

[.ConStile](#), una **guida** dedicata all'uso dei **CSS** per lo sviluppo di siti web leggeri, usabili, conformi agli **standard W3C**

---

[\[H\] Home](#) > [Tutorial](#) > [Introduzione ai css](#) > Introduzione ai CSS : Indice

# Introduzione ai CSS

I Cascade Style Sheets (CSS), noti in Italia come "Fogli di stile a cascata", sono LO strumento [definito](#) dal W3C per definire l'aspetto delle pagine web.

Prima dell'introduzione dei CSS, le opportunità per i webmaster di definire l'aspetto delle pagine web erano limitate a tag HTML (ad esempio il tag FONT) e si basavano sull'uso di tabelle per la definizione del layout. Il limite alla diffusione dei CSS è stato, inizialmente, lo scarso supporto dei CSS da parte dei browser più diffusi. A distanza di anni, i browser hanno finalmente maturato una buona capacità di interpretare i CSS e il limite al loro utilizzo è stato esclusivamente, la pigrizia e l'incapacità dei webmaster, come pure la non conoscenza dei [vantaggi offerti dai CSS](#).

Questo sito ha l'obiettivo di mostrare le potenzialità dei CSS e di aiutare i webmaster italiani ad apprendere tecniche e soluzioni per l'uso avanzato dei CSS.

Lo scopo di questo articolo è fornire le basi essenziali per la comprensione dei CSS e degli articoli pubblicati su questo sito.

## Indice

- [1. Cosa sono i fogli di stile](#)
- [2. La sintassi delle regole](#)
- [3. I selettori](#)
  - [3.1 Tag HTML](#)
  - [3.2 Classi](#)
  - [3.3 Identificatori](#)
  - [3.4 Pseudo-classi e Pseudo-elementi](#)
  - [3.5 Selettori in cascata](#)
  - [3.6 Raggruppamento](#)
- [4. Le proprietà](#)
  - 4.1 Font:
    - [font-family](#)

- [font-style](#)
- [font-variant](#)
- [font-weight](#)
- [font-size](#)
- [font](#)
- 4.2 Colore e Sfondo:
  - [color](#)
  - [background](#)
- 4.3 Testo:
  - [word-spacing](#)
  - [letter-spacing](#)
  - [line-height](#)
  - [text-indentation](#)
  - [vertical-align](#)
  - [text-decoration](#)
  - [text-transform](#)
  - [text-align](#)
  - [white-space](#)
- 4.4 Box model:
  - [margin](#)
  - [padding](#)
  - [border](#)
  - [width](#)
  - [height](#)
  - [float](#)
  - [clear](#)
- 4.5 Classificazione:
  - [display](#)
  - [list-style](#)
- 4.6 Lunghezze, percentuali, colori, url:
  - [lunghezze](#)
  - [percentuali](#)
  - [colori](#)
  - [url](#)
- [5. Associare i fogli di stile ai documenti \(X\)HTML](#)
- [6. Note](#)

*G.T.* -- ultima revisione: 03.10.2002

l'indirizzo di questa pagina è: [http://www.constile.org/tutorial/introduzione\\_ai\\_css/index.php](http://www.constile.org/tutorial/introduzione_ai_css/index.php)

[.ConStile](#), una **guida** dedicata all'uso dei **CSS** per lo sviluppo di siti web leggeri, usabili, conformi agli **standard W3C**

---

[\[H\] Home](#) > Homepage

## Editoriale:

[24.02.2003] **Questo è un sito minimalista e non me ne vergogno.** Qui non troverete inutili animazioni, flash, immagini decorative (almeno non nel codice XHTML).

**I contenuti sono il fulcro essenziale di queste pagine.** Questi contenuti sono disponibili a tutti, indipendentemente dal dispositivo web che si utilizza per navigare in Internet, indipendentemente dalle differenti abilità degli utenti. Chiunque è il benvenuto. Nessuna discriminazione.

**Forse queste pagine non sono eleganti**, forse sono anche poco originali, alcuni le troveranno non belle. A me piacciono così: navigabili con uno smartphone dotato di browser (compresi quei gioiellini realizzati in Java), come con l'ultima versione di Explorer che gira su un PC.

25 Gennaio 2003 — **vol.2 n.18**

Layout a tre colonne per  
tutti i gusti

In questo articolo vedremo come realizzare un layout a tre colonne sfruttando il posizionamento assoluto invece della proprietà **float**. Attraverso tre diversi CSS, col medesimo codice XHTML, otterremo tre diversi tipi di layout: a larghezza prefissata, liquido con colonne laterali di larghezza prefissata, completamente liquido. [ [Leggi l'articolo](#) ]

## Elenco di tutti gli articoli pubblicati

### Tutorial:

[Come formattare un articolo](#) / [Dimensionare i caratteri con i CSS](#) / [Pagine pronte per la stampa con i CSS](#) / [Progettare con i CSS](#) / [Il box model di IE5/Win](#) / [CSS e tabelle a confronto](#) /

### Clonazioni:

[Virgilio con i CSS](#) /

## Template:

[Layout a tre colonne per tutti i gusti](#) / [Come porre un box al centro della pagina](#) / [Layout di base a tre colonne](#) / [Menu verticali con i CSS](#) /

## Tips:

[Emulare gli IFRAME con i CSS](#) / [Nascondere i CSS](#) / [Diversi media, diversi stili](#) / [Ripristino in seguito all'utilizzo della proprietà float](#) /

## DHTML:

[Skin alternative in JS/CSS](#) / [Menu ad albero in JS/CSS](#) / [Ridimensionare i caratteri con un click](#) /

## Articoli essenziali

[99.9% of Websites are Obsolete](#): Il 99% dei siti web è obsoleto. / [Progettare con i CSS](#): I punti chiave per progettare un sito web basato sui fogli di stile. / [CSS e tabelle a confronto](#): Vantaggi e svantaggi di tabelle e CSS. / [Introduzione ai CSS](#): Le basi essenziali per la comprensione degli articoli pubblicati su questo sito. / [Il box model e l'errata interpretazione di IE5/Win](#): Il problema e le possibili soluzioni.

l'indirizzo di questa pagina è: <http://www.constile.org/index.php>

[.ConStile](#), una **guida** dedicata all'uso dei **CSS** per lo sviluppo di siti web leggeri, usabili, conformi agli **standard W3C**

---

[\[H\] Home](#) > [Tutorial](#) > Elenco dei tutorial pubblicati

# Elenco dei tutorial pubblicati

Guide e consigli per la realizzazione di siti web tramite i CSS. Di seguito è riportato l'elenco dei tutorial pubblicati in ordine cronologico.

- [Come formattare un articolo](#): In questo articolo vedremo come formattare un articolo rivolgendo particolare attenzione ai paragrafi, alle immagini e alle citazioni.
- [Dimensionare i caratteri con i CSS](#): I CSS vengono spesso utilizzati per dimensionare i caratteri, ma se male utilizzati possono peggiorare l'accessibilità di un sito. Una piccola guida su come dimensionare i caratteri rispettando l'accessibilità del sito.
- [CSS e tabelle a confronto](#): In questo articolo, dopo una breve introduzione sul perchè sia errato utilizzare le tabelle per impostare la grafica di un sito web, saranno messi a confronto i vantaggi e gli svantaggi dei CSS con i vantaggi e gli svantaggi delle tabelle.
- [Pagine pronte per la stampa con i CSS](#): Le pagine web, spesso pensate solo per la visione su schermo, possono risultare inadatte alla stampa. In questo articolo vedremo come, attraverso i CSS, sia possibile impostare l'aspetto di una pagina quando viene stampata.
- [Il box model di IE5/Win](#): Il box model dei CSS permette di definire dei blocchi rettangolari con specifici valori per larghezza, altezza, padding, bordi e margini. Purtroppo, IE5 per Windows non ha una interpretazione corretta del box model. Come risolvere il problema.
- [Progettare con i CSS](#): Progettare un sito web, il cui layout debba basarsi sui CSS, impone uno schema logico diverso da quello solitamente usato. In questo articolo saranno brevemente illustrati i punti chiave per progettare un sito web "con stile".

l'indirizzo di questa pagina è: <http://www.constile.org/tutorial/index.php>

[.ConStile](#), una **guida** dedicata all'uso dei **CSS** per lo sviluppo di siti web leggeri, usabili, conformi agli **standard W3C**

---

[\[H\] Home](#) > [Tutorial](#) > [Css vs table](#) > CSS e tabelle a confronto

**Articoli correlati:**

- [Progettare con i CSS](#)

**Strumenti**

- [Commenti sull'articolo](#)
- [Stampa l'articolo](#)

# CSS e tabelle a confronto

In questo articolo, dopo una breve introduzione sul perchè sia errato utilizzare le tabelle per impostare la grafica di un sito web, saranno messi a confronto i vantaggi e gli svantaggi dei CSS con i vantaggi e gli svantaggi delle tabelle.

## Tra contenuti e forma

Agli inizi, quando Internet era usato come mezzo di comunicazione fra scienziati e ricercatori, il **contenuto informativo** era l'aspetto più importante della pagina web. Per aiutare i ricercatori a riportare in maniera organizzata dati sperimentali, furono create le **tabelle** che, per loro stessa natura, servivano a **intabellare dei dati**, basti pensare al tag <td> dove l'acronimo td sta per **table data**.

Le tabelle sono però in grado di contenere molti tipi di informazioni, non solo parole e numeri, come immagini e altre tabelle. Tra i webmaster si diffuse dunque la **cattiva abitudine** di utilizzare le tabelle per la realizzazione del layout di un sito, tradendo quello che era lo scopo della loro creazione. Annidando tabelle dentro tabelle è possibile realizzare impaginazioni più o meno complesse. La cosa andò peggiorando con l'introduzione di software per la realizzazione di pagine web del tipo WYSIWYG (What You See Is What You Get: ciò che vedi è ciò che otterrai) che si basano quasi esclusivamente sull'uso delle tabelle. Il webmaster si trova di fronte a un'interfaccia grafica e "disegna" il layout del sito senza accorgersi che il software sta creando un codice che, nella migliore delle ipotesi è **ridondante e disorganizzato**, nella maggior parte dei casi risulta anche **errato**. Si ottengono così pagine pesanti (in termini di dimensione dei files) e lente da scaricare ed elaborare, spesso non utilizzabili da dispositivi come i PDA ovvero da utenti non vedenti che utilizzano screen reader, solo per fare alcuni esempi. Si pensi ad esempio ad utenti che utilizzano browser vocali o a linea di testo. Il testo sarà letto dall'inizio alla fine nell'ordine in cui compare nel codice, ed essendo il contenuto informativo indissolubilmente mescolato al codice per la grafica capita spesso che l'ordine in

cui sono lette le informazioni non sia corretto.

Per garantire una completa **accessibilità** dei contenuti, **la grafica deve essere separata dal testo**, le tabelle devono intabellare.

Per fornire un'impaginazione ai contenuti esiste una apposita tecnica, quella dei CSS (Cascading Style Sheets: fogli di stile a cascata), di cui ora mostreremo i vantaggi.

## CSS Vs <Table>

### I vantaggi dei CSS

- Accesso ai contenuti con qualsiasi dispositivo per la navigazione in Internet, anche per quelli non in grado di interpretare i CSS.
- Completa separazione fra contenuti e grafica.
- Possibilità di [organizzare il contenuto](#) in sezioni e sottosezioni.
- Codice (X)HTML semplice e pulito, privo delle ridondanze imposte dalle tabelle.
- File più leggeri, più veloci da scaricare e interpretare da parte del browser.
- Possibilità di realizzare layout più complessi di quelli possibili con le tabelle.
- Associazione di diverse impostazioni del layout per i diversi dispositivi come monitor, PDA, stampanti.
- Maggiore coerenza grafica fra le varie pagine di un sito.
- Possibilità di cambiare l'aspetto del sito, anche notevolmente, modificando solamente un file.
- Abbandono dei disastri compiuti dai software WYSIWYG, per approdare alla tecnica qui definita WYCIWYG (What You **Code** Is What You Get: ciò che **codifichi** è ciò che otterrai) che è l'unica che garantisce il **totale controllo** sul risultato finale.

### Gli svantaggi dei CSS

- I browser più datati hanno una non corretta interpretazione dei CSS, ciò richiede un minimo d'accortezza al momento della progettazione. Punto.

### I vantaggi delle tabelle

- Fanno funzionare i software WYSIWYG. Punto.

### Gli svantaggi delle tabelle

- Riducono l'accessibilità del sito.
- Impediscono di separare i contenuti dal modo in cui sono presentati
- Riducono, spesso impediscono, l'[organizzazione strutturata](#) dei contenuti.

- Codice confuso e ridondante.
- Notevole quanto inutile aumento del peso delle pagine.
- Minori possibilità grafiche.
- Un unico layout per tutti i dispositivi.
- Per ogni pagina è necessario ricreare il layout.
- Cambiare la grafica di un sito richiede spesso la totale ri-scrittura delle pagine, tutte.
- Abbandonare i disastrosi software WYSIWYG e continuare ad utilizzare le tabelle richiede uno sforzo davvero enorme.

## Un cambiamento di filosofia

Il confronto fra CSS e tabelle, che vede quest'ultime inequivocabilmente sconfitte, evidenzia la necessità di un cambiamento nel concepire la realizzazione della pagina web: prima vengono i contenuti e la loro [organizzazione strutturata](#), poi si realizza uno o più CSS che istruiscano il browser su come presentare le informazioni. La vecchia tecnica, disegno una griglia con le tabelle e realizzo la grafica e in seguito ci mescolo i contenuti appartiene al passato e non può più condurre in nessun luogo.

Progettare con i CSS, realizzando pagine standard (magari in XHTML) significa progettare per il futuro, senza però trascurare il passato: una pagina (X)HTML standard + CSS standard è accessibile con **tutti** i dispositivi per la navigazione nel web. Certo, i browser più datati richiederanno CSS specifici, magari più semplici, ma ottenere la stessa grafica su tutti i dispositivi/browser non è possibile e neppure utile. Un sito indipendente dal browser è quello in grado di presentare correttamente i contenuti su **tutti** i browser, non quello che ha la stessa grafica su IE e NN.

*G.T.* -- ultima revisione: 15.09.2002

[Commenti sull'articolo](#)

l'indirizzo di questa pagina è: [http://www.constile.org/tutorial/css\\_vs\\_table/index.php](http://www.constile.org/tutorial/css_vs_table/index.php)



[.ConStile](#), una **guida** dedicata all'uso dei **CSS** per lo sviluppo di siti web leggeri, usabili, conformi agli **standard W3C**

---

[\[H\] Home](#) > [Tutorial](#) > [Introduzione ai css](#) > Introduzione ai CSS: Cosa sono i fogli di stile

---

TOC

- [Indice](#)
  - Cosa sono i fogli di stile
  - [La sintassi delle regole](#)
  - [I selettori](#)
  - [Le proprietà](#)
  - [Associare i fogli di stile ai documenti \(X\)HTML](#)
  - [Note](#)
- 

# Introduzione ai CSS

## 1. Cosa sono i fogli di stile

I fogli di stile sono essenzialmente un'**insieme di regole** per istruire il browser su come debba essere presentata la pagina web.

Le **regole** dei CSS possono essere specificate in una pagina esterna al documento a cui le regole devono essere applicate, possono essere specificate nel documento stesso, possono essere incorporate nei tag (X)HTML. A come associare i CSS ai documenti (X)HTML è dedicata un'[apposita sezione](#) di questa guida.

Le regole specificate nei fogli di stile sono specifiche per ogni **selettore**, che può essere un tag HTML (come BODY, P, DIV) una classe (ad esempio <div class="nome-della-classe">) o un identificatore (ad esempio <div id="nome-dello-identificatore">).

Per ogni elemento, possono essere definite diverse **proprietà**. Ogni proprietà può avere un **valore** che, in associazione con la proprietà, descrive come il browser debba rappresentare il selettore.

---

TOC

- [Indice](#)
  - Cosa sono i fogli di stile
  - [La sintassi delle regole](#)
  - [I selettori](#)
  - [Le proprietà](#)
  - [Associare i fogli di stile ai documenti \(X\)HTML](#)
  - [Note](#)
-

[.ConStile](#), una **guida** dedicata all'uso dei **CSS** per lo sviluppo di siti web leggeri, usabili, conformi agli **standard W3C**

---

[\[H\] Home](#) > [Tutorial](#) > [Introduzione ai css](#) > Introduzione ai CSS: La sintassi delle regole

---

TOC

- [Indice](#)
  - [Cosa sono i fogli di stile](#)
  - La sintassi delle regole
  - [I selettori](#)
  - [Le proprietà](#)
  - [Associare i fogli di stile ai documenti \(X\)HTML](#)
  - [Note](#)
- 

# Introduzione ai CSS

## 2. La sintassi delle regole

Le regole devono essere specificate come segue:

```
selettore { proprietà: valore }
```

Qualora per uno stesso selettore si volessero specificare diverse proprietà, ogni proprietà dovrà essere separata da un punto e virgola:

```
selettore { proprietà1: valore1; proprietà2: valore2 }
```

Per maggiore comodità è possibile andare a capo:

```
selettore {  
  proprietà1: valore1;  
  proprietà2: valore2;  
  proprietà3: valore3  
}
```

Come esempio, riportiamo il seguente codice che definisce il colore e le dimensioni del testo per gli elementi H1 e H2:

```
h1 { font-size: 25px; color: #660000; }
```

```
h2 { font-size: 15px; color: #990000; }
```

Il selettore **font-size** serve a specificare la dimensione dei font, il selettore **color** serve a definire il colore del testo. font-size e color sono solo due delle [numeroso proprietà definibili](#) con i CSS. Le proprietà dei CSS (di livello 1) saranno illustrate nell'apposita sezione "[Le proprietà dei CSS](#)".

---

#### TOC

- [Indice](#)
- [Cosa sono i fogli di stile](#)
- La sintassi delle regole
- [I selettori](#)
- [Le proprietà](#)
- [Associare i fogli di stile ai documenti \(X\)HTML](#)
- [Note](#)

---

*G.T.* -- ultima revisione: 03.10.2002

l'indirizzo di questa pagina è:  
[http://www.constile.org/tutorial/introduzione\\_ai\\_css/sezione\\_2.php](http://www.constile.org/tutorial/introduzione_ai_css/sezione_2.php)

[.ConStile](#), una **guida** dedicata all'uso dei **CSS** per lo sviluppo di siti web leggeri, usabili, conformi agli **standard W3C**

---

[\[H\] Home](#) > [Tutorial](#) > [Introduzione ai css](#) > Introduzione ai CSS: I selettori

---

TOC

- [Indice](#)
  - [Cosa sono i fogli di stile](#)
  - [La sintassi delle regole](#)
  - I selettori
  - [Le proprietà](#)
  - [Associare i fogli di stile ai documenti \(X\)HTML](#)
  - [Note](#)
- 

# Introduzione ai CSS

## 3. I selettori

I selettori servono a specificare a quale elemento del documento la regola debba essere applicata. I principali selettori sono:

- [3.1 Tag HTML](#)
- [3.2 Classi](#)
- [3.3 Identificatori](#)
- [3.4 Pseudo-classi e Pseudo-elementi](#)
- [3.5 Selettori in cascata](#)
- [3.6 Raggruppamento](#)

### 3.1 Tag HTML

Ogni tag HTML è un selettore. Ad esempio:

```
p { text-indent: 2em; }  
a { font-weight: bold; }
```

Il codice sopra riportato specifica al browser che i paragrafi devono avere il capoverso con un rientro di due caratteri (come questo stesso paragrafo) e che i link devono essere in grassetto, migliorandone l'identificazione all'interno del testo.

### 3.2 Classi

Spesso si ha la necessità di associare ad uno stesso tag HTML diversi stili, per fare ciò è possibile ricorrere alle classi.

```
p.nota { font-size: small } /* carattere piccolo */  
p.evidenzia { background-color: yellow; } /* area con sfondo giallo */
```

Le due classi sono **nota** ed **evidenzia** e vengono specificate separando il nome del tag e il nome della classe con un punto. La classe verrà specificata nel codice HTML attraverso l'attributo CLASS:

```
<p class="nota">Questo paragrafo è una nota</p>  
<p class="evidenziato">Questo paragrafo è evidenziato in giallo</p>
```

La sintassi utilizzata specifica che la classe è associata al tag P. In questo modo è possibile associare lo stesso nome per classi associate a tag differenti:

```
CSS  
p.nota { font-size: small } /* carattere piccolo */  
span.nota { color: #999999; } /* testo in grigio */  
  
(X)HTML  
<p class="nota">Questo paragrafo è una nota</p>  
<p>Testo normale ( <span class="nota">nota nel testo</span> ) testo normale ...</p>
```

Le classi possono anche essere associate a nessun tag HTML, in questo modo la stessa classe può essere associata a più selettori:

```
CSS  
.nota { font-size: small } /* carattere piccolo */  
  
(X)HTML  
<p class="nota">Questo paragrafo è una nota</p>  
<div class="nota">Questa sezione è una nota</div>
```

Nota bene: per ogni selettore è possibile associare una e una sola classe. Ad esempio **non è valido** il seguente codice:

```
p.classe1.classe2 { ... }
```

## 3.3 Identificatori

Gli identificatori sono simili alle classi, ma servono a definire un'entità **unica** all'interno del documento. Se le classi, come la parola stessa indica, definiscono un insieme di elementi concettualmente omogenei (ad esempio le note di un testo possono essere accorpate nella classe `note`), l'identificatore deve definire un elemento che non si ripete all'interno del documento, ad esempio la nota per il copyright.

```
#nota-copyright { font-size: xx-small }
```

Il nome del identificatore deve essere preceduto dal carattere '#'. L'identificatore verrà specificato nel codice HTML attraverso l'attributo ID:

```
<p id="nota-copyright">Questo paragrafo è la nota per il copyright</p>
```

Nel codice HTML non devono esistere due tag HTML associati allo stesso identificatore, non possono dunque esistere due tag HTML con lo stesso valore per l'attributo ID.

```
<p id="nota-copyright">Questo paragrafo è la nota per il copyright</p>
<p id="nota-copyright">E' errato inserire questo paragrafo</p>
```

## 3.4 Pseudo-classi e Pseudo-elementi

Pseudo-classi e pseudo-elementi sono classi ed elementi speciali, automaticamente riconosciuti dai browser che supportano i CSS (Mozilla supporta maggiormente pseudo-classi e pseudo-elementi rispetto quanto non faccia Internet Explorer). Le pseudo-classi creano distinzioni fra elementi, ad esempio i link visitati e i link attivi sono distinti attraverso due pseudo-classi. Gli pseudo-elementi si riferiscono a sottoparti di elementi come il primo carattere di un paragrafo. La sintassi per specificare le regole da associare a pseudo-classi e pseudo-elementi è la seguente:

```
selettore:pseudoClasse { proprietà: valore }
selettore:pseudoElemento { proprietà: valore }
```

Come detto, pseudo-classi e pseudo-elementi sono automaticamente riconosciuti dai browser che supportano i CSS (il supporto può differire per i diversi browser), non è dunque necessario specificare nulla nel codice HTML. Si consideri il seguente esempio, relativo alle pseudo-classi:

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

```
A:visited { color: purple; }  
A:hover { color: red; }  
A:active { color: darkred; }
```

Il codice CSS indica al browser che i link devono essere blu, i link visitati devono essere viola, i link sui quali il puntatore del mouse si trova devono essere rossi (effetto roll-hover), i link attivi devono essere rosso scuro. Nel codice HTML non sarà necessario specificare nulla:

```
<a href="URL">questo link cambierà colore in base al suo stato</a>
```

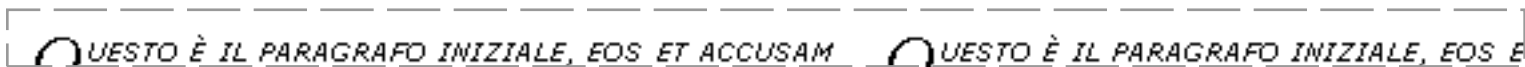
Si consideri ora il seguente esempio, in cui verrà mostrato il funzionamento degli pseudo-elementi e come pseudo-classi e pseudo-elementi possano essere associati anche a classi e identificatori:

```
p#paragrafo-iniziale { text-indent: 0em; }  
p#paragrafo-iniziale:first-letter {  
  font-size: 24px;  
  font-family: georgia;  
  font-style: italic;  
  float: left;  
  margin: -5px 3px 0 0;  
}  
p#paragrafo-iniziale:first-line {  
  text-transform: uppercase;  
  font-style: italic;  
}  
p {  
  font-size: 10px;  
  font-family: verdana;  
  text-indent: 2em;  
  width: 300px;  
  text-align: justify  
}
```

Il codice sopra riportato utilizza due pseudo-elementi: **first-letter** e **first-line**, che selezionano la prima lettera e la prima riga di testo del selettore a cui sono associate, in questo caso il paragrafo di classe **paragrafo-iniziale**. Di seguito è riportato un esempio di codice HTML con cui utilizzare le regole specificate, si noti come non ci sia alcun accenno agli pseudo-elementi ma solo alla classe paragrafo-iniziale:

```
<p id="paragrafo-iniziale">Questo è il paragrafo iniziale [...]</p>  
<p>Questo è un paragrafo normale [...]</p>  
<p>Questo è un paragrafo normale [...]</p>
```

Per comodità è di seguito riportata un'immagine che riproduce il risultato su un browser (Mozzila 1.0, in questo caso) che supporti i CSS. Si noti come la prima riga e il primo carattere del paragrafo associato alla classe paragrafo-iniziale abbiano caratteristiche diverse dal resto del testo, il risultato è stato ottenuto senza appesantire il testo se non per specificare quale fosse il paragrafo interessato d'interesse.



## 3.5 Selettori in cascata

Ci sono numerosi altri modi più o meno elaborati per specificare dei selettori, alcuni dei quali non supportati da browser largamente diffusi (ad esempio Internet Explorer 6 e inferiori). Un metodo ben supportato e molto utile è quello di utilizzare selettori in cascata:

```
selettore1 selettore2 { proprietà: valore }
```

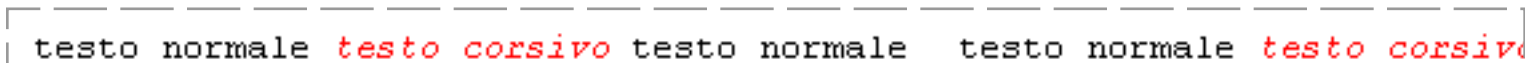
In questo caso la regola si applica al selettore2 **se** questo è **contenuto** nel selettore1. Ad esempio si consideri il seguente codice:

```
code i { color: red; }
```

Questo codice specifica che il tag I, se inserito nel tag CODE, deve contenere del testo di colore rosso. Ad esempio, si consideri il seguente codice CSS:

```
<p><code>testo normale <i>testo corsivo</i> testo normale</code> </p>  
<p>testo normale <i>testo corsivo</i> testo normale</p>
```

La seguente immagine riproduce il risultato che si ottiene su un browser (Internet Explorer 6, in questo caso) che supporti i CSS:



## 3.6 Raggruppamento

Qualora più selettori dovessero rispettare le stesse regole, è possibile raggrupparli come segue:

```
h1, h2, h3, h4, h5, h6 { font-family: sans-serif }
```



**Il codice sopra riportato è equivalente al seguente codice:**

```
h1 { font-family: sans-serif }  
h2 { font-family: sans-serif }  
h3 { font-family: sans-serif }  
h4 { font-family: sans-serif }  
h5 { font-family: sans-serif }  
h6 { font-family: sans-serif }
```

---

## TOC

- [Indice](#)
  - [Cosa sono i fogli di stile](#)
  - [La sintassi delle regole](#)
  - I selettori
  - [Le proprietà](#)
  - [Associare i fogli di stile ai documenti \(X\)HTML](#)
  - [Note](#)
- 

**G.T.** -- ultima revisione: 03.10.2002

l'indirizzo di questa pagina è:  
[http://www.constile.org/tutorial/introduzione\\_ai\\_css/sezione\\_3.php](http://www.constile.org/tutorial/introduzione_ai_css/sezione_3.php)

[.ConStile](#), una **guida** dedicata all'uso dei **CSS** per lo sviluppo di siti web leggeri, usabili, conformi agli **standard W3C**

---

[\[H\] Home](#) > [Tutorial](#) > [Introduzione ai css](#) > Introduzione ai CSS : Le proprietà

---

TOC

- [Indice](#)
  - [Cosa sono i fogli di stile](#)
  - [La sintassi delle regole](#)
  - [I selettori](#)
  - Le proprietà
  - [Associare i fogli di stile ai documenti \(X\)HTML](#)
  - [Note](#)
- 

# Introduzione ai CSS

## 4. Le proprietà

In questa sezione saranno brevemente illustrate le proprietà dei CSS di livello 1.

Verrà utilizzata la seguente sintassi (la stessa utilizzata dal W3C):

- **<Abc>**: un valore della proprietà Abc
- **Abc**: keyword richiesta
- **A B C**: A, B, C sono keyword richieste nell'ordine specificato
- **A|B**: deve apparire A o B
- **A||B**: deve apparire A o B oppure ambedue
- **[Abc]**: le parentesi quadre sono utilizzate per raggruppare gli oggetti
- **Abc\***: Abc appare zero o più volte
- **Abc+**: Abc appare una o più volte
- **Abc?**: Abc è opzionale
- **Abc{A,B}**: Abc deve apparire almeno A volte e al massimo B volte

Le proprietà possono essere raggruppate nelle seguenti categorie:

- **4.1 Font:**
  - [font-family](#)
  - [font-style](#)
  - [font-variant](#)
  - [font-weight](#)
  - [font-size](#)
  - [font](#)
- **4.2 Colore e Sfondo:**
  - [color](#)

- [background](#)
- **4.3 Testo:**
  - [word-spacing](#)
  - [letter-spacing](#)
  - [line-height](#)
  - [text-indentation](#)
  - [vertical-align](#)
  - [text-decoration](#)
  - [text-transform](#)
  - [text-align](#)
  - [white-space](#)
- **4.4 Box model:**
  - [margin](#)
  - [padding](#)
  - [border](#)
  - [width](#)
  - [height](#)
  - [float](#)
  - [clear](#)
- **4.5 Classificazione:**
  - [display](#)
  - [list-style](#)
- **4.6 Lunghezze, percentuali, colori, url:**
  - [lunghezze](#)
  - [percentuali](#)
  - [colori](#)
  - [url](#)

## 4.1 Font

### 4.1.1 font-family

**sintassi:** font-family: <nome font>\* || <font generico>

**<nome font>**

- qualsiasi nome di font (per esempio: verdana, helvetica, georgia)

**<font generico>**

- serif
- sans-serif

- monospace
- cursive [sconsigliato poiché di difficile lettura]
- fantasy [sconsigliato poiché di difficile lettura]

Esempio:

```
p { font-family: verdana, helvetica, "Trebuchet MS", sans-serif }
```

Il font adottato sarà il primo disponibile partendo da sinistra. E' bene specificare sempre un font generico, applicato qualora nessuno dei font elencati fosse disponibile. Font con nomi composta da due o più parole vanno fra virgolette: ad esempio "Trebuchet MS"

#### 4.1.2 font-style

**sintassi:** font-style: normal | italic | oblique;

Esempio:

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

Il font adottato sarà il primo disponibile partendo da sinistra. E' bene specificare sempre un font generico, applicato qualora nessuno dei font elencati fosse disponibile. Font con nomi composta da due o più parole vanno fra virgolette: ad esempio "Trebuchet MS".

#### 4.1.3 font-variant

**sintassi:** font-variant: normal | small-caps

Esempio:

```
.maiuscoletto { font-variant: small-caps }
```

Il valore small-caps crea l'effetto "Maiuscoletto", il valore normal non introduce variazioni.

#### 4.1.4 font-weight

**sintassi:** font-weight: normal | bold | bolder | lighter | 100 | 200 | 300 | 400 | 500 | 600 | 700 | 800 | 900

Esempio:

```
strong { font-weight: 900; }
```

Specifica il peso del font. I valori bolder e lighter sono relativi al valore di default. Gli altri valori sono pesi assoluti. 100-900 è una scala che va dal peso minimo al massimo. Non tutti e nove i valori da 100 a 900 sono disponibili per tutti i font, in questo caso il peso adottato è il più vicino disponibile.

#### 4.1.5 font-size

**sintassi:** font-size: <dimensione assoluta> | <dimensione relativa> | <lunghezza> | <percentuale>

<dimensione assoluta> (vedi "[Dimensionare i caratteri con i CSS](#)" per maggiori dettagli)

- xx-small | x-small | small | medium | large | x-large | xx-large

<dimensione relativa> (vedi "[Dimensionare i caratteri con i CSS](#)" per maggiori dettagli)

- larger | smaller

<lunghezza>

- vedi [sezione 4.6.1](#)

<percentuale>

- vedi [sezione 4.6.2](#)

Esempio:

```
p { font-size: 0.80em }
```

Sul dimensionamento dei caratteri è disponibile un [articolo dedicato](#)

#### 4.1.6 font

**sintassi:** font: <valore>

<valore>

- [ <[font-style](#)> || <[font-variant](#)> || <[font-weight](#)> ]? <[font-size](#)> [ / <[line-height](#)> ]? <[font-family](#)>

Esempio:

```
p.paragrafo-speciale { font: bold 1.00em/1.50em verdana, helvetica, "Trebuchet MS", sans-serif }
```

La proprietà font può essere utilizzata come un metodo veloce per definire le diverse proprietà font-family, font-style, ecc. Si noti la possibilità di specificare anche la proprietà [<line-height>](#) (una proprietà per specificare l'interlinea). Quando si utilizza la proprietà font è necessario specificare almeno i valori per le proprietà [font-size](#) e [font-family](#).

## 4.2 Colore e sfondo

### 4.2.1 color

**sintassi:** color: [<colore>](#)

Esempio:

```
a { color: #009 }  
em { color: red }
```

La proprietà color permette di specificare il colore del testo di un selettore, i possibili valori sono descritti nella [sezione 4.6.3](#).

### 4.2.2 background

**sintassi:** background-color: [<valore>](#)

[<valore>](#)

- [<colore>](#) || [<immagine>](#) || [<ripetizione>](#) || [<scrolling>](#) || [<posizione>](#)
  - [<colore>](#): i possibili valori sono descritti nella [sezione 4.6.3](#)
  - [<immagine>](#): url dell'immagine, i possibili valori sono descritti nella [sezione 4.6.4](#)
  - [<ripetizione>](#): repeat | repeat-x | repeat-y | no-repeat  
l'immagine di sfondo può essere ripetuta, ripetuta orizzontalmente, ripetuta verticalmente, non essere ripetuta (immagine di sfondo unica).
  - [<scrolling>](#): scroll | fixed  
l'immagine si muove col testo ovvero rimane fissa.
  - [<posizione>](#): [[<percentuale>](#) | [<lunghezza>](#)]{1,2} | [top | center | bottom] || [left | center | right]  
la posizione dell'immagine di sfondo può essere specificata con riferimento all'angolo superiore sinistro, sia attraverso unità [percentuali](#) o di [lunghezza](#) sia attraverso keyword.

Esempio:

```
body { background: #FFFFFF url(/images/sfondo.gif) no-repeat fixed top right; }
```

La proprietà **background** permette di specificare il colore e l'immagine di sfondo. L'immagine può essere posizionata come si desidera, può essere ripetuta oppure no, può scorrere col testo ovvero essere fissa.

## 4.3 Testo

### 4.3.1 word-spacing

**sintassi:** word-spacing: <valore>

<valore>

- normal | <[lunghezza](#)>

Esempio:

```
.parole-distanziate { word-spacing: 1.00em }  
.parole-ravvicinate { word-spacing: -0.20em }
```

La proprietà **word-spacing** permette di impostare lo spazio fra le parole, **aumentando lo spazio** fra le parole oppure **diminuendo tale spazio**. Se viene specificata una [lunghezza](#), questa va ad aggiungersi ovvero a sottrarsi alla spaziatura predefinita.

### 4.3.2 letter-spacing

**sintassi:** letter-spacing: <valore>

<valore>

- normal | <[lunghezza](#)>

Esempio:

```
.lettere-distanziate { letter-spacing: 0.50em }  
.lettere-ravvicinate { letter-spacing: -0.15em }
```

La proprietà **letter-spacing** permette di impostare lo spazio fra le singole lettere di una parola, **aumentandolo** oppure **diminuendolo**. Se viene specificata una [lunghezza](#), questa va ad aggiungersi ovvero a sottrarsi alla spaziatura predefinita.

### 4.3.3 line-height

**sintassi:** line-height: <valore>

<valore>

- normal | <[lunghezza](#)> | <[percentuale](#)>

Esempio:

```
.interlinea-ridotta { line-height: 100% }
```

La proprietà line-height permette di impostare l'interlinea. Le percentuali si riferiscono all'altezza dei font e non al valore standard dell'interlinea, valori negativi non sono permessi. L'interlinea può essere specificata anche tramite la proprietà [font](#). Questo paragrafo ha un'interlinea pari alla dimensione dei caratteri.

### 4.3.4 text-indentation

**sintassi:** text-indentation: <valore>

<valore>

- <[lunghezza](#)> | <[percentuale](#)>

Esempio:

```
p { text-indentation: 2em; }
```

La proprietà text-indentation permette di impostare il rientro del capoverso. Le percentuali si riferiscono alla larghezza del parent e sono quindi più difficili da gestire. I paragrafi di questo articolo hanno un rientro del capoverso pari 2em.

### 4.3.5 vertical-align

**sintassi:** vertical-align: <valore>

<valore>

- baseline | sub | super | top | text-top | middle | bottom | text-bottom | <[percentuale](#)>

Esempio:



```
img { vertical-align: top }  
.apice { vertical-align: super }  
.pedice { vertical-align: sub }
```

La proprietà `vertical-align` permette modificare il posizionamento verticale degli elementi inline. Le percentuali si riferiscono all'interlinea. La proprietà `vertical-align` è utile per gestire l'allineamento delle immagini come pure per creare apici ovvero pedici (per apici e pedici è bene agire anche sulla [dimensione del testo](#)).

#### 4.3.6 text-decoration

**sintassi:** `text-decoration: <valore>`

**<valore>**

- none | [ underline || overline || line-through || blink ]

Esempio:

```
#barra-di-navigazione A { text-decoration: none }  
/* link non sottolineato */
```

La proprietà `text-decoration` permette decorare il testo: i possibili valori sono:

- underline: testo sottolineato (andrebbe utilizzato **esclusivamente** per i link, poiché storicamente nelle pagine web del testo sottolineato rappresenta un link).
- overline: testo sopralineato, spesso utilizzato insieme al valore underline per evidenziare i link al passaggio del mouse per mezzo del selettore `a:hover`. La soprallineatura va usata con cautela poiché può essere confusa con la sottolineatura della riga superiore.
- line-through: testo "cancellato"
- blink: testo lampeggiante, semplicemente da evitare (e infatti non è riportato nessun esempio) poiché molto fastidioso per la vista. Persone sensibili potrebbero anche essere soggette a disturbi. "Fortunatamente" Internet Explorer non supporta il testo lampeggiante
- none: elimina le decorazioni. Spesso è utilizzato per eliminare la sottolineatura dei link (vedi esempio sopra riportato). Si ricordi che se il link non è sottolineato, la sua funzione deve essere evidenziata in altro modo, come colore (testo blu), peso del carattere (grassetto), contesto (inserito in una barra di navigazione).

#### 4.3.7 text-transform

**sintassi:** `text-transform: <valore>`

**<valore>**

- none | capitalize | uppercase | lowercase

Esempio:

```
.tutto-maiuscolo { text-transform: uppercase }
```

La proprietà text-transform permette impostare il testo maiuscolo/minuscolo: i possibili valori sono:

- capitalize: testo la prima lettera delle parole deve essere maiuscola.
- uppercase: testo le lettere delle parole devono essere tutte maiuscole.
- lowercase: testo le LETTERE delle parole devono essere tutte minuscole.
- none: elimina eventuali trasformazioni del testo.

#### 4.3.8 text-align

**sintassi:** text-align: <valore>

**<valore>**

- left | right | center | justify

Esempio:

```
h1.titolo { text-align: center }  
p.articolo { text-align: justify }
```

La proprietà text-align permette l'allineamento del testo:

- left: allineamento a sinistra.
- right: allineamento a destra.
- center: testo centrato.
- justify: testogiustificato (il testo occupa tutta la larghezza disponibile, come i paragrafi di questo articolo).

#### 4.3.9 white-space

**sintassi:** white-space: <valore>

**<valore>**

- normal | pre | nowrap

## Esempio:

```
p.testo-formattato { white-space: pre }  
p.nowrap { white-space: nowrap }
```

La proprietà `white-space` specifica come interpretare gli spazi fra gli elementi:

- **normal**: due o più spazi bianchi collassano in un unico spazio.
- **pre**: evita che gli User Agent facciano collassare sequenze di spazi bianchi. Le linee di testo vengono interrotte solo in corrispondenza delle interruzioni di linea del codice (X)HTML.
- **nowrap**: due o più spazi collassano in un unico spazio (come per il valore **normal**), ma le linee di testo vengono interrotte solo in corrispondenza delle interruzioni di linea del codice (X)HTML (come per il valore **pre**).

## 4.4 Box model

### 4.4.1 margin

**sintassi:** `margin: <valore>`

**<valore>**

- [<lunghezza>](#) | [<percentuale>](#) | `auto`

## Esempio:

```
body { margin: 10px } /* tutti i margini a 10px */  
p { margin: 10px 5px }  
/* margini sup. e inf. a 10px, margini des. e sin. a 5px */  
.esempio { margin: 1px 2px 3px 4px }  
/* margini (in senso orario -- Nord Est Sud Ovest):  
sup. 1px  
des. 2px  
inf. 3px  
sin. 4px */
```

La proprietà `margin` permette di impostare il margine di un elemento (vedi la seguente immagine per maggiori dettagli). Per avere ulteriori informazioni sul box model è disponibile il seguente articolo: "[Il box model di IE5/Win](#)".



Specificando un solo valore, si impostano tutti e quattro i margini al valore specificato. Specificando

due valori, si impostano il margine superiore e inferiore al primo valore indicato, il margine destro e sinistro al secondo valore indicato. Specificando quattro valori, si impostano il margine superiore, quello destro, quello inferiore e quello sinistro rispettivamente (in senso orario a partire dal margine superiore).

E' possibile impostare i margini singolarmente attraverso le seguenti quattro proprietà: **margin-top**, **margin-right**, **margin-bottom**, **margin-left**. Ad esempio:

```
.esempio { margin-top: 1px }  
.esempio { margin-right: 2px }  
.esempio { margin-bottom: 3px }  
.esempio { margin-left: 4px }
```

#### 4.4.2 padding

**sintassi:** padding: <valore>

<valore>

- <[lunghezza](#)> | <[percentuale](#)> | auto

Esempio:

```
body { padding: 10px } /* padding a 10px su ogni lato*/  
p { padding: 10px 5px }  
/* padding lati sup. e inf. a 10px, padding lati des. e sin. a 5px */  
.esempio { padding: 1px 2px 3px 4px }  
/* padding (in senso orario -- Nord Est Sud Ovest):  
sup. 1px  
des. 2px  
inf. 3px  
sin. 4px */
```

La proprietà padding permette di impostare il padding di un elemento (vedi la seguente immagine per maggiori dettagli). Per avere ulteriori informazioni sul box model è disponibile il seguente articolo: "[Il box model di IE5/Win](#)".



Specificando un solo valore, si imposta il padding di tutti e quattro i lati al valore specificato. Specificando due valori, si imposta il padding dei lati superiore e inferiore al primo valore indicato, il padding dei lati destro e sinistro al secondo valore indicato. Specificando quattro valori, si imposta il padding del lato superiore, del lato destro, del lato inferiore e del lato sinistro rispettivamente (in senso orario a partire dal margine superiore).

E' possibile impostare il paddin di ogni singolo lato attraverso le seguenti quattro proprietà: **padding-top**, **padding-right**, **padding-bottom**, **padding-left**. Ad esempio:

```
.esempio { padding-top: 1px }  
.esempio { padding-right: 2px }  
.esempio { padding-bottom: 3px }  
.esempio { padding-left: 4px }
```

#### 4.4.3 border

**sintassi:** border: <spessore> || <stile> || <[colore](#)>

**<spessore>**

- thin | medium | thick | <[lunghezza](#)>

Lo spessore del bordo può essere impostato attraverso le tre parole chiave thin (fino), medium (medio) e thick (spesso), ovvero specificandone la [dimensione](#)

**<stile>**

- none | dotted | dashed | solid | double | groove | ridge | inset | outset

La proprietà border permette di impostare il bordo di un elemento (vedi la seguente immagine per maggiori dettagli). Per avere ulteriori informazioni sul box model è disponibile il seguente articolo: "[Il box model di IE5/Win](#)".



Per rendere il bordo visibile è **necessario** specificarne lo stile. Se lo stile è impostato su none il bordo non è visibile (utile per le immagini).

Esempio:

```
.box1 { border: thick solid blue }  
.box2 { border: 2px dotted red }  
.box3 { border: thin dashed #000000 }  
.bottone { border: 5px outset #666 }
```

Il precedente codice (aggiungendo del margine e regolando la larghezza) genera i seguenti box:

**.box1**

.box2

.box3

.bottone

Tramite la proprietà `border` è possibile specificare solo il bordo di tutti e quattro i lati, qualora si volessero specificare i lati singolarmente è possibile utilizzare le seguenti proprietà: `border-top`, `border-right`, `border-bottom`, `border-left` che consentono, attraverso la stessa sintassi di `border`, di gestire i lati singolarmente. Esempio:

```
.box {  
  border-top: 1px dashed red;  
  border-right: 2px dotted blue;  
  border-bottom: 3px double green;  
  border-left: 4px solid #000;  
}
```

Il precedente codice (regolando la larghezza) genera il seguente box:

.box

In alternativa è possibile specificare singolarmente spessore, stile e colore del bordo attraverso le seguenti proprietà: `border-width`, `border-color`, `border-style`. Il seguente codice è equivalente a quello sopra riportato:

```
.box {  
  border-width: 1px 2px 3px 4px; /* top right bottom left */  
  border-style: dashed dotted double solid; /* top right bottom left */  
  border-color: red blue green #000; /* top right bottom left */  
}
```

.box

Quando si utilizzano le keyword `thin`, `medium` e `thick` per impostare la larghezza del bordo, lo spessore può variare leggermente in base al browser utilizzato. Lo stesso vale per i diversi stili. Si ricorda che Internet Explorer è incapace di rappresentare un bordo spesso 1 px o 1pt con lo stile `dotted`, lo stile del bordo sarà settato automaticamente su `dashed`.

Il bordo può essere aggiunto anche a parole del testo.

#### 4.4.4 width

**sintassi:** `width: <valore>`

## <valore>

- [<lunghezza>](#) | [<percentuale>](#) | auto

Esempio:

```
div.box { width: 70% }  
img.icona { width: 50px; }
```

La proprietà width permette di impostare la larghezza di un elemento (vedi la seguente immagine per maggiori dettagli). Per avere ulteriori informazioni sul box model è disponibile il seguente articolo: "[Il box model di IE5/Win](#)".



La larghezza può essere impostata per elementi block-level o gli elementi detti replaced element.

Le percentuali sono calcolate in base alla larghezza dell'elemento parent. Oltre che per la definizione dei box, la proprietà width può risultare utile nei form per impostare la larghezza dei campi e dei pulsanti.

### 4.4.5 height

**sintassi:** height: <valore>

## <valore>

- [<lunghezza>](#) | [<percentuale>](#) | auto

Esempio:

```
textarea { height: 200px; }  
img.icona { height: 50px; }
```

La proprietà height permette di impostare l'altezza di un elemento. L'altezza può essere impostata per elementi block-level o gli elementi detti replaced element.

Le percentuali sono calcolate in base all'altezza dell'elemento parent. Oltre che per la definizione dei box, la proprietà width può risultare utile nei form per impostare con precisione l'altezza di campi come TEXTAREA.

### 4.4.6 float

**sintassi:** float: <valore>

**<valore>**

- left | right | none

Esempio:

```
img.figura { float: right; }  
blockquote { float: left; }
```

La proprietà float permette di disporre il testo attorno ad un elemento. Questa proprietà funziona come gli attributi align="left" o align="right" per le immagini (attributi sconsigliati dal W3C), ma funziona con ogni elemento, non solo con le immagini.

Le percentuali sono calcolate in base all'altezza dell'elemento parent. Oltre che per la definizione dei box, la proprietà width può risultare utile nei form per impostare con precisione l'altezza di campi come TEXTAREA.

#### 4.4.7 clear

**sintassi:** clear: <valore>

**<valore>**

- none | left | right | both

Esempio:

```
p.nofloat { clear: both; }  
pre { clear: left; }
```

La proprietà clear permette di specificare se l'elemento permette la presenza di elementi floating ai suoi lati. Il valore specificato posiziona l'elemento al di sotto di eventuali elementi fluttuanti posti sul lato specificato dal valore. Ulteriori dettagli sull'utilizzo della proprietà float sono riportati nell'articolo "[Ripristino dell'allineamento in seguito all'utilizzo della proprietà float](#)".

## 4.5 Classificazione

### 4.5.1 display

**sintassi:** display: <valore>



## <valore>

- block | inline | list-item | none

Esempio:

```
img.esempio { display: block }  
img.icona { display: inline }  
img.contatore { display: none }  
li { display: list-item }
```

La proprietà display permette di definire un elemento come elemento block-level, elemento inline, elemento di lista. Se il valore della proprietà display è impostato su non l'elemento (compresi tutti gli elenti in esso contenuti!) non verrà visualizzato.

L'utilizzo dell proprietà display può alterare la normale interpretazione dello User Agent (ad esempio P è un elemento block-level mentre IMG è un elemento inline) e va dunque usato con cautela. La regola selettore { display: none } può essere utilizzata, ad esempio, per [migliorare la stampa delle pagine web](#).

### 4.5.2 list-style

**sintassi:** list-style: <marcatore> || <allineamento> || <[url](#)>

## <marcatore>

- disc | circle | square | decimal | lower-roman | upper-roman | lower-alpha | upper-alpha | none

## <allineamento>

- inside | outside

Esempio:

```
ul.square { list-style: square; }  
ul.inside { list-style: inside; }  
ul.3pt { list-style: url(/images/3pt.gif) circle; }  
ol.roman { list-style: upper-roman; }  
ol.alpha { list-style: lower-alpha; }  
ol.none { list-style: none; }
```

Il precedente codice genera le seguenti liste:

- Et harumd dereud facilis est  
er expedit distinct suscipit laboris
- Et harumd dereud facilis est  
er expedit distinct suscipit laboris
- Et harumd dereud facilis est  
er expedit distinct suscipit laboris

- Et harumd dereud facilis est  
er expedit distinct suscipit laboris
- Et harumd dereud facilis est  
er expedit distinct suscipit laboris
- Et harumd dereud facilis est  
er expedit distinct suscipit laboris

- Et harumd dereud facilis est  
er expedit distinct suscipit laboris
- Et harumd dereud facilis est  
er expedit distinct suscipit laboris
- Et harumd dereud facilis est  
er expedit distinct suscipit laboris

1. Et harumd dereud facilis est  
er expedit distinct suscipit laboris
2. Et harumd dereud facilis est  
er expedit distinct suscipit laboris
3. Et harumd dereud facilis est  
er expedit distinct suscipit laboris

1. Et harumd dereud facilis est  
er expedit distinct suscipit laboris
2. Et harumd dereud facilis est  
er expedit distinct suscipit laboris
3. Et harumd dereud facilis est  
er expedit distinct suscipit laboris

1. Et harumd dereud facilis est  
er expedit distinct suscipit laboris
2. Et harumd dereud facilis est  
er expedit distinct suscipit laboris
3. Et harumd dereud facilis est  
er expedit distinct suscipit laboris

La proprietà `list-style` permette di definire come debbano essere rappresentate le liste, permettendo di specificare sia lo stile dei marcatori, sia l'allineamento. Tramite il valore `<url>` è possibile specificare un'immagine per i marcatori. Qualora l'immagine specificata con `<url>` non fosse disponibile non viene usato nessun marcatore. Per evitare questo inconveniente è sufficiente

specificare il marcatore desiderato oltre all'immagine:

```
ul { list-style: url(/percorso/immagine.gif) circle; }
```

## 4.6 Lunghezze, percentuali, colori, url

### 4.6.1 lunghezze

Le lunghezze sono costituite da un numero (positivo o negativo) subito seguito dall'unità di misura indicata con due lettere. Non è consentito inserire degli spazi fra il numero e l'unità di misura.

Esistono due tipi di lunghezze: relative e assolute.

Le **unità relative** forniscono una lunghezza relativamente ad un'altra lunghezza e sono dunque preferibili poiché maggiormente adattabili ai diversi media. Le unità di misura relativa sono:

- **em**: l'altezza dei font dell'elemento.
- **px**: pixel, la loro dimensione dipende dal dispositivo usato per la visione delle pagine web.

Esempio:

```
.classeA { font-size: 10px }  
.classeA { line-height: 1.50em }
```

L'unità em si riferisce alla dimensione dei font dell'elemento a cui la regola si applica. Nel precedente esempio la classe 'classeA' ha un font di dimensione 10px di conseguenza 1em = 10px e la proprietà line-height impostata a 1.50em è come se fosse impostata a 15px. Quando usata per dimensionare i font, l'unità em non si riferisce alla dimensione dei font dell'elemento a cui la regola si applica, bensì alla dimensione dei font dell'elemento parent. Maggiori dettagli sul dimensionamento dei font sono riportati nell'articolo "[Dimensionare i caratteri con i CSS](#)".

Si consideri il seguente esempio:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"  
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">  
<html xmlns="http://www.w3.org/1999/xhtml">  
<head>  
<title> Esempio</title>  
<style type="text/css">  
<!--  
.classeA { font-size: 10px }  
.classeA { line-height: 1.50em }  
.calsseA h1 { font-size: 2.00em }  
-->
```

```
</style>
</head>
<body>
<div class="classeA">
<h1>Elemento H1 [ hp ]</h1>
testo delle calsse 'classeA' [ hp ]<br />
testo delle calsse 'classeA' [ hp ]
</div>
</body>
</html>
```

Il risultato del codice sopra riportato è riprodotto nella seguente immagine:



La dimensione del pixel dipende invece dalla risoluzione dispositivo. Ad esempio lo schermo di un PC ha una risoluzione pari a 96dpi mentre una stampante può avere una risoluzione pari 400dpi:



Le **lunghezze assolute** andrebbero utilizzate esclusivamente quando sono note le proprietà fisiche dei media (ad esempio la dimensione dei fogli di una stampante è tipicamente il formato A4). Le unità assolute sono:

- **in**: inch (pollici -- 1in=2.54cm)
- **cm**: centimetri
- **mm**: millimetri
- **pt**: punti (1pt=1/72in)
- **pc**: picas (1pc=12pt)

#### 4.6.2 percentuali

Le percentuali (positive o negative) sono relative alla dimensione di altri elementi, dipendendo dalla proprietà che utilizza le percentuali. Ad esempio:

```
div.halfbox { width: 50% }
/* 50% della larghezza dell'elemento parent */
p.nota { font-size: 80% }
/* 80% della dimensione dei font dell'elemento parent */
```

#### 4.6.3 colori

Un colore è cosituito da una keyword o da un codice RGB (Red Green Blue - Rosso, Verde, Blu).

Le 16 keyword sono: aqua, black, blue, fuchsia, gray, green, lime, maroon, navy, olive, purple, red,

silver, teal, white, e yellow.

I valori RGB possono essere forniti attraverso quattro sintassi:

- **#rrggbb**: ad esempio #CC6600
- **#rgb**: ad esempio #C60 (#abc = #aabbcc)
- **rgb(r,g,b)**: dove r,g,b possono variare da 0 a 255. Ad esempio rgb(204,102,0)
- **rgb(r%,g%,b%)**: dove r,g,b possono variare da 0.0 a 100.0. Ad esempio rgb(80%,40%,0)

I valori #CC6600, #C60, rgb(204,102,0), rgb(80%,40%,0) specificano tutti lo stesso colore (**arancione scuro**).

La sintassi #rgb è molto utile poiché facilita l'indicazione dei colori della palette "web safe".

#### 4.6.4 URL

Un valore URL viene fornito attraverso la seguente sintassi: url(*url*), dove *url* è l'indirizzo da specificare; *url* può essere riportato fra apici o virgolette. Esempi:

- body { background: url(/images/sfondo.gif) }
- body { background: url("/images/sfondo.gif") }
- body { background: url('/images/sfondo.gif') }
- body { background: url(../images/sfondo.gif) }

I percorsi sono relativi alla cartella contenente il foglio di stile. Se ad esempio il foglio di stile è contenuto nella cartella /inc/css/ il valore url("images/sfondo.gif") indica il file "/inc/css/images/sfondo.gif". Per evitare problemi con l'interpretazione dei percorsi relativi da parte di alcuni User Agent si consiglia di utilizzare sempre percorsi assoluti: url("/images/sfondo.gif").

---

#### TOC

- [Indice](#)
  - [Cosa sono i fogli di stile](#)
  - [La sintassi delle regole](#)
  - [I selettori](#)
  - Le proprietà
  - [Associare i fogli di stile ai documenti \(X\)HTML](#)
  - [Note](#)
- 

G.T. -- ultima revisione: 03.10.2002

l'indirizzo di questa pagina è: [http://www.constile.org/tutorial/introduzione\\_ai\\_css/sezione\\_4.php](http://www.constile.org/tutorial/introduzione_ai_css/sezione_4.php)

[.ConStile](#), una **guida** dedicata all'uso dei **CSS** per lo sviluppo di siti web leggeri, usabili, conformi agli **standard W3C**

---

[\[H\] Home](#) > [Tutorial](#) > [Introduzione ai css](#) > Introduzione ai CSS : Associare i fogli di stile ai documenti (X)HTML

---

TOC

- [Indice](#)
  - [Cosa sono i fogli di stile](#)
  - [La sintassi delle regole](#)
  - [I selettori](#)
  - [Le proprietà](#)
  - Associare i fogli di stile ai documenti (X)HTML
  - [Note](#)
- 

# Introduzione ai CSS

## 5. Associare i fogli di stile ai documenti (X)HTML

In questa sezione saranno illustrati i metodi per collegare i CSS ai documenti (X)HTML.

I metodi sono:

- Fogli di stile esterni
- Fogli di stile incorporati
- La regola @import
- CSS in linea

### Fogli di stile esterni

I fogli di stile possono essere salvati in file specifici (ad esempio <http://www.constile.org/css/screen.css>), con estensione **.css**, per essere poi collegati al codice (X)HTML tramite il tag LINK:

```
<head>
[... ]
<link rel="stylesheet" type="text/css" media="screen" href="/css/foglio_di_stile.css" />
[... ]
</head>
```

Per maggiori informazioni sull'attributo `media` si consiglia di leggere l'articolo: "[Diversi media, diversi stili](#)".

## Fogli di stile incorporati

Le regole dei CSS possono essere specificate direttamente nel codice (X)HTML tramite il tag `STYLE` posto nell'`HEAD` del documento (X)HTML:

```
<head>
[... ]
<style type="text/css">
<!--
body { font: 80% Verdana,Helvetica,sans-serif; }
p { font: 1.00em/1.20em verdana, helvetica, sans-serif }
[... ]
-->
</style>
[... ]
</head>
```

## La regola @import

La regola `@import` permette di importare regole da un altro foglio di stile. Ad esempio:

```
@import url("advanced.css");
@import special.css;
body { font: 80% Verdana,Helvetica,sans-serif; }
p { font: 1.00em/1.20em verdana, helvetica, sans-serif }
[... ]
```

Se presenti, le regole `@import` devono precedere le altre regole.

## CSS in linea

Attraverso l'attributo `STYLE` è possibile specificare le regole CSS direttamente nei tag a cui esse saranno applicate:

```
<body style="background: #CCC; color: #000; font: 1.00em verdana, helvetica, sans-serif">
<p>
  In questo paragrafo,
  <span style="background: yellow">questo testo</span>
  sarà evidenziato in giallo
</p>
</body>
```

[.ConStile](#), una **guida** dedicata all'uso dei **CSS** per lo sviluppo di siti web leggeri, usabili, conformi agli **standard W3C**

---

[\[H\] Home](#) > [Tutorial](#) > [Introduzione ai css](#) > Introduzione ai CSS : Associare i fogli di stile ai documenti (X)HTML

---

TOC

- [Indice](#)
  - [Cosa sono i fogli di stile](#)
  - [La sintassi delle regole](#)
  - [I selettori](#)
  - [Le proprietà](#)
  - [Associare i fogli di stile ai documenti \(X\)HTML](#)
  - Note
- 

# Introduzione ai CSS

## 6. Note

In questa piccola guida si è cercato di fornire la base per la comprensione degli articoli e degli esempi che costituiscono i cuore di .ConStile. In questa guida ci si è basati soprattutto sulle regole dei CSS di livello 1 (da tempo sono disponibili anche i CSS di livello 2), cercando di evidenziare gli aspetti più utili e interessanti. Una guida completa ai CSS di livello 2 è disponibile presso il sito del [W3C](#) al seguente indirizzo: <http://www.w3.org/TR/CSS21>.

Questa guida è basata sulla [documentazione disponibile](#) presso il sito del [W3C](#), sulla guida ai CSS del [WDG](#) (Web Design Group) e sull'esperienza personale dell'autore (maturata visitando numerosi siti ed eseguendo numerosi esperimenti).

---

TOC

- [Indice](#)
  - [Cosa sono i fogli di stile](#)
  - [La sintassi delle regole](#)
  - [I selettori](#)
  - [Le proprietà](#)
  - [Associare i fogli di stile ai documenti \(X\)HTML](#)
  - Note
- 

*G.T.* -- ultima revisione: 03.10.2002

l'indirizzo di questa pagina è:  
[http://www.constile.org/tutorial/introduzione\\_ai\\_css/sezione\\_6.php](http://www.constile.org/tutorial/introduzione_ai_css/sezione_6.php)