情報システム技術論'02 ＃3

久野　靖*

2002.6.13

1　はじめに

□ 前回の感想 (?)

- これを機会に自分の Web ページを作ろうと思う
- 他人がどんなサイトを見ているか/気に入っているかが興味深い
- 自分と他の人とでユーザビリティに対する見方が違っているのが興味深い

□ 今回の内容…

- CSS level 2 を一通り学ぶ
- CSS を使うとここまでできる!　という実例を見てみる
- (演習) サンプルページに自分で CSS 指定をつけてみる

2　CSS level 2

□ CSS == Cascading StyleSheet(直列スタイルシート)

- HTML におけるスタイルシートの標準として W3C が取りまとめ

□ 「直列」とは?

- Web のスタイル指定→ 1 つの要素に対して複数のスタイル指定が関与
- 例:　全体指定、個別指定、作者指定、ユーザ指定、ブラウザ指定
- これらを「順に並べて」優先度の高いものを適用→「直列」

2.1　CSS の指定方法

□ 方法 1:　個別の HTML 要素の style 属性で指定

```
<p style="color: blue">...</p>
```

□ 方法 2:　style 要素でその場に指定 (前回やった)

```
<style type="text/css">
p { color: blue }
</style>
```

□ 方法 3:　link 要素を使って別ファイルに (今回使う)

```
<link rel="stylesheet" href="st1.css">
<link rel="alternate stylesheet" titie="X" href="...">
<link rel="alternate stylesheet" titie="Y" href="...">
```

- 入れる場所は<head>...</head>の内側
- 代替スタイルシートはいくつでも指定してよい。ブラウザのメニューで選択可能に (Mozilla)

2.2　CSS の処理モデル

□ CSS 指定されたドキュメントは次のように処理される

- ドキュメント→木構造に変換
- メディア種別 (画面、紙、プロジェクタ等) を決定
- 木構造の各要素のプロパティを計算
- 整形出力される構造を生成→出力

2.3　CSS の構文

□ CSS 記述→ルールセットの集まり

```
h1 { color: blue; margin-reft: 1cm }
```

- この「{ ...　}」は「ブロック」と呼ばれる

□ 「@ルール」→特別な指定であり、次の形のいずれか

```
@名前 … ;
@名前 … { ... }
```

- 先頭部分にない (通常のルールセットが始まった後にある)@ルール、ブロックの内側の@ルールは「無視される」

□ ルールセットの冒頭→どの要素に対してルールセットを適用するかの指定→「セレクタ」と呼ばれる

```
セレクタ { 宣言 ; 宣言; … }
```

*筑波大学大学院経営システム科学専攻

- 1 つのルールセットを同じセレクタを持つ複数のルールセットに分けても意味は変わらない
- 「宣言」は「プロパティ名:値」という形をしている
- プロパティ名または値が正しくない宣言は「無視される」

□ CSS ではさまざまな種類の値を扱う→それぞれに記法が決まっている

□ 数値→整数、実数 (少数点つきの数)

□ 長さ→数値の後ろに単位をつける

- 絶対単位→ in(インチ)、cm(センチ)、mm(ミリ)、pt(ポイント)、pc(パイカ)
- 相対単位→ em(フォントサイズ)、ex(文字 x の高さ)、px(ピクセル)
- px が相対単位なのは面白い (45p)。「手を伸ばした長さで 0.28mm」

□ 百分率→「数値%」

□ URI(=URL+URN) →「url(指定)」

□ 色→「#rrggbb」または「rgb(整数, 整数, 整数)」または「rgb(百分率, 百分率, 百分率)」

□ 角度→単位 deg(度)、grad(グラディエント)、rad(ラジアン) をつける

□ 時間→単位 ms(ミリ秒)、s(秒)

□ 周波数→単位 Hz(ヘルツ)、kHz(キロヘルツ)

□ 文字列→「"..."」または「'...'」、内側に同じ文字がくるときはバックスラッシュでエスケープする

2.4 セレクタ

□ 「どの要素に対する指定か」を表す

```
  H1, H2, H3 { ... }
これは次のと同じ
  H1 { ... }
  H2 { ... }
  H3 { ... }
```

□ 「*」→すべての要素にマッチ

□ 「H1」など→その名前の要素を指定

□ 「H1 EM」など→「〜の中の〜」

□ 「OL>LI」など→「〜の直接の子供である〜」

□ 「H1+H2」など→「〜の直後にある〜」

□ 「[属性名]」、「[属性名=値]」→「その属性を持つ/その属性の値が〜である」

□ 「X. クラス名」→「X[class="クラス名"]」と同じ

□ 「X#名前」→「id 属性の値が指定した名前であるような要素」

2.5 疑似要素と疑似クラス

□ 疑似クラス→ HTML にマークアップされていない「こういう要素」という指定

- 「DIV>P:first-child」→「並んでいるうちで最初の〜」
- 「A:link」「A:visited」→未訪問/訪問すみのリンク
- 「A:hover」「A:active」「A:focus」→上にカーソルがある/上でマウスボタンが押された状態/フォーカスされている
- 「*:lang(ja)」→日本語である

□ 記事クラスに類似しているが、HTML でマークアップされていない要素の一部分等

- 「P:first-line」→最初の行
- 「P:first-letter」→最初の 1 文字
- 「X:before」「X:after」→要素 X の直前/直後

2.6 値の設定と継承

□ 値の設定は次の順番による

- 直列順により決まった値がある→それを使う
- それ以外で、値が継承されてきている→それを使う
- それ以外なら、初期値 (デフォルト値) を使う

□ 継承→プロパティの指定値が「inherit」であると、外側 (親) 要素の指定値を引き継いで来る

- 継承のあるプロパティとないプロパティとがある
- 色の場合は「透明」もある (下地が透けて見える)

2.7 直列 (Cascading)

□ スタイル指定の出どころ→ Athor(元ファイルの指定)、User(読み手の指定)、UA(ブラウザの指定)

- スタイル指定の後ろに「!important」と指定することで優先度を上げられる

□ これらのスタイル指定を次の順に並べる (直列順 --- Cascading Order)

- 各要素ごとにセレクタがマッチするすべての指定を集める
- 重み順に並べる。弱い方から UA → User → Author → Author の!important → User の!important、の順番
- 同じ順位の中では特定度の高いものを優先

- それでも同じ順位の中では後から指定されたものを優先

□ 特定度の計算→セレクタの書き方で分かる。まず ID の一致するもの→同じなら属性条件の数が多いもの→それも同じなら要素名の数が多いもの

2.8 メディア種別

□ 出力メディアに応じてスタイルを調整する機能は当然必要

□ メディア種別→ p80 参照

□ 特定種別用のスタイル指定→@import または@media で

```
@import url(print.css) print ;
@media print {
  BODY { font-size: 10pt }
}
```

2.9 箱モデル

□ それぞれの要素は内側から Content(内容) → Padding(詰めもの) → Border(枠ないし縁) → Margin(余白)、の順で並んでいる (p82)

□ 隣接する要素のマージンどうしは併合される (p84)

- ただし float な要素や postion 指定つき要素の場合は併合しない

□ ☆ margin-top、margin-right、margin-bottom、margin-left

□ ☆ margin --- 上の 4 つをまとめて指定

□ ☆ padding-top、padding-right、padding-bottom、padding-left、padding

□ ☆ border-top-width、border-right-width、border-bottom-width、border-left-width、border-width

□ ☆ border-top-color、border-right-color、border-bottom-color、border-left-color、border-color

□ ☆ border-top-style、border-right-style、border-bottom-style、border-left-style、border-style

- 取れる値→ none、hidden、dotted、dashed、solid、double、groove、ridge、inset、outset
- ☆ border --- 「幅 スタイル 色」をまとめて指定

2.10 整形モデル

□ HTML の要素は「ブロックレベル」(行単位の箱) と「インラインレベル」(行の中に入る箱) に分かれる。

- ... 、<h1>...</h1>、 ... →ブロック
- ...、...→インライン

□ 外側の「窓の箱」→箱の中に順次箱を入れて配置していく

□ ☆ display --- 各要素の整形種別を指定

- 取れる値→ none、block、inline、list-item、marker、run-in、compact、その他 table 用

□ ☆ position --- 位置指定

- 取れる値→ static(通常)、relative(相対的にずらす)、absolute (絶対位置指定)、fixed(固定→窓の特定位置に固定など)

□ ☆ top、right、bottom、left --- 外側の箱からどれくらいアケルか

□ 整形コンテキスト→ 2 つのモード

- ブロックコンテキスト→ブロックの箱を縦に積み重ねて行く
- インラインコンテキスト→要素をつめて 1 行ずつ行の箱を作る

□ 詰め込み終わった後で position: relative の場合、「ずらす」ことができる

□ フロート指定の箱→右づめ/左づめにして流し込み (p109-111)

□ ☆ float --- フロート指定

- 取れる値→ left、right、none、inherit

□ 任意の要素に「流し込みを終わらせる」機能を持たせられる

□ ☆ clear --- 流し込み終わり

- 取れる値→ left、right、both、none、inherit

□ 絶対位置指定→任意の場所に箱を配置できる

- さらに「top: auto」などを使って「本来の位置」にもできる

□ ☆ z-index → 「重なり順」を制御する数値

□ ☆ width、min-width、max-width →幅を指定

□ ☆ height、min-height、max-height →高さを指定

□ ☆ line-height →行の高さ (行間隔) を指定

□ ☆ vertical-align →縦方向の位置

- 取れる値 → baseline、sub、super、top、text-top、middle、bottom

2.11 視覚効果

□ ☆ overflow → 箱からあふれた内容をどう表示するか

- 取れる値 → visible、hidden、scroll、auto、inherit

□ ☆ clip → クリッピング領域を指定

- 取れる値→ rect(上 右 下 左)、auto、inherit

□ ☆ visibility → 見え方の指定

- 取れる値→ visible、hidden、collapse、inherit

2.12 生成コンテンツ

□ 一般に「HTML 上には書かれてないものを挿入」

□ :before、:after と content プロパティを併用→前/後に文字列を挿入可

□ ☆ content →生成するコンテンツ

- 取れる値→文字列、url(...)、attr(属性名)、open-quote、close-quote、no-open-quote、no-close-quote、カウンタ

□ ☆ quotes →引用符として使う文字列を指定

□ ☆ counter-reset、counter-increment →カウンタ機能 (略)

□ 箇条書きのマーカ→一種の生成コンテンツだが特別な位置

□ ☆ marker-offset --- マーカをどれだけとび出させるか指定

□ ☆ list-style-type --- マーカのスタイルを指定

□ 取れる値 → none, inherit, disc、circle、square、decimal、decimal-leading-zero、lower-roman、upper-roman、lower-greek、upper-greek、lower-alpha、upper-alpha、lower-latain、upper-latain、hebrew、armenian、georgian、cjk-ideographic、hiragana、katakana、hiragana-iroha、katakana-iroha

□ ☆ list-style-image --- マーカに使う画像の URL を指定

□ ☆ list-style-position --- マーカの位置

- 取れる値→ inside、outside、inherit

□ ☆ list-style --- 上 3 つをまとめて指定

2.13 ページメディア

□ ページ単位で表示される媒体→ページの大きさ、ページ替えなど (今回は使わないので省略)

2.14 色と背景

□ ☆ color、background-color --- 色、背景色

□ ☆ background-image --- 背景画像の URI

□ ☆ background-repeat → repeat、repeat-x、repeat-y、no-repeat、inherit

□ ☆ background-attachment → scroll、fixed、inherit

□ ☆ background-position → 百分率 x2、長さ x2、top/center/bottom、left/center/right

□ ☆ background --- 上記をまとめて指定

2.15 フォント

□ ☆ font-family → ファミリ名の並び

- ファミリ名と実際のフォントの対応は@font-face で指定可能
- 汎用ファミリ名→どこでも使える (はず)。serif、sans-serif、cursive、fantasy、monospace の 5 つがある

□ ☆ font-style → normal、italic、oblique、inherit

□ ☆ font-variant → normal、small-caps、inherit

□ ☆ font-weight → normal、bold、bolder、lighter、100〜900 の数値

□ ☆ font-stretch → normal、wider、narrower、…

□ ☆ font-size --- 大きさ指定

- 絶対指定→ xx-small、x-small、small、medium、large、x-large、xx-large
- 相対指定→ larger、smaller
- そのほか、長さ指定 (「10pt」)、百分率指定もあり

□ @font-face 以下→非常に細かく指定可能だが略

2.16 テキスト

□ ☆ text-indent → 長さ、百分率、inherit

□ ☆ text-align → left、right、center、justify、文字列、inherit

□ ☆ text-decoration → none、underline、overline、line-through、blink、inherit

□ ☆ text-shadow → 色、長さ x2(または 3)

□ ☆ letter-spacing、word-spacing → normal、長さ、inherit

□ ☆ text-transform → capitalize、uppercase、lowercase、none、inherit

□ ☆ wihte-space → normal、pre、nowrap、inherit

2.17 テーブル (表)

□ 内容が多いので略

- もともと表の整形は複雑 (セルに入れて、セルを並べる)
- 縦方向でスタイルを揃える面と横方向で揃える面とあるので面倒

2.18 ユーザインタフェース

□ 表示とはちょっと違う「対話的」な部分

□ ☆ cursor --- ある領域の上だけカーソルを別のものにできる

- 指定できる値 → URI、auto、crosshair、default、pointer、move、...

□ ☆ outline-width → border-width と同様の指定

□ ☆ outline-style → border-style と同様の指定

□ ☆ outline-color → 色、invert、inherit

2.19 口述スタイルシート

□ 「読み上げ」→目の悪い人に対する重要な手段

- 読み上げ時のスタイルを指定→分かりやすさを大きく改善できる
- 声の調子、早さ、ポーズ、キュー、スペルアウト、左右位置、音、…

3 各自のサイトプランを発表

4 スタイルシートでどこまでできるか?

□ 2 ちゃんねる「いけてるスタイルを作れスレ」他

- →http://www32.tok2.com/home/css/clip/joyful.cgi
- → http://members.tripod.co.jp/monar01/ (リンク集)

5 既存の HTML にスタイルシートをつけてみる

□ まず次のページを開いてみてください

```
http://smm/~kuno/ist02/sample31.html
```

□ 各自のディレクトリ WWW の下に「sample1.css」「sample2.css」を作ってください。

□ 「chmod a+r sample1.css」等で誰にでも読めるように

□ とりあえず簡単なテストとして次のようなものを入れて確認

```
body { background-color: 好きな色 }
```

□ OK だったら自分が使いたいと思うようなスタイルを入れて試す

- Mozilla で「View」→「Use Style」でスタイル指定を選ぶ
- CSS ファイルを変更したら再読み込みしてから再度選ぶ

□ 色づかいを「#rrggbb」で指定したければ次のページを参考に

```
http://smm/~kuno/ist02/colors.html
```

6 次回までの課題

□ 「簡単なサイト」の製作を開始してください。これまでにやった内容は「ページデザイン」「コンテンツデザイン」「スタイルシート」なのでそれらに配慮しながらお願いします。

□ 次回時点では完成していなくてもよいですが、どれか 1 つ以上のページのスタイルについて「こういうところを工夫した」という形で発表してもらい、議論しようと思います。