CSS Cascading Style Sheets

BOX-MODEL

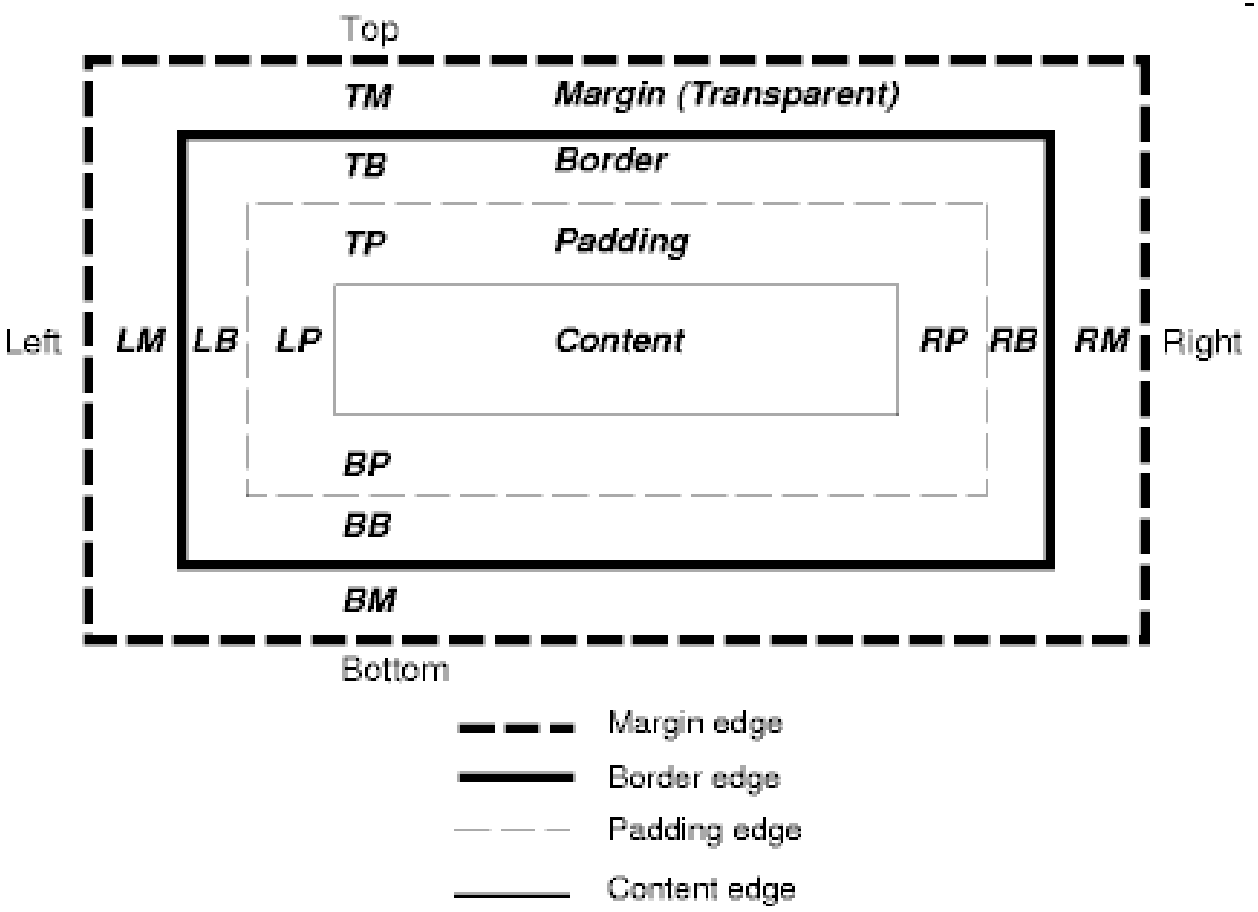
Box-Model

- Una delle cose più importanti introdotte con i CSS è il box model che permette di creare una struttura (layout) fatta da rettangoli e/o quadrati, praticamente colonne e righe con o senza bordo che servono per poter impaginare i dati quali: testi, immagini o qualsiasi altra cosa che potrebbe far parte di una pagina web.

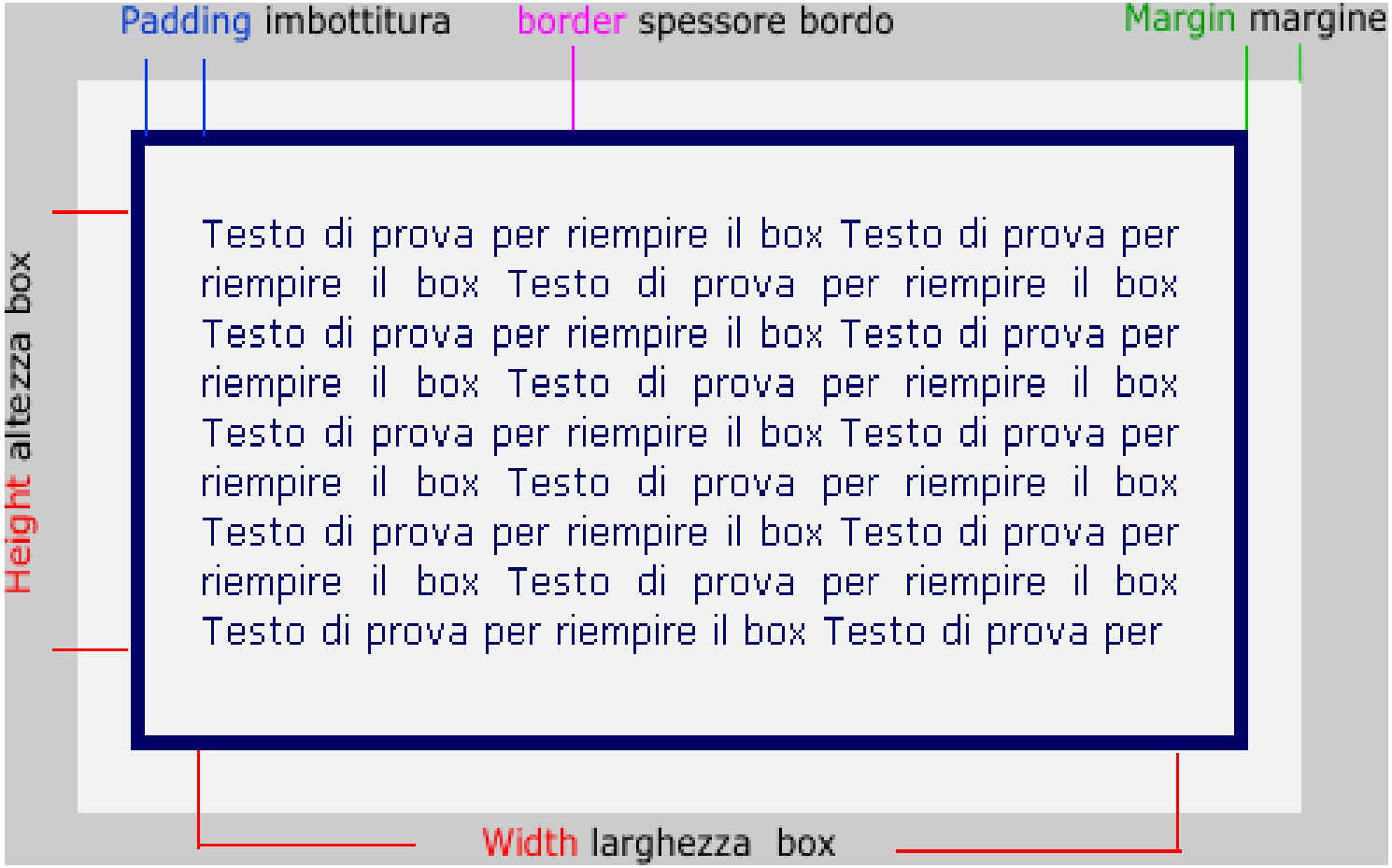
BOX-MODEL

- **Ogni elemento html viene gestito dal browser come un box al cui interno è contenuto il testo o gli altri elementi html.**
- Il box può avere o meno una cornice e le proprietà, quali lo sfondo, sono valide all'interno del box.
- Se l'elemento contenuto nel blocco sono degli altri tag html, il blocco conterrà a sua volta degli altri blocchi!

IL BOX


Top
TM
Margin (Transparent)
TB
Border
TP
Padding
Left
LM
LB
LP
Content
RP
RB
RM
Right
BP
BB
BM
Bottom
Margin edge
Border edge
Padding edge
Content edge

IL BOX


Padding imbottitura
border spessore bordo
Margin margine
Height altezza box
Testo di prova per riempire il box Testo di prova per riempire il box Testo di prova per riempire il box Testo di prova per riempire il box Testo di prova per riempire il box Testo di prova per riempire il box Testo di prova per riempire il box Testo di prova per riempire il box Testo di prova per riempire il box Testo di prova per riempire il box Testo di prova per riempire il box Testo di prova per
Width larghezza box

Box-Model: le quattro parti

Il box può essere suddiviso in quattro parti:

- Il **contenuto**, in cui è strettamente contenuto il testo o gli altri elementi html;
- Il **bordo**, è la cornice più esterna che può essere più o meno visibile con diversi spessori o colori;
- Il **padding**, è lo spazio che si trova tra il contenuto e il bordo;
- Il **margine**, è lo spazio che si trova tra il bordo e gli elementi adiacenti al blocco.

Box-Model: esempio 1

Questo il codice (esempio) in css per definire un box-model

```
#box
{
width: 350px;
 height: 150px;
padding: 20px;
border: solid 5px;
margin: 20px;
 }
```

Da notare che la larghezza e l'altezza totale di un box sono dati dalla somma dei contenuti più quello del padding più quello dello spessore del bordo. Nel nostro caso lo spazio occupato realmente sarà:

350 + (20 + 20 + 5)*2 = **440 pixel per la larghezza** e

150 + (20 + 20 + 5)*2 = **240 pixel in altezza**.

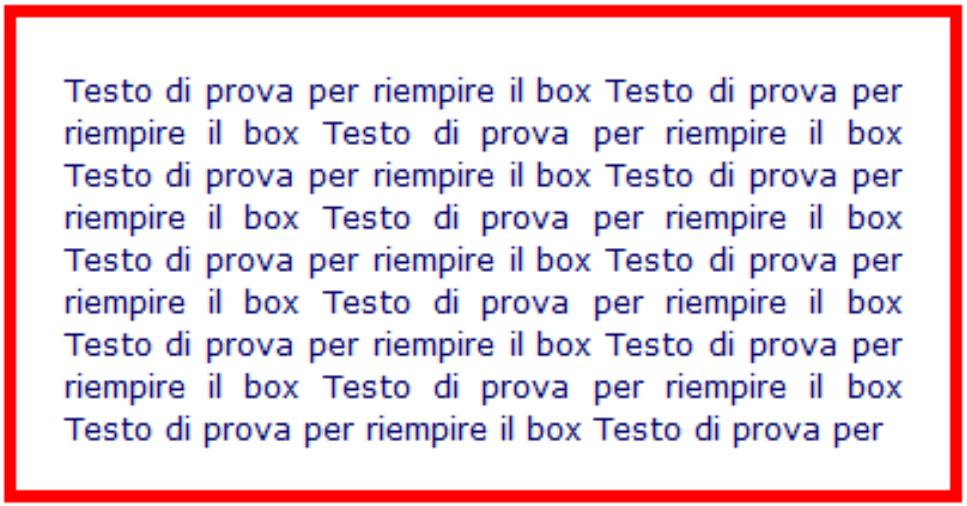
Praticamente le misure specificate (350 x 150) si riferiscono ai contenuti.

Box-Model: esempio 1

Questo il codice html per richiamarlo all'interno della pagina:

```
<div id="box">Testo di prova per riempire il box... </div>
```

Questo il risultato a video:



Bordi

- border-width – Ampiezza dei quattro bordi – 1-4 valori ()
- border-style – Quattro bordi – 1-4 valori – none, hidden, dotted, dashed, solid, double, groove, ridge, inset, outset
- border-color – Colore – 1-4 valori – color
- border – Imposta tutte le proprietà (uguali) per i quattro bordi
- border-top, border-right, … – Imposta tutte le proprietà per il bordo superiore, destro, …
- blockquote { border-left: 2px solid black; }

Border-style

Per border-style intendiamo la forma del bordo.

Anche da border-style derivano degli attributi per la gestione singola dei bordi:

border-top-style (per il bordo superiore)

border-right-style (per il bordo a destra)

border-bottom-style (per il bordo inferiore)

border-left-style (per il bordo sinistro)

- *none* (nessun bordo)
- *hidden* (nascosto)
- *dotted* testo
- *dashed* testo
- *solid* testo
- *double* testo
- *groove* testo
- *ridge* testo
- *inset* testo
- *outset* testo

Border-color

L'attributo border-color determina il colore del bordo.
Questo attributo può contenere il valore esadecimale del colore oppure il nome .

Se vogliamo modificare indipendentemente il colore dei bordi, possiamo utilizzare i seguenti attributi:
border-top-color (colore bordo superiore)
border-right-color (colore bordo a destra)
border-bottom-color (colore bordo inferiore)
border-left-color (colore bordo a sinistra)

Per utilizzare l'attributo border-color e gli altri 4 derivati è necessario definire un bordo, sia come spessore che come stile (esempio: border:5px solid)

Proprietà bordi

Property	Descrizione
border	Imposta tutte le proprietà del bordo in una dichiarazione
border-bottom	Imposta tutte le proprietà del bordo inferiore in una dichiarazione
border-bottom-color	Consente di impostare il colore del bordo inferiore
border-bottom-style	Consente di impostare lo stile del bordo inferiore
border-bottom-width	Consente di impostare la larghezza del bordo inferiore
border-color	Imposta il colore dei quattro bordi
border-left	Imposta tutte le proprietà del bordo di sinistra in una dichiarazione
border-left-color	Imposta il colore del bordo sinistro
border-left-style	Imposta lo stile del bordo sinistro
border-left-width	Imposta la larghezza del bordo sinistro
border-right	Imposta tutte le proprietà del bordo di destra in una dichiarazione
border-right-color	Consente di impostare il colore del bordo destro
border-right-style	Imposta lo stile del bordo destro
border-right-width	Consente di impostare la larghezza del bordo destro
border-style	Consente di impostare lo stile dei quattro bordi
border-top	Imposta tutte le proprietà del bordo superiore in una dichiarazione
border-top-color	Imposta il colore del bordo superiore
border-top-style	Imposta lo stile del bordo superiore
border-top-width	Consente di impostare la larghezza del bordo superiore
border-width	Imposta la larghezza dei quattro bordi

GESTIONE DELLE DIMENSIONI: ALTEZZA

In genere l'altezza di un elemento è determinata dal suo contenuto. Più testo inserisco in box, in un paragrafo o in una cella di tabella più essi saranno estesi in senso verticale. Le proprietà che analizzeremo servono in pratica a fornire un meccanismo in grado di controllare o superare in qualche modo questo comportamento standard.

1) height

Questa proprietà definisce la distanza tra il bordo superiore e quello inferiore di un elemento.

Sintassi

selettore {height: <valore>;}

Valori:

- **un valore numerico con unità di misura.**
- **un valore in percentuale**.
- **auto** (L'altezza sarà quella determinata dal contenuto).

GESTIONE DELLE DIMENSIONI: LARGHEZZA

La definizione della larghezza è in genere più problematica rispetto alla gestione dell'altezza.

1) width

Abbiamo già avuto modo di vedere come le dimensioni orizzontali di un elemento sono date dalla combinazione di varie proprietà. Un errore macroscopico quindi è ritenere che essa sia definita semplicemente dalla proprietà **width**. In pratica, bisogna sempre distinguere tra la larghezza effettiva di un elemento (i pixel di spazio che occupa sulla pagina) e la larghezza dell'area del contenuto. La prima è data dalla somma di questi valori: (margine sinistro + bordo sinistro + padding sinistro + area del contenuto + padding destro + bordo destro + margine destro)

La seconda è impostata tramite la proprietà **width**.

GESTIONE DELLE DIMENSIONI: LARGHEZZA /2

Fatta questa fondamentale premessa, possiamo analizzare nei dettagli la proprietà.

Sintassi

selettore {width: <valore>;}

Valori:

- **auto**. Valore 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.

I MARGINI /1

- Abbiamo già definito il margine come la distanza tra il bordo di un elemento e gli elementi adiacenti.
- Sono cinque le proprietà con cui è possibile definire un margine. Quattro di esse sono singole e impostano la distanza per ciascun lato del box. L'ultima, **margin,** è una proprietà a sintassi abbreviata utile per definire con una sola dichiarazione tutte le impostazioni per i quattro lati.

I MARGINI /2

1) margin-bottom
Definisce la distanza tra il lato inferiore di un elemento e gli elementi adiacenti.
2) margin-left
Definisce la distanza tra il lato sinistro di un elemento e gli elementi adiacenti.
3) margin-top
Definisce la distanza tra il lato superiore di un elemento e gli elementi adiacenti.
4) margin-right
Definisce la distanza tra il lato destro di un elemento e gli elementi adiacenti.

I MARGINI /3

5) margin

E' una proprietà a sintassi abbreviata. Con essa è possibile specificare i valori per tutti e quattro i lati di un elemento.

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

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

I MARGINI /4

Nella definizione dei valori, inoltre, è possibile mischiare percentuali con valori assoluti in unità di misura.
Un ulteriore abbreviazione della sintassi si può ottenere usando tre, due o un solo valore.

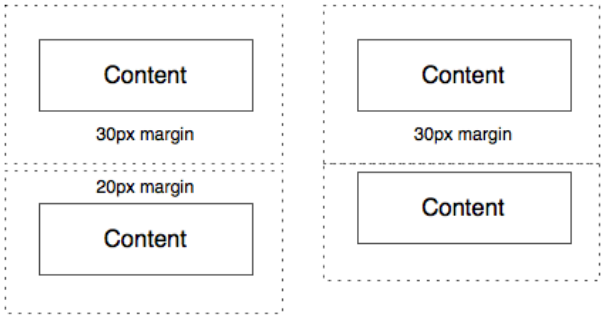
Queste le regole:

- se si usano tre valori, il primo si riferisce al margine superiore, il secondo a quelli sinistro e destro, il terzo a quello inferiore
- se si usano due valori, il primo si riferisce ai lati superiore e inferiore, il secondo al sinistro e al destro
- se si usa un solo valore, un uguale distanza sarà ai quattro lati

Un uso interessante del valore **auto** per i lati sinistro e destro è quello che consente di centrare in tal modo un elemento rispetto alla pagina o al box contenitore.

I MARGINI /5

Impostare margini negativi
Una importante particolarità dei margini è che queste proprietà accettano anche **valori negativi**, ad esempio: div.box { margin-left: -35px; }Grazie a questa importante possibilità è possibile "spostare" gli elementi della pagina facendoli uscire, ad esempio, dai confini del box genitore.
Tale facoltà si rivela estremamente interessante, ad esempio, quando si lavora con elementi flottanti o posizionati in modo assoluto.
E' appena il caso di ricordare che i *padding*, a differenza dei margini, non possono essere negativi.
Margin collapsing
Una caratteristica particolare dei margini è quella di "collassare" quando due box impilati entrano a contatto tra loro (da qui l'espressione *margin collapsing*). Questo significa che quando due elementi, ciascuno con il proprio margine, entrano a contatto, lo spazio tra loro non sarà dato (come si può pensare) dalla somma dei rispettivi margini ma dal margine maggiore (mentre l'altro margine, in un certo senso, viene annullato).
Facciamo un esempio: si supponga di avere due box uno sopra l'altro e che il box superiore abbia 30 px di margine inferiore ed il box sottostante abbia 20 px di margine superiore.
In questa situazione la distanza tra i due box non sarà di 50 px, ma di 30 in quanto il margine maggiore assorbe quello minore che, appunto, collassa.



E' molto importante precisare che questo effetto vale solo per i margini di box adiacenti in senso verticale mentre non si applica ai margini di box adiacenti sulla linea orizzontale.

IL PADDING

Un altro modo per creare spazio intorno ad un elemento è quello di usare il padding. Quando si usa il padding, lo spazio di distanza viene inserito al di qua dei bordi dell'elemento e non all'esterno. Nel caso del padding, lo spazio inserito avrà proprio il colore di sfondo dell'elemento, a differenza dei margini.
Un'analogia rispetto ai margini sta nella sintassi. Anche qui quattro proprietà singole per i lati e una a sintassi abbreviata (**padding**).

1) padding-bottom

Imposta l'ampiezza del padding sul lato inferiore di un elemento.

IL PADDING

2) padding-left
Imposta l'ampiezza del padding sul lato sinistro di un elemento.
3) padding-top
Imposta l'ampiezza del padding sul lato superiore di un elemento.
4) padding-right
Imposta l'ampiezza del padding sul lato destro di un elemento
5) padding
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 **margin**.

Posizionare il Box Model

un box è praticamente un contenitore che inserito in un punto qualsiasi della pagina ne segue il suo flusso naturale generando un ritorno a capo come tutti gli elementi di blocco, ne consegue che più box sarebbero disposti verticalmente uno di seguito all'altro.
A causa di questa caratteristica diventa impossibile creare dei layout elaborati usando più box senza fare uso di una proprietà dei fogli di style dal nome **position** che permette di posizionare il box in modo diverso da quello *naturale*.

Gli attributi ammessi per la proprietà **position** sono: *static*, *relative*, *absolute* e *fixed*.

Posizionare il Box Model

static è il posizionamento predefinito, quello naturale che segue il flusso nella disposizione della pagina. Il suo posizionamento è nel punto preciso in cui viene creato o richiamato.

```
#box1
 {
width: 400px;
height: 20px;
border: solid 1px;
color: black;
background-color: #ffff99;
}
```

Posizionare il Box Model

Il codice si riferisce al box1 ma la definizione per gli altri due box è praticamente la stessa, varia soltanto per il nome: #box2 e #box3 ed il relativo colore di sfondo: background-color.

```
<div id="box1">blocco 1 inserito nel normale flusso </div>
<div id="box2">blocco 2 inserito subito dopo l'istruzione per il blocco 1</div> <div
id="box3">blocco 3 inserito subito dopo l'istruzione per il blocco 2</div>
```

Producono questo risultato:

blocco 1 inserito nel normale flusso

blocco 2 inserito subito dopo l'istruzione per il blocco 1

blocco 3 inserito subito dopo l'istruzione per il blocco 2

Posizionare il Box Model /2

Notate come, nella diapositiva precedente, i blocchi inseriti nella pagina seguono il flusso naturale disponendosi verticalmente pur non avendo inserito alcun ritorno a capo, questo perchè il tag , così come tutti gli elementi di blocco, provoca automaticamente un ritorno a capo.

Non avendo impostato alcun **margin** i bordi risultano incollati fra loro.

L'esempio sotto è lo stesso box con **margin** impostato a 5 pixel. Ricordate quanto detto precedente riguardo al box model? si fa riferimento al box interno, per cui i 5 pixel che distanziano i vari box sono 3 pixel più i 2 pixel esterni che fanno da bordo.

blocco 1 con margin inserito nel normale flusso

blocco 2 con margin inserito subito dopo l'istruzione per il blocco 1

blocco 3 con margin inserito subito dopo l'istruzione per il blocco 2

Posizionare il Box Model /3

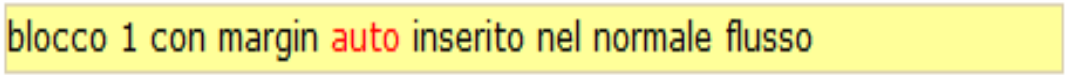
questo il codice per definirlo nel foglio di style CSS

```
#box1 {
width: 400px;
height: 20px;
 border: solid 1px;
 margin: 5px;
color: black;
background-color: #ffff99;
 }
```

Il codice si riferisce al box1 ma la definizione per gli altri due box è praticamente la stessa, varia soltanto per il nome: #box2 e #box3 ed il relativo colore di sfondo: background-color.

Posizionare il Box Model /4

L'attributo **margin** accetta anche come valore **auto** che posizionerebbe il blocco al centro del flusso visto che assegnerebbe automaticamente lo stesso margine ad entrambi i lati.

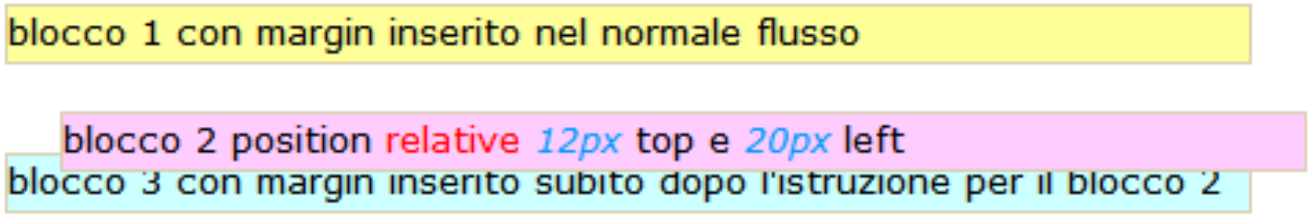


Da notare che ad essere centrato è il blocco e non il suo contenuto, da notare anche che per avere un bordo nel foglio di style non basta specificare il solo spessore (*1px*), si deve specificare anche un possibile attributo che identifica il tipo di bordo usato, in questo caso *solid*, diversamente il bordo non risulterà visibile.
Quando il contenuto del box supera le sue dimensioni, per esempio inserendo un'immagine più larga del box definito, si hanno comportamenti differenti a seconda del tipo di browser, tanto per cambiare, dove ognuno di questi interpreta a modo suo. Alcuni allargano il box per adattarlo alla dimensione del suo contenuto, altri fanno uscire l'immagine dal box.

E' possibile regolare questo comportamento al fine di unificarlo su tutti i browser grazie alla proprietà **overflow** che accetta i seguenti possibili attributi: ***visible*****:** predefinito, ***hidden*****:** nascosto, ***auto*****:** appaiono le barre di scorrimento soltanto se necessario, ***scroll*****:** appaiono le barre di scorrimento sempre, anche se non necessitano.

Posizionare il Box Model Relative /1

relative è il posizionamento modificato rispetto alla posizione predefinita, praticamente un *offset* che consente di spostare il box rispetto alla sua naturale origine, senza che questo spostamento incida o alteri il posizionamento degli altri elementi (oggetti) facenti parte della stessa pagina.



Notate come il *blocco 2* sia spostato di **12** pixel a partire dal margine alto e di **20** pixel a partire dal margine sinistro rispetto alla sua posizione naturale, senza che gli altri due box subiscano alcuna variazione di posizione. Questo perché **position relative** agisce solo sul box che ne fa uso senza alterare il normale flusso di tutto il resto

Posizionare il Box Model Relative /2

E' possibile specificare la posizione **relative** usando gli attributi *top*, *left*, *bottom* e *right*. Rispettivamente per posizionarlo a partire dall'*alto*, da *sinistra*, dal *basso* e da *destra*. Sono ammessi anche **numeri negativi** che posizionerebbero il blocco in direzione **contraria** a quella di numeri positivi.

Questo il codice nel css per definirlo.

```
#box2 {
position: relative;
top: 12px; left: 20px;
width: 450px;
height: 20px;
border: solid 1px;
 margin: 5px;
color: black;
background-color: #ffccff;
}
```

Nell'esempio sopra, il blocco 2 risulta sovrapposto al blocco 3 oscurandolo in parte, è possibile stabilire l'ordine di sovrapposizione con la proprietà **z-index** che accetta solo valori numerici positivi, il numero più alto è quello in primo piano e copre il numero più basso.

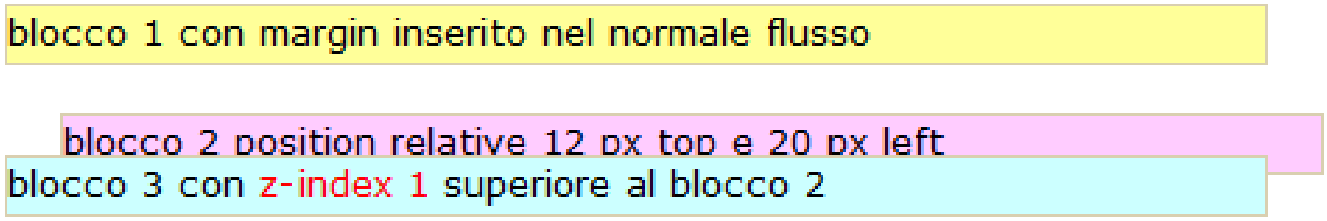
Posizionare il Box Model Relative /3

Adesso facciamo in modo che sia il blocco 3 a risultare sovrapposto al blocco 2 perchè gli viene assegnato un valore **z-index** di *1* che è superiore al valore dello z-index *0* (di default è impostato a 0) del blocco 2.

questo il codice nel css per definirlo.

```
#box3 {
 position: relative;
z-index: 1;
width: 450px;
height: 20px;
border: solid 1px;
margin: 5px; color: black;
 background-color: #ccffff;
 }
```

Questo il risultato



Posizionare il Box Model absolute

absolute è il posizionamento assoluto, cioè è possibile posizionare il blocco in un qualsiasi punto della pagina, a differenza del posizionamento relativo che abbiamo visto non alterava gli altri oggetti, pur condizionandoli con la sua presenza, **absolute** farà in modo che il blocco così definito risulti come se non facesse parte della pagina, praticamente non inciderà in alcun modo con gli altri oggetti, una specie di sovrapposizione alla pagina stessa.

E' possibile specificare la posizione **absolute** usando gli stessi attributi visti per relative *top*, *left*, *bottom* e *right*. Rispettivamente per posizionarlo a partire dall'*alto*, da *sinistra*, dal *basso* e da *destra*. Questa volta però il punto di riferimento è al primo blocco progenitore non statico e se non esiste, all'elemento <html>. Praticamente i margini della finestra del browser.

Se non viene specificato alcun valore di posizionamento il blocco assumerà valore assoluto riferendosi al punto in cui viene inserito dal normale flusso del codice. Su questa pagina è possibile vedere un esempio che chiarirà tutto quanto in modo più semplice.

Posizionare il Box Model fixed

Fixed non supportato dai browser IE è simile al posizionamento assoluto ma il riferimento è sempre e solo alla finestra del browser, quando la pagina scorre i vari box così definiti restano fissi nella loro posizione specificata.

Posizione

- bottom – Distanza del limite inferiore di un elemento rispetto all'elemento che lo contiene – auto, %, length
- left – Sinistra … – auto, %, length
- right – Destra … – auto, %, length
- top – Superiore – auto, %, length
- vertical-align – Allineamento verticale – top, middle, bottom, baseline, sub, super, text-top, text-bottom, length, %
- position – Modalità di posizionamento – static, relative, absolute, fixed, inherit
- float – Elementi posizionati a lato, circondati da testo – left, right, none, inherit

Visibilità

z-index – Ordine nello stack (rilievo, valori più alti sono in primo piano) – auto, number
overflow – Impostazione per contenuto più ampio dell'area disponibile – visible, hidden, scroll, auto
visibility – Visibilità – visible, hidden (occupa spazio)
display – Visualizzazione – none (non occupa spazio), block, inline, …

Border-radius

Con la proprietà **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**

border-top-left-radius

Definisce l'arrotondamento dell'angolo superiore sinistro di un elemento.
I valori possono essere espressi da un numero accompagnato da un'unità di misura o in percentuale.
È possibile definire uno o due valori. Se si definiscono due valori diversi, il primo imposta la misura del raggio orizzontale, il secondo quello del raggio verticale. L'uso di due valori diversi consente di ottenere angoli ellittici. Se si usa un solo valore si applica la stessa dimensione al raggio orizzontale e a quello verticale, ottenendo angoli circolari.

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

#box2 {border-top-left-radius: 20px 10px}
```

border-top-right-radius

Definisce l'arrotondamento dell'angolo superiore destro di un elemento.

Per i valori valgono le stesse considerazioni viste in precedenza.

```
#box1 {border-top-right-radius: 20px}

#box2 {border-top-right-radius: 20px 10px}
```

border-bottom-right-radius

- Definisce l'arrotondamento dell'angolo inferiore destro di un elemento.
- Per i valori valgono le stesse considerazioni viste in precedenza.

```
#box1 {border-bottom-right-radius: 20px}

#box2 {border-bottm-right-radius: 20px 10px}
```

border-bottom-left-radius

- Definisce l'arrotondamento dell'angolo inferiore sinistro di un elemento.
- Per i valori valgono le stesse considerazioni viste in precedenza.

```
#box1 {border-bottom-left-radius: 20px}

#box2 {border-bottom-left-radius: 20px 10px}
```

border-radius

È una proprietà a sintassi abbreviata (shorthand) con cui è possibile definire con una sola regola i valori di arrotondamento per tutti e quattro gli angoli di un elemento. La proprietà accetta fino a quattro valori per ciascun raggio dell'angolo (quattro per il raggio orizzontale e quattro per quello verticale). I valori relativi al raggio verticale, se espressi, vanno separati da quelli relativi al raggio orizzontale con una slash (**/**).
Il funzionamento di questa proprietà segue il principio comune nei CSS per le proprietà shorthand, lo stesso che ritroviamo nella definizione dei margini o del padding.
Dunque, se si imposta un solo valore, esso sarà applicato a tutti e quattro gli angoli; se ne usiamo due, il primo sarà applicato all'angolo superiore sinistro e all'angolo inferiore destro, il secondo all'angolo superiore destro e all'angolo inferiore sinistro, etc. Definendo quattro valori, l'ordine in cui sono applicati parte dall'angolo superiore sinistro e segue il senso orario.

border-radius esempio 1

```
#box {border-radius: 20px}
```

Essa equivale a:

```
#box {
border-top-left-radius:     20px;
border-top-right-radius:    20px;
border-bottom-right-radius: 20px;
border-bottom-left-radius:  20px;
}
```

border-radius esempio 2

```
#box {border-radius: 20px 40px}
```

equivale a:

```
#box {
border-top-left-radius:     20px;
border-top-right-radius:    40px;
border-bottom-right-radius: 20px;
border-bottom-left-radius:  40px;
}
```

border-radius esempio 3

```
#box {border-radius: 20px 40px 60px 80px}
```

che equivale a:

```
#box {
border-top-left-radius:     20px;
border-top-right-radius:    40px;
border-bottom-right-radius: 60px;
border-bottom-left-radius:  80px;
}
```

Float

Nelle intenzioni di chi ha creato i CSS probabilmente non doveva essere così, ma la proprietà **float**, in mancanza di meccanismi dedicati esclusivamente a questo scopo, è diventata nella pratica comune la base su cui costruire layout basati sui CSS. Insieme a **clear** rappresenta, insomma, un elemento essenziale dei CSS.

float
Con questa proprietà è possibile rimuovere un elemento dal normale flusso del documento e spostarlo su uno dei lati (destro o sinistro) del suo elemento contenitore. Il contenuto che circonda l'elemento scorrerà intorno ad esso sul lato opposto rispetto a quello indicato come valore di float. La proprietà non è ereditata.

La sintassi generica è questa:

selettore {float: valore;}

Float /2

float può assumere questo valori:
left l'elemento viene spostato sul lato sinistro del box contenitore, il contenuto scorre a destra
right l'elemento viene spostato sul lato destro, il contenuto scorre a sinistra
none valore iniziale e di default in mancanza di una dichiarazione esplicita; l'elemento mantiene la sua posizione normale

Clear

La proprietà **clear** serve a impedire che al fianco di un elemento compaiano altri elementi con il float. Si applica solo agli elementi **blocco** e non è ereditata.
L'origine di tale proprietà è questa: visto che il **float** sposta un elemento dal flusso normale del documento, è possibile che esso venga a trovarsi in posizioni non desiderate, magari al fianco di altri elementi che vogliamo invece tenere separati. clear risolve questo problema.

Questa la sintassi di base:
selettore {clear: valore;}

Clear /2

I valori possibili sono:
Valore Descrizione
none gli elementi con float possono stare a destra e sinistra dell'elemento;
left si impedisce il posizionamento a sinistra;
right si impedisce il posizionamento a destra;
both si impedisce il posizionamento su entrambi i lati.

```
h1 {clear: both;}
```