

Laboratorio di reti I: Cascading Style Sheets

Stefano Brocchi
brocchi@dsi.unifi.it

10 novembre, 2009

Cascading Style Sheets

- Un Cascading Style Sheets (CSS) è un documento che specifica le impostazioni grafiche di una pagina HTML
- Nel creare pagine HTML è importante conoscere i CSS in quanto ci permettono di separare la vista grafica (definita nel CSS) dal contenuto (nel codice HTML)

Uso dei CSS

- I CSS sono attualmente considerati una necessità per la grafica, tanto che tutti i tag fisici che realizzano la grafica direttamente in HTML (come ``, `<I>` o ``) sono *deprecati*
- Per validare documenti con CSS senza deprecazioni possiamo quindi usare le specifiche HTML più rigide, specificando all'inizio della pagina l'intestazione `<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">`

Vantaggio dei CSS

Potendo essere risorse esterne rispetto ad un file HTML, i CSS offrono molte possibilità:

- Cambiare la grafica senza toccare il codice che descrive il contenuto, e viceversa
- Modificare la grafica di molte pagine web modificando un unico documento
- Visualizzare più pagine web scaricando un unico foglio di stile, risparmiando così sull'occupazione di banda
- Cambiare agevolmente la grafica di una pagina, a seconda per esempio di una scelta dell'utente o del dispositivo a cui questa è destinata

Vantaggio dei CSS

- Usando i CSS quello che dovremo indicare nel codice HTML non è tanto come una determinata zona debba essere graficamente rappresentata, ma cosa rappresenti. Nel CSS verrà specificato come debbano apparire i vari oggetti
- Grazie ai CSS è possibile impostare non solo gli aspetti grafici più semplici come colori e font, ma modificare caratteristiche più complesse come il posizionamento delle varie parti della pagina

Importazione di CSS

- Ci sono due modi per collegare un CSS ad una pagina web
- Uno è tramite il tag `<style>`, da inserire all'interno del tag `<head>` del documento, dentro il quale va inserita la clausola `@import` specificante l'url del CSS, come nell'esempio:

```
<style>  
@import url(style.css);  
</style>
```

- Al posto della clausola `import` è possibile specificare il corpo del CSS stesso, imponendo nel tag `style` il valore dell'attributo `type` uguale a `text/css`:

```
<style type="text/css">  
...  
</style>
```

Importazione di CSS

- L'altro modo per collegare ad una pagina un CSS è quello tramite il tag `<link>`, sempre da specificare nel tag `<head>`, con del codice come il seguente:

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

- L'attributo `type` è obbligatorio e per i CSS deve contenere il valore `text/css`
- L'attributo `href` indica l'url del foglio di stile
- L'attributo `rel` può avere valore `stylesheet` o `alternative stylesheet` se si vuole specificare un foglio di stile alternativo, che potrà essere scambiato con quello principale tramite vari mezzi (es. JavaScript)

Importazione di CSS

- Nel tag `<link>` è possibile specificare inoltre l'attributo `media` per specificare che un determinato CSS va utilizzato solo in caso che il dispositivo che legge la pagina sia di un certo tipo.
- Alcuni possibili valori sono:
 - `all` Il CSS viene sempre applicato (valore di default)
 - `screen` Rappresentazione su schermo: valore standard per i browser
 - `print` Il CSS viene applicato quando viene richiesta la stampa della pagina
 - `handheld` Per dispositivi portatili come palmari
 - `braille` Il CSS viene usato per supporti basati sull'uso del braille

Contenuto di un CSS

- Un foglio di stile è formato da una serie di dichiarazioni nel seguente formato:

```
selettore {  
    proprietà: valore;  
    proprietà: valore;  
    ...  
}
```

- *Selettore* è una stringa che indica a quali elementi verrà applicata la regola
- Le coppie *proprietà, valore* indicano una serie di proprietà grafiche da attribuire agli elementi specificati
- E' possibile inoltre inserire commenti in un foglio di stile con la sintassi */* commento */*

Selettori di base

- Il più semplice selettore da specificare è un tag HTML
- In questo caso tutto il contenuto di quel determinato tag avrà le proprietà specificate
 - Ad esempio:

```
h3 { color:red; }
```

Imposta che tutto il testo dentro tag h3 sarà in rosso
- Si possono definire proprietà per più tag elencandoli nel selettore separati da una virgola
 - h2, h3 { color:red; }
- E' possibile specificare, anche nei selettori descritti in seguito, il carattere di wildcard *

Selettori del figlio

- E' possibile specificare che ad un tag va applicata la regola solo se questo è contenuto in un altro tag
- In questo caso usare la sintassi
tag-padre tag-figlio {...}
 - Es. `li strong { background:yellow; }` specifica che tutto il testo in un tag `` che sia anche dentro una lista (tag ``) avrà sfondo giallo
- Esiste una modalità per specificare che un tag viene considerato se è figlio *diretto* di un altro:
tag-padre > tag-figlio {...}
- Quest'ultima modalità sembra non essere attualmente supportata da explorer

Selettori di attributi

- E' possibile selezionare tag che hanno definito uno specifico attributo, o tag per cui un particolare attributo assume un determinato valore.
- In questi caso la sintassi da usare è la seguente:

tag-padre [*attributo*] { ... }

tag-padre [*attributo* = *valore*] { ... }

Uso di attributi class ed id

- Per poter definire liberamente caratteristiche di un determinato tag, esistono due attributi che è possibile specificare per ogni tag HTML: `class` e `id`
- Tramite l'attributo `class` si specifica che l'attributo appartiene ad una determinata classe di oggetti. Sarà possibile selezionare tutti gli attributi di questa classe con la sintassi

```
.nomeclasse { ... }
```

- Per esempio tramite la dichiarazione

```
.in-evidenza { background:yellow; }
```

si specifica che tutto lo sfondo del contenuto dei tag di classe `in-evidenza` sia giallo

Selettori di classi

- E' possibile specificare più classi per un singolo tag, separandole con uno spazio
- In questo caso il tag avrà le caratteristiche di entrambe le classi specificate
 - Es.
`<p class="in-evidenza enfattizzato">`
- Vedremo in seguito come vengono risolti eventuali conflitti di attributi

Selettori di id

- L'attributo `id` permette funzionalità simili all'attributo `class`. La differenza è che per ogni documento *un solo tag* deve contenere un determinato `id`
- Per selezionare un tag con un determinato identificativo usare la sintassi

```
#nomeid { ... }
```

- Es.

(nell'HTML) `<p id="intestazione">`

(nel CSS) `#intestazione {color:red;}`

Combinare i selettori

I vari selettori possono essere combinati fra loro per ottenere condizioni di selezione più complesse. Vediamo degli esempi:

- `p.importante` seleziona tutti i paragrafi di classe "importante"
- `li.enfaticizzato.importante` seleziona tutti gli elementi di una lista (``) che appartengono sia alla classe "enfaticizzato" che alla classe "importante"
- `#intestazione strong` seleziona tutti i tag strong all'interno del tag con id "intestazione"
- `div *` seleziona tutti i tag all'interno di tag `<div>`

Pseudo classi

- Esiste una particolare sintassi per indicare la selezione di elementi con un determinato stato
- In questo caso si parla di *pseudo-classi* e la sintassi da utilizzare è la seguente:

```
nome-tag:pseudo-classe { ... }
```

- Alcune delle pseudo-classi più comuni sono quelle per distinguere i collegamenti visitati o no. Le quattro pseudo classi sono:
 - `A:link` per i collegamenti non visitati
 - `A:visited` per i collegamenti già visitati
 - `A:hover` per i collegamenti sui quali è posizionato il mouse
 - `A:active` per i collegamenti nel momento in cui vengono cliccati

Ereditarietà nei CSS

- In un foglio di stile la maggior parte delle caratteristiche di un tag vengono ereditate da tutti i tag figli
- Per sapere con precisione quali vengano ereditate e quali no è necessario consultare la documentazione dei CSS. In linea di massima tuttavia le uniche caratteristiche non ereditarie sono quelle legate al contenitore che vedremo in seguito (es. dimensioni o caratteristiche del bordo)

Selezione di regole

- Nel definire un foglio di stile è possibile che alcune regole siano in conflitto tra loro
- Il visualizzatore selezionerà la regola che risulta essere la più 'specificata'
- Per la selezione, vengono eseguiti i seguenti passi a cascata (da qui il "*Cascading*" di Cascading Style Sheets):
 - La regola a prevalere è quella che contiene il maggior numero di identificativi (*#nometag*)
 - In caso di parità di identificativi, viene selezionata la regola con il maggior numero di classi e pseudo-classi
 - In caso di ugual numero di identificativi e classi viene applicata la regola con maggior numero di tag
 - Se nessuna delle precedenti è applicabile, prevale semplicemente l'ultima regola dichiarata

Selezione di regole

- Nel caso siano definiti più fogli di stile, sono definite delle regole di precedenza anche tra i vari CSS
 - Ad esempio, il foglio di stile definito dall'autore avrà chiaramente la precedenza su quello di default presente nel browser. Per le caratteristiche non definite dal programmatore, si farà comunque riferimento a quest'ultimo documento
- Per specificare che una regola deve essere selezionata prima delle altre si può specificare la clausola `!important`
- Es. `h1 {color: red !important;}`

Impostazione del colore di primo piano

- Vediamo ora alcuni degli attributi impostabili con le regole CSS ed i possibili valori che questi possono assumere
- Per modificare il colore di primo piano (cioè quello del testo) usare l'attributo `color`
- I colori da assegnare all'attributo si possono specificare o come stringhe, per 16 colori predefiniti, o come coppie di valori esadecimali
 - Es.
`color: silver;`
`color: #AAAA00;`

Impostazione dello sfondo

- Similarmente si può impostare il colore di sfondo tramite l'attributo `background-color`
- E' possibile impostare un'immagine di sfondo utilizzando la sintassi `background-image: url(urlImmagine);`
- Altri attributi per la gestione dello sfondo sono `background-attachment` e `background-repeat`

Impostazione di caratteristiche del testo

Vediamo vari attributi per la gestione del testo ed i valori che questi possono assumere:

- `font-family` per definire il tipo di font da usare. Alcuni suoi valori possono essere `Times new roman`, `Arial` o `Courier`
- `font-weight` per il peso del testo (più o meno in grassetto). Specificare un valore numerico da 100 a 900 o una keyword fra `lighter`, `normal`, `bold` o `bolder`
- `font-size` per la dimensione del testo. Indicare un numero di pixel seguito da `px` (es. `font-size: 20px;`), una dimensione in un'altra unità di misura (es. `2cm`, vedere documentazione CSS) o una keyword predefinita (es. `x-small`, `large`)

Impostazione di caratteristiche del testo

Altri attributi per lo stile del testo:

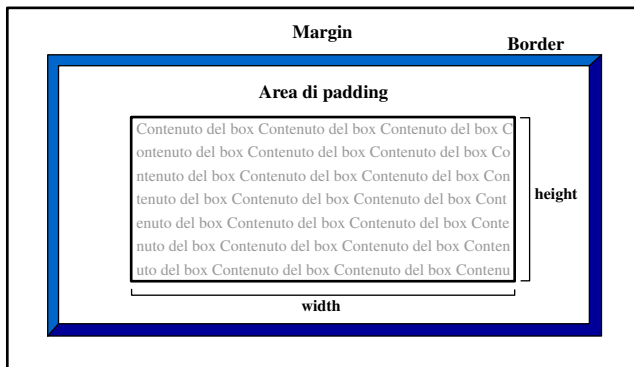
- `text-decoration` per varie decorazioni del testo: sottolineato (`underline`), barrato (`line-through`), con una riga sopra (`overline`), lampeggiante (`blink`) o senza decorazioni (`none`)
- `text-align` per specificare l'allineamento del testo nel suo contenitore. Valori possibili: `right`, `left`, `center` e `justify`

HTML box model

- Il linguaggio HTML esegue il posizionamento dei vari oggetti tramite un modello detto *box model*
- In una pagina ogni tag che rappresenta un contenitore (`img`, `p`, `table`) viene associato ad un box che definisce come verrà posizionato il contenuto nella pagina
- Il primo contenitore considerato è la zona dove viene rappresentata pagina stessa, e tutti gli altri box definiti nel codice HTML vi saranno contenuti
- Per definire un contenitore generico in HTML, usare il tag `<div>`
- Similarmente, si può usare il tag `` per definire contenitori all'interno di zone di testo detti *inline*

Struttura di un box

- La struttura di un box HTML è mostrata nella figura di seguito:



Struttura di un box

- I due valori `width` e `height` rappresentano la larghezza e l'altezza della zona con il contenuto del box
- La zona di `padding` è quella che distanzia il contenuto dai bordi del box
- Il bordo del box è rappresentato da un oggetto detto `border`
- I margini del contenitore definiscono la distanza minima che deve intercorrere tra il box e quelli adiacenti. Possono essere utilizzati per definire il posizionamento di un oggetto all'interno del suo contenitore

Struttura di un box: area contenuto

- L'altezza e la larghezza della zona con il contenuto possono essere definiti tramite gli attributi `height` e `width`
- Come per gli altri parametri che richiedono una misura, specificare una dimensione in punti o una percentuale che sarà relativa al box padre che contiene questo box
- Es.

```
width: 50%;
```

```
height: 400px;
```

Struttura di un box: padding

- La dimensione dell'area di padding può essere specificata, per quanto riguarda i quattro lati del box, con gli attributi `padding-top`, `padding-right`, `padding-bottom` e `padding-left`
- Esiste l'attributo di comodo `padding` che permette di specificare con una sola dichiarazione tutti e quattro i valori separati da uno spazio, partendo da quello in alto e procedendo in senso orario
 - Per esempio

```
padding: 10px 20px 30px 40px;
```

è equivalente a

```
padding-top: 10px; padding-right: 20px;
padding-bottom: 30px; padding-left: 40px;
```
- Se per `padding` viene specificato un unico valore, tutti e quattro i lati vengono impostati a quel valore. Definiti comportamenti di default per due o tre valori specificati

Struttura di un box: bordi

- Per quanto riguarda i bordi è possibile specificare tre proprietà riguardanti lo stile del bordo, il suo spessore ed il suo colore
- Gli attributi da utilizzare sono `border-lato-color`, `border-lato-style` e `border-lato-width` dove *lato* indica a quale dei bordi ci si sta riferendo (`left`, `right`, `bottom` o `top`)
- Del tutto analogamente alla proprietà di comodo `padding`, si possono utilizzare le proprietà `border-style`, `border-width` e `border-color` per impostare contemporaneamente più bordi
- Il valore di `border-width` deve essere una misura. Alcuni stili utilizzabili per `border-style` sono `none`, `dotted`, `dashed`, `solid`, `double`, `inset` e `outset`

Struttura di un box: margini

- Le dimensioni dei margini sono impostabili tramite le proprietà `margin-left`, `margin-right`, `margin-top`, `margin-bottom` o tramite l'attributo di comodo `margin`
- E' possibile specificare per i margini il valore `auto`. Questo imporrà un valore di default se riferito al margine in cima o in fondo al box, mentre centrerà il contenitore rispetto al contenitore padre se riferito ai due margini laterali
- Essendo il margine definito come la distanza *minima* tra due elementi, l'effettiva distanza nella visualizzazione tra due box adiacenti sarà data dal *maggiore dei due margini* e non dalla loro somma

Attributo float

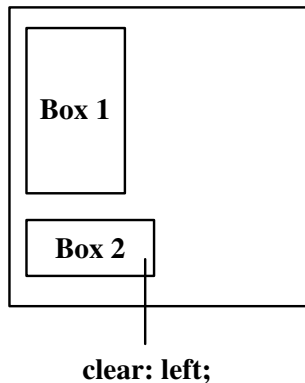
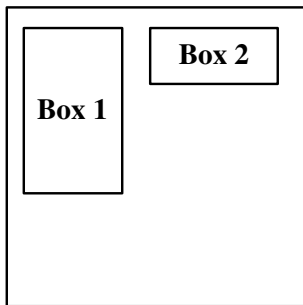
- Vedremo ora una serie di attributi utilizzabili per definire il posizionamento dei vari box
- Normalmente, i box vengono posizionati nella pagina dal browser nello stesso ordine in cui vengono incontrati nel codice
- Tramite l'attributo `float` è possibile scorrere un elemento alla lato sinistro o destro del contenitore padre.
- I tre possibili valori per l'attributo `float` sono `left`, `right` e `none`

Attributo clear

- Si può specificare che determinati box definiti con `float` siano posizionati nel lato indicato solo se non ci sono altri elementi già accostati a quel lato alla stessa altezza
- In questo caso, il browser scorrerà il box verso il basso fino a trovare una posizione dove il lato sia libero
- Per questo scopo utilizzare l'attributo `clear` con valore `none`, (valore di default) `left`, `right` o `both` (per attendere che siano liberi entrambi i lati)

Float e clear: esempio

- Vediamo un esempio con due box con attributo `float: left;`. A destra, vediamo il diverso posizionamento se definiamo l'attributo `clear: left;` nel box 2.



Attributo position

- Per definire il posizionamento di un determinato box, si può usare l'attributo `position`
- Il valore di default per `position` è `static`, che significa che l'elemento andrà piazzato normalmente nella pagina a seconda della sua posizione nel codice

Attributo `position`: `absolute`

- Se l'attributo `position` viene invece impostato a `absolute`, l'elemento viene rimosso dal normale flusso e riposizionato
- La posizione di destinazione si specifica tramite gli attributi `top`, `bottom`, `left` e `right`
- Tramite questi si indica la distanza dai margini rispetto al primo contenitore parente non statico che lo contiene
- Se il box non è contenuto in alcun altro box statico, le coordinate si riferiscono ai margini della pagina stessa
- Tramite questo riposizionamento il browser non tiene conto di eventuali sovrapposizioni; fare attenzione quindi a non coprire il contenuto di un altro box

Attributo position: absolute

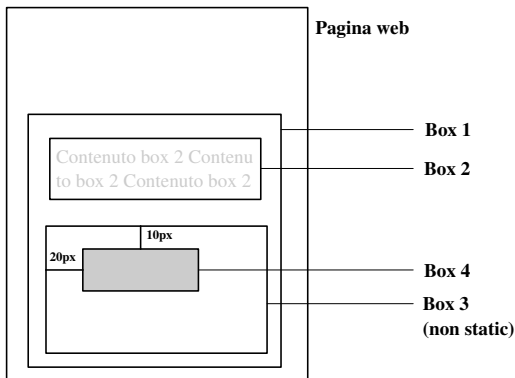
- Vediamo un esempio di posizionamento assoluto. Consideriamo il box detto Box 4 contenuto in un box detto Box 3 a sua volta contenuto nel Box 1
- Specifichiamo il seguente codice per il box considerato:

```
#box4 {  
    position: absolute;  
    width: 80px;  
    height: 20px;  
    top: 10px;  
    left: 20px;  
    background-color: silver;  
}
```

- Vediamo dove il box viene posizionato a seconda delle caratteristiche dei suoi box contenitori

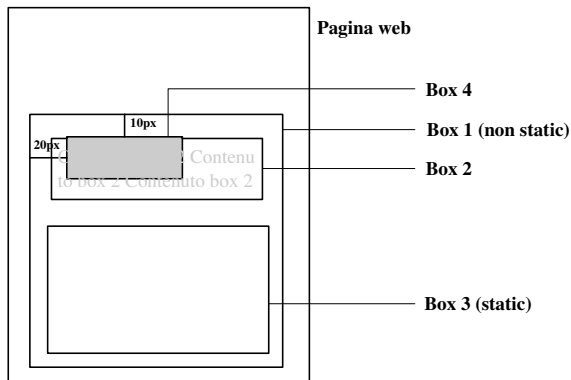
Attributo position: absolute

- Se il box 3 (il parente più vicino) non ha posizionamento statico, il box 4 è posizionato in funzione di quest'ultimo



Attributo position: absolute

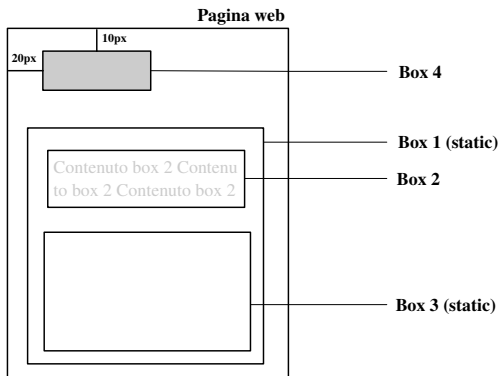
- Se il box 3 è dichiarato `static`, ma il box 1 non lo è, il box 4 verrà posizionato rispetto al box 1



- Notare che se lo spazio indicato non è libero, il box viene sovrapposto ad altri contenuti

Attributo position: absolute

- Infine se nessun box parente di box 4 ha l'attributo position diverso da static, il posizionamento avverrà rispetto alla finestra contenente la pagina web



Attributo position: relative

- Un altro valore per l'attributo position è relative
- Si indica così di posizionare il box in un determinata posizione relativa a dove il box sarebbe se fosse dichiarato statico
- La quantità e la direzione dello spostamento saranno specificati con i soliti attributi top, bottom, left e right
- Questo spostamento non ha effetto sul posizionamento degli altri box. I margini dei box circostanti, per esempio, faranno riferimento alla posizione 'originaria' del box
- Può essere utile definire box con posizionamento relativo ma con uno spostamento di zero punti in tutte le direzioni per indicare che il box deve essere posizionato come se fosse dichiarato static, ma i box absolute in esso contenuti devono far riferimento alle sue coordinate

Attributo `position: fixed`

- L'ultimo valore possibile per `position` è `fixed`
- Questo valore indica che il box deve essere posizionato ad una determinata distanza dai margini della pagina stessa (specificare dove sempre tramite `top`, `bottom`, `left` e `right`)
- Questo attributo specifica inoltre che il contenuto del box non scorrerà con il resto della pagina ma resterà fisso nei confronti della finestra

Nascondere box

- Impostando l'attributo CSS `display` a `none`, è possibile specificare che un blocco non deve essere visualizzato
 - Questo si rivela utile nella manipolazione dinamica degli attributi CSS per far apparire o scomparire box a seconda di vari eventi
- Impostando invece `display` a `block`, si impone che l'elemento venga visualizzato normalmente
 - Esiste analogamente il possibile valore `inline` che specifica che un elemento debba venir visualizzato come parte del testo e non come blocco
- La visibilità può essere manipolata anche tramite l'attributo `visibility` che può assumere valori `visible` o `hidden`. In questo caso però il layout della pagina si comporterà come se il box ci fosse anche nei casi in cui questo non è visibile

Esercizio

- Creare tramite l'uso di CSS un sito che realizza un forum
- Per ora, creare delle pagine statiche che facciano da modello per le pagine dinamiche che saranno implementate in seguito
- Progettare in anticipo l'aspetto del sito, ed assicurarsi alla fine che questo abbia l'aspetto desiderato
- Alcune pagine del sito potrebbero essere le seguenti:
 - Una pagina di login
 - Una pagina di navigazione dei vari thread
 - Una pagina di navigazione dei vari post di un thread
 - Pagine per la creazione di un nuovo post o di un nuovo thread

Esercizio

- Nelle varie pagine, inserire vari box annidati a piacere contenenti:
 - Un menu di navigazione del forum (contenente per es. 'lista thread', 'lista post di questo thread', 'logout')
 - Una serie di link esterni (per es. banner pubblicitari)
 - Un box personalizzabile da utente, contenente per esempio link o immagini relative a dei siti da lui indicati
 - Un box dovrà contenere il contenuto vero e proprio della pagina
 - A seconda del tipo di pagina, altri box per raggruppare i vari contenuti (es. i vari thread o post)
- La struttura dei vari box dovrà essere uguale nelle varie pagine ad eccezione di quella di login
- Inserire inoltre nei CSS alcune specifiche per l'aspetto del testo