Brevi cenni sui Cascading Style Sheets

Oreste Signore, <oreste@w3.org>
Responsabile Ufficio Italiano W3C
Area della Ricerca CNR - via Moruzzi, 1 - 56124 Pisa

Master in Comunicazione e New Media
Ateneo Pontificio Regina Apostolorum
Corso: Architettura del Web

Presentazione: http://www.w3c.it/education/2012/upra/css/
Versione pdf: http://www.w3c.it/education/2012/upra/css.pdf

Formato XHTML realizzato usando il tool Slidy di Dave Raggett.
Slidy dovrebbe funzionare in tutti i browser moderni con Javascript abilitato. Usare freccia destra/sinistra per muoversi da una slide all' altra.
Vedi la pagina di aiuto di Slidy per ulteriori informazioni.

CSS

Cascading Style Sheets

Fogli di Stile in Cascata

- Introdotti per aggiungere informazioni sulle proprietà dello stile ai documenti (X)HTML (font, borders,...)
- Utilizzabili anche con XML
- Un foglio di stile è un elenco di regole
- Regola è composta da un selettore (in genere il nome dell'elemento a cui si applica) e una lista di proprietà (stili) da applicare

```
pre  { display: block; font-size: 16pt; font-weight: bold; }
```

- Il contenuto dell'elemento PRE deve essere mostrato
  - *in un blocco (display: block) (scatola)*
  - *con un font di a 16 punti (font-size: 16pt) (in seguito vedremo di usare altre unità di misura...)*
  - *in grassetto (font-weight:bold)*

Contenuto e stile

- Ogni documento ha tre componenti:

  contenuto

  l' informazione da presentare

  struttura

  la marcatura del linguaggio XHTML

  stile

  l'aspetto con cui viene rappresentata l'informazione (CSS)

- HTML definisce il contenuto
- CSS definisce lo stile
- Una pagina XHTML viene rappresentata automaticamente dal browser con una sua definizione degli stili
- CSS consente al progettista un maggior controllo sulla presentazione del contenuto da parte di *tutti* i browser
- Vantaggi anche per l' accessibilità

Versioni di CSS

- CSS 1 W3C Recommendation 17 Dec 1996, revised 11 Apr 2008
  Non più mantenuta
- CSS 2.1 W3C Recommendation 7 Jun 2011
- CSS 3 vari moduli in versione “Working Draft”, alcuni “Candidate Recommendation” o “Proposed Recommendation”, altri alla “Last call”

Le regole

- Ogni stile (style) è composto da un *insieme di regole*
- Una *regola* ha due parti:

selettore (selector)

- *definisce l'elemento (o gli elementi) XHTML a cui si applica*
- *più definizioni sono separate da "`,`"*

dichiarazione (declaration)

- *descrive l'effetto*
- *è definita tra parentesi graffe "`{ }`" ed è costituita da coppie proprietà/valore (property/value)separate da "`:`", che terminano ognuna con "`;`"*
- *Il primo elemento della coppia definisce la proprietà da modificare*
- *Il valore contiene il valore che deve assumere la proprietà*

```
elemento1, elemento2 { proprietà:valore; proprietà:valore; }
```

Le regole: alcuni esempi

Esempio 1

- `p {color: red;}`
- il selettore `p` indica che la regola va applicata a tutti gli elementi `p` del documento
- la proprietà modificata è `color`
- il valore che assume la proprietà è `red`

Esempio 2

- `h1, h2 {color: #00ff00; font-size: 24px; }`
- tutti gli elementi `h1` e `h2` saranno in verde
- tutti gli elementi `h1` e `h2` saranno con caratteri di dimensione 24px
- esempio di pagina con stile *CSS internal*

“One size fits all?”

- La regola **`@media`** specifica per quali “ media types” (separati da virgole) sono valide un insieme di regole (racchiuse tra parentesi graffe)
- I tipi di media supportati
  - ***`all`**: adatto per tutti i dispositivi*
  - ***`braille`**: per i dispositivi con risposta braille tattile*
  - ***`embossed`**: per le stampanti braille*
  - ***`handheld`**: per i palmari (tipicamente schermi piccoli, larghezza di banda limitata)*
  - ***`print`**: per materiale presentato a pagine e per la visione su schermo in modalità “print preview”. Vedi paged media per le informazioni sulla formattazione specifiche per i dispositivi paginati.*
  - ***`projection`**: per le presentazioni mediante proiettore. Vedi paged media.*
  - ***`screen`**: essenzialmente per gli schermi a colori dei PC*
  - ***`speech`**: per sintetizzatori vocali (CSS2 aveva il media type “`aural`”). Vedi aural style sheets per dettagli*
  - ***`tty`**: per dispositivi con caratteri di dimensione fissa (telescriventi, terminali, dispositivi mobili con display dalle potenzialità limitate)*
  - ***`tv`**: per dispositivi di tipo televisivo (bassa risoluzione, colore, scrolling limitato, suono disponibile)*

```
@media print {
  body { font-size: 10pt; }
}
@media screen {
  body { font-size: 13px; }
}
@media screen, print {
  body { line-height: 1.2; }
}
```

Definire sempre un CSS per la stampa!

```
@media print
  {
      body {
      margin-left: 3cm;
      margin-right: 3cm;
      margin-top: 3cm;
      margin-bottom: 3cm;
      background-color: #ffffff;
      color: #000000;
      font-size: 10.5pt;
      font-family: Arial, Helvetica, sans-serif;
      }
      h1, h2, h3 {
        color: #0000ff;
      }
      h1.title {
        font-size: 18pt;
        font-family: Arial, Helvetica, sans-serif;
        font-weight: bold
      }
      …
      .leftMenu {display: none;}
      …
  }
```

Dove definire gli stili

inline o embedded

All'interno del tag di riferimento

internal

All'interno del tag style

external

Importandolo con il tag style o indicandolo nel tag link

Stile inline (o embedded)

- direttamente in un tag XHTML

```
<h2 style="color: red; text-transform:uppercase;">Testo intestazione di livello 2</h2>
```

- è *sconsigliabile* (**Deprecato in XHTML1.1 e 1.0 Strict**)

Stile internal

- si codifica inserendo uno *style block* nella parte **head** di un documento HTML

```
...
<head>
  <title>titolo della pagina</title>
  <style type="text/css">
     h1 { color: red;  }
  </style>
</head>
<body>
  <h1>Intestazione</h1>
  ...
```

- metodo *molto usato*
- utilizzo *semplice*
- utile per *test*
- *non garantisce* la coerenza dello stile

Stile external

- usando il tag **<link>** nella componente **head** del documento HTML

```
...
<head>
  <title>titolo della pagina</title>
  <link type="text/css" rel="stylesheet" href ="/stili/nomefile.css" />
</head>
...
```

- oppure con il tag **style**

```
...
<head>
  <title>titolo della pagina</title>
  <style type="text/css">
        @import url(/stili/nomefile.css);
  </style>
</head>
...
```

- *coerenza* e più agevole *manutenzione* dello stile
- i file css esterni vengono trattati dal browser come file separati, quindi il caricamento della pagina è *più veloce* (non occorre scaricare ogni volta il file css)

Tipi di regole del css

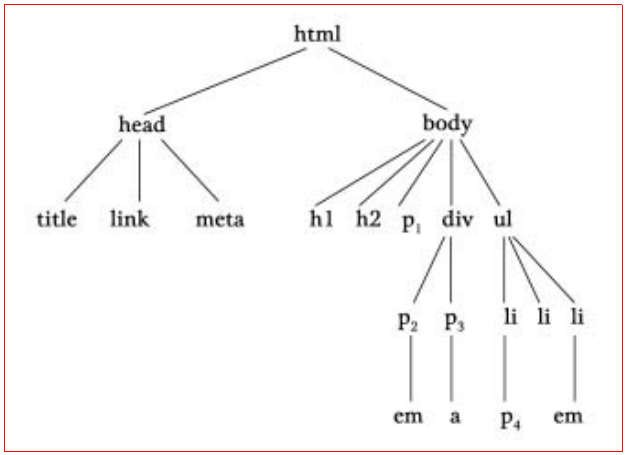
- Su quali *proprietà* agiscono?
  - *praticamente su tutte*
- Su quali tipi di *elementi*?
  - *è possibile selezionare su quali particolari elementi*

Come individuare gli elementi su cui agisce il css

- *Tutti* gli elementi di un certo tipo
- Tutti gli elementi di un certo tipo assegnati ad uno *stesso gruppo o classe*
- Tutti gli elementi di un certo tipo *contenuti in altri elementi* di un certo tipo
- Tutti gli elementi di un certo tipo *contenuti* in altri elementi di un certo tipo *e* assegnati ad uno *stesso gruppo o classe*
- Tutti gli elementi di un certo tipo solo quando compaiono *immediatamente dopo* elementi di qualche altro tipo
- Solo uno specifico elemento di un determinato tipo al quale viene assegnato un *ID univoco*

Le regole di inheritance (ereditarietà)

- Ogni elemento appartiene al *document inheritance tree*
- L'elemento radice è sempre l'elemento **`html`**
- I suoi discendenti diretti sono **`head`** e **`body`**
- Molte proprietà (es. **`font-family`**) prendono il valore specificato per l' elemento padre



Ereditarietà

- Con CSS non è necessario specificare le proprietà di ogni elemento
- Ove non presenti regole specifiche, ogni elemento erediterà lo stile dell'elemento padre
- Con questa regola

```
h1 { color: red; }
```

  e questa parte di codice xhtml

```
<h1>Un <em>titolo</em><h1>
```

  "*titolo*" sarà rosso e corsivo (corsivo perché *em* per definizione viene cosi rappresentato ma potrebbe non essere così)

- Ma se specificata anche la seguente regola

```
em { color: green; font-style: normal; }
```

- Allora questa sostituirà lo stile ereditato da **<h1>** e "titolo" sarà di colore verde, non rosso, e non corsivo ma normale

Valori ereditati

- Se non specificata una proprietà, CSS assume un valore di default
- A parte pochi casi, questo è sempre ereditato, ovvero la proprietà assume lo stesso valore che ha nella scatola contenitore dell'elemento in questione
- Tra i valori non ereditati:
  - *display (per HTML è sempre il valore naturale dell'elemento, block per* `p` *o* `h1`*, inline per* `strong`*,* `em` *o* `a`*, mentre per XML è inline)*
  - *background (sempre transparent)*

La dichiarazione !important

- CSS permette sia agli autori che agli utenti di esprimere preferenze sulle regole di presentazione
- Le regole degli autori sostituiscono le regole dei lettori a meno che il lettore non usi una dichiarazione !important. Per esempio la seguente regola diche che l'elemento p deve essere colorato di blu anche se l'autore del documento ha richiesto che sia in un colore differente
- Daltra parte font-family deve essere serif solo se l'autore non ha detto diversamente

```
p { color: blue !important; font-family: serif;}
```

- Se però la regola dell'autore è stata dichiarata !important con CSS1 vinceva la regola dell'autore. Con CSS2 è cambiato e vince la regola del lettore
- E' possibile definire regole multiple per gli stessi elementi, e adottare un *meccanismo a cascata* per la loro applicazione:
  - *User Agent: il browser definisce (o esplicitamente o implicitamente codificandole nel software) le regole di default per gli elementi dei documenti*
  - *User: l'utente può fornire un ulteriore foglio di stile per indicare regole di proprio piacimento. Tipicamente è una funzione del browser*
  - *Author: l'autore delle pagine fornisce, nei modi visti in precedenza, i fogli di stile del documento specifico*
  - *Regole !important : Quando una regola utente (tipicamente) è seguita dalla keyword !important, essa sopravanza una analoga regola di senso diverso dell'autore*

Ordine della cascata

- In generale le regole più specifiche vincono. Esempio:

```
<p> bla bla
<cite id="id1" class="particolare">
    bla bla bla
</cite>
</p>
```

- Le regole più specifiche sono preferite. Pertanto quella che seleziona l'elemento `cite` per il suo `id` sarà preferita a quella che seleziona `cite` per il suo `class`
- Una regola che seleziona `cite` per il suo `class` sarà preferita a quella che seleziona `cite` contenuto negli elementi `p`
- Infine, se nessuna di queste regole è applicata, sarà selezionata una regola generica `cite`
- Se non c'è un selettore che si appaia, viene ereditata la proprietà dall'elemento padre
- Se non c'è nessun valore ereditato dal padre viene usato il valore di default
- A parità di specificità si applica l'ultima in ordine di apparizione
- L'ordine di priorità è
    1. *User defined style*
    2. *Embedded or inline style sheet*
    3. *Internal style sheet*
    4. *External style sheet*
    5. *Browser default style*

Valori di proprietà in CSS

Le proprietà CSS possono essere nomi o valori - I nomi sono tutte parole chiave CSS

- Per quanto riguarda i valori esistono diverse possibilità:
  - *Parole chiave come "`display: none;`" in display o come solid in "`border-style: solid;`"*
  - *Misure espresse come*
    - *Numeri interi*: valori numerici assoluti
    - *Valori numerici*: espressi in unità di misura: 0,5in (`margin-top: 0.5in`), 12pt (`font-size: 12pt`)
    - *Percentuali*: espressi in relazione al contenitore padre
  - *Altri valori possono essere:*
    - URL come in: `{background-image:url(http://www.isti.cnr.it/immagini/carta.gif}`
    - oppure colori RGB come #CC0033 `{color: #CC0033}`
- Esistono quattro tipi di valori di proprietà:
  1. *length*
  2. *URL*
  3. *color*
  4. *keyword*

Esempi: nomi di proprietà e valori

Nome	Valore
display	none
font-style	italic
margin-top	0.5in
font-size	12pt
border-style	solid
color	#CC0033
background-color	white
background-image	url(http://www.isti.cnr.it/immagini/sfondoblu.gif)
list-style-image	url(/immagini/pallinorosso.png)
line-height	120%

Unità di misura

- Usate per definire larghezza, altezza, dimensione di parole e lettere, spazi, indentazione di testo, altezza di linee, margini, padding, larghezze di bordi, …
- Le misure possono essere specificate in tre modi:
    1. *Unità assolute: in (inches), cm, mm, pt (points, 1 pt corrisponde a 1/72 pollice), pc (pica, corrispondente 12pt, quindi ad 1/6 di pollice)*
    2. *Unità relative: bigger, +1*
    3. *Percentuali: 50%*

(vedi: http://www.w3.org/TR/CSS2/syndata.html#length-units)

Unità di misura relative

- CSS mette a disposizione tre metodi per specificare le unità relative di misure
  1. ***em**: la larghezza della lettera m del font corrente*
  2. ***ex**: l'altezza della lettera x del font corrente*
  3. ***px**: la grandezza di un pixel (si assume uguale a 0.75pt)*
- In questo esempio lo spessore dei bordi destro e sinistro dell'elemento p corrisponderanno alla larghezza della lettera m del font corrente, i bordi superiore e inferiore alla metà dell'altezza della lettera x del font corrente:

```
p  { border-right-width: 1em; border-left-width: 1em;
     border-top-width: 0.5ex; border-bottom-width: 0.5ex; }
```

Unità di misura in percentuale

- È *sconsigliato* usare la misura di lunghezza in pixel per vari motivi:
  - *La grandezza dei pixel varia con la risoluzione*
  - *Con il passare del tempo i monitor incrementano la densità di punti*
  - *Con l'aumentare della risoluzione la specifica del pixel può rendere illeggibile il testo*
- Le misure possono essere specificate in percentuale rispetto al valore della grandezza dell'elemento padre

```
p:first-letter  { font-size: 300%; }
em              { font-size: 120%; }
```

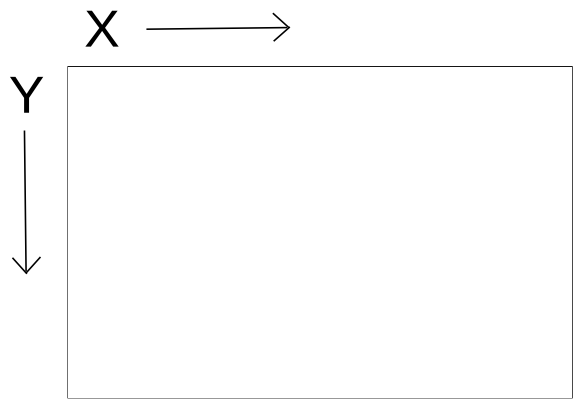
Il testo

Del testo è possibile controllare sia gli aspetti relativi al font che quelli relativi all'organizzazione del testo nella scatola di riferimento:

- font-family: il/i nomi del/dei font
- font-style (*normal* | *italic* | *oblique*), font-variant (*normal* | *small-caps*), font-weight (*normal* | *bold* | *bolder* | *lighter* | *100<-> 900*), font-stretch (*normal* | *wider* | *narrower* | *ultra-condensed* | *extra-condensed* | *condensed* | *semi-condensed* | *semi-expanded* | *expanded* | *extra-expanded* | *ultra-expanded*): caratteristiche del font
- text-indent, text-align (*left* | *right* | *center* | *justify*), line-height: indentazione, allineamento e interlinea delle righe della scatola
- text-decoration (*none* | *underline* | *overline* | *line-through* | *blink*)
- text-shadow: ulteriori stili applicabili al testo
- letter-spacing e word-spacing: spaziatura tra lettere e tra parole
- text-transform (*capitalize* | *uppercase* | *lowercase* | *none*): trasformazione della forma delle lettere
- white-space (*normal* | *pre* | *nowrap*): specifica la gestione dei ritorni a capo e del collassamento dei whitespace all'interno di un elemento

Modello di formattazione

- CSS definisce un modello di visualizzazione bidimensionale
- Gli elementi da visualizzare sono inseriti in rettangoli immaginari detti scatole (box)

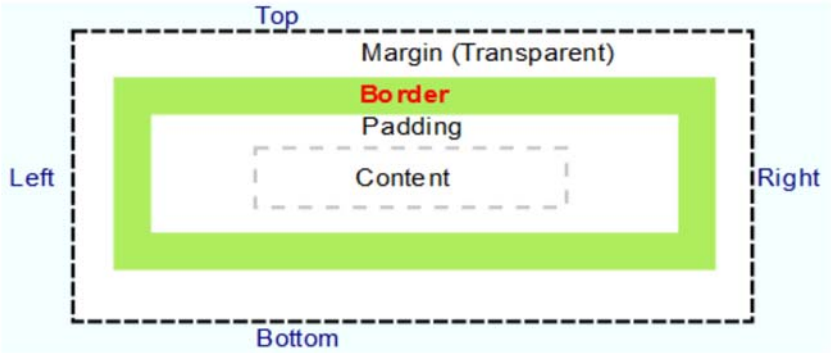


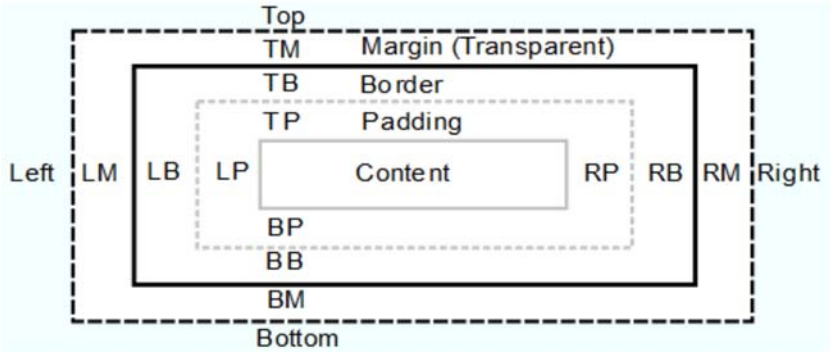
Box

Le scatole hanno delle proprietà che permettono di definire

- altezza, larghezza, dimensione e posizione
- aree relative a margini, cuscinetti (margini interni - padding), bordi (trasparenti per default)

La larghezza della scatola è data dalla somma della larghezza del contenuto (ovvero dell'elemento testo o immagine) + quella delle aree dei padding, bordi e margini.





Box

- *Margini*: la regione che separa una scatola dall'altra, necessariamente trasparente
  - *margin-top, margin-bottom, margin-left, margin-right: dimensioni del margine della scatola*
- *Border*: la regione ove visualizzare un bordo per la scatola (trasparente per default)
  - *border-top, border-bottom, border-left, border-right, border-width, border-color: dimensioni ed aspetto del bordo*
  - *border-style: può assumere come valori none, dotted, dashed, solid, double, groove (incavo), ridge (rilievo), inset, outset*
- *Padding*: la regione di respiro tra il bordo della scatola ed il contenuto - Ha il colore dello sfondo
  - *padding-top, padding-bottom, padding-left, padding-right: dimensioni del padding della scatola*
- *Content*: la regione dove sta il contenuto dell'elemento
  - *background-color, background-image, background-repeat, background-attachment, background-position: colore, immagine e meccanismo di ripetizione dell'immagine di sfondo della scatola*

Tipi di elementi

- Elementi "**Block"**; (blocco) sono elementi che sono mostrati in un blocco per conto proprio
  - *p (paragrafo), div (blocco generico), pre (blocco preformattato), address (autore della pagina), blockquote (citazione lunga)*

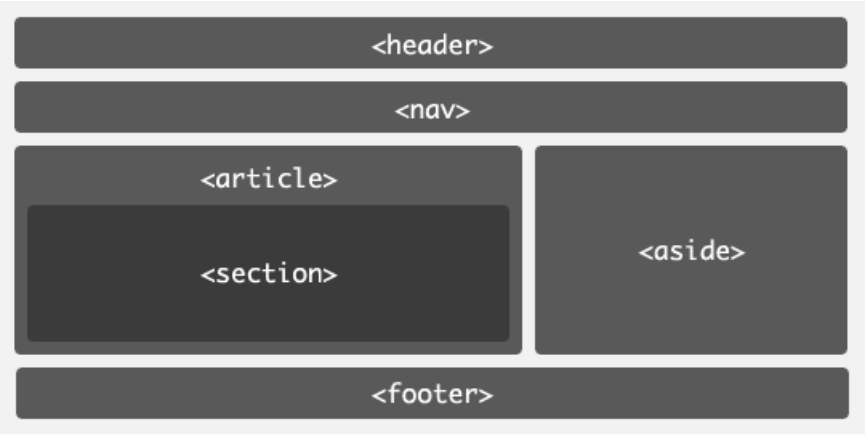
  possono essere inclusi solo da altri elementi blocco.
- Elementi "**Inline**" sono elementi che non provocano un a capo e possono essere inclusi da qualsiasi altro elemento
  - *a (ancora, collegamento), em (enfasi), strong (maggiore enfasi) span (generico elemento inline), dfn (definizione), code (frammento di programma), samp (output d'esempio), kdb (testo inserito dall'utente), var (variabile di programma), cite (breve citazione), q (citazione lunga), abbr, acronim (abbreviazioni ed acronimi), sup, sub (testo in apice e in pedice), bdo (bidirectional override)*
- Elementi "**List-item**" sono elementi con un marcatore (punto elenco, numero)

e

- ** ** (*division*) è un elemento *block-level*
- **** (*span of words*) è un elemento *inline*
- molto utili se usati con l'attributo **class**
- permettono, in pratica, di creare nuovi *HTML element*

```
<div class="nomeclasse">
<p>
In una div possono comparire elementi inline, e anche
<span class="specificaSpan"> parole che assumono le caratteristiche</span>
assegnate alla classe "specificaSpan".<br />
La div può avere caratteristiche di visualizzazione particolari.
</p>
</div>
```

html5 introduce delle specializzazioni di div: nav, footer, … sono elementi strutturali che potranno essere usati per specificare parti strutturali della pagina



Posizionamento

Come si inseriscono le scatole nello spazio di visualizzazione:

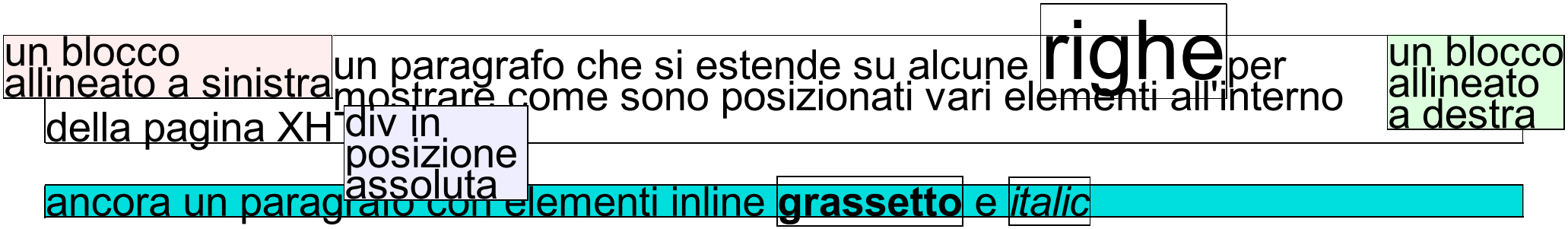
- *Flusso normale di tipo blocco*: una sopra l'altra in successione verticale (come paragrafi)
- *Flusso normale di tipo inline*: una accanto all'altra in successione orizzontale (come parole della stessa riga)
- *Flusso di tipo float*: all'interno del contenitore e poi spostate all'estrema sinistra o destra della scatola, lasciando che le altre scatole vi girino intorno
- *Posizionamento assoluto*: nella posizione indicata indipendentemente dal flusso e da quel che la zona già visualizza (eventualmente nascondendo ciò che sta sotto)

Modello visuale di CSS

Proprietà che controllano il tipo di posizionamento e quello di scatola:

- **display** *(inline | block | list-item | ... | none)*: il tipo di scatola da utilizzare per l'elemento: blocco, inline, lista, cella di tabella, ...
- **position** *(static | relative | absolute | fixed)*: il posizionamento rispetto al flusso del documento
- **float** *(left | right | none)*: un float è una scatola scivolata all'estrema destra o sinistra dell'area disponibile all'interno del contenitore
- **z-index**: la posizione nello stack di scatole potenzialmente sovrapposte. Il valore più alto è più vicino al lettore, e quindi nasconde gli altri - default background delle scatole = trasparente
- **top, bottom, left, right**: coordinate della scatola
- **width, height**: dimensioni usabili invece di right e bottom

Un esempio di posizionamento



Un esempio di posizionamento

```
<div>
<div style="display:block; float:right;background:#ffeeee;border-style: solid; border-width: 1px;">un blocco <br/> allineato<br/> a destra</div>
<div style="display:block; float:left;background:#ffeeee;border-style: solid; border-width: 1px;">un blocco <br/> allineato a sinistra </div>
<p style="display:block; border-style: solid; border-width: 1px;">un paragrafo che si estende su alcune <span style="display:inline; border-style: solid; border-width: 1px; font-size: 200%;">righe</span>per mostrare come sono posizionati vari elementi all'interno della pagina XHTML</p>
<p style="display:block; border-style: solid; border-width: 1px;">ancora un paragrafo con elementi inline <span style="display:inline; font-weight:bold; border-style: solid; border-width: 1px;">grassetto</span> e <span style="display:inline; font-style:italic; border-style: solid; border-width: 1px;">italic</span>
<div style="display:block; position:absolute; top:180px; left: 280px; width: 150px; background:#eeeeff;border-style: solid; border-width: 1px;">div in posizione assoluta</div>
</p>
</div>
```

Il layout “fluido”

- L'attributo `float` permette di posizionare un elemento in modo “fluido”
- `float:left;` o `float:right;` posiziona l'elemento il più a sinistra o il più a destra dello *spazio disponibile*
- Se non c'è spazio sufficiente, l'elemento verrà posizionato secondo il flusso normale, quindi *sotto* l'elemento corrente
- Tenere sempre conto dell'*ingombro totale* del box (margini, padding)
  - *i vari browser si comportano in modo diverso!*
- Vediamo un esempio

Raggruppamento di Selettori

- Se abbiamo regole identiche che si applicano ad elementi diversi

```
p  { display: block; margin-bottom: 10px; }
h1 { display: block; margin-bottom: 10px; }
```

- Possiamo raggruppare gli elementi separandoli con una virgola

```
p, h1 { display: block; margin-bottom: 10px; }
```

I selettori (quadro sinottico)

Tipo	Significato (la regola si applica a ...)	Esempio
selettore universale	tutti gli elementi nel documento	`*`
element type	tutti gli HTML elements del tipo del selettore	`h1` `p`
class selector	tutti gli elementi HTML che precedono il punto (o tutti, se non è specificato nulla) la cui definizione li rende elementi della classe il cui nome segue il punto	`.articletitle` `h1.important`
ID selector	l' unico elemento nel documento il cui attributo coincide con la stringa che segue il simbolo #	`#special3` `p#special52`

pseudo-element selector (CSS2)	le istanze dello pseudo-element	`p:first-letter`
		`p:first-line`
		`h1:first-child`

pseudo-class selector (CSS2)	istanze della pseudo-classs, la cui presentazione può variare a seguito dell' interazione dell' utente con la pagina	`a:hover`
		`a:active`
		`a:focus`
		`a:link`
		`a:visited`
		`body:lang(d)`

descendant selector	tutti gli elementi del tipo più a destra di una lista (separata da spazi), solo quando l' element type discende dal tipo alla sua sinistra	`p em` `p.wide em`
parent-child selector (CSS2)	tutti gli elementi del tipo specificato a destra del simbolo ">", che sono figli degli elementi a sinistra del simbolo ">" (è una forma più specifica del descendant selector)	`body > p`

Tipo	Significato (la regola si applica a ...)	Esempio
adjacent selectors (CSS2)	tutti gli elementi del tipo specificato a destra del segno "+", adiacenti (nel codice HTML) agli elementi a sinistra del segno "+"	`h1+h2` `p+h3`

attribute selectors (CSS2)	attributi che corrispondono al profilo specificato nelle parentesi quadre	`p[align]`
		`input[type="text"]`
		`img[alt~="none"]`
		`body[lang|="en"]`

Universal Selector

La regola si applica a:
tutti gli elementi nel documento

```
* {
    color: red;
    font-style: ;
  }
```

Esempio

Element Type Selector

La regola si applica a:
tutti gli HTML elements del tipo del selettore

```
body {
      color: white;
      background-color: black;
     }
```

Esempio

Class Selector

La regola si applica a:
tutti gli elementi HTML che precedono il punto (o tutti, se non è specificato nulla) la cui definizione li rende elementi della classe il cui nome segue il punto

```
.blue {
       font-family: verdana, arial, sans-serif;
       color: #0000ff;
       }
```

Saranno in blu tutti gli elementi per i quali si specifica
`class="blue"` (esempio)

Si possono definire gli elementi che possono appartenere a una classe (esempio):

```
h1,h2.blue {
            font-family: verdana, arial, sans-serif;
            color:#0000ff;
           }
```

ID Selector

La regola si applica a:
l' unico elemento nel documento il cui attributo coincide con la stringa che segue il simbolo #

esempio

Pseudo-Element Selector

La regola si applica a:
le istanze dello pseudo-element
(è una proprietà CSS2)

Pseudo-elementi

- La prima lettera di un paragrafo

```
p:first-letter { font-size: 300%; }
```

- La prima linea di un paragrafo

```
p:first-line { font-variant: small-caps; }
```

- before, after: si applica prima / dopo il contenuto dell'elemento.

esempio

Pseudo Class Selector

La regola si applica a:
*istanze della pseudo-class, la cui presentazione può **variare a seguito dell' interazione** dell' utente con la pagina*
(è una proprietà CSS2)

- **`a:link`** non visitato, link non attivo
- **`a:hover`** mouse sopra
- **`a:active`** attivato (cliccato col mouse o attivato con altro dispositivo)
- **`a:visited`** visitato
- **`:focus`** "l'oggetto a fuoco" (l'oggetto selezionato)
- **`:first-child`** il primo figlio dell'elemento
- **`:left`** le pagine a sinistra di un documento (generalmente per la stampa)
- **`:right`** le pagine a destra

esempio

Selettori Contestuali

- Se la formattazione dipende dalla posizione all'interno di altri elementi (per esempio se dipende dall'elemento padre) è possibile scrivere regole che si applicano agli elementi contenuti all'interno di altri specifici elementi
- Anteporre il nome dell'elemento avo (e padre) a quello dell'elemento al quale vogliamo applicare la proprietà di stile specifica.

```
p cite { color: red; }

<p>paragrafo con <cite>citazione</cite> interna</p>
```

- Risultato:

  paragrafo con *citazione* interna

Altri selettori

- Selettore universale (*):

```
*:   La regola si applica a tutti gli elementi nel documento
```

- Selettori contestuali (A D P>F F + PF):
  Basati sugli elementi figli, fratelli, genitori

```
Avo Discendente    D discendente di A
Padre>Figlio      F figlio di P
Fratello+PrimoFratello PF primo fratello di F

H2+P { color: red; }
```

- Selettori di attributi (E[attr] E[attr="valore"]):

```
E[attr]           elementi E che hanno un attributo attr
E[attr="valore"]   elementi E che hanno un attributo attr il cui valore è "valore"

A[NAME] { color: green; }
```

Descendant Selector

La regola si applica a:
tutti gli elementi del tipo più a destra di una lista (separata da spazi), solo quando l' element type discende dal tipo alla sua sinistra

esempio

Parent-Child Selector

La regola si applica a:
tutti gli elementi del tipo specificato a destra del simbolo ">", che sono figli degli elementi a sinistra del simbolo ">" (è una forma più specifica del descendant selector)
(è una proprietà CSS2)

esempio

Adjacent Selector

La regola si applica a:
tutti gli elementi del tipo specificato a destra del segno "+", adiacenti (nel codice HTML) agli elementi a sinistra del segno "+"
(è una proprietà CSS2)

esempio

Attribute Selector

La regola si applica a:
attributi che corrispondono al profilo specificato nelle parentesi quadre
(è una proprietà CSS2)

esempio

- **`[attribute]`** – matches if the attribute is defined at all for the element(s)
- **`[attribute="setting"]`** – matches only if the attribute is defined as having the value of setting
- **`[attribute~="setting"]`** – matches only if the attribute is defined with a space-separated list of values, one of which exactly matches "setting"
- **`[attribute|="setting"]`** – matches only if the attribute is defined with a hyphen- separated list of "words" and the first of these words begins with setting

Liste e sottoliste numerate (CSS2)

```
ul, ol { counter-reset: item }
li { display: block; }
li:before { content: counters(item, "."); counter-increment: item }
```

numera liste e sottoliste nel formato
1
1.1
1.1.1
...

Cambio dinamico dello stile CSS

Tecniche di accessibilità WCAG 2.0

Esempio modifica dinamica di parti del documento

Esempio cambio del file CSS

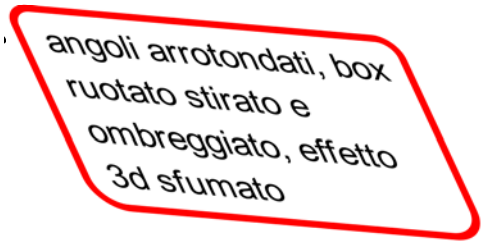
Gli standard non limitano la creatività

- Un esempio tra tanti, il subacqueo, e una sua spiegazione
- e un sito come riferimento: CSS Zen Garden
- vi sono poi altri standard grafici come SVG, Webgl e il nuovo elemento canvas di HTML5
- Per rendersi conto degli effetti grafici possibili (notizia del 6/4/2010): porting di DOOM2 in HTML5

Inoltre: alcuni cenni di CSS3

molte sono le nuove funzionalità che aggiunge CSS3, alcuni esempi di effetti che in genere troviamo in programmi di grafica professionali (provati solo con firefox 3.5):

- testo ombreggiato



- riflessi

- ...

Strumenti e stato implementazione

- validatore CSS (W3C)
- Total validator
- WAVE - Web accessibility evaluation tool
- Juicystudio -Analizzatore di contrasto
- ..
- Stato di implementazione HTML5 e CSS3

Alcune buone prassi

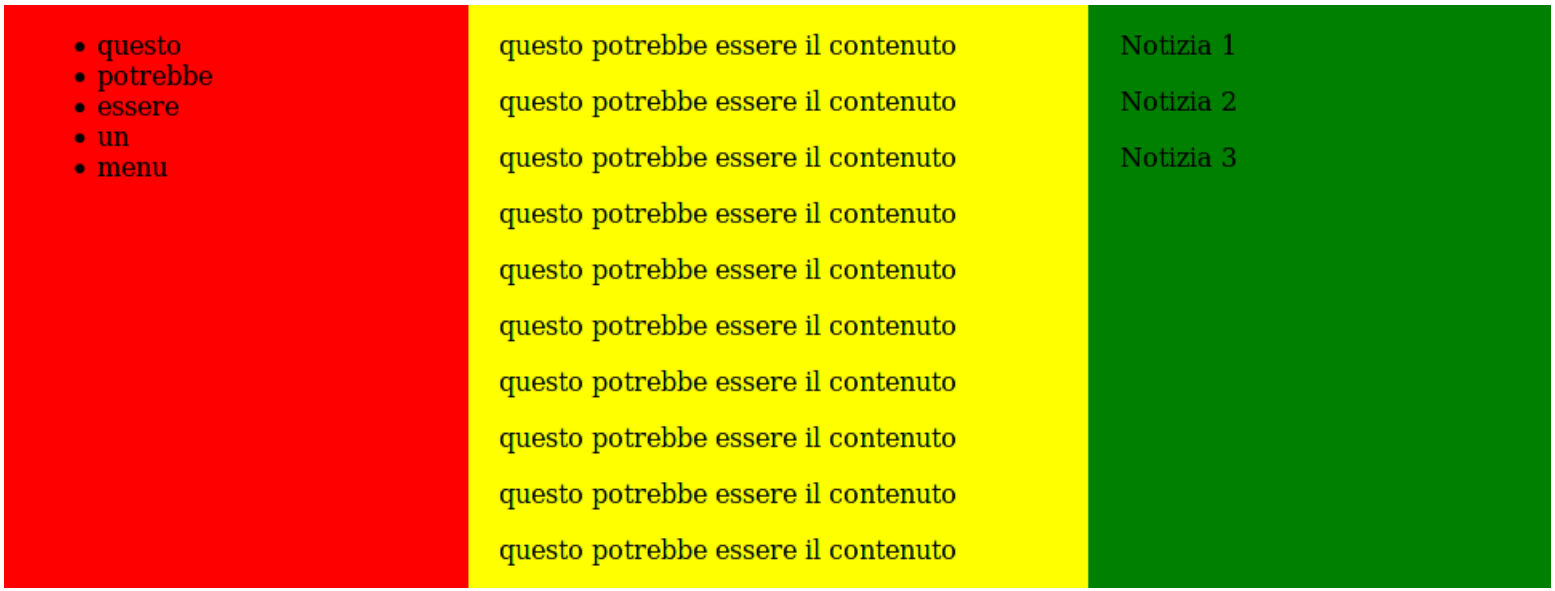
(rif. cap. 3 Wium Lee & Boss)

- Usare le unità **em** per creare style sheet scalabili
- Usare sempre le unità **em** per impostare il font-size
- Usare unità relative per le lunghezze
- Usare unità assolute di lunghezza solo quando sono note le caratteristiche fisiche del dispositivo di output
- Usare elementi *floating* invece delle tabelle
- Disporre il contenuto in ordine logico
- Assicurarsi che il documento sia leggibile anche senza style sheets
- Provare il documento con diversi browser
- Specificare sempre un font generico come alternativa
- Fermarsi in tempo (evitare troppi effetti sulla stessa pagina)

Esercizi

Esercizio

Realizzare una pagina XHTML 1.1 con tre blocchi di testo
senza tabelle (a meno che non contengano dati e siano all'interno di un blocco)
senza posizionamenti assoluti



Una soluzione

Esercizio

Collegarsi al sito http://jigsaw.w3.org/css-validator/ e validare lo stile di una pagina del proprio ente

Eliminare tutti gli elementi e gli attributi di presentazione della pagina e realizzare un foglio di stile

validare il foglio di stile (http://validator.w3.org)

Trasformare la pagina (con un editor testi) in XHTML1 1.1

Avvalersi di Tidy per correggere il codice

Esercizio

Realizzare una pagina XHTML 1.1 con un form impaginato senza tabelle

Una soluzione realizzata utilizzando varie tecniche WCAG 2.0

Esercizio (svolto)

- Codificare una pagina web come in figura
- La pagina dovrà:
    1. *essere valida rispetto alla grammatica formale XHTML Strict*
    2. *utilizzare un layout "fluido" che occupi il 100% delle dimensioni orizzontali dello schermo*
    3. *essere ridimensionabile per dimensione della finestra e dei caratteri*
    4. *contenere immagini (in particolare il logo in alto a destra e quello in basso a sinistra) scalabili al ridimensionamento della finestra*
    5. *utilizzare gli pseudo-elementi `<div>` per la l'impaginazione (non sono ammesse tabelle di layout)*
    6. *implementare il menù di sinistra utilizzando elementi `<h1>` e `<h2>`*
    7. *rispettare le regole di accessibilità dei siti Web, come definite dal W3C*
    8. *avere un CSS di print per evitare la stampa del menù e di altri elementi accessori e decorativi (vedi l' esempio)*
- il sito in HTML
- il CSS, anche in una forma stampabile

Esercizio per "casa" (mandatemelo via email o ne discutiamo la prossima volta)

Preparare un mini sito Web costituito da tre documenti XHTML1.1 collegati tra loro:

- una prima pagina in cui si descrivono le finalità del sito, una sezione contenente i collegamenti alle altre pagine, dei link che consentono di saltare blocchi di testo e di collegamenti, un help, ...
- una seconda pagina che raccoglie le notizie ed eventi del mese
- una terza pagina con un form per richiesta informazioni

Grazie per l' attenzione

Domande?

... e risposte

Se non è sul Web non esiste ...

... troverete sul sito dell' Ufficio (http://www.w3c.it/)
le *slide* (http://www.w3c.it/education/2012/upra/css)

Queste slide fanno parte del materiale predisposto per il corso Architettura del Web