情報基礎
CSSを用いたWebページ作成

CSSとは

- Cascading Style Sheetの省略表記。シーエスエスと読む
- Webページのレイアウト（視覚的構造）を定義する「スタイルシート」の規格の一つ
- Webの標準化団体であるW3C(World Wide Web Consortium)によって標準化

W3Cで推奨される考え方

- 論理構造：マークアップ言語
  - HTML, XHTML, XMLなど
- レイアウト（見た目）：スタイルシート言語
  - CSSなど

HTML&CSSの経緯

- 1990頃　Web登場
  - HTMLはWebページの論理構造を記述するもので、レイアウトを記述するものではない
  - レイアウトを記述するためのスタイルシート言語は規定されなかった
- 1993頃以降　ブラウザ普及
  - サイト製作者は見た目をよくしたい
  - ブラウザにあわせてレイアウト記述
    - 例）文字を大きくしたいときに<h3>を使う
  - ブラウザによって異なる独自要素も登場　例）<font>
  - HTMLの混乱と複雑化

HTML&CSSの経緯

- 1996 レイアウトを記述する枠組としてCSSが規定(CSS1)
- 1997 HTML3.2：折衷的なHTML
  - 暫定的措置としてレイアウトに関する独自要素の取り入れ
- 1997 HTML4.0：論理構造とレイアウトの分離
- 1998 CSS2
- 1999 HTML4.01
- 2000 XHTML1.0
- 2001 XHTML1.1
- 2011 CSS2.1
- 2011/2012 CSS3
- 2014 HTML5：元はWHATWGによる

CSS利用のメリット

- 詳細なレイアウトを記述できる
  - HTMLよりレイアウト記述力が高い
  - 例）テキストや画像を好きな位置に配置できる
- HTMLから、レイアウトに関する記述を除去できる
  - HTMLがシンプルに
  - 文法間違いを減らせる→情報を正しく伝達できる、表示の間違いを減らせる、維持管理が楽に
  - アクセシビリティ向上
    - 例）HTMLのtable要素を使ってレイアウトすると、音声読み上げソフトを利用してWebページを聞く視覚障害者に正しく情報が伝わりにくい
  - 検索エンジン対策

CSS利用のメリット

- レイアウトに関する記述を外部CSSファイルにできる
  - サイトの維持管理が楽に（後述）
- ユーザが自分の好きなスタイルシートでページを見ることができる
- 正しい（推奨される）考え方

CSSのメリットの例(1)

- 「New」という文字列の色を赤から黄色にかえたい場合
- 100ページ（100個のHTMLファイル）のWebサイト

HTMLのみで記述

- HTMLファイル

```
<font color="red">New</font>
```

100個のHTMLファイルで、redをyellowに変更

HTML+CSSで記述

- HTML

```
<span class="attention">New</span>
```

"attention（名前は自由）" というclassを設定
100個のHTMLファイルは変更不要

- CSS（別ファイル）

```
.attention
  color: yellow;
```

1個のCSSファイルにおいて "attention"の色をredからyellowに変更

CSSの基本

- セレクタ {プロパティ:プロパティ値;}
- 例
- p {color: #ff0000; }

課題

- 1. トップページをCSSを用いて作り直す
- 2. 二つ以上のブラウザで確認
- 3. W3C HTML ValidatorでHTML文法チェック, W3C CSS ValidatorでCSS文法チェック
- 4. トップページを置き換える

課題

- 5. 画像ファイルをimagesディレクトリにまとめる
- 6. 2ページ目以降もCSSを用いて作り直す
  - 文法チェックも忘れずに
- 7. サイト全体を二つ以上のブラウザで確認
  - スマホを持っている人はスマホでも確認

課題（余裕のある人）

- 8. HTML5+CSS3対応にする（最初からHTML5+CSS3でもよい）

HTMLとCSSの組合せ（参考）

- HTMLとCSSの組合せは自由
- 現在 HTML4.01 Transitional
- HTML4 + CSS2（授業ではこれを取り上げる）
- HTML4 + CSS3
- HTML5 + CSS2
- HTML5 + CSS3（余裕のある人へ）

課題の目標

- 論理構造(HTML)とレイアウト(CSS)を分ける
  - HTMLからレイアウトに関する記述をなくす
- 文法チェックに合格する(HTML/CSS)
- ディレクトリ構成を整理する
- 複数ページ（サイト全体）をCSS化する

一般的なディレクトリ構成

- public_html（ディレクトリ）
  - css（ディレクトリ）
    - index.css（名前は自由）
  - images（ディレクトリ）
  - index.html
  - 他のHTMLファイル

課題の手順（準備）

- 0. 見本の確認（HTML4.01+CSS2）
  - http://www.ex.media.osaka-cu.ac.jp/~harumi/mihon/以下
  - index.html, profile.html
  - css/index.css
  - images/photo.gif

  - コピーして使用してもよい

課題の手順

- 1. トップページをCSSを用いて作り直す
  - 1.1 cssという名前のディレクトリをpublic_htmlの下に作成
    - アクセス権を変更
  - 1.2 index.cssという名前（名前は自由）のファイルをcssの下に作成
    - 見本のindex.cssをダウンロード＆保存すると便利
    - アクセス権を変更
  - 1.3 index.htmlをtest.htmlという名前で保存（バックアップのため）
  - 1.4 test.htmlからindex.cssを参照
    - <link rel="stylesheet" type="text/css" href="css/index.css">

課題の手順

- 1. トップページをCSSを用いて作り直す
  - 1.5 test.htmlからレイアウトに関する記述を取り除く (HTML)
  - 1.6 レイアウトをindex.cssに記述する (CSS)
  - 注）1.5, 1.6 は、上から順番に少しずつ行う
  - 注）デザインを変更してもよい

Tips

- 少しずつ簡単なところから、上から下へ
- <h1>の上下の<hr>⇒<h1>
- <font color="…">⇒
- <body bgcolor="…">や<body background="…">⇒<body>
- ⇒
- レイアウト調整に用いた
,<pre>, <table>などをなくす

課題の手順

- 2. 複数のブラウザで確認
- 3. HTMLとCSSの文法チェック
  - 3.1 W3C HTML Validator
  - 3.2 W3C CSS Validator

課題の手順

- 4. トップページを置き換える
  - test.htmlが完成したらindex.htmlをindex.html.bakなどの名前で保存し（バックアップのため）、test.htmlをindex.htmlの名前で保存

課題の手順

- 5. 画像ファイルをimagesディレクトリにまとめる
  - 5.1 public_htmlの下にimagesディレクトリを作成
  - 5.2 画像ファイル(.jpg, .gifなど）をimagesディレクトリに移動（リンク切れが心配な場合はコピーする）
  - 5.3 HTMLとCSSファイルの中の画像へのパスを変更して保存
    - 例）<img src="photo.gif">を<img src="images/photo.gif">に

課題の手順

- 6. 2ページ目以降もCSSを利用して書き直す
  - 手順は1.と同じ
- 7. サイト全体を二つ以上のブラウザで確認
  - スマホを持っている人はスマホでも確認

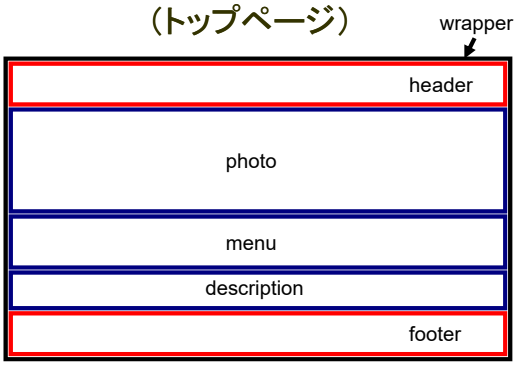
ブロックレベル要素とインライン要素

- ブロックレベル要素
  - 文書や段落を構成する基本要素
  - 例）見出し要素、div要素
- インライン要素
  - 特定の部分に何らかの役割を持たせる要素
  - 中にブロック要素を含むことができない
  - 例）アンカー要素、span要素

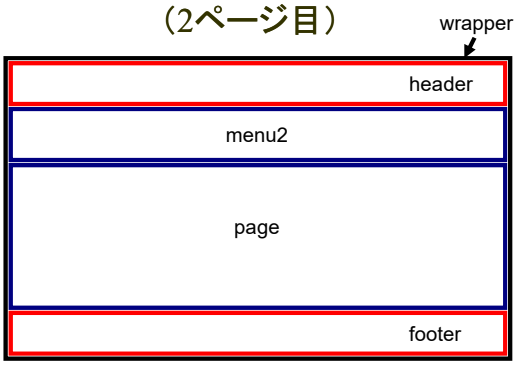
idとclass

- id
  - 要素に固有の（一つしかない）identifier（名前）をつける
- class
  - 要素にクラス（分類）をつける

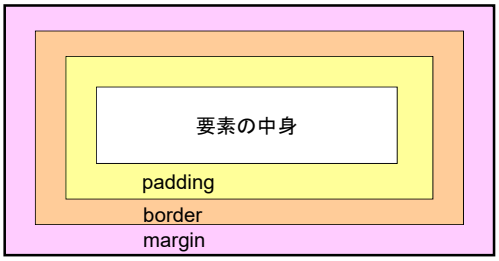
見本の基本レイアウト（トップページ）



見本の基本レイアウト（2ページ目）



marginとpadding



marginとpaddingの記述方法

- margin: 0;　　上下左右が0
- margin: 0 auto;　上下が0 左右は中央揃え
- margin: 0 0 0 0;　上、右、下、左が0

- 個別に指定するときは
  - margin-left, margin-right, margin-top, margin-bottom
  - 例) margin-left: 100;

CSSファイルの見方

- HTML：ブラウザで「ドキュメントのソースを表示」
- CSS:HTMLを見てCSSファイルのURLをブラウザで入力

Webにおける絶対パスと相対パス

- 絶対パス
  - ファイルのURL
  - 例）http://www.ex.media.osaka-cu.ac.jp/~学籍番号/
  - 例）http://www.ex.media.osaka-cu.ac.jp/~学籍番号/index.html
  - http://www.ex.media.osaka-cu.ac.jp/~学籍番号/profile.html
  - 例）http://www.ex.media.osaka-cu.ac.jp/~学籍番号/css/index.css
  - 例）http://www.ex.media.osaka-cu.ac.jp/~学籍番号/images/photo.gif

Webにおける絶対パスと相対パス

- 相対パス
  - 自分の位置から見た相手の位置
  - 自分「.」、自分の上「..」（ただし「./」は省略可能）
  - 例）index.htmlからprofile.htmlを見る場合：
    - ./profile.html または profile.html
  - 例）index.htmlからindex.cssを見る場合：
    - ./css/index.css または css/index.css
  - 例）index.cssからphoto.gifを見る場合：
    - ../images/photo.gif

パーミッション

- ディレクトリやファイルのアクセス権
  - d: directory; r: readable; w: writable; x: executable

d	r	w	x	r	w	x	r	w	x
	Owner			Group			Other		
-	r	w	-	r	-	-	r	-	-

（例）このファイルは
自分（オーナー）は読み書き可能
他人は読むことができるだけ

ファイル等が表示できないとき

- ファイル名
  - 空白を含むファイル名、日本語等のファイル名は×
- 画像ファイルの形式（PNG, JPG, GIF）
- ファイルが壊れていないか
- パス名
  - file: では×
- ファイルの置き場所
  - public_htmlに下にあるか、指定された場所にあるか
- パーミッションが「読めない」設定になっていないか

ファイル等が表示できないとき（文字化け）

- <meta>で指定された文字コードと実際のファイルの文字コードが合っているか
- ファイルを文字化け状態で保存していないか

他の見本

- HTML5+CSS2
  - http://www.ex.media.osaka-cu.ac.jp/~harumi/mihon/以下
  - index5.html, profile5.html
  - css/index2.css
  - images/photo.gif
  - コピーして使用してもよい
  - 注）HTML5のIE対策はしていない

他の見本

- HTML5+CSS3
  - http://www.ex.media.osaka-cu.ac.jp/~harumi/mihon2/以下
  - index.html
  - css/css3.css
  - images/photo-1000-01.jpg, photo-1000-02.jpg
  - scrollsmoothly.js
  - コピーして使用してもよい

CSS利用の問題

- HTMLより仕様が複雑
- ユーザがスタイルシートを利用しないことがある
- ブラウザによってCSSの対応が異なる。ブラウザのバグもある
  - CSSの文法通りに記述しても正しく表示されないことがある。ブラウザの対応度合いの差はHTMLのより大きい
- 複数のブラウザで美しく正しく表示させるためには、多くの手間と時間とノウハウが必要