

CSS - CASCADING STYLE SHEETS

CSS

- Standard W3C
- Definisce la presentazione del documento HTML (o in generale XML)
 - Cioè come un documento viene visualizzato in contesti diversi
- CSS è un linguaggio indipendente con la sua sintassi

Perché CSS

- **Controllo fine di carattere e layout**
 - a livello del mondo della stampa
- **Meno lavoro**
 - cambio l'aspetto di molte pagine in un solo foglio di stile
- **Migliore accessibilità**
 - nell'html è rimasto solo il contenuto semantico
- **Supportato da tutti i browser**
 - sicuramente dalla versione 2

Regole CSS

```
h1 { color: green; }
```

```
p { font-size: small; color: black; }
```

- **Selector (Selettore)**
 - identifica l'elemento o gli elementi a cui applicar lo stile
- **Declaration (Dichiarazione)**
 - costituita da una coppia proprietà valore separati dai :
 - fornisce l'istruzione di rendering
 - ogni dichiarazione è delimitata da un ;

declaration block

```
selector {  
  property1: value1;  
  property2: value2;  
  property3: value3;  
}
```

Come funziona

1. Scrivere un documento HTML
 2. Scrivere le regole CSS
 3. "Aggancio" le regole all'HTML
- Il browser visualizza lo stile definito

Stili nel head

- External style sheet

- collegato mediante il tag `<link>`
- file txt con estensione .css dove scriviamo le regole

```
<head>
<link rel="stylesheet" type="text/css" href="mystyle.css">
</head>
```

```
# mystyle.css > ...
```

```
1  body {
2      background-color: lightblue;
3  }
4
5  h1 {
6      color: navy;
7      margin-left: 20px;
8  }
```

- Internal style element

- regole nell'head tra i tag
`<style>...</style>`

```
<head>
<style>
body {
    background-color: linen;
}
h1 {
    color: maroon;
    margin-left: 40px;
}
</style>
</head>
```

Elemento `<link>`

- Allega un file alla pagina corrente
 - va posizionato nella sezione head del file html

- CSS

```
<head>  
  <link rel="stylesheet" type="text/css" href="theme.css">  
</head>
```

- Favicon

```
<link rel="icon" href="demo_icon.gif" type="image/gif">
```

Stile Inline

```
<!DOCTYPE html>
<html>
<head>
</head>
<body>
  <h1 style="color:red;">This is a Heading</h1>
  <p style="font-size: 1.1em; font-family: sans-serif;">This is a
paragraph.</p>
</body>
</html>
```

- Definisco lo stile usando l'attributo style
 - Viene applicata solo all'elemento
 - dichiarazioni multiple separate con ;
- Non andrebbe mai usato perché mischiano la struttura con la presentazione!!!
 - Si usa in casi molto particolari
 - per fare override mirati di regole esterne

SELETTORI SEMPLICI

Selettore elemento

- Seleziono tutti gli elementi di quel tipo

Esempio: seleziono tutti i **p**

```
p {  
    color: red;  
    text-align: center;  
}
```

```
body {  
    background-color: lightblue;  
}
```

```
h1 {  
    color: navy;  
    margin-left: 20px;  
}
```

```
p{  
    font-size: large;  
}
```

```
section{  
    margin-top: 10px;  
}
```

Selettore classe

- Seleziono tutti gli elementi di una classe
 - li individuo con l'attributo `class` ed un **identificatore**
 - uso come selettore l'identificatore preceduto dal punto

Esempio

```
<h1 class="center">Heading</h1>
<p class="center">A paragraph</p>
```

```
.center {
  text-align: center;
  color: red;
}
```

```
p.center {
  text-align: center;
  color: green;
}
```

https://www.w3schools.com/css/tryit.asp?filename=trycss_syntax_class

https://www.w3schools.com/css/tryit.asp?filename=trycss_syntax_element_class

Selettore id

- Selezione l'elemento con quell'id
 - scrivo l'id preceduto dal cancelletto (#)
- Esempio: `<p id="para1">...</p>`

```
#para1 {  
    text-align: center;  
    color: red;  
}
```

Esempio

```
<article>
  <p>Ciao a tutti</p>
  <p>Abbiamo iniziato il CSS</p>
  <p>come fai a cambiare lo stile a me?</p>
  <p>e me no?</p>
  <p>ma me si!!!!</p>
  <p>Io voglio uno stile tutto mio</p>
  <section>
    <p>non mi scordate di me
      dentro la section</p>
  </section>
</article>
```

- Quale selettore CSS permetterebbe di modellare il contenuto del solo 6° <p> tag senza modificare l'HTML?
- Quale selettore CSS ti permetterebbe di modellare il contenuto del 3° e 5° <p> tag senza modificare l'HTML?

Esempio

```
<article>
  <p>Ciao a tutti</p>
  <p>Abbiamo iniziato il CSS</p>
  <p class="stile-cl">come fai a cambiare lo stile a me?</p>
  <p>e me no?</p>
  <p class="stile-cl">ma me si!!!</p>
  <p id="stile-id">Io voglio uno stile tutto mio</p>
  <section>
    <p>non mi scordate di me
      dentro la section</p>
  </section>
</article>
```

```
.stile-cl{
  color: ■red;
}

#stile-id{
  color: ■green;
}
```

Raggruppare selettori

- Si possono anteporre più selettori ad un blocco di dichiarazioni

```
h1 {
  text-align: center;
  color: red;
}

h2 {
  text-align: center;
  color: red;
}

p {
  text-align: center;
  color: red;
}
```



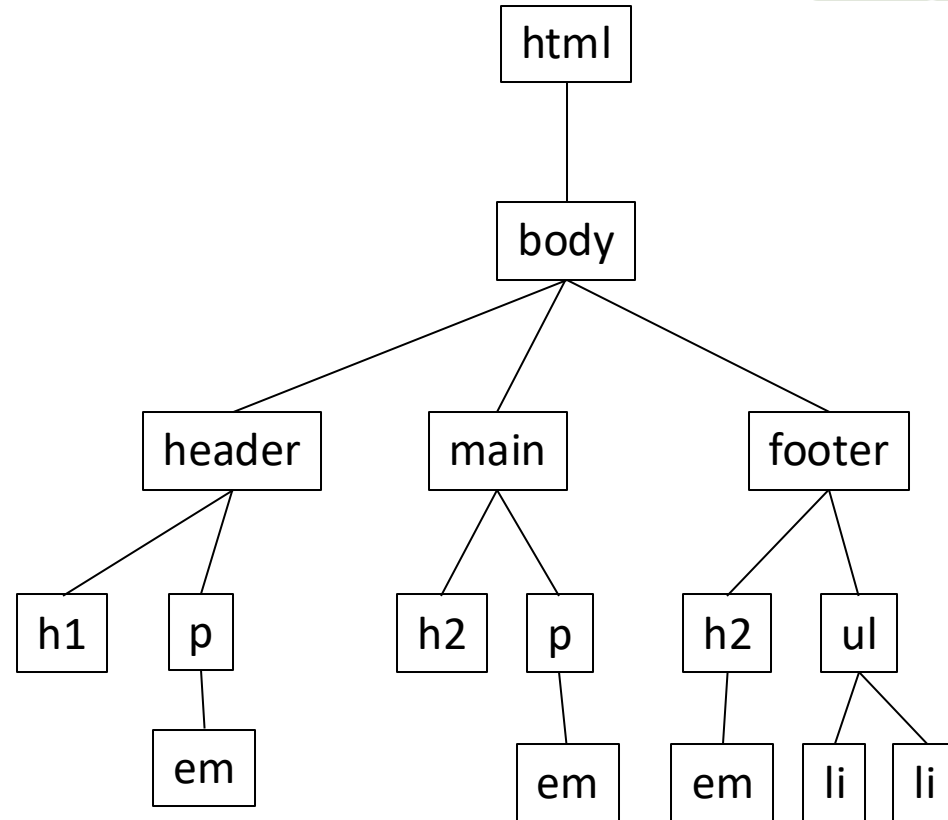
```
h1, h2, p {
  text-align: center;
  color: red;
}
```

I selettori si radunano con la virgola!!

SELETTORI E DOM

Relazioni nel DOM

- **Descendant** - discendenti
 - Gli elementi contenuti in un elemento sono i suoi discendenti
- **Child** - figli
 - Discendenti diretti e viceversa si dice genitore (parent)
- **Ancestor** - antenati
 - Gli elementi sopra nell'albero
- **Parent** – genitore
 - Elementi direttamente sopra
- **Sibling** – fratelli
 - Elementi con lo stesso parent



Selettori composti

1. Selettori per **descendant** (spazio)
2. Selettori per **child** (>)
3. Selettori per **Adjacent sibling** (+)
 - il fratello immediatamente **successivo**
4. Selettori per **General sibling** (~)
 - tutti i fratelli **successivi**

Selezionare i discendenti 1

p em {color: grey;}

header p {font-size: 20px;}

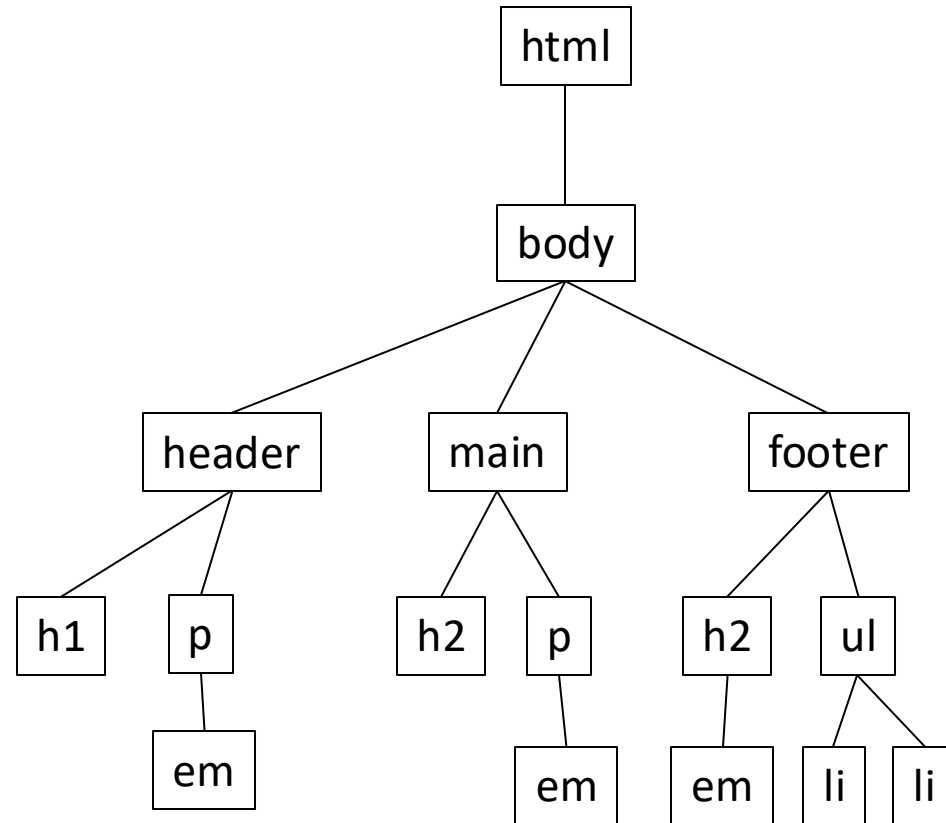
main p {font-size: 16px;}

footer h2 {...}

main h2 {...}

header em {...}

main em {...}



Selezionare i discendenti 2

p.ingredienti {...}

div.ricetta {...}

.ricetta p {...}

.ricetta h2 {...}

.ricetta .ingredienti {...}

div.ricetta p.ingredienti {...}

```
<div class="ricetta">
  <h2>...</h2>
  <p>...</p>
  <p class="ingredienti"> ...</p>
</div>

<div class="ricetta">
  <h2>...</h2>
  <p>...</p>
  <p class="ingredienti"> ...</p>
</div>
```

Selezionare i child

```
div > p {  
    background-color: yellow;  
}
```

- Tutti i **p** figli (child) di un **div**

https://www.w3schools.com/css/tryit.asp?filename=trycss_sel_element_gt

Selezionare fratelli

```
div + p {  
  background-color: yellow;  
}
```

- Tutti gli elementi **p** immediatamente successivi ad un **div**
https://www.w3schools.com/css/tryit.asp?filename=trycss_sel_element_pluss

```
div ~ p {  
  background-color: yellow;  
}
```

- Tutti gli elementi **p** fratelli successivi degli elementi **div**
https://www.w3schools.com/css/tryit.asp?filename=trycss_sel_element_tilde

Pseudo Classi

- Una pseudo-classe viene utilizzata per definire uno stato speciale di un elemento.
 - mouse over
 - visited e unvisited link
 - focus

Esempio: stili ancora

https://www.w3schools.com/css/tryit.asp?filename=trycss_link

Elenco

https://www.w3schools.com/css/css_pseudo_classes.asp

```
selector:pseudo-class {  
    property:value;  
}
```

```
/* visited link */  
a:visited {  
    color: #00FF00;  
}
```

```
/* mouse over link */  
a:hover {  
    color: #FF00FF;  
}
```

Altre pseudo classi

- :root
- :only-child
- :empty
- :first-of-type
- :first-child
- :last-of-type
- :last-child
- :only-of-type
- :nth-child()
- :nth-last-child()
- :nth-of-type()
- :nth-last-of-type()

https://www.w3schools.com/css/css_pseudo_classes.asp

Pseudo elementi

- Uno pseudo-elemento viene utilizzato per dare uno stile alle parti specifiche di un elemento.
 - Disegna la prima lettera o riga di un elemento
 - Inserire un contenuto prima o dopo un elemento
- Selettori
 - ::after
 - ::before
 - ::first-letter
 - ::first-line
 - ::selection

```
selector::pseudo-element {  
  property:value;  
}
```

```
p::first-letter {  
  color: #ff0000;  
  font-size: xx-large;  
}
```

https://www.w3schools.com/css/css_pseudo_elements.asp

Selettori con attributi

- È possibile impostare lo stile su elementi HTML con attributi o valori di attributo specifici.

```
[attribute] {  
  property: value;  
}
```

```
img[alt] {  
  background-color: grey;  
}
```

```
[attribute=value] {  
  property: value;  
}
```

```
a[target="_blank"] {  
  color: red;  
}
```

- è possibile selezionare parte del valore:
 - parola, inizio, fine, etc.

https://www.w3schools.com/css/css_attribute_selectors.asp

Selector	Example	Example description
<code>[<u>attribute</u>]</code>	<code>[target]</code>	Selects all elements with a target attribute
<code>[<u>attribute=value</u>]</code>	<code>[target="_blank"]</code>	Selects all elements with target="_blank"
<code>[<u>attribute~=value</u>]</code>	<code>[title~="flower"]</code>	Selects all elements with a title attribute that contains a space-separated list of words, one of which is "flower"
<code>[<u>attribute =value</u>]</code>	<code>[lang "en"]</code>	Selects all elements with a lang attribute value starting with "en"
<code>[<u>attribute^=value</u>]</code>	<code>a[href^="https"]</code>	Selects all <a> elements with a href attribute value starting with "https"
<code>[<u>attribute\$=value</u>]</code>	<code>a[href\$=".pdf"]</code>	Selects all <a> elements with a href attribute value ending with ".pdf"
<code>[<u>attribute*=value</u>]</code>	<code>a[href*="w3schools"]</code>	Selects all <a> elements with a href attribute value containing the substring "w3schools"

https://www.w3schools.com/css/css_attribute_selectors.asp

EREDITARIETÀ

Ereditarietà

- Alcune proprietà sono ereditate dai **discendenti**
 - la dichiarazione color viene passa allo span
 - le altre no

```
<!DOCTYPE html>
<html>
<head>
<style>

p{
  color:white;
  background:grey;
  border: medium solid black;
}

</style>
</head>
<body>
  <h1>This is a Heading</h1>
  <p>This is a paragraph <span>with a span</span> inside.</p>
</body>
</html>
```

This is a Heading

This is a paragraph with a span inside.

DOM - Albero degli elementi

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
  <meta charset="UTF-8">
```

```
  <title>Titolo</title>
```

```
  <style>
```

```
    h1{ color: red;}
```

```
  </style>
```

```
</head>
```

```
<body>
```

```
  <h1>Titolo 1</h1>
```

```
  <p>paragrafo</p>
```

```
  <p>paragrafo con un elemento <em>importante</em></p>
```

```
  <h2>Titolo 2</h2>
```

```
  <p>paragrafo con immagine<img src=""></p>
```

```
  <p>paragrafo</p>
```

```
  <p>paragrafo con due <em>elementi</em> <em>importanti</em></p>
```

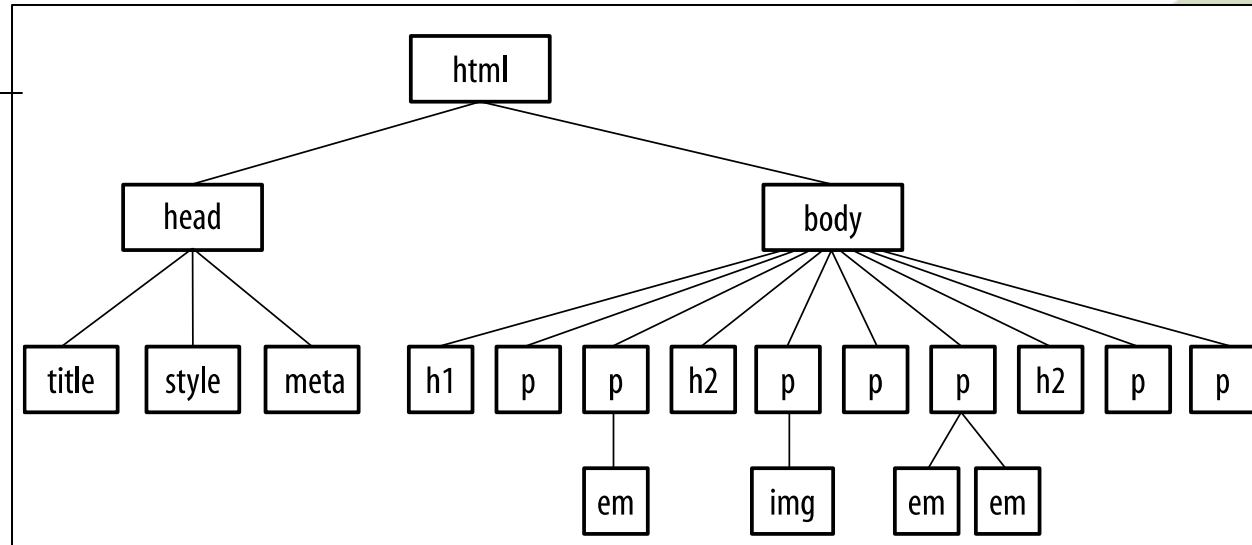
```
  <h2>Titolo 2</h2>
```

```
  <p>paragrafo</p>
```

```
  <p>paragrafo</p>
```

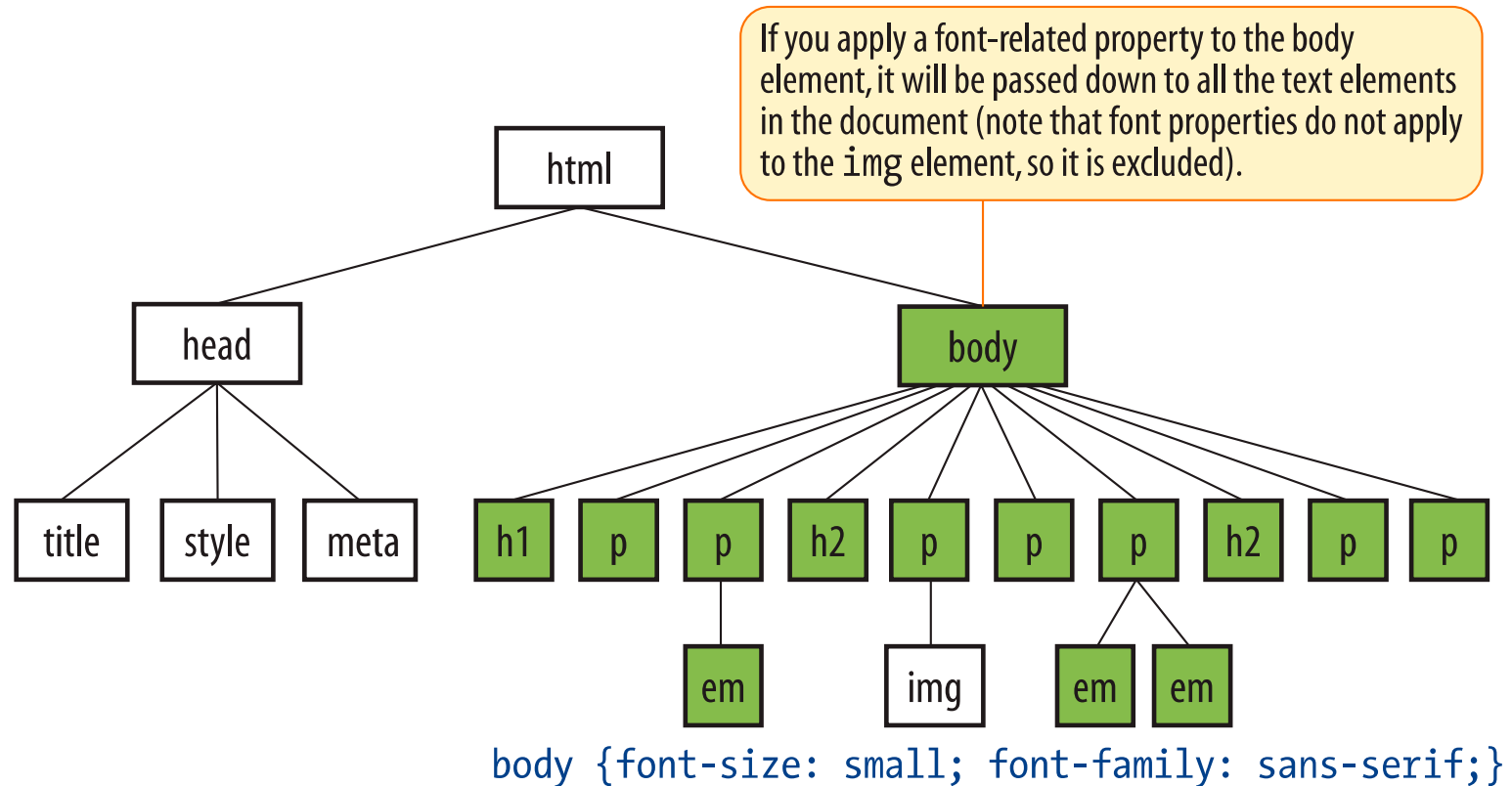
```
</body>
```

```
</html>
```



Dichiarazioni ereditate

- Gli stili relativi ai font sono ereditati dai descendant



Proprietà ereditate

- azimuth
- border-collapse
- border-spacing
- caption-side
- color
- cursor
- direction
- elevation
- empty-cells
- font-family
- font-size
- font-style
- font-variant
- font-weight
- font
- letter-spacing
- line-height
- list-style-image
- list-style-position
- list-style-type
- list-style
- orphans
- pitch-range
- pitch
- quotes
- stress
- text-align
- text-indent
- text-transform
- visibility
- voice-family
- white-space
- widows
- word-spacing

Conflitti

- I **conflitti** di dichiarazione sono la normalità
 - Posso applicare più stili allo stesso elemento
 - Alcune proprietà le eredito
- Es: conflitto diretto fra due regole o proprietà

```
p{
  color: blue;
  margin-left: 30px;
}

p{
  color: green;
}
```

```
p{
  color: blue;
  color: green;
}
```

La seconda dichiarazione cancella la prima!!!

- Domina quello più vicino all'elemento
- Il p sarà verde.

ATTENZIONE
Non è sempre così!!!!

Esempio 1

```
<!DOCTYPE html>
<html>
<head>
<style>

p{
  color: blue;
  margin-left: 30px;
}

p{
  color: green;
}

</style>
</head>
<body>

<h1>This is a Heading</h1>
<p>This is a paragraph.</p>
<p>This is another paragraph.</p>

</body>
</html>
```

- I paragrafi sono verdi
- Il margine rimane

This is a Heading

This is a paragraph.

This is another paragraph.

```
element.style {
}

p {
  color: ■ green;
}

p {
  color: ■ blue;
  margin-left: 30px;
}
```

devtool

Esempio 2

```
<!DOCTYPE html>
```

```
<html>
<head>
<style>
```

```
p{
  color: blue;
  margin-left: 30px;
}
```

```
p{
  color: green;
}
```

```
</style>
</head>
<body>
```

```
<h1>This is a Heading</h1>
<p>This is a paragraph.</p>
<p style="color: red;">This is another par

</body>
</html>
```

- Il secondo paragrafo è rosso!!
 - domina lo style inline

This is a Heading

This is a paragraph.

This is another paragraph.

```
element.style {
  color: red;
}

p {
  color: green;
}

p {
  color: blue;
  margin-left: 30px;
}
```

devtool

Esempio 3

```
<!DOCTYPE html>
<html>
<head>
<style>

#para3{
  color: orange;
}

.red{
  color: red;
}

p{
  color: green;
}

</style>
</head>
<body>

<h1>This is a Heading</h1>
<p>This is a paragraph.</p>
<p class="red">This is a paragraph with class.</p>
<p id="para3" class="red">This is a paragraph with id and class.</p>

</body>
</html>
```

This is a Heading

This is a paragraph.

This is a paragraph with class.

This is a paragraph with id and class.

Non tutti i selettori sono uguali, alcuni hanno più "peso" di altri

Esempio 1

```
<!DOCTYPE html>
<html>
<head>
<style>
p{
  color: red;
}
p span{
  color: green;
}

</style>
</head>
<body>

<h1>This is a Heading</h1>
<p>This is a paragraph.</p>
<p>This is a <span>paragraph with a span</span> inside.</p>

</body>
</html>
```

This is a Heading

This is a paragraph.

This is a paragraph with a span inside.

Esempio 2

```
<!DOCTYPE html>
<html>
<head>
<style>
.news p{
  color: green;
}

.blog p{
  color: red;
}

</style>
</head>
<body>
<h1>This is a Heading</h1>
<section class="news">
  <p>This is a paragraph in sec news.</p>
  <p>This is a another paragraph in sec news.</p>
</section>
<section class="blog">
  <p>This is a paragraph in sec blog.</p>
  <p>This is a another paragraph in sec blog.</p>
</section>
</body>
</html>
```

This is a Heading

This is a paragraph in sec news.

This is a another paragraph in sec news.

This is a paragraph in sec blog.

This is a another paragraph in sec blog.

Esempio 3

```
<!DOCTYPE html>
<html>
<head>
<style>
#news p{
    color: green;
}

#blog p{
    color: red;
}

</style>
</head>
<body>

<h1>This is a Heading</h1>
<section id="news">
    <p>This is a paragraph in sec news.</p>
    <p>This is a another paragraph in sec news.</p>
</section>
<section id="blog">
    <p>This is a paragraph in sec blog.</p>
    <p>This is a another paragraph in sec blog.</p>
</section>
</body>
</html>
```

This is a Heading

This is a paragraph in sec news.

This is a another paragraph in sec news.

This is a paragraph in sec blog.

This is a another paragraph in sec blog.

Esempio 4

```
<!DOCTYPE html>
<html>
<head>
<style>
section{
  color: grey;
}

.news .mark{
  font-weight: bold;
}

.blog .mark{
  color: orange;
}

</style>
</head>
<body>

<h1>This is a Heading</h1>
<section class="news">
  <p>This is a paragraph in sec news.</p>
  <p class="mark">This is a another paragraph in sec news.</p>
</section>
<section class="blog">
  <p class="mark">This is a paragraph in sec blog.</p>
  <p>This is a another <span class="mark">paragraph</span> in sec blog.
</p>
</section>
</body>
</html>
```

This is a Heading

This is a paragraph in sec news.

This is a another paragraph in sec news.

This is a paragraph in sec blog.

This is a another **paragraph** in sec blog.

Esempio 5

```
<!DOCTYPE html>
<html>
<head>
<style>
.news.mark{
  font-weight: bold;
}

.news .mark{
  color: orange;
  font-weight: normal;
}

</style>
</head>
<body>

<h1>This is a Heading</h1>
<section class="news mark">
  <p>This is a paragraph in sec news.</p>
  <p class="mark">This is a another paragraph in sec news.</p>
</section>
</section>
</body>
</html>
```

This is a Heading

This is a paragraph in sec news.

This is a another paragraph in sec news.

Classi multiple

```
<!DOCTYPE html>
<html>
<head>
<style>
.btn {
  font-size: 18px; color: white; margin: 10px;
  padding: 10px; display: block; width: 100px;
}

.btn-danger {
  background: red;
}

.btn-success {
  background: green;
}

</style>
</head>
<body>

<h1>This is a Heading</h1>
<a class="btn btn-danger" href="">Pericolo</a>
<a class="btn btn-success" href="">Hai vinto</a>

</body>
</html>
```

This is a Heading

Pericolo

Hai vinto

!important

```
<!DOCTYPE html>
<html>
<head>
<style>

#para3{
  color: orange;
}

.red{
  color: red;
}

p{
  color: green !important;
}

</style>
</head>
<body>

<h1>This is a Heading</h1>
<p>This is a paragraph.</p>
<p class="red">This is a paragraph with class.</p>
<p id="para3" class="red">This is a paragraph with
id and class.</p>

</body>
</html>
```

- Se non voglio che una regola venga sovrascritta la posso dichiarare **!important**
- Viene usato in casi molto particolari
 - di solito è sempre possibile evitarlo

This is a Heading

This is a paragraph.

This is a paragraph with class.

This is a paragraph with id and class.

Cascade

- Il Cascade "**cascata**" è un algoritmo che definisce come combinare valori di proprietà provenienti da fonti diverse.
- Stili:
 - **browser** - user-agent stylesheets
 - **author** - quello che scriviamo noi
 - **reader** - the user of the browser, may have a custom style sheet to tailor its experience.

<https://developer.mozilla.org/it/docs/Web/CSS/Cascade>

Cascading Order

Precedence Order (low to high)	Origin	Importance
1	user-agent (browser)	normal
2	user	normal
3	author (developer)	normal
4	CSS @keyframe animations	
5	author (developer)	!important
6	user	!important
7	user-agent (browser)	!important
8	CSS transitions	

https://developer.mozilla.org/it/docs/Web/CSS/Cascade#cascading_order

Calcolo della specificità

- Ad ogni dichiarazione è attribuita una specificità misurata con quattro valori [a,b,c,d]
 - ogni valore è indipendente dagli altri
 - "a" è il valore più importante e "d" il meno importante
 - si confrontano i valori più importanti e se uguali si passa a quelli successivi altrimenti termina
 - Es. 0,1,0,0 è più specifico di 0,0,5,5
- Calcolo dei valori
 - a. 1 se la dichiarazione è inline, 0 altrimenti
 - b. numero di selettori id
 - c. numero di selettori di classe, attributo o pseudo-classe
 - d. numero di selettori elemento o pseudo elemento

Esempi

- Dallo standard **(6.4.3 Calculating a selector's specificity)**

<code>* {}</code>	<code>/* a=0 b=0 c=0 d=0 -> specificity = 0,0,0,0 */</code>
<code>li {}</code>	<code>/* a=0 b=0 c=0 d=1 -> specificity = 0,0,0,1 */</code>
<code>li::first-line {}</code>	<code>/* a=0 b=0 c=0 d=2 -> specificity = 0,0,0,2 */</code>
<code>ul li {}</code>	<code>/* a=0 b=0 c=0 d=2 -> specificity = 0,0,0,2 */</code>
<code>ul ol+li {}</code>	<code>/* a=0 b=0 c=0 d=3 -> specificity = 0,0,0,3 */</code>
<code>h1 + *[rel=up]{}</code>	<code>/* a=0 b=0 c=1 d=1 -> specificity = 0,0,1,1 */</code>
<code>ul ol li.red {}</code>	<code>/* a=0 b=0 c=1 d=3 -> specificity = 0,0,1,3 */</code>
<code>li.red.level {}</code>	<code>/* a=0 b=0 c=2 d=1 -> specificity = 0,0,2,1 */</code>
<code>#x34y {}</code>	<code>/* a=0 b=1 c=0 d=0 -> specificity = 0,1,0,0 */</code>
<code>style=""</code>	<code>/* a=1 b=0 c=0 d=0 -> specificity = 1,0,0,0 */</code>

Esempi

- Dallo standard (6.4.3 Calculating a selector's specificity)

<code>* {}</code>	<code>/* a=0 b=0 c=0 d=0 -> specificity = 0,0,0,0 */</code>
<code>li</code>	<code>/* a=0 b=0 c=0 d=1 -> specificity = 0,0,0,1 */</code>
<code>html body div#pagewrap ul#summer-drinks li.favorite</code>	<code>/* a=0 b=0 c=0 d=4 -> specificity = 0,0,0,4 */</code>
<code>ul li {}</code>	<code>/* a=0 b=0 c=0 d=2 -> specificity = 0,0,0,2 */</code>
<code>ul ol+li {}</code>	<code>/* a=0 b=0 c=0 d=3 -> specificity = 0,0,0,3 */</code>
<code>h1 + *[rel=up]{}</code>	<code>/* a=0 b=0 c=1 d=1 -> specificity = 0,0,1,1 */</code>
<code>ul ol li.red {}</code>	<code>/* a=0 b=0 c=1 d=3 -> specificity = 0,0,1,3 */</code>
<code>li.red.level {}</code>	<code>/* a=0 b=0 c=2 d=1 -> specificity = 0,0,2,1 */</code>
<code>#x34y {}</code>	<code>/* a=0 b=1 c=0 d=0 -> specificity = 0,1,0,0 */</code>
<code>style=""</code>	<code>/* a=1 b=0 c=0 d=0 -> specificity = 1,0,0,0 */</code>