CSS(Cascading Style Sheet)

1. CSS とは

CSS(Cascading Style Sheet)は HTML や XHTML，XML 文書の見栄えを調整するための仕組みである。HTML が単なるテキストであるのと同じ意味で，CSS も単なるテキストで表現する。HTML/XHTML/XML 文書に CSS を適用する方法は複数ある。ここでは，まず HTML(XHTML)文書に対して CSS を適用する場合について説明し，XML 文書に対する CSS の利用については後述する(予定である)。

2. CSS が必要な理由

一般に，HTML/XHTML 文書を表示する際には，特別な指定がなければデフォルトの(標準の)表示体裁で画面に描画される。デフォルトの表示体裁は閲覧ソフトに依存するものであり，文書製作者が指定するものではない。したがって，見る人によっては違う見え方をすることが一般的である。これに対して文書作成者が表示体裁を指定するために用いるのが CSS である。

文書の価値には，含まれているデータとプレゼンテーションという二つの面があるとされる。提供する情報，閲覧する装置が異なれば，最適な表示体裁も異なるので，文書作成者が，表示体裁／プレゼンテーションを明示的に指定することは，内容を適切に伝えるために重要な要素と考えられる。

近年ではこのようなデータと表示形式の分離という考え方が定着してきており，一時的に作成するページのような場合を除いて，HTML で記述するのはできるだけデータだけを記述しておき，レイアウトに関しては CSS で記述するというのが普通になってきている。また，それを一歩進めてデータだけを XML で記述し，レイアウトは XSL-FO または XSLT+CSS で記述するという方法をとることも多くなってきている。

さらに，CSS を用いることは，サイト内の Web ページ全体に統一感を持たせたり，更新が頻繁に行われる Web ページの管理を楽にするという用途にも有用である。特に，ビジネスの現場においては HTML を更新できる人員が限られている場合も多い。そのような場合には最初に外注で基本的な飾り付けを作成してもらい，データ部分だけを修正するという方法をとることでメンテナンスを容易にすることもできる。

3. 一般的な考え方

スタイルシートは，基本的に HTML タグの各要素に対して詳細な飾り付けを追加するものだと考えればわかりやすい。たとえば，

HTML ファイルで　　<h2> ドナドナの歌詞の秘密 </h2>　　のように指定されているとき，

CSS スタイルシートで　　h2 {font-size: 8pt;　color: red}　　と指定することによって，

h2 タグで囲まれた部分の文字の大きさや色が指定したものになるというものである。

したがって，CSS スタイルシートで指定した HTML タグは，本来そのタグが持っていた文字の大きさなどのスタイルは意味を持たず，あくまで CSS スタイルシートで指定したものが採用されることになる。これにより，たとえば h1 ～ h7 タグなどは(文字の大きさを変えるという役割を意識することなく)ヘッダーの指定という本来の意味を意識して使用することになるだろう。一方，文字の大きさや配置を指定するために使用されていたタグの一部は，その存在意義が小さくなり，場合によっては使われなくなるということもあるだろう。また，従来はあまり使用されなかった div タグなどが CSS の指定をするために多様されることもありえる。

ここで問題になるのが，同じタグが同一ページの中で複数回使用されており，そのレイアウト

や修飾を変えたいという状況への対応である。たとえば，全体が２つのパートにわかれており，上部の見出し h1 と下部の見出し h2 の色をそれぞれ変えたいというような場合である。この解決策としては，各 HTML タグ(CSS ファイルでは TYPE と呼ぶ)に，サブカテゴリー(CSS では CLASS と呼ぶ)を指定し，その組み合わせに対してスタイルシートの指定をするという方法をとる。

具体的には，たとえば

HTML ファイルで　　<h2 class="song"> ドナドナの歌詞の秘密 </h2>
　　<h2 class="picture"> ドナドナの写真　　</h2>　のように指定し，

CSS スタイルシートで　h2.song {font-size: 8pt;　color: red}
　　h2.picture {font-size: 8pt;　color: green と指定することによって，

同じ h2 というタグで指定されたテキストの色を，それぞれ変えることができる。

4. CSS を指定する３つの方法

代表的な方法としては，以下の３つの方法があげられる。

1）外部スタイルシート　：HTML/XML 文書とは別の CSS ファイルを用意する方法。
　　複数の文書でスタイル指定を呼び出して適用するのが一般的

2）head 内指定 style 要素　：HTML/XML 文書の内部に stye 要素として指定。
　　その文書全体に対してスタイル指定を適用

3）インライン style 属性　：HTML/XML 文書の内部に stye 属性として指定。
　　特定の要素だけに対してスタイル指定を適用

この３つの方法で指定されたスタイルが異なる場合(たとえば，外部スタイルシートでは段落の背景色を青にしているが，style 要素では赤にしているなど)，原則としては「style 属性」が，「style 要素」や「外部スタイルシート」よりもの順に優先される。また，もし同じタグに対するスタイルが２度指定されている場合には，後ろに記述した方が優先となるのが基本である(実際にはいろいろな決まり事があり，どのように適用されるかはもう少し複雑)。

一般的には，外部スタイルシートを使用して全体のスタイルを指定し，あるページ特有のスタイルを style 要素で指定，さらに例外的なものについてはインライン style 属性を使用するという使い方をする。

4.1 CSS スタイルシートを使用するにあたって

CSS スタイルシートを指定する場合，その HTML ファイルの最初(head タグの中)にスタイルのタイプ(CSS を使うということ)を指定しておく必要があることに注意が必要である。

具体的には，HTML ファイルの head タグの部分に以下の記述を書くことになる。

```
<meta http-equiv="Content-Style-Type" content="text/css" />
```

この記述を書いた上で，上記の３種類の方法でスタイルシートの設定を行う。

注)これを記述しなくても良い場合など，いろいろなケースもあるがあっても邪魔にならないので...

4.2　外部スタイルシートの利用

CSS を別ファイルに保存し，HTML/XML 文書から CSS ファイルへのリンクを指定する事で当該文書に適用する方法である。１つの CSS ファイルを複数の HTML/XML 文書に適用できるという特徴がある。その場合，CSS ファイルを変更すれば，そのファイルを適用していた全ての HTML 文書表示の体裁が一斉に変わることになる。

CSS ファイルは外部スタイルシートとも呼ぱれる。HTML と同様に単なるテキストなので，テキストエディタで作成・編集することができる。一般的にはファイル名は拡張子を*.css とい

う形で指定する。CSS ファイルを作成する際には，1)全て半角英数字を使用する，2)改行／スペース／タブの挿入は任意であるが，全角スペースは厳禁という２つの注意に留意すること。

CSS ファイルを呼び出す指定は以下の通りである。複数の外部スタイルシートも指定でき，その中で重複がある場合は後に指定した内容が有効となる。

<link rel="stylesheet" href="./atomic.css" type="text/css" media="screen" />

ここで，　rel="stylesheet"　　は「読み込む文書がスタイルシートである」
　　href="./atomic.css"　は「読み込む文書名は "./atomic.css" である」
　　type="text/css"　　は「スタイルシートの種類が CSS である」
　　media="screen"　　は「読み込ませる対象の出力装置は画面である」

ことを意味する。href の指定は絶対 URL だけではなく相対 URL も使用可能である。

4.3　head 内に指定する style 要素の利用

たとえば，外部スタイルシートを読み込んだ HTML 文書内で当該文書だけに適用したいスタイルがあるような場合には，当該 HTML ソースの head　要素内に style 要素を記述するなどの方法で，特定の文書だけに有効なスタイルを記述することができる。

style 要素の記述は，head 要素内に<style>タグを使用して，次のように style 要素を直接記述する。一般的には，外部スタイルシートへの link 要素よりも後ろに書く。

```
<style type="text/css">
h1｛ background: #dddd ｝
</style>
```

4.4 インライン style 属性の指定

インライン style 属性は，従来の HTML タグを指定するところで，タグの引数としてスタイルを指定するものである。全体を通した指定をした上で，ごく一部だけ例外的にスタイルを変更したい場合などに有効である。具体的には，以下のように指定する。

```
<h2 style="color: red">これは赤い文字の段落。</h2>
```

これによって，この<h2>タグで指定された部分の文字色を赤色に変更することができる。

5. スタイルの指定方法

CSS の書式は　　セレクタ ｛ プロパティ: 値; プロパティ: 値 ｝　のように指定する。

ここで，セレクタはスタイルを適用する対象を示す。一般的には HTML/XML のタグを指し示すと考えればわかりやすい。プロパティは，'font-size' や 'line-height' などのように，どのような項目について指定をしたいのかという項目名を指し示す。各プロパティのうしろに ： (コロン) をつけて値を続けて指定することとなる。

たとえば，h1 タグの先頭の字下げを二文字分という指定をする場合には，

```
h1｛ text-indent: 2em ｝
```

のように指定する。この例では h1 がセレクタ，text-indent がプロパティ，2em がその値となる。また，複数のセレクタに対して共通のスタイルを宣言する場合は，複数のセレクタをカンマ "," で区切って指定することができる。 例:　h1｛ color: maroon; text-indent: 2em ｝

例 : h1, h2, h3｛ font-family: sans-serif ｝

5.1　全称セレクタ

全ての要素に適用したい場合には，セレクタにワイルドカード "*"　が使える。たとえば，全ての要素のマージンとパディングを 0 にしたい場合には，以下のように指定する。

```
* { margin: 0;  padding: 0; }
```

5.2 クラス・セレクタ

3 章で示したように，実際の CSS 指定に際しては，元の HTML ファイルで同一のタグが各所に使用されていて，その場所によってスタイルを分けて指定したいことが数多くある。HTML では，body 要素内の全ての要素に class 属性を指定することで，これを実現できる。

たとえば，HTML 文書として以下のように指定されており，

```
<h1 class="note">ここは h1 要素の内容部です。この h1 要素には,
        class 属性が与えられており，その属性値は note です。</h1>
<h1 class="head">ここは h1 要素の内容部です。この h1 要素には,
        class 属性が与えられており，その属性値は head です。</h1>
```

対応するスタイルシートとして以下のように指定した場合，

```
h1.note { color: green }
h1.head { color: red }
```

前者の h1 は緑色となり，後者は赤色で表示される。

さらに，上記では h1.note　のように h1 という TYPE と note という CLASS を"."(ピリオド)でつないで指定していたが，TYPE を限定せずに，note という CLASS 全てについて指定を行うことも可能であり，　.note { color: green } のように，"."(ピリオド)の後ろに CLASS だけを指定することも可能である。

5.3 ID

クラスセレクタとほぼ同様の使い方をすることも多いが，基本的には１回しか出現しない要素について指定する場合に用いる。

たとえば，HTML 文書として以下のように指定されており，

```
<h1 id="idno1">ここは h1 要素の内容部です。この h1 要素には,
        class 属性が与えられており，その属性値は note です。</h1>
```

対応するスタイルシートとして以下のように指定した場合，

```
h1#idno1 { color: green }
```

のように # で指定する。

6. CSS で指定する内容の例

ここでは，以下をとりあげる。

1）色，カラー
   a）背景の指定 - background
2）字体，フォント
   a）フォントのスタイルの指定 - font-style
   b）フォントの大文字と小文字の指定 - font-variant
   c）フォントの太さの指定 - font-weight
   d）フォントの大きさの指定 - font-size
   e）行の高さ　- line-height
   f）フォントの種類 - font-family
3）テキスト
   a）段落の字下げ(インデント）- text-indent

b）文字の整列 - text-align
c）文字の間隔 - letter-spacing
d）テキストの装飾 - text-decoration
4）リンク・アンカー
5）マージン - margin
6）パッディング - padding
7）境界線(ボーダー) - border
a）ボーダーの太さ - border-width
b）ボーダーの種類 - border-style
c）ボーダーの色 - border-color
8）ボックスの配置 - float

1）色，カラー
a）背景の指定 - background
・単純に, body に背景色をつけたい場合には, body { background: red } のように指定する。
・画像ファイルを背景に貼りたい場合には，URI を指定することもできる。その場合には，body { background: url("画像ファイルの URL") } のようになる。背景色と画像ファイルの両方を指定することも可能である。その他，背景画像の表示体裁としては以下がある。
- どの方向に繰り返すか：
repeat | repeat-x(横に繰り返し) | repeat-y(縦に繰り返し) | no-repeat
- スクロールに対して固定するかどうか： scroll | fixed
- どこに配置するか：
[[<percentage> | <length>]{1,2}|[[top | center | bottom]||[left | center | right]]]
・body ではなく，p や h1 などを指定することで段落に対する背景なども指定可能である。
・背景の上に描写されるものの色を前景色と呼ぶ。文字の色などが代表的。
・文字を表示するタグ(セレクタ)ごとに指定可能。プロパティー"color"を使用。
・たとえば，strong 要素に対して赤色を指定する場合には以下のように記述する。
strong { color: red }
・全体についての文字の色を指定する場合には, セレクタ body にプロパティ color を指定。
・指定できる色の例
black, maroon, green, navy, silver, red, lime, blue,
gray, purple, olive, teal, white, fuchsia, yellow, aqua
RGB の 16(00-FF)進数で指定することもできる

2）字体，フォント
a）フォントのスタイルの指定 - font-style
・多くのブラウザでは，address 要素をイタリック(斜体)で表現する。このようなタグによるフォントのスタイルを指定するものである。
・たとえば，strong { font-style:italic } のように指定する。
・指定できるのは以下の通り
- イタリック(斜体)： italic
- ノーマル(通常) ： normal
- 少し傾けたフォント(普通はイタリックと変わらない) ：oblique
- 親要素の指定を継承 ：inherit
b）フォントの大文字と小文字の指定 - font-variant

・実際問題としては，小文字を背の低い大文字で表現する場合に指定することが多い
・たとえば，address｛ font-variant:small-caps ｝のように指定する。
・指定できるのは以下の通り
- ノーマル(通常)　：　normal
- 小文字を背の低い大文字で表すもの ：small-caps
- 親要素の指定を継承 ：inherit

c）フォントの太さの指定 - font-weight
・実際問題としては，太字(bold)の指定だけをすることが多い
・たとえば，address｛ font-weight:bold ｝のように指定する。
・指定できるのは以下の通り
- ノーマル(通常)　：　normal
- ボールド(太字)　：　bold
- 親要素の指定を継承 ：inherit
- 親要素より太く/細く　：　bolder/lighter
- 文字の太さを 100 きざみで　：100 ～ 900
　ただし，実際には数字の指定では違いがあまり出ないので使われないことが多い

d）フォントの大きさの指定 - font-size
・キーワードで指定する方法，絶対値で指定する方法，標準要素との相対比で指定する方法などがある。
・たとえば，address｛ font-size:small ｝，address｛ font-size:150% ｝のように指定する。
・指定できるのは以下の通り
[キーワードで指定する方法]
- 大きさを示すキーワード　：　xx-small, x-small, small, medium, large, x-large, xx-large
- 親要素より大きく/小さく　：larger, smaller

[絶対値で指定する方法]
- インチ　：　in
- センチメートル　：　cm
- ミリメートル　：　mm
- ポイント(1/72 インチ)　：　pt
- ピカ(12 ポイント)　：　pc

[相対値で指定する方法]
- 親要素のフォントの大きさを 1 として指定　：　em
- 親要素での小文字の x の高さを 1 として指定　：　ex
- ピクセル数(解像度に依存する)　：　px
- 親要素のフォントの大きさに対する比率　：　%

inherit 親要素の指定を継承
・ただし，指定してもブラウザによっては反映されないこともある
・body 要素には font-size を指定しないこと。

e）行の高さ　- line-height
・標準の文字の大きさを変化させた場合に，行の高さまで変化してしまうのを補正するためにも使用される。
・normal, 絶対値で指定する方法，標準時との相対比で指定する方法などがある。
・たとえば，p｛ line-height:0.8 ｝のように指定する。
・指定できる例は以下の通り
- ノーマル(通常)　：　normal

- inherit 親要素の指定を継承

[絶対値で指定する方法]

- インチ ： in
- センチメートル ： cm
- ミリメートル ： mm
- ポイント(1/72 インチ) ： pt
- ピカ(12 ポイント) ： pc

[相対値で指定する方法]

- 親要素のフォントの大きさを 1 として指定 ： em
- 親要素での小文字の x の高さを 1 として指定 ： ex
- ピクセル数(解像度に依存する) ： px
- 親要素のフォントの大きさに対する比率 ： % 120%など
- 親要素のフォントの大きさに対する比率 ： 倍率を示す数字 1.2 など

f) フォントの種類 - font-family

・基本的には，フォントの種類を一般的なキーワードで指定し(generic-family)，ブラウザが環境にあわせて具体的なフォントを設定するという方法をとることが多い。直接，具体的なフォント名を指定することもできるが望ましくない。

・指定できる例は以下の通り

- 明朝体フォント(ひげ付文字) : serif
- ゴシックフォント(ひげなし文字) : sans-serif
- 等幅フォント : monospace
- 装飾主体の文字 : fantasy
- 手書き風フォント : cursive

・ブラウザが適切な文字を見つけられなかった場合には，serif に対応する文字を表示する。

3) テキスト

a) 段落の字下げ(インデント) - text-indent

・印刷物の場合，段落の一文字目をへこませて表現する。このようなものを，「字下げ」，「インデント」と呼ぶ。

・指定できるのは以下の通り

[絶対値で指定する方法]

- インチ ： in
- センチメートル ： cm
- ミリメートル ： mm
- ポイント(1/72 インチ) ： pt
- ピカ(12 ポイント) ： pc

[相対値で指定する方法]

- 当該要素(フォント)の大きさを 1 として指定 ： em
- 当該要素の小文字の x の高さを 1 として指定 ： ex
- ピクセル数(解像度に依存する) ： px
- 親要素のフォントの大きさに対する比率 ： %

b) 文字の整列 - text-align

・指定できるのは以下の通り

- 左揃え : left
- 中央揃え : center

- 右揃え　　: right
- 均等割付　　: justify

・ただし現在のところ，多くのブラウザで justify の指定は無効となる。

c）文字の間隔 - letter-spacing

・行間を広げている場合には文字同士の間隔を広げた方がゆったりした印象になることがある。このような指定を行うものである。

・指定できるのは以下の通り

[絶対値で指定する方法]

- インチ　：　in
- センチメートル　：　cm
- ミリメートル　：　mm
- ポイント(1/72 インチ)　：　pt
- ピカ(12 ポイント)　：　pc

[相対値で指定する方法]

- 当該要素(フォント)の大きさを 1 として指定　：　em
- 当該要素の小文字の x の高さを 1 として指定　：　ex
- ピクセル数(解像度に依存する)　：　px
- 当該要素のフォントの大きさに対する比率　：　%

d）テキストの装飾 - text-decoration

・上線，下線などの，文字列に対する装飾の指定ができる。

・指定できるのは以下の通り

- 装飾なし　　: none
- 下線　　: underline
- 上線　　: overline
- 取り消し線　　: line-through
- 点滅　　: blink

4）リンク・アンカー

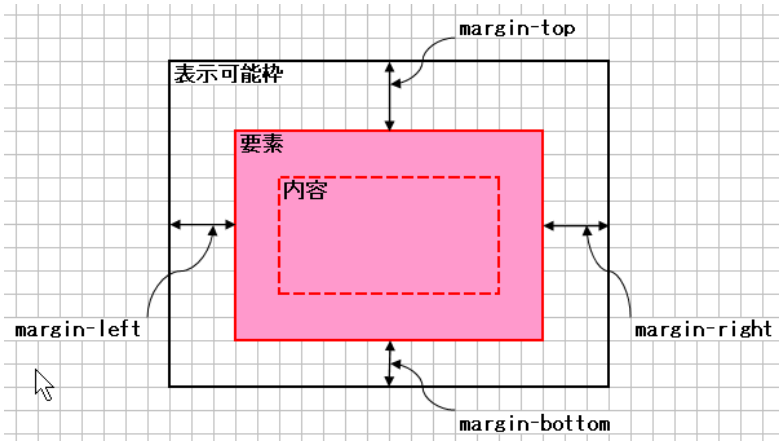
ここまでは，各タグ(セレクタ)ごとの指定の仕方について説明してきた。しかし，ハイパーリンク(a href-=... という形で a タグを使用する)については，単に a タグについて指定するだけではなく，「そのリンク先を１度訪問した状態」「まだ未訪問の状態」「カーソルがトリガーの上に乗っている状態」という三つについて別々にスタイルシートを設定できることが望ましいと考えられる。

そこで，CSS では以下の通りのセレクタを指定することができるようになっている。

- 訪問済み状態　　:　a:link　　ex.　a:link｛ color: maroon ｝
- 訪問済み状態　　:　a:visited　　ex.　a:visited｛ color: navy ｝
- カーソルが上にある状態　:　a:hover　　ex.　a:hover｛ color: red ｝

5）マージン - margin

マージンは，表示要素全体と表示可能枠の間の距離を指定するものであり，表示内容と他の要素との距離を指定するものではない。また，要素の表示可能枠は，周囲の要素との関係で決定される。したがって，たとえば画像を表示する場合に，その表示される画像自体に含まれる余白は指定するマージンには含まれないし，表示可能枠自体もまわりの要素との関係で予想と異なることは十分にありえる。



マージンの指定は，p｛ margin: 2em　3em　1em ｝のように行うが，その指定する要素の数によって解釈が大いに異なる。

1）パラメータを１つだけ指定した場合　　ex.　p｛ margin: 1em ｝
この場合，上下左右すべてが，指定された値で等しく余白をとることになる。

2）パラメータが２つの場合　　ex.　p｛ margin: 1em 2em ｝
この場合，第１のパラメタが上下のマージン，第２のパラメータが左右のマージンとなる。

3）パラメータが３つの場合　　ex.　p｛ margin: 2em　3em　1em ｝
この場合，第１のパラメータが上のマージン，第２のパラメータが左右のマージン，
第３のパラメータが下のマージンということになる。左右の幅は同じにしたい時など。

4）パラメータが４つの場合　　ex.　p｛ margin: 2em　3em　1em　0｝
この場合，第１のパラメータが上のマージン，第２のパラメータが右のマージン，
第３のパラメータが下のマージン，第４のパラメータが左のマージンとなる。
簡単に言うと，上から時計回りの順番に各マージンを指定することになる。

・指定できるのは以下の通り

[絶対値で指定する方法]

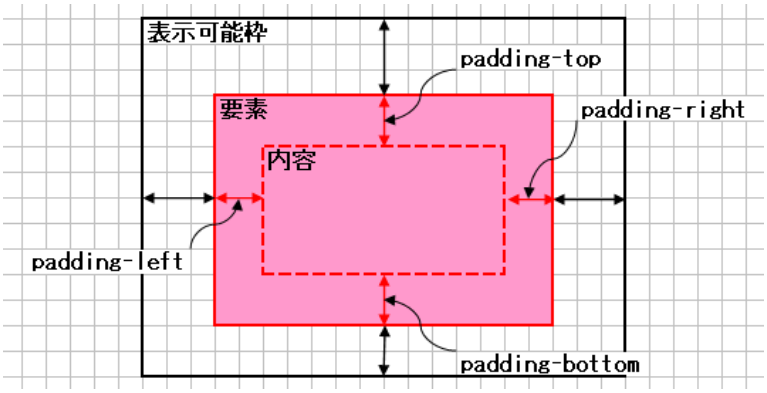
- インチ　：　in
- センチメートル　：　cm
- ミリメートル　：　mm
- ポイント(1/72 インチ)　：　pt
- ピカ(12 ポイント)　：　pc

[相対値で指定する方法]

- 当該要素(フォント)の大きさを 1 として指定　：　em
- 当該要素の小文字の x の高さを 1 として指定　：　ex
- ピクセル数(解像度に依存する)　：　px
- 当該要素のフォントの大きさに対する比率　：　%

6）パッディング　- padding

パッディングは，パツド(詰め物)の挿入を意味し，表示内容まわりの表示要素をふくらませる感じを演出する。マージンが要素の開始線をずらして表示可能領域の大きさを変更するものであるのに対し，パッディングは要素の大きさを膨らませるものであり，膨らんだ範囲の背景色は要素の背景が使われる。



パッディングの指定は，p { padding: 2em 3em 1em }のように行うが，その指定する要素の数によって解釈が大いに異なる(マージンの場合と考え方は同じ)。

1) パラメータを１つだけ指定した場合　　ex. p { padding: 1em }
この場合，上下左右すべてが，指定された値で詰め物をすることになる。

2) パラメータが２つの場合　　ex. p { padding: 1em 2em }
この場合，第１のパラメタが上下の詰め物，第２のパラメータが左右の詰め物となる。

3) パラメータが３つの場合　　ex. p { padding: 2em 3em 1em }
この場合，第１のパラメータが上の詰め物，第２のパラメータが左右の詰め物，
第３のパラメータが下の詰め物ということになる。

4) パラメータが４つの場合　　ex. p { padding: 2em 3em 1em 0}
この場合，第１のパラメータが上の詰め物，第２のパラメータが右の詰め物，
第３のパラメータが下の詰め物，第４のパラメータが左の詰め物となる。
簡単に言うと，上から時計回りの順番に各パッディングの値を指定することになる。

・指定できるのは以下の通り

[絶対値で指定する方法]

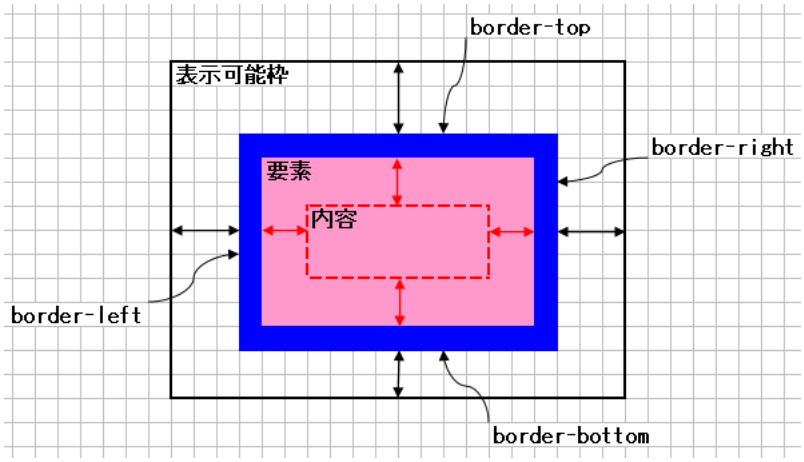
- インチ　：　in
- センチメートル　：　cm
- ミリメートル　：　mm
- ポイント(1/72 インチ)　：　pt
- ピカ(12 ポイント)　：　pc

[相対値で指定する方法]

- 当該要素(フォント)の大きさを 1 として指定　：　em
- 当該要素の小文字の x の高さを 1 として指定　：　ex
- ピクセル数(解像度に依存する)　：　px
- 当該要素のフォントの大きさに対する比率　：　%

7) 境界線(ボーダー)　　- border

要素の周りに表示する境界線(ボーダー)を指定する。言いかえれば，パッディング領域とマージン領域の間ということもできる。境界線のの太さ，種類，色を個別に指定することができる。



境界線の指定は，p { border-width: 2em 3em 1em }のように行うが，その指定する要素の数によって解釈が大いに異なる(マージンの場合と考え方は同じ)。

1）パラメータを１つだけ指定した場合　　ex. p { border-width: 1em }
この場合，上下左右すべてが，指定された値ということになる。

2）パラメータが２つの場合　　ex. p { border-width: 1em 2em }
この場合，第１のパラメタが上下，第２のパラメータが左右の指定値となる。

3）パラメータが３つの場合　　ex. p { border-width: 2em 3em 1em }
この場合，第１のパラメータが上，第２のパラメータが左，第３のパラメータが下の値を指定していることになる。

4）パラメータが４つの場合　　ex. p { border-width: 2em 3em 1em 0}
この場合，第１のパラメータが上，第２のパラメータが右，第３のパラメータが下，第４のパラメータが左の値を指定していることになる。簡単に言うと，上から時計回りの順番に値を指定していることになる。

a）ボーダーの太さ - border-width
・指定できるのは以下の通り
[キーワードで指定する方法]
- 大きさを示すキーワード ： thin(細い線)，medium(中くらいの線)，thick(太い線)
[絶対値で指定する方法]
- インチ ： in
- センチメートル ： cm
- ミリメートル ： mm
- ポイント(1/72 インチ) ： pt
- ピカ(12 ポイント) ： pc
[相対値で指定する方法]
- 親要素のフォントの大きさを 1 として指定 ： em
- 親要素での小文字の x の高さを 1 として指定 ： ex
- ピクセル数(解像度に依存する) ： px
- 親要素のフォントの大きさに対する比率 ： %

b）ボーダーの種類 - border-style

・指定できるのは以下の通り

- 境界線なし(隣接要素で指定があればそれを使用）　: none
- 境界線なし(隣接要素で指定があっても強制的に非表示）　: hidden
- 点線　　: dotted
- 波線　　: dashed
- 実線　　: solid
- 二重線　: double
- 窪み線（凹線）: groove
- 浮き線（凸線）: ridge
- 内側への下り傾斜(要素がキャンバスに埋もれた底面のように表現）　: inset
- 内側への上り傾斜(浮き上がった天面のように表現)　: outset
- 親要素での指定を継承　: inherit

c）ボーダーの色 - border-color

・指定できる色の例

black, maroon, green, navy, silver, red, lime, blue,
gray, purple, olive, teal, white, fuchsia, yellow, aqua
RGB の 16(00-FF)進数で指定することもできる

8）ボックスの配置　　- float

div で指定された文字列のかたまりや，箇条書き，表などのような，いわゆるボックスを右か左にフロートさせる(フロートとは「寄せる」の意味)。HTML の img 要素の align 属性と似た感じの役割と思えばわかりやすいだろう。あくまで，ボックス全体を寄せることを指定するものであって，中身のテキストだけ位置ぞろえをする場合には'text-align'を使う。

- 左フロートのブロックボックス　: left　　ex.　div { floatt: left }
- 右フロートのブロックボックス　: right　　ex.　div { float: right }
- フロートしない(標準)　: none　　ex.　div { float: none }
- (センタリング)　: center　　ex.　div { float: center }

(正確には center はフロートではない)

なお，指定した要素に内在寸法が無い場合は'width'で幅を指定する必要がある。その指定がない場合には幅ゼロとして無視されることになってしまう。

7. スタイルシート演習

7.1 スタイルの統一

たとえば，http://www.slis.keio.ac.jp/~ushi/hobby.html で表示される原田のページのいくつかは，以下のような形でスタイルが統一されている。

a）背景に牛柄が表示される

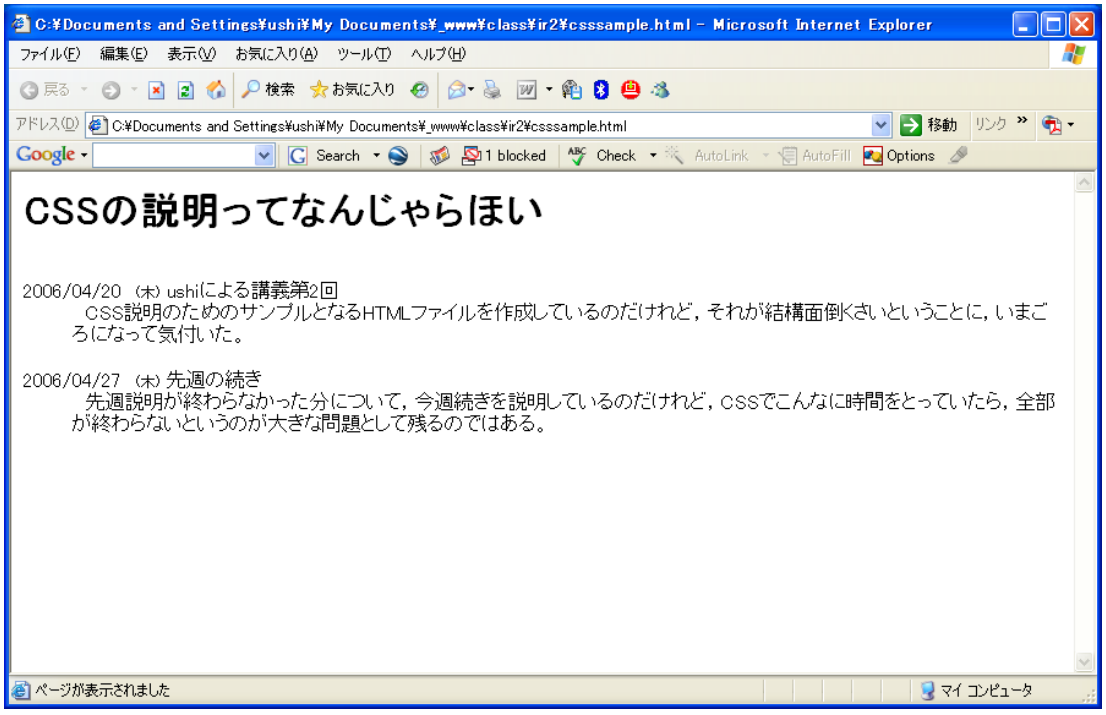
b）ページのタイトルを少し大きめの文字で，左に青太線，上に青細線の飾りを付して表記

c）各項目の見出しを，左に緑太線，上に緑細線の飾りを付して表記

このようなページを，単純な HTML ファイルを元に順に飾り付けをしていくことにする。まず最初に作成する HTML ファイルは以下のような内容で十分である。

```
<html>
<head>
   <meta http-equiv="Content-type" content="text/html; charset=Shift_JIS">
   <meta http-equiv="Content-Style-Type" content="text/css">
   <link rel="stylesheet" href="./common.css" type="text/css">
   <link rel="stylesheet" href="./local.css" type="text/css">
   <title>CSS のサンプルページ</title>
</head>
<body><div class="text">
<h1>CSS の説明ってなんじゃらほい</h1>      <br />
<dl>
<dt> 2006/04/20   <small>(木)</small> <strong>ushi による講義第 2 回 </strong></dt>
<dd>  CSS 説明のためのサンプルとなる HTML ファイルを作成しているのだけれど,
      それが結構面倒くさいということに，いまごろになって気付いた。 </dd>
<p />
<dt>2006/04/27   <small>(木)</small> <strong>先週の続き</strong> </dt>
<dd>  先週説明が終わらなかった分について, 今週続きを説明しているのだけれど, CSS
      でこんなに時間をとっていたら，全部が終わらないというのが大きな問題として残
      るのではある。   </dd>
</dl>
</div></body>
</html>
```

この例に見られるように，表示する内容を規定している<body>部には，単に表示する文章がれているだけで，飾り付け情報はほとんど指定されていない。したがって，この head 部を取り除いて表示すると以下のようになる。



ここで，head に指定した CSS ファイルを作成する。上記の例では，CSS ファイルが２つ設定されているが，これは複数のページ全体に適用して Web サイトの統一をはかるための common.css ファイルと，このファイルだけの例外的な指定を行った local.css の両方を使うという指定がされているためで，場合によっては，どちらかひとつだけでも問題はない。

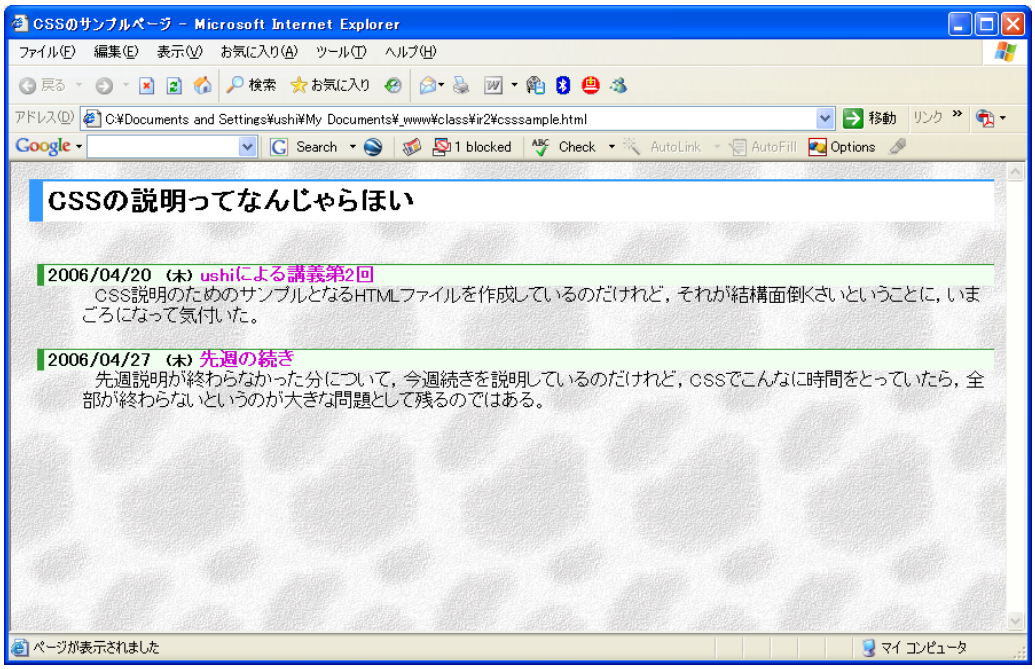
```
<link rel="stylesheet" href="./common.css" type="text/css">
<link rel="stylesheet" href="./local.css" type="text/css">
```

作成する CSS ファイル(今回は common.css というファイルだけを作成し，local.css ファイルは作成しないものとするが，わけてもよい)は以下の通りの内容とする。

```
/* BODY 全体 */
body {      color: #000000;
            background-image: url("image/cowskin.gif");
            text-align: left;                                   }
/* 本文 */
div.text {  margin-left: 0.4em;
            padding-bottom: 0.4em;                              }
/* 強調 */
strong {    color: #cc00cc;
            font-weight: bold;                                  }

/* 見出し */
h1 {        font-size: 1.5em;
            font-weight: bold;
            width: 100%;
            padding: 4px 4px 4px 4px;
            border-width: 2px 0px 0px 12px;
            border-style: solid;
            border-color: #3399ff;
            background-color: #ffffff;                          }
/* dl リスト */
dt {        font-size: 1.0em;
            font-weight: bold;
            width: 100%;
            padding-left: 4px;
            border-width: 1px 0px 0px 6px;
            border-style: solid;
            border-color: #339933;
            background-color: #f0fff0;                          }
small {     font-size: 0.8em;                                   }
```

この結果, 以下のように飾り付けがされることになる(cowskin.gif は指定の場所にあるとして)。



このような飾り付けは，ほかにもいろいろな指定が可能である。たとえば，一部の文字だけに蛍光色ペンで色をつけたようにしたい場合には，HTML ファイルのソースで

```
<span class="yellow">蛍光ペンを引きたい箇所</span>はここです。
```

のように指定しておき，CSS ファイル中で

```
span.yellow { background-color: #ffff55; }
```

と指定すれば可能になるし，たとえば，strong タグで示した強調箇所に二重線を引く場合には，

```
strong { border-bottom: double red 3px; }
```

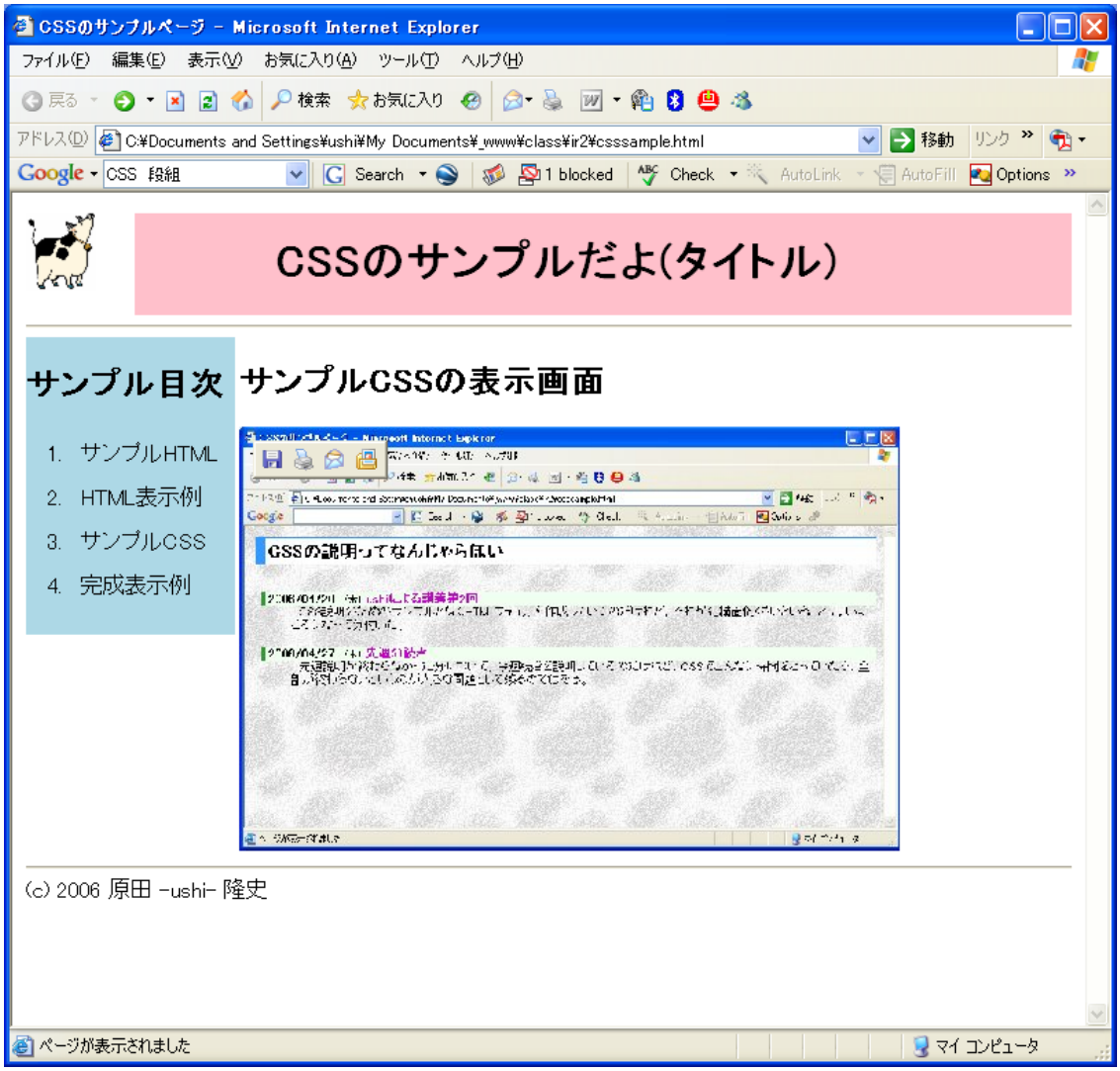
を指定すれば，二重下線を引くこともできる。ただし，この二重下線はある程度の幅がある場合に表示されるので，上記の例の strong の指定部分をこのように変更しても二重下線は引かれない。

7.2 スタイルシートで段組を表示する

近年は，HTML の frame タグは使われない傾向にある。実際問題として，frame タグの使用は HTML ソースの可読性を損ねることになるほか，ページを閲覧した人がページの登録がしにくいなど，様々な点で問題があるという指摘もある。

そこで，近年は段組をスタイルシートで作成することが多くなってきた。このために CSS の float が使われることが多い。少し考えると，このようなレイアウトを CSS で記述することは非常に複雑であるように思えるかもしれないが，実際には段組の校正というのは非常に単純であることが多い。特に，デザインはスタイルシートで記述することになるのであるから，HTML ファイルは非常に単純な構成となる。

まず，以下のような６つのブロックからなるページを作成したい場合を考える。
1）左上の「牛のロゴ」
2）タイトル部分
3）ロゴおよびタイトルと下とを分ける横線
4）サンプル目次部分
5）「サンプル CSS の表示画面」と書かれたメイン画面のようなところ
6）下の下線と著作権表示



この６つのブロックそれぞれを div タグで分け，それぞれの部分に表示内容を記述した HTML ファイルを作成する。

従来の frame を使用した方法では，各ブロックの内容を別のファイルに分割して記述し，それらを組み合わせる HTML ファイルを別に作成して，それぞれ呼び出すという方式をとっていたのと比較して，表示すべき内容が１つの HTML ファイル中に全て記述されているという点で，非常にわかりやすいものとなっている。

```
<html>
<head>　(前の例と同じなので省略) </head>
      <body>
       <div class="block_a">
         <img src="logo.gif" />
       </div>
       <div class="block_b">
         <h1>CSS のサンプルだよ(タイトル)</h1>
       </div>
       <div class="block_c">
         <hr />
        </div>
       <div class="block_d">
         <h2 class="contents">サンプル目次</h2>
            <ol>
              <li /> サンプル HTML
              <li /> HTML 表示例
              <li /> サンプル CSS
              <li /> 完成表示例
            </ol>
       </div>
      <div class="block_e">
              <h2 class="subtitle">サンプル CSS の表示画面</h2>
              <img src="sample.jpg" />
      </div>
      <div class="block_f">
        <hr />
          (c) 2006 原田 -ushi- 隆史
      </div>
</body>
</html>
```

その上で次ページのような CSS ファイルを作成して呼び出す。この CSS の特徴は以下の通り。

1) ブロック A(block_a)で float left;を指定することで，次のブロック block_b の左側に配置することを指定している。この際，width の指定をする必要があることに注意。
2) ブロック B(block_b)では，配置に関する指定は行っていない。何も指定しなければ，前に float left;を指定したブロックの右に配置される。ここではブロック内の文字の配置のみを指定している。
3) ブロック C(block_c)では，clear: both; を指定することで，ブロック A の float の影響を脱して上の２つのブロックの下に配置されるようになっている。
4) ブロック D,E は再度ブロック A,B と同様の指定
   ただし，ここで注意すべきはブロック E(block_e)に，margin-left:20%; が設定されていることである。これは，ブロック D よりもブロック E の高さが高い場合にブロック D の下ま

でテキストがまわりこんでしまうのを防止するために指定している。逆に回り込んで表示(ブロック E の中の左上にブロック D が表示される感じ)表示したいという場合は，この指定をしなければよい。

5) ブロック F は，再度ブロック C と同様の指定

```
/* common.css */
        div.block_a {
            float: left;
            width: 10%;
        }
        div.block_b {
            width: 100%;
            height: 60px;
            padding-left: 15%;
            padding-top: 18px;
            background-color: pink;
        }
        div.block_c {
            clear: both;
            width: 100%;
        }
        div.block_d {
            float: left;
            width: 20%;
            line-height: 2em;
            padding-top: 18px;
            background-color: lightblue;
        }
        div.block_e {
            margin-left: 20%;
        }
        div.block_f {
            clear:
            width: 100%;
        }
```

このように，複雑に見えてもブロックの関係を上から順に見ていけば，比較的簡単に設定をすることができる。

8. その他

CSS の指定に関しては，ほかにも数多くの指定可能項目があるので，各自で参考書などを参照されたい。