CSS 3

Cosa è il CSS

L’acronimo **CSS** sta per **Cascading Style Sheets** (fogli di stile a cascata) e designa un linguaggio di stile per i documenti web. I CSS istruiscono un browser o un altro programma utente su come il documento debba essere presentato all’utente

albero del DOM

Elementi blocco (block) ed elementi in linea (inline)

Gli elementi blocco sono box che possono contenere altri elementi, sia di tipo blocco che di tipo inline. **Quando un elemento blocco è inserito nel documento viene automaticamente creata una nuova riga nel flusso del documento**

<h1>Titolo</h1>

Paragrafo

Elementi Inline

Gli elementi inline non possono contenere elementi blocco, ma solo altri elementi inline

Tramite i CSS possiamo modificare tale modalità attraverso la proprietà **display**. Grazie a quest'ultima, per fare solo un esempio, possiamo fare in modo che un titolo h1 (elemento blocco) venga mostrato come un elemento in linea

albero del DOM

Elementi rimpiazzati e non rimpiazzati

Un'altra distinzione da ricordare è quella tra elementi rimpiazzati ed elementi non rimpiazzati,

i **rimpiazzati** sono quelli in cui altezza e larghezza sono definite dall'elemento stesso e non da ciò che lo circonda, ad es:

img

input

text

textarea

select

Non rimpiazzati (non hanno dimensioni)

div

section

Elementi Padri e figli

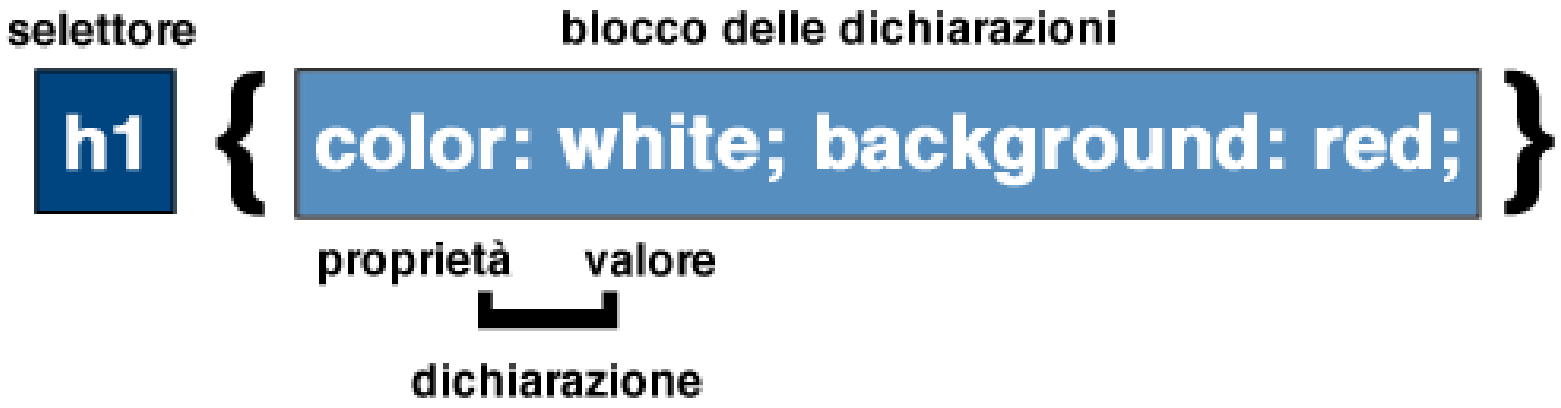
Un elemento si dice **genitore** (**parent**) quando contiene altri elementi. Si dice **figlio** (**child**) quando è racchiuso in un altro elemento

Se si scende di due livelli: diciamo allora che è un **antenato** e che questo è rispetto al primo un discendente.

Gli elementi che sono posti sullo stesso livello, ovvero quelli che hanno lo stesso genitore, si dicono **fratelli** (ingl: siblings).

Il capostipite (quello che non ha padri [<html>]) si dice che è **l'elemento radice**

Com’è fatto un CSS




```
h1 {
 color: red;
 font: 36px Helvetica, Arial, sans-serif;
}
```

Gli spazi bianchi lasciati all’interno di una regola non influiscono sul risultato

p {font: 12px Verdana, arial;}

Proprietà singole e a sintassi abbreviata

è possibile fare uso di **proprietà singole** e **proprietà a sintassi abbreviata**
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;} TOP – RIGHT – BOTTOM – LEFT

div {margin: 10px 5px 10px ;} TOP – RIGHT+LEFT - BOTTOM

div {margin: 10px 5px;} TOP+BOTTOM – RIGHT+LEFT

div {margin: 10px;} TOP+BOTTOM+RIGHT+LEFT

Commenti

Le parti racchiuse tra i segni /* e */, rappresentano commenti al codice

```
/* Stili per i titoli h1 */

/* Colore del testo delle liste */

/* Colore dei titoli h1 per la stampa */
```

Valori e unità di misura nei CSS

I valori di una proprietà **non vanno mai messi tra virgolette**.

Le uniche eccezioni riguardano i valori espressi da stringhe di testo e i nomi dei font formati da più di una parola.

```
p {font-family: "Times New Roman", Georgia, serif;}
```

Nei CSS i valori possono essere espressi da:

- **numeri** definiti come:
  - **numeri interi** (1, 23, 45, etc.)
  - **in virgola mobile** (1.2, 3.45, 4.90, etc.)
- **unità di misura**
- **percentuali**
- **codici** per la definizione dei colori
- **URI**
- parole chiave (**keywords**)
- **stringhe** di testo

```
/* Altezza di linea con un numero */
p {line-height: 1.2;}

/* Larghezza con unità di misura */
div {width: 300px;}

/* Larghezza in percentuale */
div {width: 60%;}

/* Colore con codice esadecimale */
body {background-color: #2795b6;}

/* URL per un'immagine di sfondo */
body {background-image: url(sfondo.jpg);}

/* Ripetizione dello sfondo con una keyword */
body {background-repeat: no-repeat;}

/* Stringa di testo */
content: "Viva i CSS";
```

unità di misura

Le più utilizzate:

px: unità più utilizzata ed ideale per gli schermi

em: unità poco utilizzata, è relativa alla dimensione standard (2 em= 2 volte la dimensione dell'attuale dimensione del font)

Percentuale

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.

h1 { line-height: 1.2em }
Questa regola significa che la *line-height* dell'elemento *h1* sarà del 20% superiore alla *font-size* dell'elemento h1.

Unità em vs rem

Quando si utilizzano **unità rem**, la dimensione in pixel in cui vengono tradotte dipende dalla dimensione del carattere dell'elemento principale della pagina, ovvero l'elemento html. Quella dimensione di carattere viene moltiplicata per qualsiasi numero stiate usando con le unità rem.

Ad esempio, con una dimensione di carattere dell'elemento principale di 16px, 10rem sarebbe pari a 160px, cioè 10 x 16 = 160.

Quando si utilizzano **unità em**, il valore in pixel finale è una moltiplicazione della dimensione del font sull'elemento cui si applica lo stile.

Ad esempio, se un div ha una dimensione di carattere di 18px, 10em sarebbe pari a 180px, cioè 10 x 18 = 180.

Se imposto ad una proprietà ad esempio height: 4rem e il valore del font-size dell'elemento html è 12px, l'altezza del contenitore sarà 48px;

```
html{
    font-size: 12px;
}
div{
    min-height: 4rem; /*=48px*/
}
```

Unità vw e vh

- **VW (Viewport Width):** gestisce il dimensionamento di un elemento in relazione alla larghezza della finestra del browser. L'unità vw è pari all'1% della larghezza della viewport.
- In queste condizioni, dunque, per rendere un elemento ampio sempre quanto l'intera larghezza della finestra del browser sarà necessario impostare la sua width sul valore 100vw.

- **VH (Viewport Height):** gestisce il dimensionamento di un elemento in relazione all'altezza della finestra del browser. L'unità vh è pari all'1% dell'altezza della viewport.

  In queste condizioni, dunque, per rendere un elemento alto sempre come l'intera finestra del browser sarà necessario impostare la sua height sul valore 100vh

CSS esterni e interni

È **esterno** un foglio di stile **definito in un file separato** dal documento. Si tratta di semplici documenti di testo modificabili anche con un editor di testo ai quali si assegna **l'estensione .css.**

```
<html>
 <head>
  <link href="css/style.css" rel="stylesheet" type="text/css">
 </head>
 <body>
 [...]
</html>
```

@import

Un altro modo per caricare CSS esterni è usare la **direttiva @import** all'interno **dell'elemento <style>**:

```
<html>
 <head>
  <style>
   @import url(style.css);
  </style>
 </head>
 <body>
 [...]
</html>
```

La direttiva import deve essere la prima all'interno dell'attributo style

@import

Un principio fondamentale è che all'interno del tag <style>, @import deve essere la prima regola definita
@import viene usata innanzitutto per **collegare un foglio di stile esterno al documento**.

La sintassi generica è la seguente:

```
<style type="text/css">
  @import url(stile.css);
</style>
```

```
@import url("stile.css");
```

L'url del foglio di stile può essere relativo, come negli esempi precedenti, o assoluto, come in questo:

```
<style type="text/css">
  @import url(http://www.miosito.it/stile.css);
</style>
```

Direttiva senza l'indicazione url:

```
@import "stile.css";
```

Le @-rules – media queries

Permettono di definire stili diversi per differenti tipi di media e di screen.

```
@media print {
 h1 {color: black;}
}
@media screen  and (min-width:576px) and (max-width:768px){
 h1 {color: red;}
}
```

Smartphone: < 576px

Tablet: <768px;

Notebook: <992px;

Desktop: <1200px;

Come dichiarare una media query

- attributo media nel link

```
<link rel="stylesheet" media=" screen and (min-width: 480px) " href="colore.css" />
```

Permette di avere diversi fogli di stile per media differenti.

- dentro il foglio di stile

```
@media screen and (min-width: 480px) {
/* qui vanno le regole CSS */
}
```

- all'interno di un altro foglio di stile

```
@import url(colore.css) screen and (min-width: 480px));
```

CSS Interni

I fogli incorporati sono quelli inseriti direttamente nel documento HTML tramite il **tag <style>**. Anche in questo caso la dichiarazione va posta **all'interno della sezione <head>**

```
<html>
 <head>
  <style type="text/css">   (eliminato in HTML5)
   body {background: white;}
   p {color: black;}
   [...]
  </style>
 </head>
 <body>
 [...]
</html>
```

CSS in linea

L'ultimo modo per formattare un elemento con i CSS consiste nell'uso dell'**attributo** HTML **style.**

Esso fa parte della collezione di attributi HTML definiti globali: si tratta di quegli attributi **applicabili a tutti gli elementi**.

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

La sintassi generica è la seguente:

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

Ereditarietà

le impostazioni di stile applicate ad un elemento **vengono ereditate anche dai suoi discendenti. Almeno fino a quando, per un elemento discendente, non si imposti esplicitamente un valore diverso per quella proprietà**.

body {color: #222;}

Tutti gli elementi discendenti di body, erediteranno questa impostazione. Ma se ad un certo punto definiamo nel codice del CSS un selettore con la proprietà color: white; l'ereditarietà viene spezzata:

body {color: #222;}

li {color: white;}

Peso

1- User CSS:

Sono i fogli di stile dell'utente: con un CSS in locale l'utente può ridefinire i CSS del browser e quelli dell'autore.
I CSS utente hanno priorità massima, e sono stati pensati soprattutto per l'accessibilità, ma non solo.

Strumenti->Opzioni Internet->Generale-> Accesso Facilitato->Fogli di stile utente

2- Author CSS:

Ovvero i fogli di stile specificati dall'autore della pagina. Questi andranno a ridefinire i CSS del browser, e ci sono tre sottotipi, nell'ordine dal più influente al meno influente:

- **quelli inlinea**.
- **quelli incorporati**
- **i CSS esterni,**

Possono inoltre essere definiti per diversi *media*, ovvero diversi dispositivi.

3- User Agent CSS:

Ovvero **il foglio di stile *di default* del dispositivo con cui si sta visualizzando la pagina**. In particolare, per quanto riguarda i browser, è il foglio di stile con cui viene visualizzata una pagina senza alcun altro CSS.

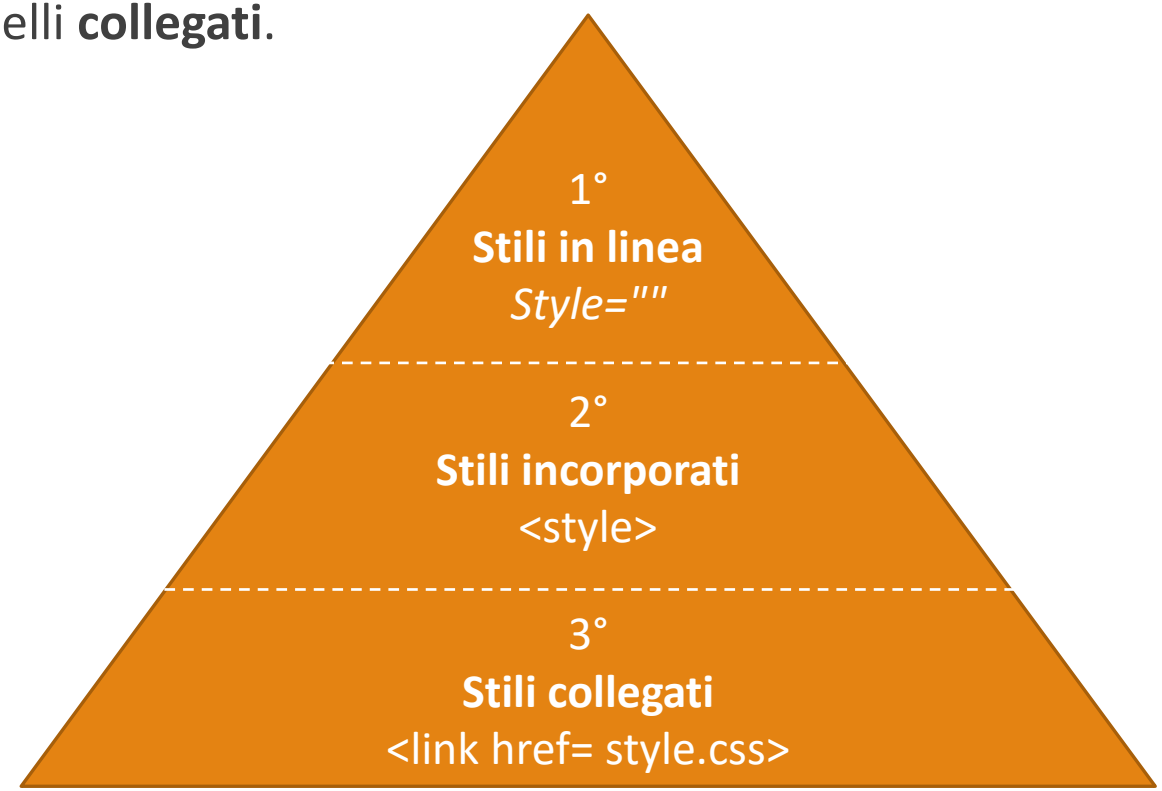
Stili in cascata

L'ordine, se le dichiarazioni degli stili sono fatte nell'ordine più corretto e logico, è quindi il seguente: gli stili **in linea** prevalgono su quelli **incorporati** che a loro volta prevalgono su quelli **collegati**.

Style=""

<style>

<link href=



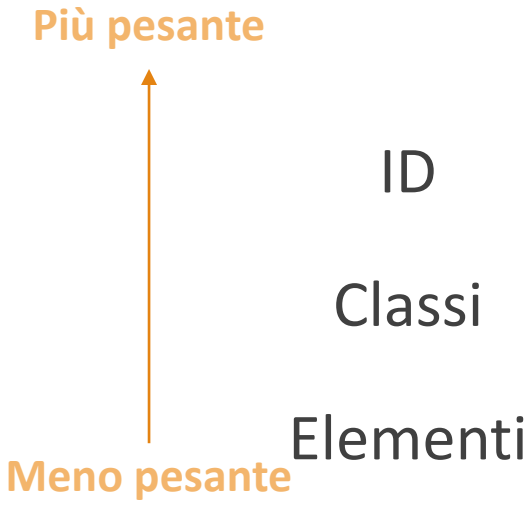
Specificità

Se ci sono due o più regole CSS che puntano allo stesso elemento, quella con il peso maggiore verrà applicata a quell'elemento.
La specificità ha prevalenza sulla provenienza dello stile (interno o file esterno)
(1,0,0)#id (0,1,0).class (0,0,1)span

I fattori del calcolo sono tre e ciascuno di essi rappresenta il valore di una tripletta.

I. Per prima cosa si conta il numero di selettori id presenti nella regola.
II. Si passa quindi a verificare la presenza di classi e pseudo-classi.
III. Infine si conta il numero di elementi definiti nella regola

gli **id pesano più delle classi che pesano più dei singoli elementi**



Importanza !important

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

Selettori di Base: Selettore universale

Il selettore universale è definito sintatticamente da un asterisco: *****. La sua funzione è quella di selezionare tutti gli elementi presenti in un documento

```
* {color: red}
```

Selettori di Base: Selettori di tipo

rappresentati dal **nome di uno specifico elemento** HTML

```
h1 {color: red}
p {color: green}
li {color: blue}
```

Selettori di Base: Selettori di classe

Nel codice HTML può essere assegnata una classe usando l'**attributo class** e assegnando ad esso un valore a nostra scelta

```
<h1 class="titolo">Testo</h1>
```

Nei CSS, per selezionare gli elementi a cui sia stata assegnata una classe, si utilizza questa sintassi:

```
.titolo {color: red}
```

Selettori di Base: Selettori di ID

Anche id è un attributo universale in HTML. Significa che tutti gli elementi presenti nel documento possono avere un loro id.

A differenza delle classi, però, **uno specifico id può essere assegnato solo ad un elemento. L'ID DEVE ESSERE UNIVOCO!!!**

Nei CSS, per selezionare un elemento cui sia stato assegnato un certo id, si usa questa sintassi, facendo precedere il valore dell'id dal simbolo del cancelletto (**#**):

#titolo

È anche possibile usare prima del cancelletto il nome dell'elemento:

h1#titolo

Selettori combinatori o di relazione

Una categoria fondamentale di selettori CSS è rappresentata dai cosiddetti **combinatori** (detti anche **selettori di relazione**). Hanno la funzione di **mettere in relazione elementi** presenti all'interno dell'albero del documento. Sono quattro:

- Selettore di **discendenti** (spazio)
  ES: **p strong**

- Selettore di **figli diretti(>)**
  ES: .**top>p**

- Selettore di **fratelli adiacenti (+)**: seleziona elementi che vengono immediatamente dopo l'elemento specificato. Elementi fratelli devono avere lo stesso elemento genitore. Adiacenti vuol dire immediatamente successivo.

- Selettore **generale di fratelli (~)** [ALT+126] _ e successori

Selettori discendenti

SELETTORE DI DISCENDENTI

Il selettore di discendenti è sicuramente quello più utilizzato dei quattro. Non è presente solo nella specifica CSS3 ma anche nelle precedenti versioni ed è utilissimo per evitare l'abuso delle classi per assegnare stili agli elementi

div#container p {color: red}

serve ad assegnare lo stile solo ai paragrafi contenuti nel div#container (tutti i p discendenti del div con id = container)

```
<div id="container">
        <p class="titolo">Questo testo è in un paragrafo discendente da un div con
        id <code>container</code> e sarà rosso.</p>
</div>
<div id="main">
        <p>Questo testo è in un paragrafo discendente da un div con id
        <code>main</code> e non sarà rosso.</p>
</div>
```

Selettore di figli

Il selettore di figli (>) consente di selezionare un elemento che è **figlio diretto** dell'elemento padre.

```
body > p {color: red}
```

```
<body>
        <p>Primo paragrafo</p>
        <div>
                <p>Secondo paragrafo</p>
        </div>
        <p>Terzo paragrafo</p>
</body>
```

solo il primo e il terzo sono **figli diretti di body**.
Il secondo è invece figlio diretto di un elemento div

Selettore di fratelli adiacenti

Serve a scorrere in orizzontale l'albero del DOM assegnando le regole CSS agli elementi che si trovano allo stesso livello di un altro elemento.
Consente di assegnare uno stile all'elemento fratello immediatamente adiacente

h1 + h2 {color: red; text-decoration: underline}

```
<div>
  <h1>1. Questo è il titolo principale.</h1>
  <h2>1.1 Questo è il primo sottotitolo.</h2>
  <p>...</p>
  <h2>1.2 Questo è il secondo sottotitolo.</h2>
  <p>...</p>
</div>
```

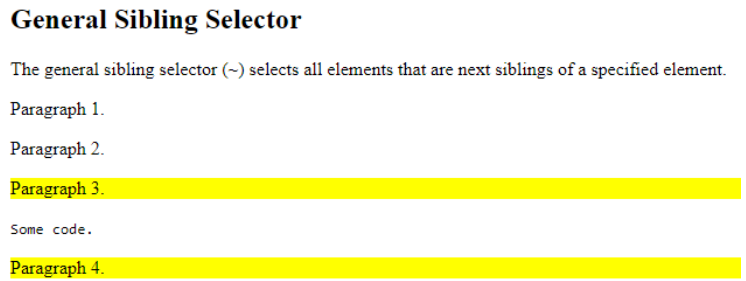
verrà selezionato solo il primo <h2> dato che è immediatamente adiacente al tag <h1>.

Selettore generale di fratelli

(**~**) è una generalizzazione di quello visto in precedenza. Esso assegna uno stile a **tutti gli elementi che sono fratelli ALT+126**

div ~ p { background-color: yellow;}

```
<h2>General Sibling Selector</h2>
<p>The general sibling selector (~) selects all elements that are next siblings of a specified element.</p>

<p>Paragraph 1.</p>

<div>
    <p>Paragraph 2.</p>
</div>

<p>Paragraph 3.</p>
<code>Some code.</code>
<p>Paragraph 4.</p>
```



andremo a selezionare tutti gli elementi <h2> dello stesso livello di <h1> indipendentemente dalla posizione che occupano.

(DEVONO ESSERE FRATELLI E SUCCESSORI)

Selettori di Attributo

Consentono di selezionare gli elementi all'interno di una pagina in base ai loro attributi e assegnare così lo stile desiderato

E[attribute]

Questo selettore individua **tutti gli elementi E che possiedono l'attributo attribute**, indipendentemente dal contenuto dell'attributo.

CSS:

```
a[title] {color: blue; text-decoration: underline}
```

HTML:

```
<a title="Lorem Ipsum" href="#">Lorem Ipsum</a>
```

E[attribute=value]

Questo selettore individua **tutti gli elementi E** che possiedono l'attributo **attribute che al proprio interno contiene il valore value**

```
a[title="Lorem"] {color: blue; text-decoration: underline}
```

```
<a title="Lorem" href="#">Lorem Ipsum</a>
```

```
h2[data-indirizzo]{color:#23197c;}
h2[data-indirizzo="mio"]{color:#c4c5b4;}
```

```
<h2 data-indirizzo="mio">titolo</h2>
<h2 data-indirizzo="no">titolo</h2>
```

CSS Custom Properties

Le CSS Custom Properties, altrimenti chiamate CSS Variables, consentono di introdurre le variabili nelle dichiarazioni CSS

Nei CSS, una una variabile è una qualsiasi "proprietà" il cui nome inizia con due trattini (dash dash in inglese).

```
:root {
  --primary-color: red;
}

p {
color: var(--primary-color);
}
```

CSS Custom Properties e @media

Le variabili CSS possono essere utilizzate con le regole @media condizionali.

Le Variabili CSS sono case-sensitive.

Le variabili CSS possono essere utilizzate direttamente nell'HTML.

```
<html style="--color: blue">
```

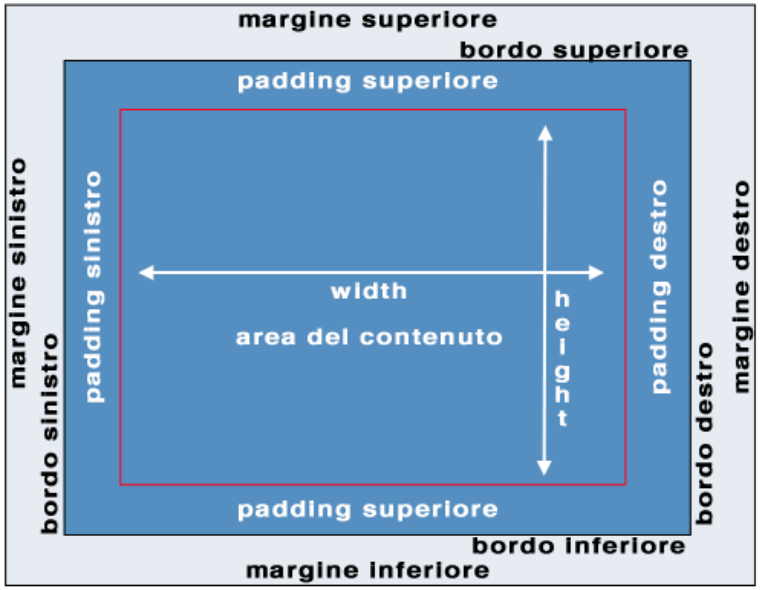
```
:root {
--padding: 20px;
}

@media screen and (min-width:768px) {
--padding: 30px;
}
```

Box Model

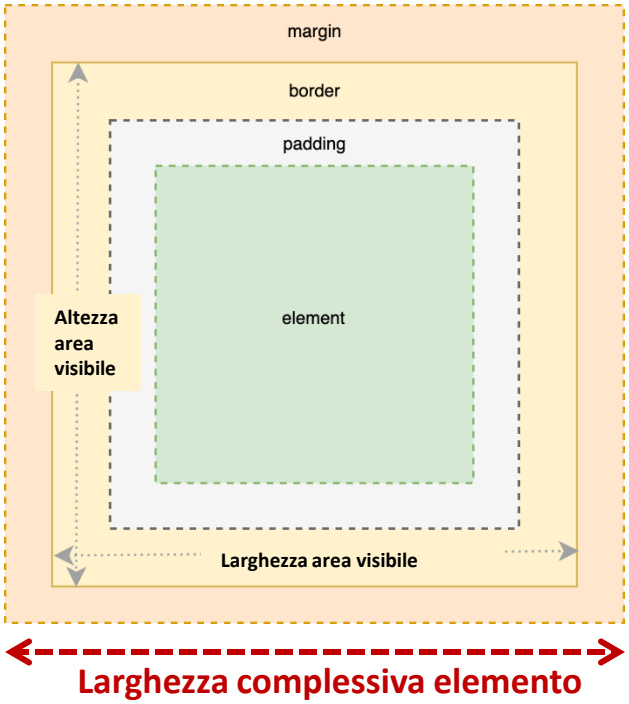
Componenti del box model

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

Ogni box comprende un certo numero di componenti di base, ciascuno modificabile con proprietà dei CSS. La figura qui sotto mostra visivamente tali componenti:



Box Model



Larghezza del box

Bisogna distinguere tra tre concetti:

la larghezza dell'area del contenuto;
la larghezza complessiva;
la larghezza dell'area visibile.

La prima è data dal valore della proprietà **width**.

La seconda corrisponde allo spazio occupato sulla pagina compresi i margini ed è data da questa somma:

margine sinistro + bordo sinistro + padding sinistro + area del contenuto
+ padding destro + bordo destro + margine destro

La terza corrisponde allo spazio occupato sulla pagina esclusi i margini, parliamo insomma della parte del box delimitata dai bordi e a cui si può applicare uno sfondo. È data da questa somma:

+ area del contenuto + padding destro + bordo destro +bordo sinistro + padding sinistro

auto

Solo per tre proprietà è possibile impostare il valore auto: margini (**margin**), altezza (**height**) e larghezza (**width**).

L'effetto dell'uso di auto è quello di lasciar calcolare al browser l'ammontare del valore per ciascuna di queste proprietà

Solo i margini possono avere **valori negativi**. Ciò 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 sarà data dalla somma delle due distanze**. A prevalere sarà invece la **distanza maggiore** tra le due. È il meccanismo del cosiddetto **margin collapsing.**

Le regole sono uguali se i div sono uno sotto l'altro:

```
#my {margin-bottom: 50px}
#my2 {margin-top: 50px}

/*#my {margin-bottom: 50px}*/
#my2 {margin-top: 50px}
```

La proprietà height

Definisce la distanza tra il bordo superiore e quello inferiore di un elemento. **Non è ereditata** e si applica a tutti gli elementi tranne:

colonne di tabelle;

elementi inline non rimpiazzati.

Il valore può essere espresso da:

- un **valore numerico** con unità di misura;
- un valore in **percentuale**: il valore in percentuale è sempre definito **rispetto all'altezza del blocco contenitore**, purché esso abbia un'altezza esplicitamente dichiarata; diversamente, la percentuale viene interpretata come auto;
- **auto**: l'altezza sarà quella determinata dal contenuto.

```
div {height: 250px;}
ul {height: 50%;}
p {height: auto;}
```

La proprietà min-height

Imposta **un'altezza minima** per un elemento. Valgono per questa proprietà le stesse osservazioni fatte per height relativamente al contenuto. Non è ereditata.

Sintassi ed esempi

selettore {min-height: valore;}

I valori possono essere:

- un valore numerico con unità di misura;
- un valore in percentuale.

div {min-height: 200px;}
p {min-height: 30%;}

La proprietà max-height

La proprietà **max-height** serve a impostare l'altezza massima di un elemento. Anche per essa valgono le osservazioni già fatte per il contenuto eccedente. Non è ereditata.

Sintassi ed esempi

selettore {max-height: valore;}

Per i valori possiamo ricorrere a:

- **none**: valore iniziale e di default, l'altezza dell'elemento non è limitata;
- un **valore numerico** con unità di misura;
- un valore in **percentuale**.

```
div {max-height: 400px;}
p {max-height: 40%;}
form {max-height: none;}
```

La proprietà overflow

Fornisce un modo per **gestire il contenuto che superi i limiti imposti con height**. Serve infatti per definire il comportamento di un **elemento blocco nel caso il suo contenuto ecceda dalle sue dimensioni esplicite.**

selettore {overflow : valore;}

I **valori** possono essere espressi con le **parole chiave**:

- **visible**: valore iniziale, il contenuto eccedente rimane visibile;
- **hidden:** il contenuto eccedente non viene mostrato;
- **scroll**: il browser crea barre di scorrimento che consentono di fruire del contenuto eccedente;
- **auto**: il browser tratta il contenuto eccedente secondo le sue impostazioni predefinite; di norma dovrebbe mostrare una barra di scorrimento laterale.
- **initial**: imposta la dimensione del contenitore al contenuto

div {overflow: **auto**;}
p {overflow: **hidden**;}
div {overflow: **visible**;}
p {overflow: **scroll**;}

Overflow

overflow: visible

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

overflow: hidden

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea

overflow: scroll

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud

```
#box2 {
background: #4c74be;
width: 300px;
height: 200px;
padding: 30px;
margin-bottom: 40px;
overflow-x: scroll;
}
```

overflow-x: scroll

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquipgrtrytfrewuffrrfigreigreigrekgreikeriiciigreirg ex ea commodo consequat.

width

Con la proprietà width, dunque, impostiamo la larghezza dell'area del contenuto di un box**, esclusi padding e bordi**.

Selettore

{width: valore;}

Il valore per width può corrispondere a:

- **auto**: **valore iniziale e di default**; se non si impostano margini, bordi e padding la larghezza dell'elemento sarà uguale all'area del contenuto dell'elemento contenitore;
- un **valore numerico** con unità di misura;
- un valore in **percentuale:** la larghezza sarà calcolata rispetto a quella dell'elemento contenitore.

La proprietà width non è ereditata.

```
div {width: auto;}
p {width: 90px;}
div.box {width: 50%;}
```

La proprietà min-width

Imposta la **larghezza minima** di un elemento. Si applica a tutti gli elementi, tranne a quelli in linea non rimpiazzati e agli elementi di tabelle.

Proprietà non ereditata.

Sintassi ed esempi

selettore {min-width: valore;}

I valori possono essere:

- un **valore numerico** con unità di misura;
- un valore in **percentuale**: la larghezza sarà come minimo quella espressa dalla percentuale riferita alla larghezza dell'elemento contenitore.

div {min-width: 400px;}

p {min-width: 40%;}

La proprietà max-width

Imposta la **larghezza massima** di un elemento. Non è ereditata.
l'elemento può assumere una larghezza inferiore rispetto al valore impostato ma non un valore superiore

Sintassi ed esempi

selettore {max-width: valore;}

Per quanto riguarda i valori, essi possono essere rappresentati da:

- **none**: valore di default, non c'è un limite per larghezza dell'elemento;
- un **valore numerico** con unità di misura;
- un valore in **percentuale**.

div {max-width: 400px;}
p {max-width: 40%;}

margin-top, left, right, bottom

Imposta la distanza tra il lato (bordo) di un elemento e gli elementi adiacenti. Si applica a tutti gli elementi e non è ereditata.

Selettore

{margin-top: valore;}

I valori possibili sono:

- un **valore numerico** con unità di misura: il valore è espresso in termini assoluti;
- un valore in **percentuale**: il valore è calcolato come percentuale rispetto alla larghezza (width) del blocco contenitore;
- **auto**: il browser calcola automaticamente la distanza.

div {margin-top: 20px;}
p {margin-top: 10%;}
img {margin-top: auto;}
Oppure margin-left ecc…

Margin – sintassi abbreviata

È una proprietà a sintassi abbreviata. Con essa è possibile specificare in una sola regola i **valori per tutti e quattro i lati** di un elemento. Si applica a tutti gli elementi e non è ereditata. I tipi di valori esprimibili sono gli stessi visti per le proprietà singole.

Sintassi ed esempi

La sintassi di base per questa proprietà è la seguente:

selettore {margin: valore-1, valore-2, valore-3, valore-4;}
Top, right, bottom, left

L'ordine di lettura va inteso **in senso orario**. Per cui: il primo valore si riferisce al lato superiore, il secondo a quello destro, il terzo al lato inferiore, il quarto a quello sinistro. In pratica, usare la sintassi vista nell'esempio equivale a scrivere:

```
div {
margin: 10px 15px 10px 20px;
}
```

=

```
div {
 margin-top: 10px;
 margin-right: 15px;
 margin-bottom: 10px;
 margin-left: 20px;
}
```

margin

Un'ulteriore abbreviazione della sintassi si può ottenere usando tre, due o un solo valore. Queste le regole:

- se si usano **tre valori**, il primo si riferisce al margine superiore, il secondo a quelli sinistro e destro, il terzo a quello inferiore;

  selettore {margin: valore-1, valore-2, valore-3;}

  Top, right + left, bottom

- se si usano **due valori**, il primo si riferisce ai lati superiore e inferiore, il secondo al sinistro e al destro;

  selettore {margin: valore-1, valore-2;}

  Top + bottom, right + left

- se si usa **un solo valore**, un uguale distanza sarà applicata ai quattro lati.

  selettore {margin: valore-1}

  Top + right+ bottom + left

padding-top, left, right, bottom

Imposta l'ampiezza del padding di un elemento. Si applica a tutti gli elementi e non è ereditata

selettore

{padding-top: valore;}

I valori possono essere:

- un **valore numerico** con unità di misura;
- un **valore in percentuale** calcolato come percentuale rispetto alla larghezza (width) del blocco contenitore.

div {padding-top: 40px;}

p {padding-top: 20%;}

Oppure padding-left ecc…

Padding – sintassi abbreviata

Proprietà a sintassi abbreviata. Serve a impostare i valori del padding per tutti e quattro i lati di un elemento. Valgono per essa tutte le osservazioni e le regole sintattiche viste per la proprietà margin.

Sintassi ed esempi

La sintassi di base per questa proprietà è la seguente:

selettore {margin: valore-1, valore-2, valore-3, valore-4;}
Top, right, bottom, left

I valori possono essere:

- un elenco di **valori numerici** con unità di misura;
- un elenco di valori in **percentuale**.

Nella definizione dei valori è possibile mischiare percentuali con valori assoluti in unità di misura.

padding

Un'ulteriore abbreviazione della sintassi si può ottenere usando tre, due o un solo valore. Queste le regole:

- se si usano **tre valori**, il primo si riferisce al padding superiore, il secondo a quelli sinistro e destro, il terzo a quello inferiore;

selettore {padding: valore-1, valore-2, valore-3;}

Top, right + left, bottom

- se si usano **due valori**, il primo si riferisce ai lati superiore e inferiore, il secondo al sinistro e al destro;

selettore {padding: valore-1, valore-2;}

Top + bottom, right + left

- se si usa **un solo valore**, un uguale distanza sarà applicata ai quattro lati.

selettore {padding: valore-1}

Top + right+ bottom + left

Border: lato e valori

In entrambi gli esempi di sintassi sostituite a **<lato>** uno degli indicatori dei quattro lati: **top, right, bottom o left**

Per quanto concerne i valori, come si vede dall'elenco delle proprietà, di ciascun lato si possono definire per il bordo tre aspetti:

il colore (color);
lo stile (style);
lo spessore (width).

Border

In linea di massima possiamo suddividere le proprietà relative ai bordi in due categorie: **proprietà singole** e **proprietà a sintassi abbreviata**

proprieta' singole	proprieta' sintassi abbreviata
border-top-color	border
border-top-style	border-bottom
border-top-width	border-top
border-bottom-color	border-right
border-bottom-style	border-left
border-bottom-width	border-color
border-right-color	border-style
border-right-style	border-width
border-right-width	
border-left-color	
border-left-style	
border-left-width	

border-color, border-width

BORDER-COLOR

I valori possibili per il color sono:

- un qualsiasi colore;
- la parola chiave inherit.

ESEMPI:

```
/* top | right | bottom | left */
 border-color: red yellow green blue;

/* top | left and right | bottom */
border-color: red rgb(240, 30, 50, 0.7) green;

/* <color> values */
border-color: red;
```

BORDER-WIDTH

il **width**. Esso può essere modificato secondo i seguenti valori:

- **un valore numerico con unità di misura**;
- **thin**: bordo sottile;
- **medium**: bordo di spessore medio;
- **thick**: bordo di spessore largo.

border-style

Lo stile di un bordo può invece essere espresso con una delle seguenti parole chiave:

Stile bordo	Descrizione
none	L'elemento non presenta alcun bordo e lo spessore equivale a 0.
hidden	Equivalente a none
dotted	Bordo a puntini
dashed	Bordo a lineette
solid	Bordo solido e continuo
double	Bordo solido, continuo, doppio
groove	Bordo in rilievo
ridge	Altro tipo di bordo in rilievo
inset	Effetto 'incastonato'
outset	Effetto 'sbalzato'

Esempio border

Come scrivere, dunque, una regola per impostare uno solo dei bordi? Si può fare così, usando le proprietà singole:

```
div {
 border-left-color: black;
 border-left-style: solid;
 border-left-width: 1px;
}
```

Ma è molto più comodo scrivere così, facendo ricorso alla **proprietà a sintassi abbreviata**:

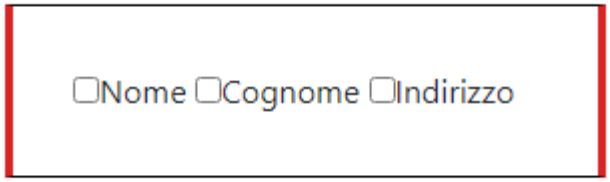
div {border-left: 1px solid black;}

Stili per tutti e 4 i bordi

Se si vogliono impostare stili per tutti e quattro i bordi del box, si hanno ancora una volta **due opzioni**. La prima è da usare quando si vogliono impostare insieme i quattro bordi ma si vuole assegnare a ciascuno uno stile diverso in quanto a colore, spessore, stile:

```
selettore {
 border-width: <valori>;
 border-style: <valori>;
 border-color: <valori>;
}
```

```
div {
 border-width: 1px 4px;
 border-style: solid;
 border-color: black red;
}
```



Usare la proprietà border

L'ultima proprietà a sintassi abbreviata è **border.** Con essa possiamo definire con una sola regola le **impostazioni per i quattro bordi**. Il suo uso è però limitato a un solo caso, peraltro molto comune: che i quattro bordi abbiano tutti lo stesso colore, lo stesso stile e lo stesso spessore.

Questa la **sintassi**:

```
selettore {
 border: <valore spessore>  <valore stile>  <valore colore>;
}
```

Che tradotto in codice reale diventa:

```
div {border: 2px solid black;}
```

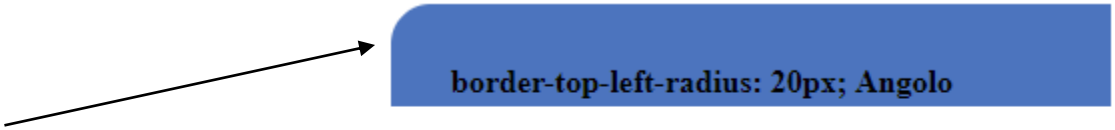
Border-radius

Si possono realizzare in maniera semplice e intuitiva **angoli arrotondati.**
Vediamo nei dettagli come la specifica definisce questa funzionalità.

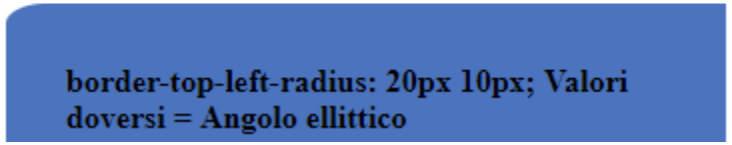
Le proprietà coinvolte sono cinque:

- border-**top-left**-radius
- border-**top-right**-radius
- border-**bottom-right**-radius
- border-**bottom-left**-radius
- border-radius

#box1 {border-top-left-radius: 20px}



#box2 {border-top-left-radius: **20px 10px**}
Se si definiscono due valori diversi, il primo imposta la misura del **verticale**



Per concludere un esempio con quattro valori:

#box {border-radius: 20px 40px 60px 80px
(top-left, top-right, bottom-right, bottom-left)

Gestione dello sfondo

La lista delle proprietà per lo sfondo applicabili a tutti gli elementi:

background-color
background-image
background-repeat
background-attachment
background-position
background-size
background-origin

background-color

Definisce il colore di sfondo di un elemento. Questa proprietà non è ereditata.

Sintassi

selettore {background-color: valore;}

Valori

- un qualunque colore
- la parola chiave **transparent.**
  Usando transparent come valore un elemento avrà come colore quello dell'elemento parente.

body { background-color: white; }
p { background-color: #FFFFFF; }
.classe1 { background-color: rgb(0, 0, 0)


```
Div{
background-color: black;
color: #ed069b;
}
```

background-image

Definisce l'URL di un'immagine da usare come sfondo di un elemento. Questa proprietà non è ereditata.

Sintassi

selettore { background-image: url(valore); }

Valori:

- un **URL assoluto** o **relativo** che punti ad un'immagine
- la parola chiave **none**. Valore di default.

```
body {
background-image: url(sfondo.gif);
}
div {
 background-image: url("https://www.sito/css3-
 logo.png");
}
```

background-repeat

Consente di definire la direzione in cui l'immagine di sfondo viene ripetuta. Proprietà non ereditata.

Sintassi

selettore {background-repeat: valore;}

Valori

- **repeat**. L'immagine viene ripetuta in orizzontale e verticale. È il comportamento standard.
- **repeat-x**. L'immagine viene ripetuta solo in orizzontale.
- **repeat-y.** L'immagine viene ripetuta solo in verticale.
- **no-repeat**. L'immagine non viene ripetuta.

background-attachment

Si imposta il comportamento **dell'immagine di sfondo rispetto all'elemento cui è applicata e all'intera finestra del browser.**

Si decide, in pratica, se essa deve scorrere insieme al contenuto o se deve invece rimanere fissa. Proprietà non ereditata.

Sintassi

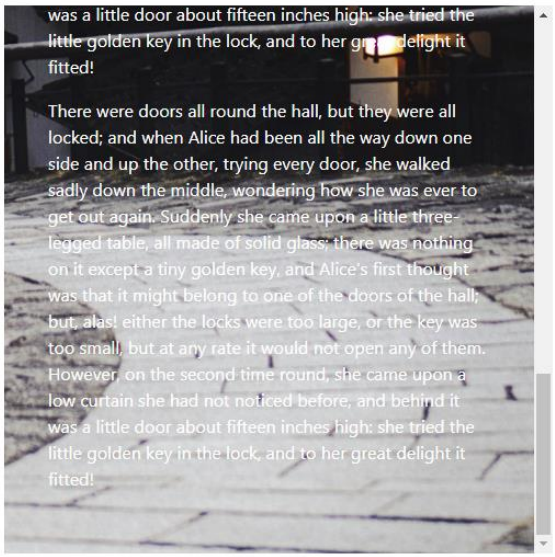
selettore {background-attachment: valore;}

Valori:

- **scroll**. L'immagine scorre con il resto del documento quando si fa lo scrolling della pagina. Scorre con il viewport ma non con il container. (esempio)
- **fixed**. L'immagine rimane fissa mentre il documento scorre. Immagine fissata al viewport (esempio)
- **local**. L'immagine scorre sia con il container che con il viewport.

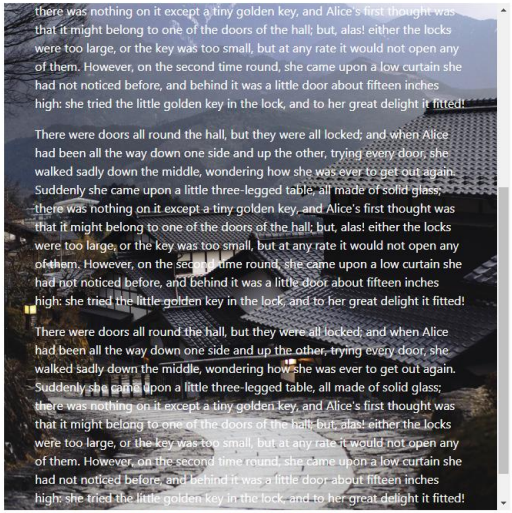
```
body { background-image: url(back_400.gif);
background-repeat: repeat-x;
background-attachment: fixed; }
```

background-attachment



LOCAL

L'immagine scorre con il contenuto



Scroll

L'immagine non scorre con il contenuto del contenitore ma con il viewport.

ALTRO TESTO PER RIEMPIRE LA PAGINA

There were doors all round the hall, but they were all locked; and when Alice had been all the way down one side and up the other, trying every door, she walked sadly down the middle, wondering how she was ever to get out again. Suddenly she came upon a little three-legged table, all made of solid glass; there was nothing on it except a tiny golden key, and Alice's first thought was that it might belong to one of the doors of the hall; but, alas! either the locks were too large, or the key was too small, but at any rate it would not open any of them. However, on the second time round, she came upon a low curtain she had not noticed before, and

background-position

Definisce il punto in cui verrà piazzata un'immagine di sfondo non ripetuta o da dove inizierà la ripetizione di una ripetuta. Si applica **solo agli elementi blocco o rimpiazzati**

selettore {background-position: valoreOriz | valoreVert;}

I valori specificano le **coordinate** di un punto sull'asse verticale e su quello orizzontale e possono essere espressi con diverse unità di misura e modalità:

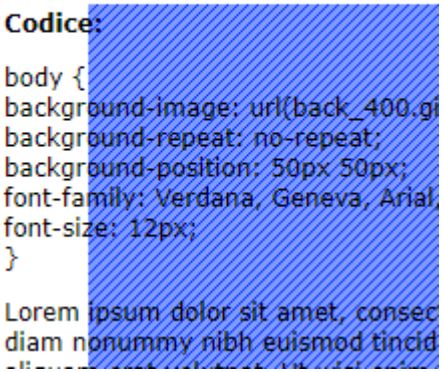
x% y%

xpos ypos

left top
left center
left bottom
right top
right center
right bottom
center top
center center
center bottom

```
body {
  background-image:
url(back_400.gif);
  background-repeat: no-
repeat;
  background-position:
50px 50px;
}
```

Background-position:



background-size

Definisce la dimensione dell'immagine di background.

I valori possono essere espressi con diverse unità di misura e modalità:

- con valori in **percentuale**
- Con **valori numerici** e unità di misura
- **cover**: scala l'immagine più grande possibile per riempire il contenitore, ne fa vedere anche solo una parte
- **contain**: scala l'immagine più grande possibile per riempire il contenitore, ma la mostra sempre intera

background

Possiamo definire in un colpo solo 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.

Sintassi

background:
bg-color
bg-image position/bg-size
bg-repeat
bg-origin
bg-clip
bg-attachment ;

```
body {
   background: #00ff00 url("smiley.gif") no-repeat fixed center;
}
```

La proprietà display

Il valore può essere rappresentato unicamente da una parola chiave. Nella pratica comune

- **block** l'elemento viene reso come un elemento blocco
- **inline** l'elemento a cui viene applicata assume le caratteristiche degli elementi inline
- **inline-block** l'elemento **può assumere, come gli elementi blocco, dimensioni esplicite (larghezza e altezza), margini e padding, ma come tutti gli elementi inline, si disporrà orizzontalmente** e non verticalmente, potendo essere circondato dal testo ed essendo sensibile all'allineamento verticale
- **none** l'elemento non viene mostrato; o meglio: è come se non fosse nemmeno presente nel documento, in quanto non genera alcun box (non occupa spazio); l'uso del valore none è uno dei mezzi con cui, nei CSS, si può nascondere un elemento

Gestione del testo

La gestione del testo e della tipografia è un aspetto essenziale dei CSS. Le proprietà che definiscono il modo in cui il testo appare sullo schermo sono tante e abbiamo deciso di suddividere l'argomento in due lezioni. Iniziamo quindi dalle proprietà di base

il font da usare;
la sua dimensione;
la sua consistenza
l'interlinea tra le righe;
l'allineamento del testo:
la sua decorazione (sottolineature, etc.).

Font-family

font-family

La proprietà font-family serve a impostare il tipo di carattere tipografico per una qualunque porzione di testo. Si applica a tutti gli elementi ed è ereditata.

p {**font-family**: Arial, Verdana, sans-serif;}

Quando la pagina viene caricata, il browser tenterà di usare il primo font della lista. Se questo non è disponibile sul dispositivo dell'utente userà il secondo. In mancanza anche di questo, verrà utilizzato il font principale della famiglia sans-serif presente sul sistema.

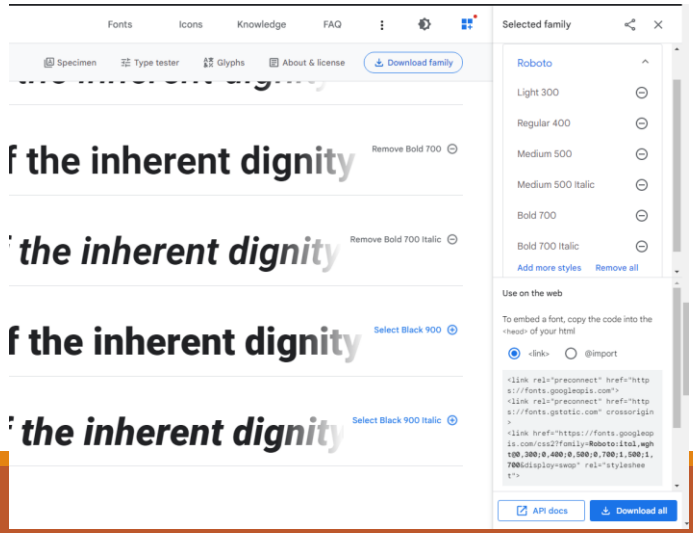
serif (Times New Roman);
sans-serif (Arial);
cursive (Comic Sans);
fantasy (Allegro BT);
monospace (Courier).

usare google font

- andare su **google font**
- Cercare il font desiderato.
- Copiare il link nella barra laterale a destra e inserirlo all'interno del tag «head» del documento.

```
<link rel="preconnect" href="https://fonts.googleapis.com">
<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
<link href="https://fonts.googleapis.com/css2?family=Roboto&display=swap"
rel="stylesheet">
```

- Nel file CSS usare la regola
  font-family: 'Roboto', sans-serif;



inserire un font dal ttf

- scaricare il font ad esempio fontsquirrel
  https://www.fontsquirrel.com/

- Mettere il file scaricato nella cartella (nell'esempio /fonts)

- includerlo nel css

```
@font-face{
src: url("fonts/SinkinSans-100Thin.otf");
font-family: SinkinSans;
}
body {
font-family: SinkinSans;
}
```

font-size

è la proprietà considerata essenziale nella definizione dell'aspetto del testo, di cui definisce le dimensioni. È applicabile a tutti gli elementi ed ereditata.

Dimensione assoluta significa che essa non dipende da nessun altro elemento ed è quella definita dall'unità di misura usata.

Dimensione relativa significa che essa viene calcolata in base alla dimensione del testo dell'elemento parente.

Valori dimensione assoluta:

- 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).
  Di tutte queste unità, le uniche proponibili per il testo sono **punti e pixel**. Si consiglia di usare la prima solo per CSS destinati alla stampa.

font-size

Sono **valori relativi**:

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

Nelle pratiche più comuni, la scelta del dimensionamento dei font viene fatta tra pixel, em, rem e percentuale.

* Se la dimensione di un font dovrebbe essere 12px, mettere 2em lo fa diventare 24px, mentre metterlo 3em significa metterlo 36px

font-weight

Serve a definire la **consistenza** o "peso" visivo del testo. Si applica a tutti gli elementi ed è ereditata

Il "peso" visivo di un carattere può essere espresso con una **scala numerica** o con **parole chiave**:

- **valori numerici**: 100 – 200 – 300 – 400 – 500 – 600 – 700 – 800 – 900 ordinati in senso crescente (dal più leggero al più pesante);
- **normal**: valore di default, è 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; serve a specificare che una determinata porzione di testo dovrà apparire più pesante a livello visuale rispetto al testo dell'elemento parente;
- **lighter**: misura relativa; il testo sarà più leggero di quello dell'elemento parente.

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

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. E' possibile specificare un angolo di inclinazione che va da -90 a 90 gradi. Valore di default: 14deg.

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

Sintassi ed esempi

selettore {font-style: valore;}

p {font-style: italic;}

This paragraph is normal.

This paragraph is italic.

This paragraph is oblique.

line-height

Serve a definire **l'altezza di una riga di testo** all'interno di un elemento blocco. Ma l'effetto ottenuto è appunto quello di impostare uno **spazio tra le righe**

- **normal**: il browser separerà le righe con uno spazio ritenuto "ragionevole"; dovrebbe corrispondere a un valore numerico compreso **tra 1 e 1.2**;
- un **valore numerico**: usando valori numerici tipo **1.2, 1.3, 1.5** si ottiene questo risultato: l'altezza della riga sarà uguale alla **dimensione del font moltiplicata per questo valore**;
- un valore numerico con **unità di misura**: l'altezza della riga sarà uguale alla dimensione specificata;
- **percentuale**: l'altezza della riga viene calcolata come una **percentuale della dimensione del font**.

```
p {line-height: 1.5;}
body {line-height: 15px;}
```

Line-height: normal

Far out in the uncharted backwaters of the unfashionable end of the western spiral arm of the Galaxy lies a small unregarded yellow sun.

Line-height:2.5

Far out in the uncharted

backwaters of the

unfashionable end of the

western spiral arm of the

Galaxy lies a small

unregarded yellow sun.

font

La proprietà font è una proprietà a sintassi abbreviata che serve ad impostare con una sola dichiarazione tutte le principali caratteristiche del testo. Le proprietà definibili in forma abbreviata con font sono:

font-family;
font-size;
line-height;
font-weight;
font-style;
font-variant;

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

text-decoration

I **valori** che è possibile usare sono:

- **none**: il testo non avrà alcuna decorazione particolare;
- **underline**: il testo sarà **sottolineato**; happy than right
- **overline**: il testo avrà una **linea superiore**; happy than right
- **line-through:** il testo sarà attraversato da una **linea orizzontale al centro**; ~~happy than right~~

```
p {text-decoration: none;}
a { text-decoration: underline;}
```

text-decoration-style ()
text-decoration-color solid, wavy, dotted, dashed, double
text-decoration-line (underline, overline, line-through))
text-decoration-thickness (valore num con unità misura, %)). Imposta lo spessore del text-decoration.

Sintassi abbreviata:
Selettore{text-decoration: underline wavy green 3p

Confusion kissed me on the cheek, and left a taste so bittersweet

font-variant

Consente di **trasformare il testo in maiuscoletto** (lettere in maiuscolo rese con dimensioni uguali ai caratteri minuscoli). Proprietà ereditata.

selettore {font-variant: valore;}

I valori possibili sono solo due:

- **normal**: il testo ha il suo aspetto normale; valore iniziale e di default
- **small-caps**: trasforma il testo in maiuscoletto.
- **all-small-caps**: trasforma tutto il testo in maiuscoletto.

h2 {font-variant: small-caps;}

normal	*Small-caps*	*All-small-caps*
Difficult waffles	DIFFICULT WAFFLES	DIFFICULT WAFFLES

text-indent

Definisce **l'indentazione della prima riga in ogni elemento contenente del testo**. Proprietà ereditata.

selettore {text-indent: valore;}

Si può esprimere il **valore** con:

- un **valore numerico** con unità di misura;
- un valore in **percentuale**.

Come al solito, il valore con unità di misura è assoluto, quello in percentuale è relativo. In questo caso il valore è **relativo alla larghezza dell'area del contenuto**. In pratica, se per un paragrafo largo 200px imposto un'indentazione uguale al 10%, essa sarà uguale a 20px.

div {text-indent: 10%;}

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt.

div {text-indent: -3em;}

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt.

text-transform

Questa proprietà serve a cambiare gli attributi del testo relativamente a **tre aspetti**: **maiuscolo**, **minuscolo**, **prima lettera maiuscola**. È una proprietà ereditata.

selettore {text-transform: valore;}

- la keyword **none**: valore di default; nessuna trasformazione viene applicata;
- **capitalize**: la **prima lettera** di ogni parola viene trasformata in **maiuscolo**;
- **uppercase**: **tutto** il testo diventa **maiuscolo**;
- **lowercase**: **tutto** il testo è **minuscolo**.

```
p {text-transform: capitalize;}
h1 {text-transform: uppercase;}
```

Text-shadow

Consente di creare un testo ombreggiato grazie alla proprietà **text-shadow.**

```
text-shadow: h-shadow v-shadow blur-radius color
```

- il primo (2px) definisce lo spostamento dell'ombra sull'asse orizzontale (x), **horizontal shadow** ;
- il secondo (2px) definisce lo spostamento dell'ombra sull'asse verticale (y), **vertical shadow**;
- il terzo valore (3px) imposta il livello di **sfocatura (blur)** dell'ombra: più alto è questo valore, più sfocata apparirà l'ombra; se si usa 0 otterremo un'ombra netta e senza sfocatura; **OPZIONALE**
- il quarto valore (#333) definisce il **colore** dell'ombra. **OPZIONALE**

```
text-shadow: 2px 2px 3px #333;
```

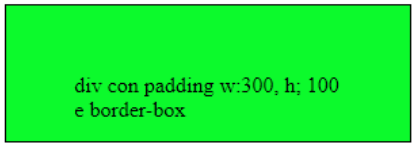
Lorem ipsum dolor sit amet, consectetur

Box sizing

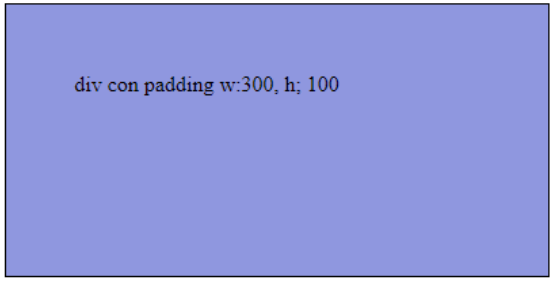
Per rendere più flessibile il modello di base, i CSS3 aggiungono, nel modulo Basic User Interface, la proprietà **box-sizing.**

Valori:

- **content-box**: il valore di width fa riferimento **all'area del contenuto** (il che equivale a seguire la regola e il comportamento qui sopra citati);
- **border-box**: il valore di width fa riferimento al box nella sua interezza, **comprendendo cioè anche il padding e il bordo**.




```
#id1{background-color: #0cfa2c;width:
300px;height: 100px;padding:50px;
border: 1px solid black;
box-sizing: border-box;}
#id2{background-color: #8f97df;width:
300px;height: 100px;
border: 1px solid black;padding: 50px;
}
```



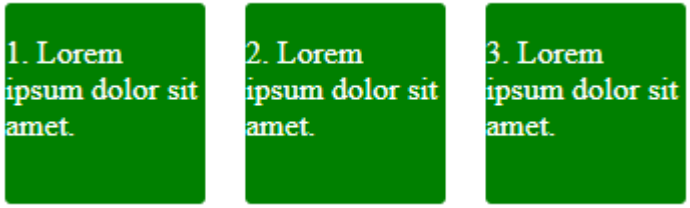
Flexible box layout (**flexbox**)

La struttura minima per agire con il **flexbox** prevede **un elemento contenitore** e degli **elementi figli**

```
<div id="container" >
<div id="box1" >
<p>1. Lorem ipsum dolor sit amet.</p>
</div>
<div id="box2" >
<p>2. Lorem ipsum dolor sit amet.</p>
</div>
<div id="box3" >
<p>3. Lorem ipsum dolor sit amet.</p>
</div>
</div>
```

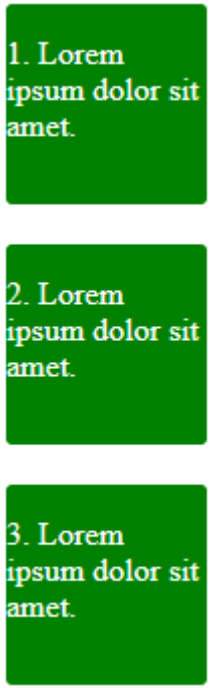
Flexbox – orientamento verticale

```
#container{
width: 500px;
display:flex;
flex-direction: row;
flex-wrap: nowrap;
}
#box1,#box2,#box3{
width: 100px;
height: 100px;
background-color: green;
color:white;
border-radius: 3px;
margin: 10px;
}
```



Flexbox – orientamento orizzontale

```
#container{
width: 500px;
display:flex;
flex-direction: column;
flex-wrap: nowrap;
}
#box1,#box2,#box3{
width: 100px;
height: 100px;
background-color: green;
color:white;
border-radius: 3px;
margin: 10px;
}
```



Flexbox justify-content

Determina l'allineamento lungo l'asse principale del container flex

- **flex-start** – Valore di default. Gli elementi flex sono posizionati all'inizio del container
- **flex-end** – Gli elementi sono posizionati alla fine del container.
- **center:** gli elementi vengo centrati all'interno del container
- **space-between** – gli elementi sono posizionati con uno spazio tra di loro
- **space-around** – gli elementi sono posizionati con spazion prima, tra e dopo di loro
- **space-evenly** – gli elementi hanno uguale spazio attorno a loro.

Flexbox justify-content

```
#container{
width: 500px;
height: 500px;
display:flex;
flex-direction: row;
flex-wrap: nowrap;
justify-content: space-around;
background-color: #dad8d8;
padding: 10px;
}
#box1,#box2,#box3{
width: 100px;
height: 100px;
background-color: green;
color:white;
border-radius: 3px;
}
```

row-reverse, column-reverse

Inverte l'ordine degli item (ex row reverse)

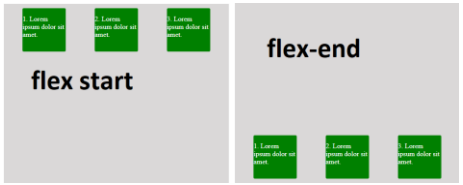

```
flex-direction: row-reverse;
```

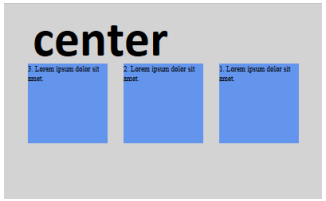
per le colonne: **column-reverse**

align-items: allineamento verticale

In un flexbox, la proprietà align-items regola la distribuzione degli elementi lungo l'asse perpendicolare all'asse principale (quindi asse verticale se flex-direction: row; asse orizzontale se flex-direction: column).

- **stretch** – valore di default. Gli elementi vengono allungati per adattarsi al contenitore.
- **flex-start** – gli elementi sono posizionati in cima al container.
- **flex-end** – gli elementi vengono disposti alla fine del container
- **center** – gli elementi sono posizionati al centro del container (verticalmente se flex-direction: row)
- **baseline** – gli elementi sono posizionati alla baseline del container






```
#container{
width: 500px;
height: 500px;
display:flex;
flex-direction: row;
flex-wrap: nowrap;
justify-content: space-around;
align-items: flex-end;
background-color: #dad8d8;
padding: 10px;
}
#box1,#box2,#box3{
width: 100px;
height: 100px;
background-color: green;
color:white;
border-radius: 3px;
}
```

Riferimenti bibliografici

I contenuti sono tratti dal sito html.it, w3schools.com, developer.mozilla.org e rielaborati dal docente