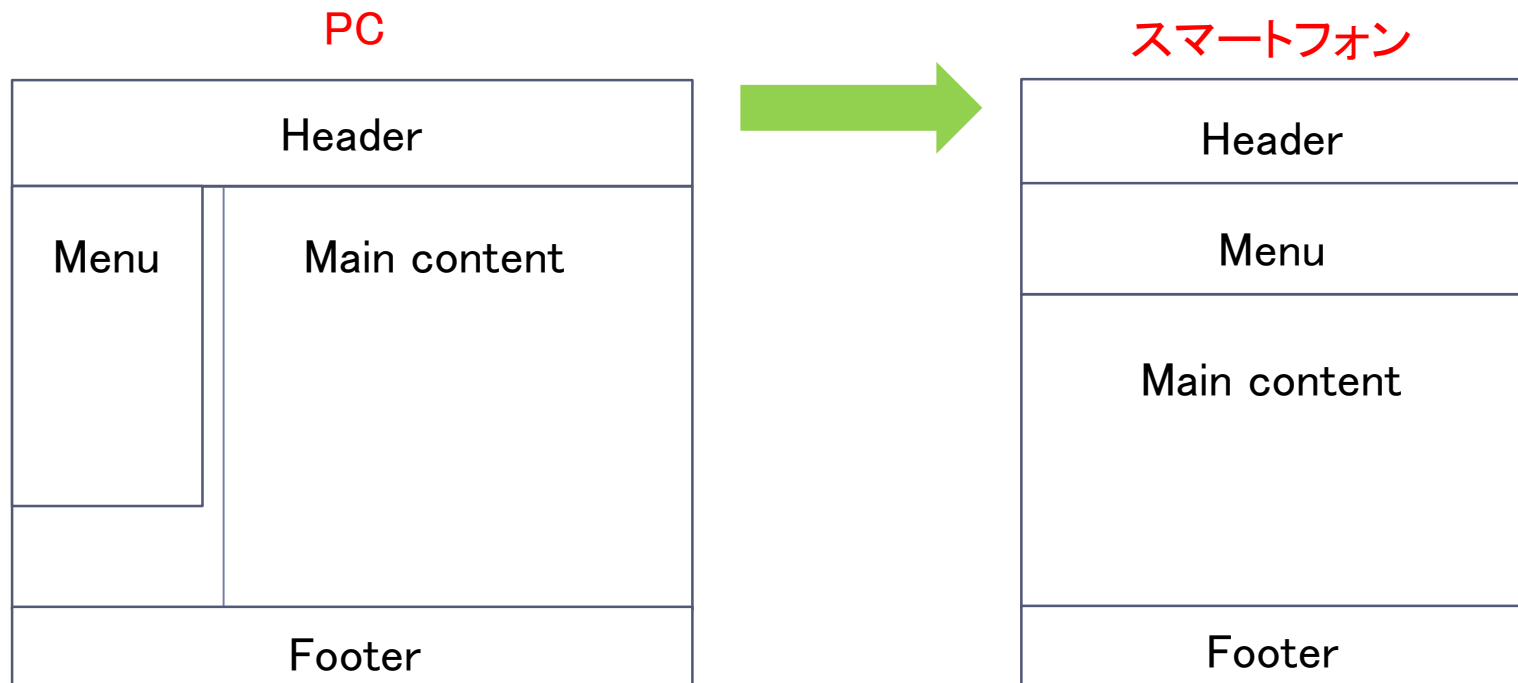
```
@media screen and (max-width: 600px) {  
    nav {  
        display: none;  
    }  
    article {  
        margin-left: 0px;  
    }  
}
```

レスポンシブデザイン

▶ 表示メディア(デバイス)に応じてレイアウトを変更する

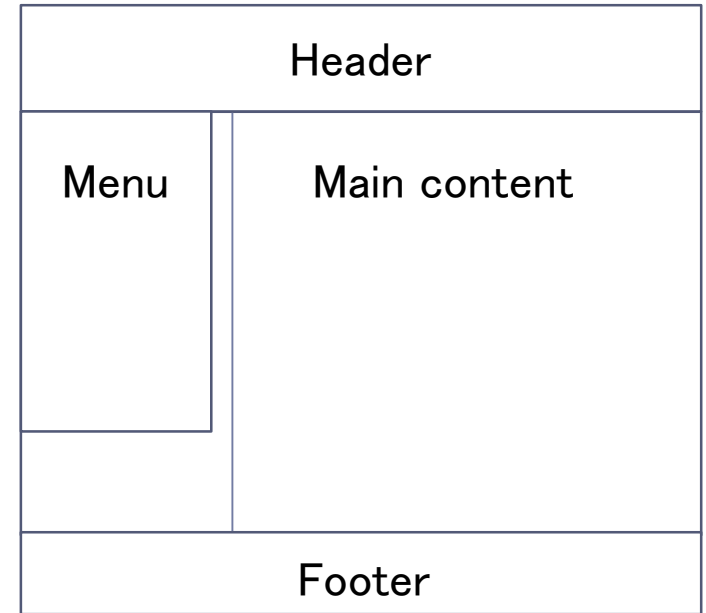
▶ 例

- ▶ PCではメニューが左に出て、メインの内容が右にある
- ▶ スマホではメニューは上に出て、メインの内容がその下にある



課題：CSSでレイアウト

- ▶ 架空のオンラインショップのトップページにCSSを使って右のようにレイアウトしなさい
 - ▶ CSSは別ファイルとして用意し、リンクすること。
- ▶ メニュー
 - ▶ メニューは本文の左に配置しなさい
 - ▶ メニューの項目がボタンのようになるようにしなさい
- ▶ 提出
 - ▶ <https://vu5.sfc.keio.ac.jp/kadai/>
 - ▶ レイアウトしたオンラインショップのHTMLとCSSをCNSにおいて、そのURLを提出してください。
 - ▶ 締め切り：5月27日正午



まとめ

- ▶ 原理

- ▶ 宣言的 vs 手続き的

- ▶ スタイルシート

- ▶ 内容と表現の分離

- ▶ CSS

- ▶ セレクタ
- ▶ カスケード
- ▶ 継承