

## LAYOUTS

Abbiamo già visto nei capitoli precedenti come creare una cella allineata ad un paragrafo ([vedi](#)) o come sia possibile creare dei menù usando le proprietà delle tabelle ([vedi](#)) o del Tag DIV ([vedi](#)).

Il Tag DIV è molto usato per realizzare dei layer, il cui posizionamento è relativo al Tag contenitore, che a sua volta può essere il BODY oppure no.

### Tipi di Posizionamento

- **position:static**  
L'elemento è posizionato secondo il normale svolgimento del codice HTML all'interno della pagina, senza variazioni. È il valore di default.
- **position:relative**  
Il contenitore SPAN viene spostato rispetto al suo naturale posizionamento (cioè senza definizione della posizione tramite il "position") all'interno della pagina, secondo una misura variabile indicata dai valori espressi nelle proprietà di posizionamento ( *top:50; left:50;* ).
- **position:absolute**  
Le coordinate ( *top:50; left:50;* ) indicano la posizione variata rispetto al box genitore, che nell'esempio è il Tag TABLE e non il BODY. Così, in questo esempio, la scritta non si vedrà più all'interno della seconda cella, bensì in un'altra posizione.
- **position:fixed**  
L'elemento è fissato all'interno della pagina: le coordinate in questo caso fanno realmente riferimento alla finestra principale del browser e l'elemento è fermo nel punto indicato dalle coordinate. Questa proprietà non è supportata da Internet Explorer.

È inoltre possibile inserire dei livelli annidati l'uno dentro l'altro; per ottenere ad esempio una struttura di pagina che si ridimensioni al variare della risoluzione dello schermo, basterebbe quindi definire un livello in posizione relativa nella pagina ( che si sposterà secondo il normale andamento del codice ), e al suo interno annidare poi uno o più livelli in posizione assoluta con le debite variazioni di posizionamento per ottenere la visualizzazione desiderata.

### Il "box model"

Ovvero: sapere come ciascuno dei blocchi che formano una pagina possa essere impostato secondo le sue principali proprietà: larghezza e altezza, margini e bordi, padding e colori di sfondo.

### La proprietà float

Il floating è un altro modo per definire la posizione di un elemento.

*Cosa succede quando applico questa proprietà?*

Semplificando, l'elemento viene rimosso dal normale flusso dei dati e viene portato all'estremità sinistra ( **FLOAT: left;** ) o a destra ( **FLOAT: right;** ) del suo box contenitore. Il testo eventualmente presente fuori dall'elemento scorre intorno ad esso. Ecco un [esempio](#).

### Impaginare tenendo presente la risoluzione del monitor

Un problema frequente in cui si imbattono i webmaster principianti è quello della risoluzione del monitor: di solito infatti incominciano a sviluppare il loro primo sito prendendo come riferimento soltanto la risoluzione del proprio monitor, il browser che stanno utilizzando in quel momento.

L'utente invece può avere una risoluzione del monitor ( 800x600, 1024x768, 1280x1024 ) che non rispetta minimamente la configurazione del monitor del webmaster; infine il browser o il sistema operativo possono essere i più svariati.

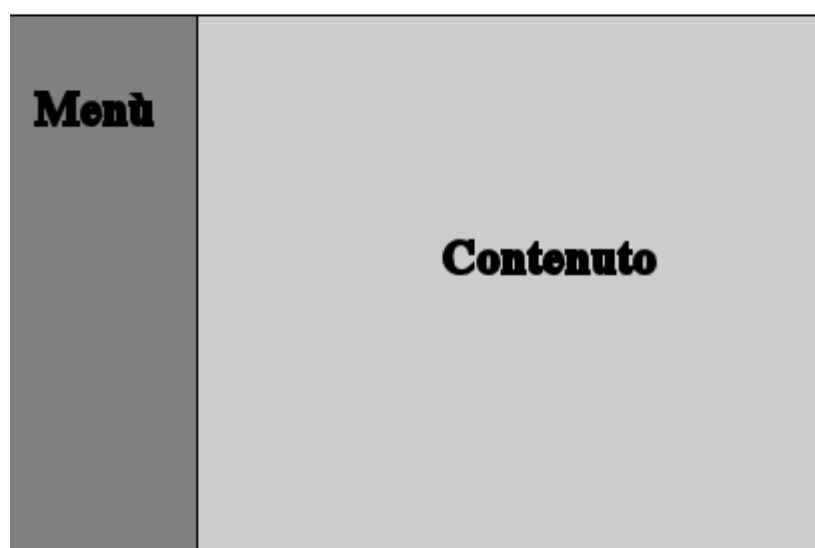
Progettare per il web significa quindi tenere conto di tutti questi fattori: magari si può ignorarli ( volutamente ), ma bisogna sapere comunque che esistono.

Nel caso della [risoluzione del monitor](#) il webmaster ha sostanzialmente le seguenti possibili scelte di impaginazione:

1. *Ignorare il problema e allineare il sito sulla sinistra della pagina:*  
A risoluzioni del monitor superiori di quella per cui si sta sviluppando (es: 1024x768) la pagina presenta uno spazio vuoto sulla destra.
2. *Centrare il sito:*  
A risoluzioni superiori rispetto a quelle per cui sono progettate, verranno visualizzate due colonne vuote (rispettivamente a sinistra e a destra).
3. *Impostare la pagina al 100% in altezza ed in larghezza:*  
In questo modo non avrei alcun problema per l'aspetto della risoluzione, ma è sconsigliabile per siti con contenuti molto lunghi, a meno che non si adottino determinati accorgimenti ( uso di DIV o iFRAME interni alla pagina ).
4. *Impostare la pagina solo al 100% in larghezza:*  
In questo modo non avrei alcun problema per l'aspetto della risoluzione, ed a differenza della soluzione precedente, la pagina diventerà scrollabile verticalmente. E' quindi molto usata per siti con contenuti molto lunghi.

Vediamo ora degli esempi, per illustrare la differenza tra tutti questi Layouts:

## Header



Supponiamo di voler realizzare una pagina simile a quella superiore, allora possiamo impostare le seguenti strutture di pagina:

	800x600	>= 1024x768
Allineare la pagina a sinistra <a href="#">esempio</a>	La pagina si vede a tutto schermo.	La pagina si vede allineata a sinistra con una colonna a destra vuota.
Allineare la pagina al centro <a href="#">esempio</a>	La pagina si vede a tutto schermo.	La pagina si vede allineata al centro con una colonna vuota sia destra e sia a sinistra.
Impostare il 100% sia in altezza e sia in larghezza <a href="#">esempio</a>	La pagina si vede a tutto schermo.	La pagina si vede a tutto schermo.
Impostare il 100% solo in larghezza	La pagina si vede a tutto schermo, ma scrollabile	La pagina si vede a tutto schermo, ma scrollabile verticalmente.

esempio	verticalmente.	
---------	----------------	--

Quanto sopra mostrato, è stato realizzato tramite l'uso di Tabelle, ma un Layout può essere realizzato anche con il Tag DIV; è sufficiente infatti inserire un livello (coincidente con un tag DIV) in posizione relativa nella pagina (che si sposterà secondo il normale andamento del codice), e al suo interno annidare poi uno o più livelli in posizione assoluta con le debite variazioni di posizionamento per ottenere la visualizzazione desiderata. In questo modo faremo sì che i livelli mantengano una normale posizionamento all'interno della pagina (evitando nel contempo le difficoltà che avremmo utilizzando il posizionamento relativo per ogni singolo livello). Ad esempio con il seguente codice, otterrei il seguente [esempio](#):

```
<table style="width:100%; height:100px">
<tr>
<td style="width:60%; border:1px solid red"></td>
<td style="width:40%; border:1px solid red; vertical-align:top">
<div style="position:relative">
<span style="position:absolute; top:0; left:0; width:100px; z-index:2;
background-color:blue"> primo livello </span>
<span style="position:absolute; top:0; left:50px; padding-left:70px; width:250px;
z-index:1; background-color:red"> secondo livello </span>
</div>
</td>
</tr>
</table>
```

Vediamo ora come realizzare il [layout a 3 colonne con il Tag DIV](#):

	800x600	>= 1024x768
Allineare la pagina a sinistra <a href="#">esempio</a>	La pagina si vede a tutto schermo.	La pagina si vede allineata a sinistra con una colonna a destra vuota.
Allineare la pagina al centro <a href="#">esempio</a>	La pagina si vede a tutto schermo.	La pagina si vede allineata al centro con una colonna vuota sia destra e sia a sinistra.

La larghezza dei vari DIV può essere fissata anche in %, ottenendo così un layout perfettamente liquido con la risoluzione. Come avete potuto notare, l'altezza delle due colonne laterali è data dal loro contenuto; questo può creare un problema nel caso in cui il colore di sfondo della pagina sia diverso da quello delle due colonne e del Div contenuto. Per risolvere, quindi, tale problema in modo parziale, si può inserire un colore di sfondo al Body.

**N.B:** Per il div #testata/#banner, visto che l'elemento parente è body, la larghezza sarà sempre uguale a quella di quest'ultimo elemento. In pratica, usate *width* solo quando volete specificare una misura e ricordate che la larghezza massima di un elemento non può essere superiore a quella dell'elemento parente.

Vediamo ora come realizzare il [layout a 2 colonne con il Tag DIV TOTALMENTE LIQUIDO](#):

	800x600	>= 1024x768
<a href="#">esempio</a>	La pagina si vede a tutto schermo.	La pagina si vede a tutto schermo, poiché il DIV Contenuto si ridimensiona in base al suo contenuto e alla risoluzione stessa.