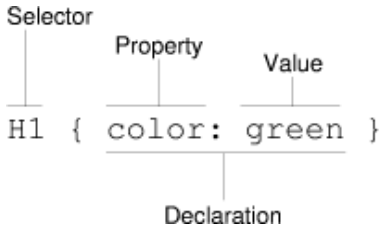
I CASCADING STYLE SHEETS

Attualmente si consiglia di pensare prima ai contenuti e solo in un momento successivo allo **stile di presentazione**. Questa è la filosofia dei fogli di stile che sostituiscono i tag di formattazione (come <FONT FACE> inserito direttamente nel codice) mantenendo un aspetto uniforme in tutte le pagine associate.

In un foglio di stile, ogni istruzione si suddivide in *selettore* (il tag) e *dichiarazione* che suggerisce come presentarlo: le dichiarazioni delle *proprietà* hanno la forma "nameProperty : value".



I CSS si aggiungono alle pagine web in uno dei quattro modi seguenti:

- Con aggiunta "**inline**"
- Con **inclusione** nel documento (CSS **embedded**)
- Con **collegamento** ad un **foglio di stile esterno**
- **Importando** uno o più fogli di stile **esterni**

Nel dettaglio:

- Con aggiunta "**inline**" si inserisce la modifica nel singolo tag mediante **attributo** STYLE)

```
<body style="background-color: #FF0000">
```

- Con **inclusione** nel documento HTML : uso della metaistruzione <STYLE ><!--…--></STYLE> di solito all'interno della sezione HEAD dopo TITLE

```
<head>
      <title>Esempio</title>
      <style type="text/css">
        body {background-color: #FF0000}
      </style>
</head>
```

```
selector {property: value }
```

a quale tag HTMl viene applicata la proprietà (per esempio "body")

Una proprietà potrebbe essere per esempio il colore di sfondo ("background-color")

Rosso potrebbe essere per esempio il valore della proprietà del colore di sfondo ("#FF0000")

- Con **collegamento** ad un **foglio di stile esterno**: uso della metaistruzione <LINK href = "*nome*.css" rel = "stylesheet" type = "text/css"> all'interno della sezione HEAD).

style.css

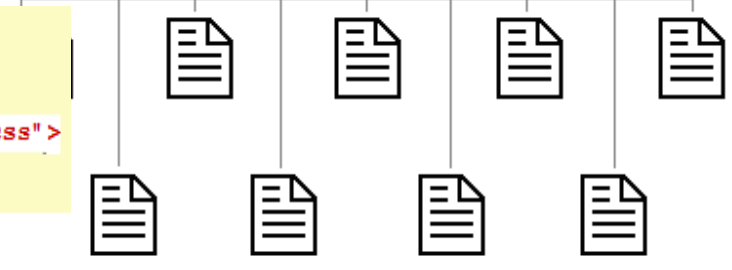
```
body {
  background-color: #FF0000
}
```

cambiamenti nel foglio di stile esterno avranno effetto su tutti i documenti HTML

default.htm

```
<head>
      <title>Il mio documento</title>
      <link rel="stylesheet"
            type="text/css" href="style.css">

</head>
```



documenti HTML che linkano allo stesso foglio di stile

L'elemento **<link>** presenta una serie di attributi:

Attributo	Descrizione
rel	descrive il tipo di relazione tra il documento e il file collegato. È **obbligatorio**. Per i CSS due sono i valori possibili: **stylesheet** e **alternate stylesheet**. Approfondimenti nell'uso di Fogli di stile alternativi
href	serve a definire l'URL assoluto o relativo del foglio di stile. È **obbligatorio**
type	identifica il tipo di dati da collegare. Per i CSS l'unico valore possibile è **text/css**. L'attributo è **obbligatorio**
media	con questo attributo si identifica il supporto (schermo, stampa, etc) cui applicare un particolare foglio di stile. Attributo **opzionale**. L'argomento può essere approfondito

Se si vogliono combinare più CSS separati ed applicarli alla stessa pagina basta assegnare lo stesso nome (TITLE) a tutti gli stili (ad es: <LINK href = "*nome1*.css" rel = "stylesheet" type = "text/css" **title** = "Stili"> <LINK href = "*nome2*.css" rel = "stylesheet" type = "text/css" **title** = "Stili">).

- **Importando** uno o più **fogli di stile esterni** (con uso di @import url(*nome*.css); all'interno di altro foglio di stile o del metatag STYLE in modo da creare una "cascata"). Le dichiarazioni del successivo hanno la precedenza su quelle del primo e lo stesso Style Sheet che importa ha precedenza sugli altri.

Si possono impostare più proprietà tutte applicate allo stesso tag separandole col carattere "**;**"

H1 { color: green; text-align: center }

Possono essere impostati più valori di ogni proprietà, separandoli con una virgola: il precedente ha priorità sul successivo (valori consigliati in alternativa al browser che può non implementarli).

Ad esempio : I { font-family: courier, arial, times }

Si possono raggruppare sia selettori sia dichiarazioni per rendere più compatto il codice, ad esempio:

BLOCKQUOTE, EM, I { color : marron; font-family: courier }

è equivalente a:

```
BLOCKQUOTE { color : marron;  font-family: courier }
EM { color : marron;  font-family: courier }
I { color : marron;  font-family: courier }
```

I selettori possono essere concatenati. Ad esempio H1 STRONG { color : blue } agisce sul testo in ogni tag STRONG all'interno di tag H1.

L'**ordine di precedenza**[1] con cui i browser supportano l'interpretazione dei fogli di stile è diverso cambiando tra una versione e quella più recente con la seguente tendenza :

1. Inline (livello locale)
2. Embedded (inclusi o livello globale)
3. Linked (collegati)
4. Imported (importati)
5. Reader
6. Browser default

[1] In realtà i browser applicano lo stile definito per ultimo (sono nate dunque regole di stile per definire prima a livello più esterno ad esempio @import deve precedere ogni definizione di stile inclusa). Una corretta strategia vuole che i fogli esterni siano **sempre** dichiarati prima di quelli incorporati.

L'elemento *important* serve per forzare l'importanza di una scelta di apparenza nel caso di collisione di stili all'interno dello stesso foglio ad esempio:

```
<HTML>
        <HEAD><TITLE>Quando gli stili collidono</TITLE>
                <STYLE TYPE= "text/css"  MEDIA = screen >
                <! --
                BODY {  color : yellow;
                          font-family: impact  }
                P  {  color : red ! important;
                       font-family: arial ! important  }
                CODE { color : marron;
                          font-family: courier  }
                -->
                </STYLE>
        </HEAD>
        <BODY>
                <P><CODE>Testo </CODE></P>
        </BODY>
</HTML>
```

EFFETTO: il testo sarà scritto in Arial ed in colore rosso; si noti che le istruzioni sono **inserite come commento** per compatibilità con browser che non supportano i CSS. Verificare che la forzatura inline prevale ed il testo appare in corsivo di stile times in verde modificando il body:

```
<P><CODE><I STYLE="color: green; font_family: times">Testo</I> </CODE></P>
```

Il **CSS2** "eredita" il valore **!important** per tutte le proprietà degli stili che governano parte o tutto il documento. Per esempio, lo stile seguente forza tutti gli sfondi in bianco e tutto il testo in nero:

```
/* Configura il colore del testo in nero ed il colore di sfondo del documento in bianco */

body {  color: black ! important ;
          background: white ! important
        }
/* Fa in modo che i valori di "color" e "background" debbano essere ereditati da tutti gli altri elementi,
rafforzandoli con "!important".  Notare che questo può essere annullato da un altro più specifico operatore di
stile. */

        *{color: inherit ! important ;
          background: inherit ! Important
         }

         /* Uso di selettore UNIVERSALE: il simbolo asterisco significa "ogni elemento" */
```

I **CSS2** includono anche queste **caratteristiche di controllo**:

- Colori (**color**, **background-color**, **border-color**, **outline-color**) e caratteri (**font**): l'utente può applicare le sue preferenze al colore e ai font del documento Web .

- I contorni dinamici (la proprietà **outline**) permette gli utenti (ad esempio gli ipovedenti) di creare dei profili attorno al contenuto che non cambiano il layout ma che forniscono informazioni importanti.

Per esempio, per disegnare uno spessa linea nera attorno ad un elemento quando su di esso c'è il focus, ed una spessa linea rossa quando è attivo, si possono usare queste righe:

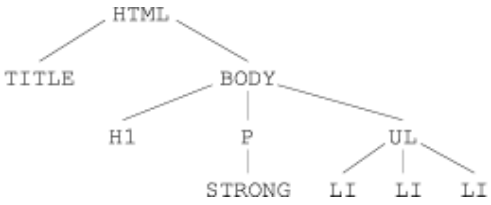
```
:focus { outline: thick solid black }
:active { outline: thick solid red }
```

I **CSS3** introducono novità: dagli effetti visuali ai layout, dai selettori alle media queries e alle animazioni.

Il concetto di inheritance (ereditarietà)

```
<HTML>
<HEAD>
 <TITLE>Bach's home page</TITLE>
 <STYLE TYPE="text/css">
          BODY { color: green }
  </STYLE>
</HEAD>
 <BODY>
  <H1>Bach's home page</H1>
  <P>Johann Sebastian Bach was a <STRONG>prolific</STRONG> composer. Among his  works are:
  <UL>
    <LI>the Goldberg Variations
    <LI>the Brandenburg Concertos
    <LI>the Christmas Oratorio
  </UL>
 </BODY>
</HTML>
```

La struttura di questo documento è:



Una proprietà CSS che setta il valore di un elemento sarà trasferita attraverso l'albero a tutti i discendenti. Nell'esempio se si vuole che il tag H1 sia associato ad altro colore basta aggiungere un istruzione che scritta prima o dopo ha comunque precedenza nel conflitto:

```
<STYLE TYPE="text/css">
 BODY { color: green }
 H1 { color: navy }
</STYLE>
```

NB: Il tag background non ammette inheritance.

Le classi

Le **classi** permettono di impostare apparenza diversa per lo stesso tag o per diversi tag. Un selettore di classe che permette di stabilire più di uno stile per un tag/elemento usa la sintassi punto nel CSS : "**. nomeclasse**" e si applica con l'attributo CLASS = "nomeclasse" nel codice HTML.
Ad esempio:

```
<STYLE TYPE= "text/css">
        p.bluetext {  color : blue }
</STYLE>
```

imposterà il colore blu usando nel documento HTML <p class = "bluetext">

Se si usa invece:

```
<STYLE TYPE= "text/css">
        .bluetext {  color : blue }
</STYLE>
```

imposterà il colore blu sia usando <H3 CLASS= "bluetext">…</H3> sia <EM CLASS= "bluetext"> ..</EM> nel documento HTML.

E' prevista la dichiarazione di **classi multiple**. Ad esempio la regola seguente applicherà gli stili impostati a tutti gli elementi in cui siano presenti (in qualunque ordine) almeno tutti i nomi delle classi definiti nel selettore:

h1**.testorosso.grassetto** {color: red; font-weight: bold;}

L’attributo id

Per limitare lo scope delle informazioni di stile ad una singola istanza di un elemento, si setta l’**attributo “id”** che usa la sintassi cancelletto nel CSS: “**#nomeid**” e si applica con l’attributo ID = “nomeid” nel codice HTML.

```
<HTML> <HEAD> <STYLE type="text/css">
                        #myid {border-width: 1; border: solid; text-align: center}
                </STYLE>
        </HEAD>
        <BODY>  <H1 id="myid"> Questo H1 e’ influenzato dallo style </H1>  </BODY>
</HTML>
```

Un **id** è simile ad una classe, ma con una fondamentale differenza: una classe può essere applicata a più elementi nella stessa pagina, mentre si consiglia di usare un id per riferirsi ad un solo elemento definendone un preciso stile. In pratica, posso avere nello stesso documento: <p class="testo"> e <table class="testo">, ma si consiglia di usare ad esempio <table id="sfondo"> senza più usarlo per altri elementi.

Gli attributi di stile inline hanno precedenza su **id**, **id** sulla class, e class sugli elementi HTML definiti come stylesheet.

***Regole**: Usare un numero minimo di CSS per ogni sito*
Usare i CSS esterni piuttosto di stili interni, ed evitare i CSS in riga (inline).
Se vi sono più CSS, usare lo stesso nome di classe per lo stesso concetto in tutti.

Le pseudo-classi

Il concetto di pseudo-classe ha qualcosa di "filosofico". Una pseudo-classe non definisce infatti un elemento ma un particolare stato di quest'ultimo. Imposta uno stile per un elemento al verificarsi di certe condizioni.

A livello sintattico le pseuso-classi non possono essere mai dichiarate da sole, ma per la loro stessa natura devono sempre appoggiarsi ad un selettore. Il primo costrutto che esaminiamo è quello con un elemento semplice:

```
a:link {color: blue;}
```

La regola vuol dire: i collegamenti ipertestuali (**<A>**) che non siano stati visitati (**:link**) avranno il colore blue. Da qui risulta più chiaro il concetto espresso all'inizio: la pseudo-classe **:link** definisce lo stile (colore blue) solo in una determinata situazione, ovvero quando il link non è stato attivato. Appena ciò dovesse avvenire, il testo non sarà più blue, perché la condizione originaria è venuta meno.

Torniamo alla sintassi. La pseudo-classe (tutte iniziano con i **due punti**) segue senza spazi l'elemento. Subito dopo si crea nel modo consueto il blocco delle dichiarazioni.

Una pseudo-classe può anche essere associata a selettori di tipo classe. I costrutti possibili sono due. Il primo è quello sancito nella specifica **CSS1**. La pseudo-classe doveva seguire la dichiarazione della classe:

```
a.collegamento:link {color: green;}
```

Questa regola fissa il testo verde (green) solo per i link non visitati che abbiano come attributo **class="collegamento"**. Sarà verde questo collegamento:

```
<a href="pagina.htm" class="collegamento">
```

ma non questo:

```
<a href="pagina2.htm">
```

A partire dalla specifica **CSS2** è consentita anche questa sintassi:

a:link.collegamento

in cui la classe segue la pseudo-classe. Significato e risultati sono comunque identici al primo esempio. Il primo tipo di sintassi garantisce una maggiore compatibilità con i browser più datati. Gli esempi e la sintassi presentati valgono per tutte le pseudo-classi.

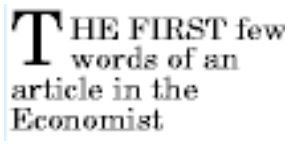
Pseudoelementi: **:first-letter, :first-line, :before, :after**

nb: dalla **CSS3** si introduce la notazione :: per distinguere pseudoelementi da pseudoclassi
ad esempio con p::first-line { text-transform: uppercase } si cambiano le lettere della prima linea di ogni paragrafo in maiuscolo

L'effetto in figura può essere realizzato col segmento

```
<P><SPAN><P::first-letter>T</P::first-letter>he first</SPAN>
few words of an article in the Economist.
</P>
```

con scelta di stile SPAN { text-transform: uppercase }



Come esempio di **pseudo-classe** si usa **:link** per impostare diversi **effetti sui Link**:

- **Link senza sottolineatura:**
```
<style type=text/css>
 a:link, a:visited { text-decoration: none}
</style>
```
- **Link sottolineato solo al passaggio del mouse:**
```
<style type=text/css>
 a:link, a:visited { text-decoration: none}
 a:hover { text-decoration: underline }
</style>
```
- **Link che cambia colore al passaggio del mouse:**
```
<style type=text/css>
 a:link, a:visited {color: #00ff00}
 a:hover {color: #ff0000 }
</style>
```
- **Link che s'ingrandisce al passaggio del mouse:**
```
<style type=text/css>
 a:link { font-size: 15px }
 a:visited { font-size: 15px }
 a:hover { font-size: 18px }
</style>
```

Si applica solo all'elemento (X)HTML **<A>** che abbia anche l'attributo **href**. Quindi, non alle cosiddette ancore invisibili ma solo ai link ipertestuali. Definisce lo stile per questo elemento quando il collegamento punta ad un sito o ad una pagina non ancora visitati.

:active quando si clicca su un collegamento ipertestuale
:visited quando si ritorna dal collegamento ipertestuale andato a buon fine
:hover Azione al passaggio del mouse
:focus Su **Explorer Win non noterete nulla,** ma è attivo quando il campo di testo riceve il **focus**
:lang stile attivo per data lingua

CSS per specificare il posizionamento

Oltre a rimpiazzare i tag di formattazione, gli Style Sheet si propongono come **sostituti di tabelle e gif trasparenti** per specificare il posizionamento degli elementi della pagina HTML. E' possibile specificare che ogni oggetto sia contenuto in un rettangolo (*bounding box*) del quale si possono indicare posizione (sia come distanza dai bordi della finestra sia relativamente ad altri oggetti), dimensione, sovrapporre ad altri oggetti e rendere invisibile.

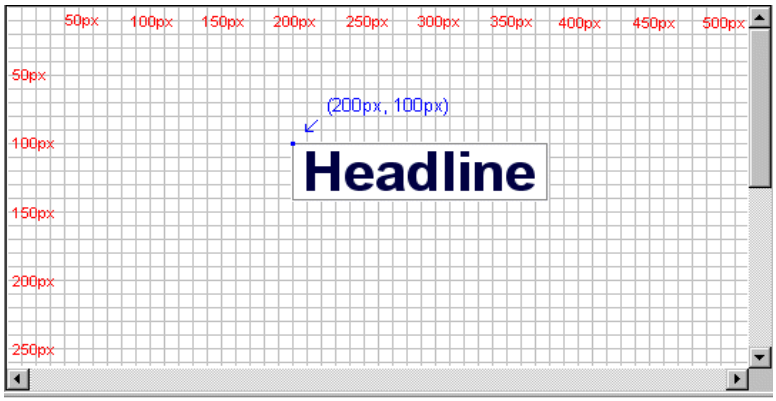
La posizione di un elemento si specifica nello style sheet con la *proprietà **position***, ad esempio si usa :

UL { position : absolute; top: 100px; left: 50px; width: 400px}

per ottenere tutte le liste larghe 400 pixel, a 50 pixel dal bordo sinistro e 100 da quello superiore della finestra del browser.

Per posizionare l'angolo in alto a sinistra di un'intestazione a 100 pixel di distanza dal bordo superiore e a 200 pixel di distanza dal bordo sinistro

```
h1 {
        position:absolute;
        top: 100px;
        left: 200px;
}
```



Per ottenere il posizionamento separando il contenuto nella pagina si può ricorrere ai **tag DIV** e agli **stili inline**:
<DIV STYLE= "position : absolute; top: 150px; left: 150px; background-color: blue">Questo testo e' contenuto in rettangolo blu a 150 pixel dai bordi superiore e sinistro della finestra del browser.
</DIV>
Esempio che posiziona ai quattro angoli del documento box con sfondo colorato (usando id e CSS esterno)

Float per avvolgere altro elemento

Il floating è un'operazione che veniva fatta in HTML sulle immagini. Bastava usare nel tag IMG l'attributo align e impostare come valore left, right, middle, etc.

Sintassi: <selettore> {float: <valore>}

Valori **left.** L'elemento viene spostato sul lato sinistro del box contenitore, il contenuto scorre a destra.
right. L'elemento viene spostato sul lato destro, il contenuto scorre a sinistra.
none. Valore iniziale e di default in mancanza di una dichiarazione esplicita. L'elemento mantiene la sua posizione normale.

Se usate il float con le immagini non avete problemi perché esse hanno una dimensione intrinseca che il browser riconosce. Ma se lo applicate ad altri elementi dovete esplicitamente impostare una dimensione orizzontale con la proprietà width.
Esempi

```
div {width: 200px; float:right;} // obbligatoria dimensione orizzontale
img {float: left;}
```

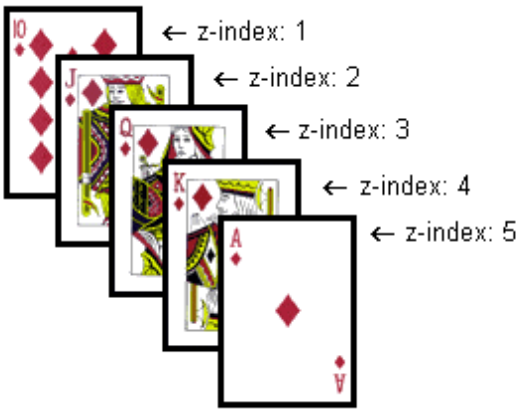
Vedi anche **clear** ed esempi o altri (float, clear … display)

Per ottenere **effetti di sovrapposizione**[2] tra elementi si usa la *proprietà* ***z-index***, che specifica il livello in cui si trova l'elemento cui e' applicata (più il valore e' alto, più vicino all'osservatore e' l'oggetto) ad esempio con:

.sotto{ position : absolute; top: 100px; left: 20px; z-index: 1; background-color: yellow}
.sopra { position : absolute; top: 115px; left: 40px; z-index: 2; background-color: red}

si potranno impostare livelli <P CLASS= sotto> e <P CLASS=sopra>.

Esempio con CSS esterno



Per effetti di sovrapposizione e **per effetti nascosti** (rettangoli in IE erano zone in Netscape) esempi

Le proprietà singole sono la maggior parte: impostano per un dato elemento o selettore un singolo aspetto.

Con le **shorthand properties** (*proprietà abbreviate*), è possibile invece definire con una sola dichiarazione un insieme di proprietà.

Chiariamo con un esempio.

Ogni elemento presenta sui suoi quattro lati un certo margine rispetto a quelli adiacenti. È possibile definire per ciascuno di essi un valore usando quattro proprietà singole separate:

- **margin-top**
- **margin-right**
- **margin-bottom**
- **margin-left**

La regola sarebbe questa:

```
div {
        margin-top: 10px;
        margin-right: 5px;
        margin-bottom: 10px;
        margin-left: 5px;
}
```

Lo stesso risultato si può ottenere usando la *proprietà a sintassi abbreviata* **margin**:

```
div {margin: 10px 5px 10px 5px;}
```

[2] Esempi tratti dal tutorial http://it.html.net/tutorials/css/

Esercizio[3]**:**

Ricerca le proprietà (usi e costrutti sintattici di ciascuna) dell'elenco:
background | border | border-top | border-right | border-bottom | border-left | cursor | font | list-style | margin | padding.

Scaricabile test.rar con esempi d'uso di proprietà: background-attachment con valori fixed o scroll, background-color, background-image, background-position, background-repeat, color, cursor; proprietà singole ed abbreviate per bordi, margini, padding, liste e posizione, chiarimenti sulla struttura di box, display, ereditarietà, float e clear, overflow per compensare superamento dei limiti delle dimensioni, diversi supporti (at_media.html), pseudoclassi e pseudoelementi ….

nb:
gli esempi d'uso di proprietà CSS, memorizzati in forma compressa sono anche consultabili con navigazione nella guida on-line "CSS di base" di HTML.it

Ad esempio, all'indirizzo http://html.it/guide/esempi/guida_html/esempi/body/08.html si usa attributo di style *inline* per gestire lo sfondo; all'indirizzo http://www.html.it/guide/esempi/css/test/zindex.html si può vedere l'effetto nell'uso della proprietà z-index.

Usando CSS embedded o esterni si può verificare l'attenzione a ridurre il "peso" delle immagini ad esempio dalla pagina http://html.it/articoli/esempi/articoli_pro/495/gradient.html

Aspetto diverso a seconda del "supporto"

L'attributo **media** può accompagnare sia l'elemento <LINK> che l'elemento <STYLE>. Ecco due esempi di sintassi:

```
<link rel="stylesheet" type="text/css" media="print, tv, aural" href="print.css" >
<style type="text/css" media="screen">...</style>
```

Una versione per il video e una per la carta (http://www.fucinaweb.com/design/stampacss.asp)
Esempio che usa la creazione i due css: **video.css** e **printer.css.** L'unica differenza tra i due css è la presenza della regola **display:none** per la versione di stampa. La regola display posta a none nasconde i tag che la usano. Si crea dunque una diversa classe .screen {display: none} e la si associa a tutte le parti del documento Html che non si vogliono stampare.

Uso di **id** per diversificare la visualizzazione su dispositivi diversi

(http://professoressa.altervista.org/style/esempio01.html e http://css.html.it/guide/lezione/89/ridefinire-il-foglio-distile-per-lo-schermo/)

NB: è possibile definire all'interno della direttiva @import anche il supporto cui applicare il CSS, in modo simile a quanto abbiamo visto a proposito dell'attributo **media**. Per fare ciò basta far seguire all'url del CSS l'indicazione di uno dei media previsti nella specifica:

```
<style type="text/css">
@import url(foglio_stampa.css) print, tv, aural;
@import url(foglio_schermo.css) screen;
</style>
```

Le precedenti regole hanno lo stesso effetto dell'uso della notazione @media ma sono preferiti nei download. Anche in questo modo, invece di costruire due fogli di stile, si creano in un unico CSS esterno queste regole per diversificare l'aspetto dell'elemento:

```
@media print,tv, aural {
  h1 {color: black;}
  }
@media screen {
h1 {color: red;}
}
```

Per approfondimenti http://www.webaccessibile.org/argomenti/documento.asp?DocID=133 (suoni)
Un altro modo per impostare stili alternativi è ricorrere all'uso di JavaScript.

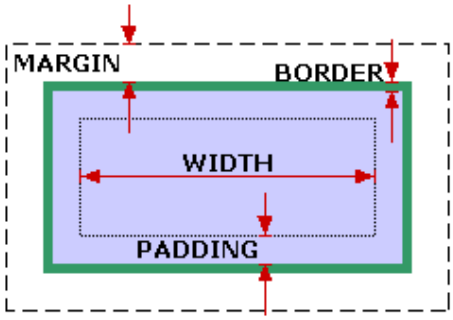
[3] Reference anche http://www.morpheusweb.it/html/manuali/css.asp con esempi

Il box model dei CSS

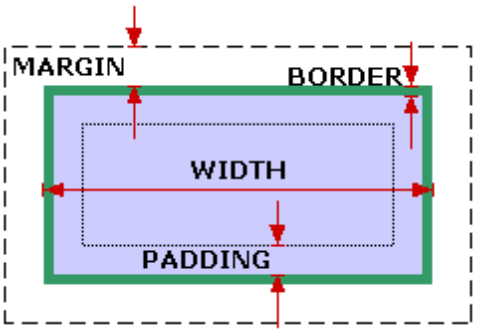
Il box model[4] dei CSS permette di definire dei blocchi rettangolari con specifici valori per la larghezza e l'altezza della sezione contenuti, del padding, dei bordi, dei margini. Purtroppo, IE5 per Windows non ha una interpretazione corretta del box model. E' molto importante conoscere il problema, essendo il box model uno dei migliori strumenti offerti dai CSS ed essendo IE5 per Windows ancora il browser più diffuso.

Il box model viene specificato impostando la larghezza e l'altezza dei contenuti, il padding, dimensioni e stile del bordo e il margine. Il padding è lo spazio fra i contenuti e il bordo, mentre il margine è lo spazio fra il bordo e gli altri contenuti della pagina.

Quando, con la proprietà '**width**', si specifica la larghezza del box, si specifica la larghezza dell'area che conterrà il testo del box. La larghezza del box fino al bordo è la somma della larghezza specificata con **width + ampiezza del padding + spessore del bordo** (vedi immagine successiva).



Questo secondo gli **standard CSS**



IE5 per Windows (non IE5 per Mac) considera invece la larghezza specificata con width, come la larghezza del box fino al bordo. Spessore del bordo e padding vengono scalati dalla larghezza specificata

nb: con i CSS3 si possono realizzare in modo facile bordi arrotondati senza ricorrere ad immagini. Si vedano esempi e compatibilità

Bordi					
border-radius	9.0+	4.0+	5.0+	4.0+	10.5+

[4] Vedi anche http://css.html.it/guide/lezione/28/il-box-model/

Link utili

Word Wide Web Consortium http://www.w3.org/ in particolare la sezione Style Sheets (W3C)
o la versione in italiano **Fogli di stile**

o un semplice Tutorial http://it.html.net/tutorials/css/ (**italiano**)
con link d'esempio per validare (usando il servizio del W3C)

Errori nella costruzione di un sito: importante è fare tutto il contrario rispetto all'autore del **sito più brutto del mondo**... **Il sito internet più brutto del mondo**

AskTog http://www.asktog.com/menus/designMenu.html con l'illustrazione degli ***errori più comuni*** nel progetto di web design "Top 10 Mistakes of Web Design" (inglese)

In lingua italiana:

http://web-link.it/css/css.htm (guida semplice)
http://www.html.it/css/ o http://www.risorse.net/css/ o http://www.fauser.edu/fau/css/tuttocss.htm
http://www.usabile.it/cssdesign.htm usabilità ed accessibilità
http://lau.csi.it/risorse/CSS2/index.shtml

con link di approfondimento:

Per approfondire l'argomento CSS:

W3-School: a scuola di CSS direttamente dal W3C con esempi guidati (idea *CSS Navigation Bar*)

http://www.w3.org/Style/CSS/current-work (Raccomandazioni del W3C)

W3C-CSS2: Raccomandazioni del W3C sui CSS2 vs CSS3

Web Page Design for Designers: Joe Gillespie

Con Stile: risorsa italiana ricca di tutorial ed esempi originali

Uso di CSS per introdurre **creatività** http://www.meyerweb.com/eric/css/edge/index.html
in particolare esempi d'uso effetto float (Eric meyer) :
http://www.meyerweb.com/eric/css/edge/gallery.html
con reference CSS2 in http://www.meyerweb.com/eric/css/references/css2ref.html

CSS per emulare immagini mappate
http://stclassi.altervista.org/HTML_CSS/Emulare_le_immagini_mappate_con_i_CSS.pdf

CSS per modificare barre a scorrimento
http://stclassi.altervista.org/HTML_CSS/MODIFICARE_LA_BARRA_DI_SCORRIMENTO_CON_I_CSS.pdf

Uso di JavaScript per stili alternativi

```
<html>
<head>
        <title>Stili alternativi</title>
        <script type="text/javascript" src="styleswitcher.js"></script>
        <link rel="stylesheet" type="text/css"  title="principale" href="screen.css" />
        <link rel="alternate stylesheet" type="text/css"  title="alternativo" href="print.css" />
</head>
<body>
<h1>Stili alternativi</h1>
<p>A questo documento abbiamo applicato due fogli di stile, uno principale e uno
  alternativo. Questo il codice HTML: </p>
<p>&lt;link rel=&quot;stylesheet&quot; type=&quot;text/css&quot; title=&quot;principale&quot;
  href=&quot;screen.css&quot; /&gt;<br />  &lt;link rel=&quot;alternate stylesheet&quot;
type=&quot;text/css&quot; title=&quot;alternativo&quot;   href=&quot;print.css&quot; /&gt;<br />
</p>
<p>Al documento abbiamo collegato uno script tramite il quale possibile cambiare
  il foglio di stile. Verificate cliccando sul link qui sotto.</p>
<p><a onclick="setActiveStyleSheet('alternativo'); return false;" href="#">Alternativo</a></p>
</body>
</html>
```

Con file "**styleswitcher.js**":

```
function setActiveStyleSheet(title) {
  var i, a, main;
  for(i=0; (a = document.getElementsByTagName("link")[i]); i++)
   {       if(a.getAttribute("rel").indexOf("style") != -1 && a.getAttribute("title"))
                {  a.disabled = true;
                    if(a.getAttribute("title") == title) a.disabled = false;
                 }
     }
   }
function getActiveStyleSheet() {
  var i, a;
  for(i=0; (a = document.getElementsByTagName("link")[i]); i++)
   {    if(a.getAttribute("rel").indexOf("style") != -1 && a.getAttribute("title") && !a.disabled)
              return a.getAttribute("title");
     }
  return null;
}
```

Con file "**print.css**":

```
body {            font-family: "Times New Roman", serif; font-size: 12pt;
background: White;}
                        H1 {background: White; color: Black;}
```

Con file "**screen.css**":

```
body { font-family: "Verdana",  sans-serif; font-size: 12px; background:
White;}
                        H1 {background: Black; color: Red;}
```