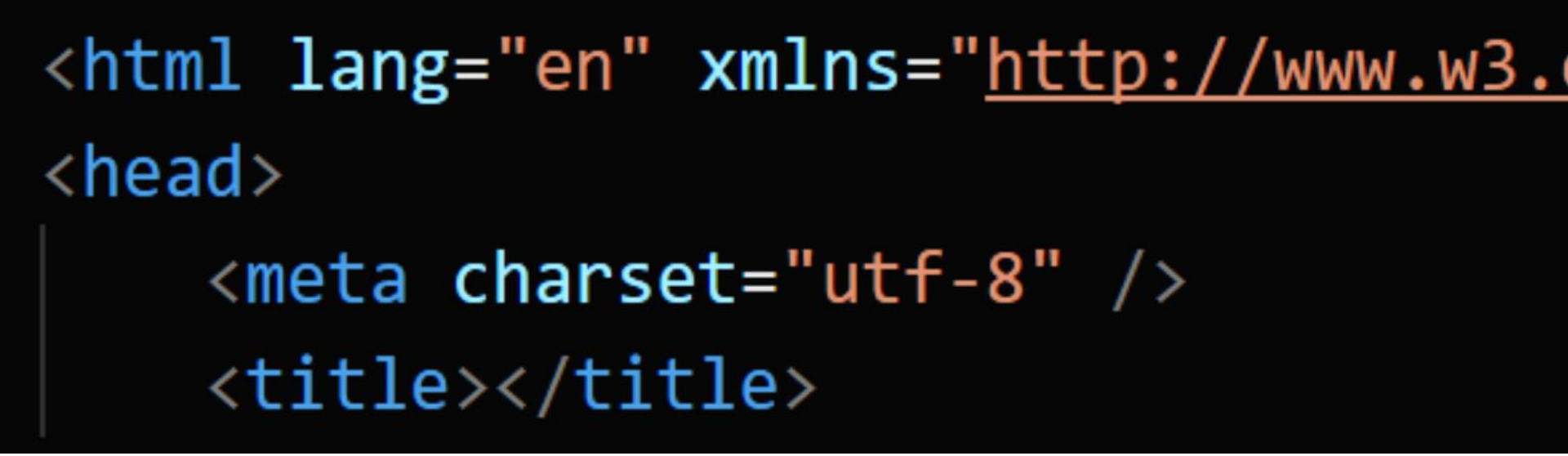


Web Coding

Prototipazione di ipertesti e siti web in HTML
ed introduzione alla creazione di stili grafici con fogli di stile CSS.

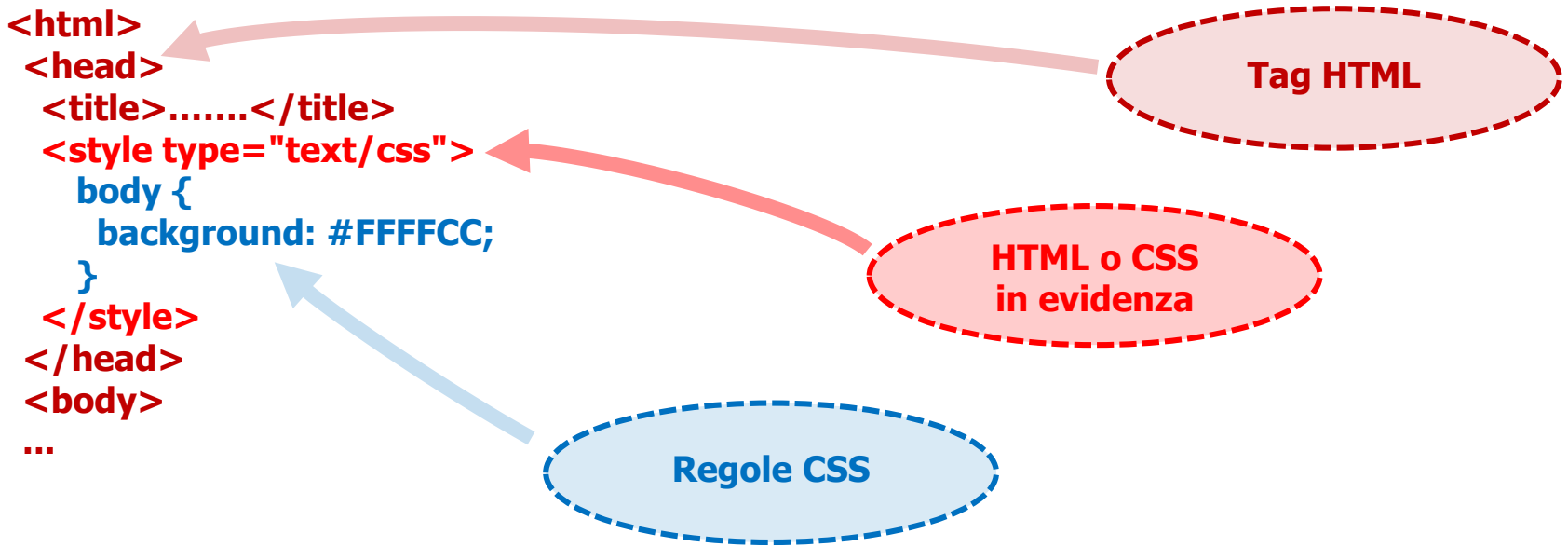
Unità Didattica UD03: Introduzione ai fogli di stile CSS: box model, colori e sfondi

prof. Giovanni Borga

Convenzioni grafiche utilizzate in questa e le successive presentazioni

Distinzione tra HTML e CSS nelle slides

Per facilitare la distinzione tra i due diversi linguaggi, in queste slides il codice viene riportato in 3 colorazioni diverse: rosso, rosso scuro, blu, secondo lo schema sotto riportato



Introduzione ai fogli di stile

Cascading Stile Sheet (CSS) – logica e sintassi

Cascading Style Sheet - CSS

Dietro il semplice acronimo CSS (Cascading Style Sheets - Fogli di stile a cascata) si nasconde **uno dei fondamentali linguaggi standard del W3C**.

E' l'ideale complemento dell'HTML. Infatti, nelle intenzioni del W3C Consortium, l'HTML, così come la sua evoluzione, XHTML, deve essere visto semplicemente come un linguaggio strutturale privo di qualsiasi elemento riguardante la presentazione dei contenuti. Per questo obiettivo, ovvero definire l'aspetto visuale ed grafico di una pagina, lo strumento designato sono appunto i CSS.

Questo approccio è noto anche mediante l'espressione: **separare il contenuto dalla formattazione**.

Le versioni del linguaggio:

- **CSS1**: dicembre 1996.
- **CSS2**: maggio 1998 (non porta grandi stravolgimenti, ma molte aggiunte rispetto alla 1)
- **CSS3**: pubblicato come raccomandazione ma già ampiamente supportato

Vantaggi dell'uso dei fogli di stile

Utilizzando i CSS si capisce ben presto che le possibilità offerte da questo linguaggio sono numerosissime.

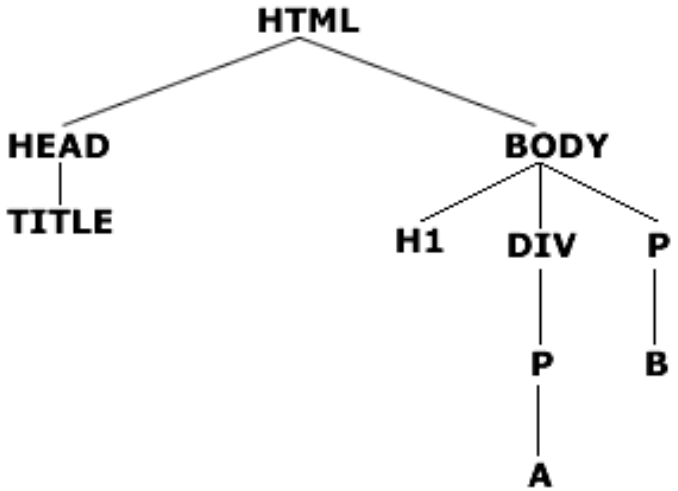
Sinteticamente vediamo i principali **punti di forza**:

- Quantità di **opzioni grafiche** e di layout estremamente più ampia di quanto offerto dall'HTML
- Gestione efficiente e versatile del **DOM** (Document Object Model)
- **Ottimizzazione del codice** (si definisce uno stile e lo si applica a tutti gli elementi – le modifiche possono poi essere effettuate solo sulle definizioni)
- **Interazione con lo scripting**

Struttura del documento ed ereditarietà

Osserviamo il codice a sinistra; la sua rappresentazione gerarchica può essere l'immagine a destra:

```
<html>
 <head>
  <title>Struttura del documento</title>
 </head>
 <body>
  <h1>Titolo</h1>
  <div>
   <p>
    Primo <a href="pagina.htm">paragrafo</a>
   </p>
  </div>
  <p>Secondo<b>paragrafo</b></p>
 </body>
</html>
```

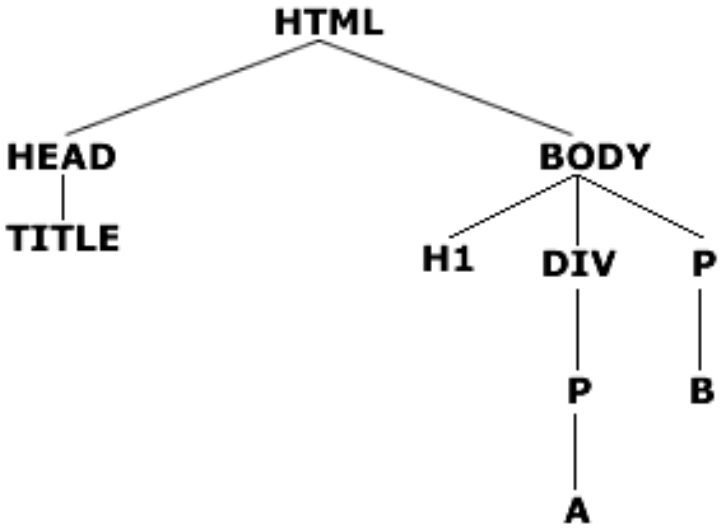


Gerarchia degli elementi di un documento

Il documento è in effetti una perfetta forma di gerarchia ordinata in cui tutti gli elementi hanno tra di loro una relazione del tipo **genitore-figlio.**

PARENT-CHILD in inglese; dicitura importante perchè utilizzata nei linguaggi come DOM o Javascript: **ogni elemento è genitore e/o figlio di un altro**.

- Un elemento si dice genitore (parent) quando contiene altri elementi.
- Si dice figlio (child) quando è racchiuso in un altro elemento.



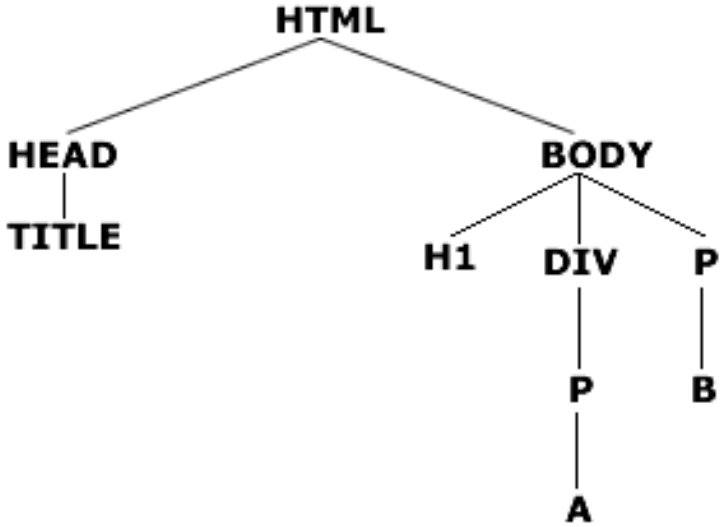
In base a queste indicazioni il nostro documento ha le seguenti caratteristiche:

1. *BODY è figlio di HTML,*
2. *BODY è anche genitore di H1, DIV e P.*
3. *P è a sua volta genitore di un elemento B.*

Gerarchia ed ereditarietà

Anche BODY è genitore di B? Non proprio.

Serve un'altra distinzione, quella tra **antenato** (*ancestor*) e **discendente** (*descendant*).



La relazione parent-child è valida solo se tra un elemento e l'altro si scende di **UN livello**. (Esattamente come in un albero familiare si indica la relazione tra padre e figlio).

- HEAD è figlio di HTML
- A è figlio di P
- … ecc.

Tra DIV e A, invece si scende di due livelli: diciamo allora che DIV è un antenato di A e che questo è rispetto al primo un discendente.

C'è un solo elemento che racchiude tutti e non è racchiuso: HTML. Si dice infatti che HTML è l'elemento radice (root).

NB: attenzione a non fraintendere: l'elemento radice di un documento (X)HTML non è BODY. HTML inoltre, non è una semplice dichiarazione ma a tutti gli effetti un TAG alla stregua di tutti gli altri.

Fogli di stile interni ed esterni

Abbiamo diversi modi per inserire i fogli di stile CSS in un documento.
Per capire il meccanismo è necessario chiarire la fondamentale distinzione tra **fogli esterni** e **interni**.

E' esterno un foglio di stile definito in **un file separato** dal documento

Come per gli HTML, si tratta di semplici documenti di testo ai quali si assegna **l'estensione .css**.

Un foglio di stile **è invece interno** quando il suo **codice è compreso in quello del documento**

A seconda che si lavori con un CSS esterno o interno **ci sono alcune variazioni nella sintassi e nella modalità di inserimento**.

I fogli interni sono di 2 tipi per cui in totale abbiamo TRE DIVERSE TIPOLOGIE di fogli di stile:

a) **Collegati** (esterni)
b) **Incorporati** (interni)
c) **In linea** (interni)

CSS collegati

Per collegare un foglio esterno ad un documento si utilizza il tag <LINK>.
La dichiarazione va sempre collocata all'interno della sezione <HEAD> del documento HTML:

```
<html>
  <head>
   <title>Inserire i fogli di stile in un documento</title>
   <link rel="stylesheet" type="text/css" href="stile.css" />
  </head>
<body>
...
```

Attributi del tag <link>:

- **rel**: descrive il tipo di relazione tra il documento e il file collegato. (*per i CSS due sono i valori possibili: «stylesheet» e «alternate stylesheet»*).
- **href**: indica l'URL del foglio di stile.
- **type**: indica il tipo di dati da collegare. (*Per i CSS l'unico valore possibile è text/css*).
- **media**: indica il supporto (schermo, stampa, etc) cui applicare un particolare foglio di stile.
  L'attributo è opzionale.

CSS incorporati

I fogli incorporati sono quelli inseriti direttamente nel documento (X)HTML tramite l'elemento <STYLE>. Anche in questo caso la dichiarazione va posta all'interno della sezione <HEAD>:

```
<html>
 <head>
  <title>Inserire i fogli di stile in un documento</title>
  <style type="text/css">
    body {
     background: #FFFFCC;
    }
  </style>
 </head>
 <body>
 ...
```

(in rosso il tag, in blu le regole di stile)

Attributi del tag <style>:

- **type** (obbligatorio)
- **media** (opzionale)

(Valgono le indicazioni viste in precedenza)

Dopo l'apertura di <style> si scrivono le regole del CSS seguite dalla normale chiusura </style>.

CSS in linea

L'ultimo modo per formattare un elemento con un foglio di stile consiste nell'uso dell'**attributo style**.

Esso fa parte della collezione di attributi (X)HTML definita «**Common attributes**», ovvero la famiglia di attributi applicabili a tutti i tag HTML.

La dichiarazione avviene a livello dei singoli tag contenuti nella pagina e per questo si parla di fogli di stile in linea (inline).

La sintassi generica è la seguente: **<tag style="regola_di_stile1; regola_di_stile2; ...">**

Se, ad esempio, si vuole formattare un titolo H1 in modo che abbia il testo di colore rosso e lo sfondo nero, scriveremo:

<h1 style="color: red; background: black;">...</h1>

NB: Come valore di style si possono dichiarare **più regole di stile**. Esse vanno separate dal punto e virgola. I due punti si usano invece per introdurre il valore della proprietà da impostare.

Note sul collegamento/incorporamento dei CSS

Quando usare l'una o l'altra soluzione?

i risultati nella formattazione del documento non cambiano. La giusta soluzione sarà quindi quella richiesta dalla nostra applicazione.

Occorre dunque **progettare**, pensare in anticipo a quella che dovrà essere la struttura delle pagine del sito.

Le fasi da seguire per la progettazione possono essere le seguenti:

1. Si costruisce un **foglio di stile generico ed esterno**, da applicare a tutte le pagine del sito. Esso conterrà le regole per formattare gli elementi o le sezioni presenti in tutte le pagine.
2. Si analizzano sezioni ed elementi presenti solo in certe pagine o che vogliate modificare solo in determinati casi. (ad esempio, se occorre cambiare in rosso il colore di un titolo iniziale solo in una pagina delle decine che formano il sito).
   In questo caso si usa uno **stile incorporato solo in quella pagina**.
3. Infine se in particolari punti ci sono da introdurre formattazioni «spot» ed eccezioni si utilizza la tecnica dei **fogli di stile inline**.

L'attributo MEDIA

Abbiamo accennato all'attributo MEDIA (opzionale). Grazie ad esso è possibile **impostare un foglio di stile per ogni supporto** su cui la nostra pagina verrà visualizzata.

Uno degli utilizzi più diffusi è quello di differenziare gli stili di visualizzazione da quelli di stampa.

L'attributo media può essere applicato sia a <link> sia a <style>:

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

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

E' possibile usare più di un valore, separando i nomi della lista con una virgola:

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

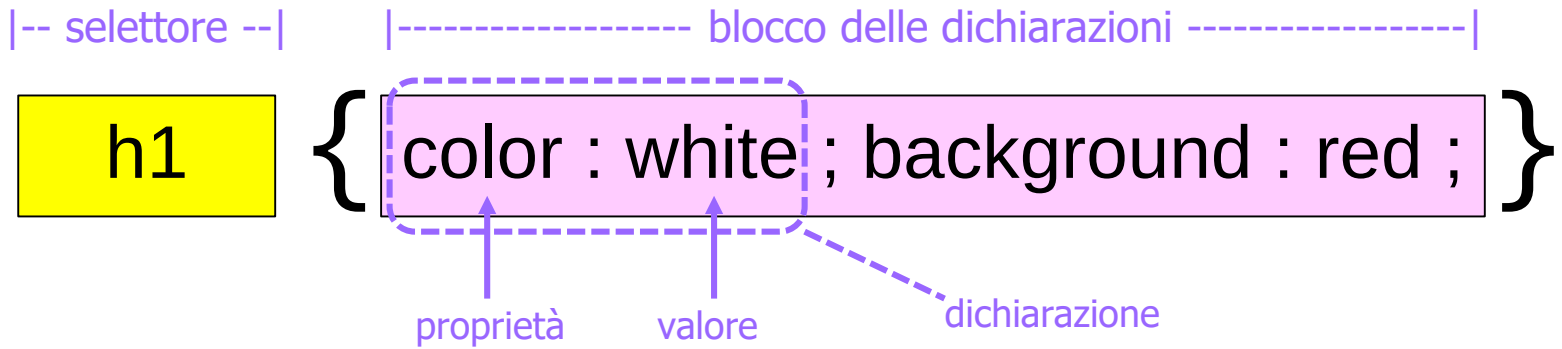
Valori validi per MEDIA:

- **all** (default). *Il CSS si applica a tutti i dispositivi di visualizzazione*.
- **screen**. *Stili per i normali browser web.*
- **print**. *Stili per le stampe del documento*.
- **projection**. *Stili per la presentazione / proiezione del documento.*
- **aural**. *Stili per dispositivi a sintesi vocale*.
- **braille**. *Stili per supporti braille*.
- **embossed**. *Stili per stampanti braille*.
- **handheld**. *Stili per dispositivi mobili*.
- **tty**. *Stili per dispositivi a carattere fisso*.
- **tv**. *Stili per le Web-tv*.

Sintassi dei fogli di stile

Struttura di una REGOLA

Questa è la tipica struttura di una **REGOLA** CSS.
Una regola esprime come deve essere l'aspetto di un elemento della pagina HTML.



Il SELETTORE indica **quale elemento** deve essere renderizzato con la regola.
Il BLOCCO delle dichiarazioni indica **come** renderizzare l'elemento. Il blocco può contenere una o più dichiarazioni composte da proprietà dell'elemento e relativo valore.

E' importante imparare fin da subito il ruolo dei TRE DELIMITATORI: la parentesi graffa, i due punti e il punto e virgola.

Blocchi delle dichiarazioni

Il blocco delle dichiarazioni è **delimitato rispetto al selettore da due parentesi graffe**.
Al suo interno possono trovare posto più dichiarazioni che sono sostanzialmente delle coppie:

- **proprietà**
- **valore**

La proprietà definisce un aspetto dell'elemento (margini, colore di sfondo, etc) mentre il valore definisce il carattere di quell'aspetto.
Proprietà e valore devono essere separati dai due punti.

Valgono inoltre le seguenti regole sintattiche:

1. Non è possibile indicare più di una proprietà con riferimento allo stesso valore,
2. E' ammesso invece, per determinati casi, specificare più valori per la stessa proprietà:

~~**body {color background: black;}**~~

p {font: 12px Verdana;}

Più dichiarazioni vanno separate dal punto e virgola. (il punto e virgola dopo l'ultima dichiarazione è facoltativo. Alcuni browser meno recenti lo richiedono quindi è in generale consigliato metterla).

Gli spazi lasciati all'interno di una regola non influiscono sul risultato.

Commenti

Per inserire parti di commento in un CSS si utilizzano i seguenti segni:

/*	come segno di apertura
*/	come segno di chiusura

Questa sintassi vale sia per commenti di una sola riga sia per più righe di commento. Ecco un esempio:

```
<style type="text/css">
  /* questo è lo stile per il body */
  body {
    background: #FFFFCC;
  }
</style>
```

NB: nei fogli di stile incorporati questa sintassi è ammessa solo all'interno del tag <style>, al di fuori si deve utilizzare la sintassi dei commenti HTML.

Proprietà singole e a sintassi abbreviata

Nelle definizione delle regole è possibile, per alcune proprietà, utilizzare il riferimento ai **singoli valori** oppure utilizzare una **sintassi abbreviata**. Vediamo ad esempio la proprietà margin; la sintassi a proprietà singole è la seguente:

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

Ma è anche possibile quest'altra sintassi (attenzione la sequenza: top-right-bottom-left) :

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

Le proprietà per le quali è possibile utilizzare la sintassi abbreviata sono le seguenti:
background | border | border-top | border-right | border-bottom | border-left | cue | font | list-style | | margin | outline | padding | pause |

Selettori di elementi (type selector)

Fondamentalmente una regola CSS viene applicata ad un selettore. Il selettore è una semplice dichiarazione che serve a **selezionare la parte** o le parti di un documento soggette ad una specifica regola.

Il più semplice dei selettori è il **TYPE SELECTOR**. E' una dichiarazione che **coincide con i nomi dei tag HTML**. Vediamo un esempio:

```
h1 {color: #000000;}
p {background: white; font: 12px Verdana, arial, sans-serif;}
table {width: 200px;}
```

È possibile nei CSS **raggruppare diversi elementi** al fine di semplificare il codice. Gli elementi raggruppati vanno separati da una **virgola**.

```
h1, h2, h3 {background: white;}
```

Questa regola attribuisce il colore bianco ai primi tre livelli di heading.

Selettori speciali (classi e ID)

Esistono due attributi fondamentali applicabili a tutti gli elementi HTML: **CLASS** e **ID**.

Dichiarare questi attributi a prescindere dai CSS non ha alcun senso e non modifica in alcun modo la presentazione della pagina. Dobbiamo infatti definire i corrispondenti selettori in un foglio di stile.

I valori per CLASS e ID sono etichette che possiamo liberamente definire. Vediamo un esempio chiarificatore; un paragrafo caratterizzato da un attributo CLASS valorizzato con «testorosso»:

```
<p class="testorosso">....</p>
```

Nell'HEAD del documento potremmo trovare un CSS incorporato con il seguente codice:

```
<style type="text/css">
  .testorosso {
      font: 12px arial, Helvetica, sans-serif;
      color: #FF0000;
  }
</style>
```

In tutto il documento i paragrafi avranno il testo di colore rosso.

ID ha lo stesso comportamento di CLASS; la scelta tra i due dipende dalla logica del documento.

Il selettore speciale CLASS

Per definire una classe si usa far precedere il nome da un semplice **punto**:

.nome_della_classe

Questa è la sintassi di base. Un selettore classe così definito può essere applicato a tutti gli elementi di un documento HTML.

Il nome della classe può essere combinato con un type selector; ad esempio:

```
p.testorosso {color: red;}
```

lo stile verrà applicato solo ai paragrafi che presentino l'attributo class="testorosso".

E' importante stabilire delle strategie.
Questo secondo tipo di sintassi combinata va usata solo se si pensa di applicare una classe ad uno specifico tipo di elemento (solo paragrafi o solo div, e così via).
Se invece si intende applicare la classe a tipi diversi va utilizzata la sintassi base.

Si possono **attribuire più classi allo stesso elemento** utilizzando lo spazio come separatore:

```
<p class="testorosso testobold testoitalic">....</p>
```

Il selettore speciale ID

La sintassi di un selettore ID è semplice come la precedente: è sufficiente far precedere il nome dal simbolo di cancelletto **#**:

#nome_id

Il markup HTML corrispondente sarà: ** .... **

Con questa regola usiamo il selettore ne modo classico:

#titolo {color: blue;}

assegniamo il colore blue al tag che possiede l'attributo id="titolo".

Come per le classi è possibile usare una sintassi combinata:

p#nome_id {color: blue;}

In realtà il selettore ID è generalmente utilizato per etichettare ed individuare in modo univoco un elemento all'interno di un documento. Di conseguenza la sintassi combinata, seppur corretta, non viene di fatto mai applicata.

NB: per lo stesso motivo **NON è ammessa l'attribuzione di più ID allo stesso elemento**!

Valori e unità di misura

Regole di base per i valori e le unità di misura

1. I **valori** di una proprietà **non vanno mai messi tra virgolette**.
   Uniche eccezioni i valori espressi da stringhe di testo e i nomi dei font formati da più di una parola (esempio: "Times New Roman").

2. Quando si usano **valori numerici con unità di misura**, **non bisogna lasciare spazio tra numero e sigla dell'unità**.
   E' corretto quindi scrivere 15px oppure 5em.
   E' invece sbagliato usare 15 px o 5 em. In questi casi la regola sarà ignorata o mal interpretata.

Unità per le dimensioni

Le unità di misura usate per definire dimensioni, spazi o distanze sono 8. (Nella pratica comune solo alcune di esse sono realmente usate).

- **in** (inches - pollici) classica misura del sistema metrico americano, quasi mai utilizzato per definire il render a video, molto più utile per gli stili orientati alla stampa.
- **cm** (centimetri): come per i pollici, si tratta di un'unità molto poco utilizzata.
- **mm** (millimetri): valgono le considerazioni fatte per pollici e centimetri.
- **pt** (points - punti): unità di misura tipografica destinata essenzialmente a definire la dimensione dei font.
- **pc** (picas): unità poco usata. 1 pica equivale a 12 punti.
- **em** (em-height): unità di misura molto usata dagli autori CSS. 1 em equivale all'altezza media di un carattere per un dato font. E' un unità di misura relativa.
- **ex** (ex-height): poco usata. 1 ex equivale all'altezza del carattere x minuscolo del font scelto.
- **px** (pixels): unità di misura ideale su monitor. E' quella più usata e facile da comprendere.

Percentuali, colori, stringhe, URI

Percentuali

Un valore espresso in percentuale è da considerare **sempre relativo rispetto ad un altro valore**, in genere quello espresso per l'elemento parente. Si esprime con un valore numerico seguito (senza spazi) dal segno di percentuale: **60%** è pertanto corretto, 60 % no.

Colori

I diversi modi per esprimere il valore di un colore sono esattamente corrispondenti alle **regole dell'HTML**.

Stringhe

Alcune proprietà dei CSS possono avere come valore una stringa di testo. I valori espressi da stringhe vanno sempre racchiusi **tra virgolette**. Le proprietà in questione sono tre: content, quotes, text-align (ma solo per le tabelle definite con i CSS).

Valori URL

Si tratta di indirizzi che puntano a documenti esterni (in genere immagini, come negli sfondi). Possono essere assoluti o relativi. Nel secondo caso il path fa sempre riferimento alla posizione del foglio di stile e non del documento HTML. La definizione dell'indirizzo è sempre introdotta dalla **parola chiave url** e va inserita tra parentesi tonde **senza virgolette**. Esempio: **url(immagini/sfondo.gif)**.

Ereditarietà, cascade, conflitti tra stili

Le leggi fondamentali dei fogli di stile

Le leggi di

Ereditarietà
Cascade
Conflitti tra stili

sono alla base del meccanismo di funzionamento dei fogli di stile.

Sono le leggi che definiscono l'interdipendenza delle regole di stile all'interno di un documento HTML

Ereditarietà

Secondo il meccanismo dell'ereditarietà:

le impostazioni stilistiche applicate ad un elemento ricadono anche sui suoi discendenti.

Questo **fino a quando**, per un elemento discendente, non si imposti esplicitamente un valore diverso per quella proprietà.

Se impostiamo ad esempio il colore rosso per il testo (color: red;) a livello dell'elemento BODY tutti gli altri elementi suoi discendenti erediteranno questa impostazione. Ma se ad un certo punto definiamo nel codice del CSS un selettore con la proprietà color: black; l'ereditarietà viene spezzata.

Non tutte le proprietà sono ereditate; in genere sono quelle attinenti la formattazione del box model: margini, bordi, padding, background le più importanti.

I motivi sono in generale abbastanza logici: nel caso del bordo, ereditarlo è semplicemente senza senso. Se ne imposto uno, diciamo, per un paragrafo sarebbe assurdo che un elemento <A> o un testo in grassetto vengano circondati dallo stesso bordo!

Il peso nei conflitti tra stili

In una situazione in cui esistono le seguenti regole in un CSS:

```
p {color: black;}
.testo {color: red;}
```

E in una pagina HTML scrivo questo codice: ** Testo del paragrafo **
Avremo il testo del paragrafo rosso, non nero.

Questo perchè il selettore CLASS prevale sul TYPE SELECTOR.

Introduciamo dunque il concetto di **peso** che è di fatto la maggiore o minore importanza da assegnare a ciascuna regola.

In generale ricordiamo che, a parità di peso **gli stili in linea prevalgono su quelli incorporati che a loro volta prevalgono su quelli collegati.**

Specificità

La specificità descrive il peso relativo delle varie regole all'interno di un foglio di stile. Quando il peso relativo di una regola è maggiore di una seconda regola che è in conflitto con la prima fa sì che la prima regola prevalga sulla seconda. Ma come calcoliamo questo peso specifico?

Per ottenere la specificità di una regola si compone una tripletta di numeri: a-b-c, dove a è il numero di selettori ID, b il numero di selettori CLASS, c il numero di TYPE SELECTOR.

Ad esempio:

#titolo {color: black;} ha un ID, 0 classi, 0 elementi. La tripletta è dunque 1-0-0

.classe1 {color: green;} ha invece 0 ID, 1 classe, 0 elementi. La tripletta è 0-1-0

h1 {color: red;} ha infine 0 ID, 0 classi, un elemento. La tripletta è: 0-0-1

Vale la regola per cui gli ID pesano più delle CLASS che pesano più dei TYPE SELECTOR.

Per cui il tag **<h1 id=class="titolo" class="classe1">** apparirà in nero.

NB: il calcolo della specificità si effettua a prescindere dalla numerosità degli elementi di ordine inferiore.
La regola **#paragrafo {color: green;}** presenta la seguente specificità 1-0-0 ed è più importante di **div p {color: red;}** che ha la tripletta 0-0-2.

Importanza

Infine il concetto di importanza. Si tratta di una legge semplice e lineare:

Se una dichiarazione viene accompagnata dalla **parola chiave !important** essa balza al primo posto nell'ordine di applicazione a prescindere da peso, origine, specificità e ordine.

La sintassi di utilizzo è la seguente:

```
p { color: red  !important }
```

La parola chiave va posizionata dopo il valore della proprietà separata da uno spazio. Attenzione a non dimenticare il punto esclamativo!

Principali proprietà dei CSS

Box Model e proprietà di base del testo

Il BOX MODEL

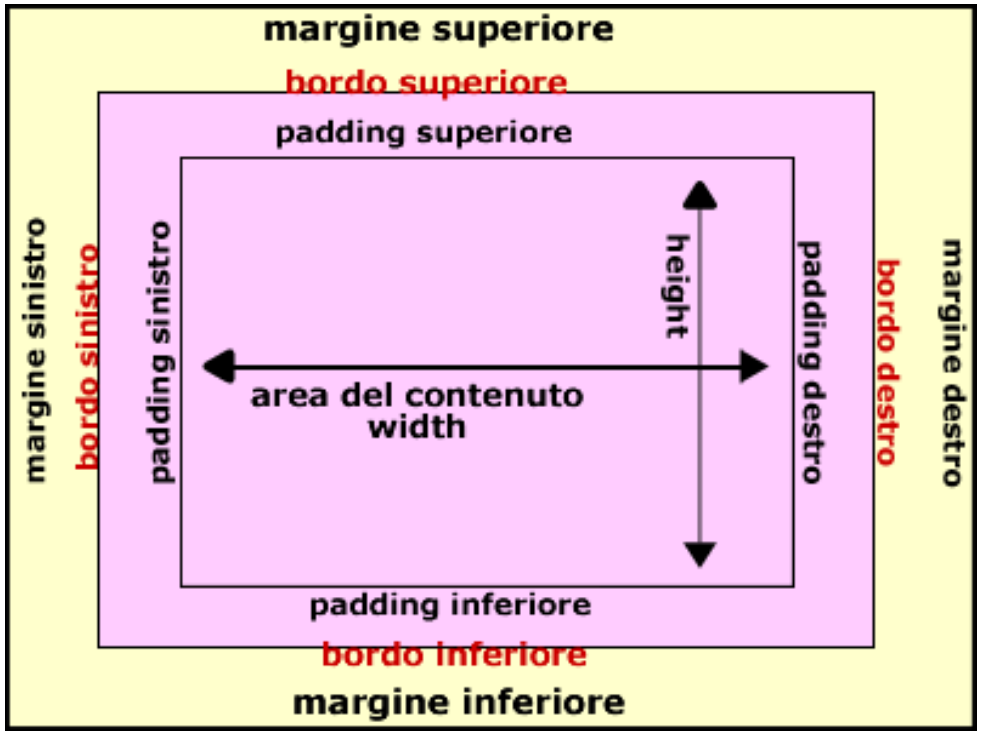
Il BOX MODEL

Se si vuole usare i CSS per scopi che vadano oltre la semplice gestione di sfondo e testo occorre avere ben chiaro il meccanismo che governa la presentazione dei vari elementi di una pagina.

Abbiamo visto che una pagina (X)HTML non è altro che un insieme di box rettangolari (che si tratti di elementi blocco o di elementi inline non fa differenza).

Tutto l'insieme di regole che gestisce l'aspetto visuale degli elementi blocco viene in genere riferito al cosiddetto **box model**.

Ogni box comprende un certo numero di componenti di base, ciascuno modificabile con proprietà dei CSS.



Componenti del box model

- **Area del contenuto** - È la zona in cui trova spazio il contenuto vero e proprio, testo, immagini, animazioni Flash. Le dimensioni orizzontali dell'area possono essere modificate con la proprietà width. Quelle verticali con height.
- **Padding** - È uno spazio vuoto che può essere creato tra l'area del contenuto e il bordo dell'elemento. Come si vede dalla figura, se si imposta un colore di sfondo per un elemento questo si estende dall'area del contenuto alla zona di padding.
- **Bordo** - È una linea di dimensione, stile e colore variabile che circonda le zone del padding e del contenuto.
- **Margine** - È uno spazio di dimensioni variabili che separa ogni elemento da quelli adiacenti.

NB: queste componenti non sono state introdotte con i CSS, ma fanno parte del normale meccanismo di rendering di un documento. Quando realizziamo una pagina (X)HTML senza fogli di stile è il browser ad applicare per alcune di queste proprietà le sue impostazioni predefinite. Per esempio, introdurrà un certo margine tra un titolo e un paragrafo o tra due paragrafi.

La novità è che con i CSS possiamo controllare con precisione al pixel tutti questi aspetti. Il box model è governato da una serie di regole di base concernenti la definizione di un box e il suo rapporto con gli altri elementi.

Larghezza del BOX

Bisogna distinguere tra la larghezza dell'area del contenuto e la larghezza effettiva di un box .

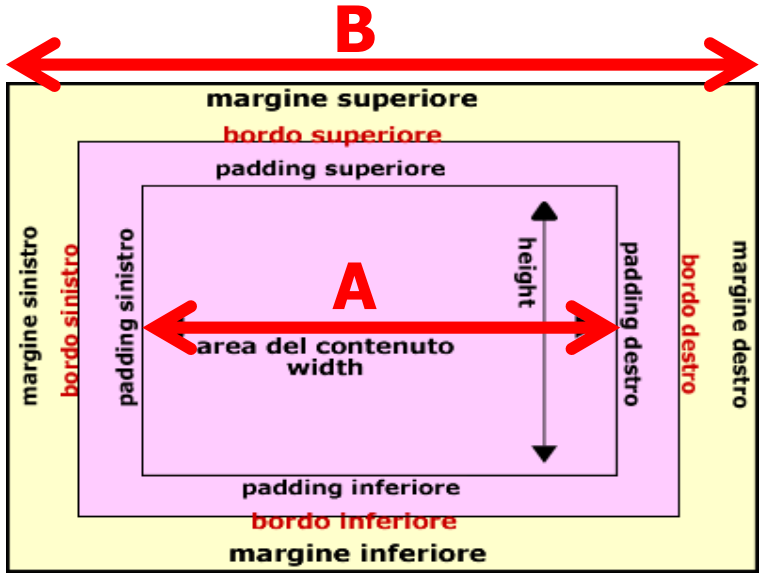
A - LARGHEZZA DELL'AREA DEL CONTENUTO = valore della proprietà width.

B - LARGHEZZA DEL BOX = margine sinistro + bordo sinistro + padding sinistro + area del contenuto + padding destro + bordo destro + margine destro

Margini, padding e bordi devono considerarsi a tutti gli effetti parte dell'area complessiva dell'elemento.

Se non si imposta WIDTH, o se si imposta a «auto» la larghezza di un box è uguale a quella dell'area del contenuto dell'elemento contenitore.

Cioè, ogni elemento «invade» automaticamente lo spazio in larghezza a sua disposizione che corrisponde allo spazio del contenuto dell'elemento che lo contiene.



Il valore AUTO

Per **tre** proprietà del box model è possibile utilizzare il valore «**auto**»: Le proprietà sono:

margini, altezza e larghezza

L'effetto è quello di lasciar calcolare al browser l'ammontare di ciascuna di esse in base alla risoluzione dello schermo e alle dimensioni della finestra.

Valori negativi

Sono ammesi valori negativi ma **solo per i margini**, Non è consentito per padding, bordi, altezza e larghezza.

Margini verticali e orizzontali tra gli elementi

Per due box adiacenti in senso verticale, che abbiano impostato un margine inferiore e uno superiore, **la distanza non è data dalla somma** delle due distanze.

A prevalere sarà invece la distanza maggiore tra le due.

Il meccanismo è detto del «**margin collapsing**» e non si applica nel senso orizzontale.

Proprietà di base del testo

Proprietà COLOR

La proprietà color si applica a tutti gli elementi ed è ereditata.

Se si imposta il colore per un elemento esso sarà ereditato da tutti gli elementi discendenti per cui non si definisca esplicitamente un altro colore.

I valori possibili sono:

- qualunque **valore di un colore** definito con i metodi descritti più avanti
- la parola chiave **inherit**. *(Con essa si dice esplicitamente al browser di ereditare le impostazioni definite per l'elemento parente)*.

Nell'esempio che segue dichiariamo che tutti i paragrafi di un documento devono avere il testo in rosso:

```
p {color: red }
```

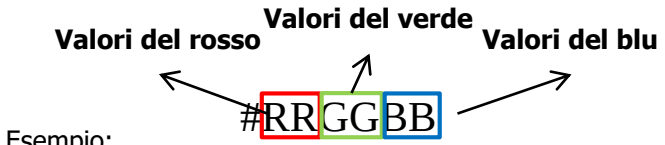
PROPRIETA' COLOR

La definizione dei colori ha la sintassi di base dell'HTML. Ecco il riepilogo.

PAROLE CHIAVE

BLACK
NAVY
BLUE
MAROON
PURPLE
GRENN
RED
TEAL
FUCHSIA
OLIVE
GRAY
LIME
AQUA
SILVER
YELLOW
WHITE

CODICI ESADECIMALI



Esempio:

#CC0000 =

Alcuni colori:

Colore	Hex	Colore	Hex
black	000000	silver	C0C0C0
navy	000080	blue	0000FF
green	008000	lime	00FF00
teal	008080	aqua	00FFFF

VALORI RGB: sintassi abbreviata

E' possibile usare per essi una sintassi abbreviata in cui i valori per il rosso, il verde e il blue sono definiti solo dalla prima lettera o numero. Il colore dell'esempio precedente può essere definito anche così: #C00

Proprietà COLOR: specificità dei CSS

Con i CSS è possibile utilizzare anche una tecnica specifica per definire un codice colore, utilizzando la parola chiave RGB seguita da valori in percentuale o in numeri interi da 0 a 255.

PERCENTUALI RGB

#RGB

rgb(rrr%, ggg%, bbb%)

rgb(0%,0%,0%)

rgb(100%,100%,100%)

VALORI RGB

#RGB

rgb(rrr, ggg, bbb)

rgb(0,0,0)

rgb(255,255,255)

Per ogni elemento si possono definire almeno tre colori:
il colore di **primo piano** (foreground); il colore del **bordo** (border), il colore di **sfondo** (background).

La proprietà COLOR si riferisce solo al primo piano, non allo sfondo o al bordo.

Gestione dello sfondo

Si tratta di una delle più importanti novità introdotte dai CSS.

Lo sfondo si gestisce con 5 proprietà specifiche che **possono essere applicate a tutti gli elementi**:

1. **background-color**
2. **background-image**
3. **background-repeat**
4. **background-attachment**
5. **background-position**

Sono tutte proprietà NON ereditate. Ciascuna di essa definisce un solo, particolare aspetto relativo allo sfondo di un elemento.

La proprietà **background**, invece, è la proprietà a sintassi abbreviata con cui possiamo definire sinteticamente e con una sola dichiarazione tutti i valori per lo sfondo.

Proprietà singole dello sfondo

background-color (colore di sfondo)

Valori: *un qualunque colore* | transparent (corrisponde al colore dell'elemento parente)

Esempi:

body { background-color: white }
p { background-color: #FFFFFF }

---

background-image (URL di un'immagine come sfondo)

Valori: *una URL assoluta o relativa che punti ad un'immagine* | none (default)

Esempi:

body { background-image: url(sfondo.gif) }
div { background-image: url(http://www.server.it/images/sfondo.gif) }

---

background-repeat (direzione in cui l'immagine di sfondo viene ripetuta)

Valori: repeat (default) | repeat-x | repeat-y | no-repeat

Esempi:

body { background-image: url(sfondo.gif) ; background-repeat: repeat }
div { background-image: url(sfondo.gif) ; background-repeat: repeat-x }

Proprietà singole dello sfondo

background-attachment (scorrimento dello sfondo rispetto al testo).
Valori: scroll | fixed
Esempio: **body { background-image: url(back.gif) ; background-attachment: fixed }**

---

background-position (punto di inserimento dell'immagine di sfondo)
Valori: coordinate di un punto sugli assi verticale e orizzontale con tre modalità:
valori in percentuale | *valori espressi con unità di misura* | top | left | bottom | right
Esempi:
body {background-image: url(back.gif) ; background-repeat: no-repeat ; background-position: 50px 50px }
(l'immagine apparirà a 50px dal bordo superiore e a 50px da quello sinistro della finestra)

body {background-image: url(back.gif) ; background-repeat: no-repeat ; background-position: center center }
(l'immagine apparirà centrata in entrambe le direzioni)

NB: se si imposta un solo valore esso verrà usato sia per l'asse orizzontale sia per quello verticale.

Proprietà a sintassi abbreviata per lo sfondo

Con la proprietà **background** possiamo definire in una unica dichiarazione tutti gli aspetti dello sfondo.
Per essere valida, la dichiarazione non deve contenere necessariamente riferimenti a tutte le proprietà viste finora, ma deve contenere almeno la definizione del colore di sfondo.

La struttura sintattica è la seguente:

selettore {background:background-color background-image background-repeat background-attachment background-position; }

I valori non vanno separati da virgole. L'ordine delle caratteristiche non è influente.

Esempio: **body { background:white url(sfondo.gif) repeat-x fixed }**

Proprietà di base del testo

Un aspetto essenziale dei CSS riguarda il nuovo approccio alla gestione del testo. Con i CSS viene sostanzialmente **abolito il tag <font>**.

Vengono introdotte molte nuove proprietà che potremmo dividere tra proprietà di base e proprietà estese.

Le proprietà di base sono quelle che permettono di gestire i seguenti aspetti:

- Il **font** da usare
- La **dimensione** del testo
- La **consistenza** del testo
- L'**interlinea** tra i paragrafi
- L'**allineamento** del testo
- La cosiddetta «**decorazione**» del testo, ovvero sottolineature, barrati ecc.

Carattere del testo

Per impostare il tipo di carattere di una porzione di testo si utilizza la proprietà **font-family**.
Font-family è applicabile a tutti i tag ed è ereditata. La sintassi permette di impostare non solo un singolo font ma un elenco di font alternativi; nell'esempio che segue:

p { font-family: arial, Verdana, sans-serif }

il browser tenterà di usare il primo font della lista. Se questo non è disponibile userà il secondo. In mancanza anche di questo verrà utilizzato il font principale della famiglia sans-serif presente sul sistema.

Quando si imposta la proprietà font-family si possono usare tutti i font che si vuole, ma **è opportuno inserire alla fine l'indicazione di una famiglia generica** tra le cinque elencate: (tra parentesi riportiamo i caratteri predefiniti sui sistemi Windows):

- **serif** (Times New Roman)
- **sans-serif** (arial)
- **cursive** (Comic Sans)
- **fantasy** (Allegro BT)
- **monospace** (Courier)

I nomi nella lista vanno separati dalla virgola.
I caratteri con nomi composti da più parole vanno inseriti tra virgolette.

Esempio: **div { font-family: Georgia, "Times New Roman", serif }**

Dimensione del testo

Font-size, insieme a font-family è la proprietà considerata essenziale nella definizione dell'aspetto del testo, di cui definisce le dimensioni. Applicabile a tutti gli elementi ed ereditata.

Le dimensioni possono essere espresse in senso **assoluto** o in senso **relativo**.

(*Assoluto significa che essa non dipende da nessun altro elemento ed è quella definita dall'unità di misura usata. Relativo significa che essa viene calcolata in base alla dimensione del testo dell'elemento parente*).

I valori assoluti ammessi sono:

- le sette parole chiave **xx-small, x-small, small, medium, large, x-large, xx-large**
- quelli espressi con le seguenti unità di misura: pixel (**px**), centimetri (**cm**), millimetri (**mm**), punti (**pt**), picas (**pc**), pollici (**in**), x-height(**ex**). Per i testi a video si consigliano però sono punti e pixel. Le altre sono più adatte per i CSS destinati alla stampa.

I valori relativi ammessi sono:

- le parole chiave **smaller** e **larger**
- quelli espressi in **em** (em-height)
- quelli espressi in **percentuale**

Esempi: **p { font-size: 12px }** ; **div.titolo { font-size: 50% }** ;
#div1 { font-size: large } ; **h2 { font-size: 1.2em }**

Consistenza del testo

Font-weight serve a definire la consistenza (il «peso visivo») del testo.
Si applica a tutti gli elementi ed è ereditata.

I valori ammessi per font.weight possono appartenere ad una scala numerica o essere delle parole chiave; anche in questo caso abbiamo valori assoluti o relativi:

- valori numerici **100** - **200** - **300** - **400** - **500** - **600** - **700** - **800** - **900** ordinati in senso crescente (dal leggero al pesante)
- **normal**. Valore di default. E' l'aspetto normale del font ed equivale al valore 400
- **bold**. Il carattere acquista l'aspetto che definiamo in genere grassetto. Equivale a 700
- **bolder**. Misura relativa. Il testo sarà più pesante rispetto al testo dell'elemento parente
- **lighter**. Misura relativa. Il testo sarà più leggero di quello dell'elemento parente

Esempi:

```
p { font-weight: 900 } ;  div { font-weight: bold }
```

Stile del testo

La proprietà **font-style** imposta le caratteristiche del testo in base ad uno di questi tre valori:

- **normal**: il testo mantiene il suo aspetto normale
- **italic**: formatta il testo in corsivo
- **oblique**: praticamente simile a italic

La proprietà si applica a tutti gli elementi ed è ereditata.

L'esempio è semplicissimo:

```
p { font-style: italic }
```

Interlinea

Grazie a **line-height** è possibile impostare dimensioni di interlinea in modo molto flessibile.

La proprietà, in realtà, serve a definire l'altezza di una riga di testo all'interno di un elemento blocco, ma l'effetto ottenuto è appunto quello ricercato da molti editor, ovvero un modo per impostare uno spazio tra le righe. La proprietà si applica a tutti gli elementi ed è ereditata.

Valori ammessi:

- **normal**. Il browser separerà le righe con uno spazio «standard».
  Generalmente corrispondente a un valore numerico compreso tra 1 e 1.2
- **valore numerico intero o decimale**. Usando valori numerici con dei decimali (es. 1.2, 1.3, 1.5) si un risultato in cui l'altezza della riga è uguale alla dimensione del font moltiplicata il valore espresso.
- **valore numerico con unità di misura**. L'altezza della riga sarà uguale alla dimensione specificata.
- **valore percentuale**. L'altezza della riga viene calcolata come una percentuale della dimensione del font.

E' sconsigliato l'utilizzo dei valori in percentuale e in unità esplicite; in generale è più controllabile la resa con valori numerici.

Esempi:

```
p { line-height: 1.5}
```

```
body { line-height: 15px }
```

Sintassi abbreviata per il testo

La proprietà **font** può essere paragonata a background. Si tratta di una proprietà a sintassi abbreviata che serve a impostare con una sola dichiarazione tutte le principali caratteristiche del testo.

Le proprietà definibili in forma abbreviata con font sono quelle che abbiamo descritto finora, ovvero:

font-family* | *font-size* | *line-height* | *font-weight* | *font-style* | *font-variant

Valgono le seguenti regole:

- I **valori** delle singole proprietà NON vanno separati da virgole.
- I valori definiti per la **font-family** VANNO separati da virgole *(anche in questo caso i nomi dei font costituiti da più parole vanno racchiusi tra virgolette)*
- Per quanto riguarda l'**ordine** delle proprietà, la dichiarazione dovrebbe sempre finire con la coppia font-size > font-family.
- Se si vuole indicare un valore di **line-height** è necessario mettere in valore dopo quello di font-size separandolo da uno slash. *(es. 10px/1.2, 24pt/1.5 ecc.)*

Nell'esempio:

```
p { font: bold 12px/1.5 Georgia, "Times New Roman", serif }
```

Nell'ordine abbiamo definito: font-weight, dimensione/ interlinea, font-family.

Allineamento del testo

La proprietà **text-align** serve a impostare l'allineamento del testo. E' ereditata e si applica a tutti gli elementi.

Sono ammessi i valori:

- **left**. Allinea il testo a sinistra
- **right**. Allinea il testo a destra
- **center**. Centra il testo
- **justify**. Giustifica il testo

Esempio:

```
p { text-align: center }
```

«decorazione» del testo

Text-decoration imposta particolari decorazioni e stili per il testo.
Ereditabile e applicabile a tutti gli elementi.

Valori ammessi:

- **none**. Il testo non avrà alcuna decorazione particolare
- **underline**. Il testo sarà sottolineato
- **overline** Il testo avrà una linea superiore
- **line-through**. Il testo sarà attraversato da una linea orizzontale al centro
- **blink**. Testo lampeggiante *(molto sconsigliato perché poco supportato dai vecchi browser e per il fatto che va valutato con molta attenzione l'inserimento di elementi «animati» in una pagina perché se sono troppi rendono poco efficace la comunicazione)*

Esempi:

```
p { text-decoration: overline }
a { text-decoration: none }
```

Esercizio n.3

Modificare la pagina dell'esercizio precedente introducendo CSS

Modifiche da apportare:

- Aggiunta di una sezione di contenuti a piacere con testo dentro a riquadri con sfondi colorati senza usare il tag <table> ma con impiego di tag ** **;
- Modifica del font del **titolo con un carattere senza grazie** senza usare il tag <font> ma con i CSS.
- Inserimento di un ** contenente de testo e una immagine di sfondo**; non inserire l'immagine come <img> ma sempre tramite CSS.

Suggerimento sulle fasi di integrazione dei CSS:

1. **Rimozione di tutte le parti di codice relative alla formattazione** dall'HTML dell'esercizio già svolto lasciando solo quelle di contenuto. (Es: il tag Data di consegna diventa semplicemente Data di consegna . *L'allineamento andrà successivamente indicato nel CSS*)
2. **Creazione del foglio di stile** contenente gli stili per applicare la formattazione.
3. **Applicazione degli stili** utilizzando i tre tipi di selettore: type selector, class e id. Utilizzare inoltre tutte e tre le modalità di incorporamento: CSS esterno, interno e inline.