
Il Linguaggio CSS

Lorenzo Sommaruga

Nov 2018

Obiettivi

- (BI-2) Comprendere l'utilità del linguaggio CSS e i suoi vantaggi
- (BI-2) Capire come i fogli di stile sono strutturati
- (BI-2) Capire come si associano a pagine HTML e come si relazionano a diversi media
- (BI-3) Applicare le tecniche apprese in applicazioni web HTML

Contenuti

- CSS: perché e cos'è
 - **vantaggi nell'utilizzo di CSS**
 - **cosa sono i fogli di stile**
 - **specifiche CSS**
- Come sono fatti i fogli di stile CSS
 - **le regole**
 - **i selettori**
- CSS e HTML
 - **come si associano ad una pagina HTML**
 - **fogli di stile per diversi media**
 - **fogli di stile alternativi**
 - **compatibilità tra i browser**

CSS: perché e cos'è

Struttura, presentazione, contenuto

- La **struttura** di un documento è come questo è organizzato logicamente (e.g. capitoli, sezioni, etc.)
- La **presentazione** di un documento è il modo in cui il documento è riprodotto (e.g. come stampato, su schermo, etc.)
- Il **contenuto** di un documento è ciò che questo comunica all'utente attraverso linguaggio naturale, immagini, suoni, filmati, animazioni, etc.

A cosa serve CSS (1/2)

- Separare la presentazione

A cosa serve CSS (2/2)

- In X(HTML) ci sono elementi strutturali ed elementi di presentazione
 - un elemento che specifica la struttura, e.g. *h1*, *p*, *blockquote* in (X)HTML, si chiama **elemento strutturale**
 - un elemento che specifica la presentazione del documento, e.g. *i*, *b*, *em*, (*font*, *center*) in (X)HTML, si chiama **elemento di presentazione**
- Pur essendo presenti in (X)HTML elementi e attributi di presentazione, è buona regola NON usarli. La maggior parte sono stati deprecati da HTML5
- Per questo è nato CSS
- CSS ha lo scopo di definire gli aspetti visuali, i.e. di presentazione di una pagina (X)HTML

Aspetti visuali o di presentazione

- Sfondo
- Margini, bordi
- Colori
- Spaziatura, stili del testo
- Posizione degli elementi
- Visualizzazione di liste e tabelle
- ...

Usando HTML5 si evita di avere elementi ed attributi di presentazione

Vantaggi nell'utilizzo di CSS

- Riduce il peso della pagina e quindi i tempi di caricamento
- Consente di specificare presentazioni alternative (e.g. schermo, stampa, sintetizzatori vocali, etc.)
- Consente di ottenere un codice HTML più lineare e pulito

Cosa sono i fogli di stile CSS

- I fogli di stile sono documenti che descrivono come trattare la formattazione
- Offrono un controllo preciso sulla presentazione di pagine (X)HTML
- Consentono di separare i contenuti dalla presentazione
- CSS (Cascading Stylesheet Language)
 - meccanismo di fogli di stile che permette di associare uno stile a documenti strutturati (HTML e XML)

Specifiche CSS: le diverse versioni

Ref: http://www.w3.org/TR/#tr_CSS

<http://www.w3.org/Style/CSS/current-work.../learning>

- **CSS1:** Cascading Style Sheets, level 1
W3C Rec. Dic 1996, revised Gen. 1999, Apr. 1998
<http://www.w3.org/TR/REC-CSS1> prima versione
- **CSS2:** Cascading Style Sheets Level 2 Revision 1 (CSS 2.1) Specification
W3C Rec. 07 June 2011 (W3C Rec. May1998)
<http://www.w3.org/TR/REC-CSS2/>
 - basato su CSS1
 - aggiunge supporto per diversi dispositivi (browser visuali, dispositivi audio, stampanti, dispositivi braille, ecc)
 - migliore posizionamento di elementi nella pagina
 - nuovi font scaricabili, aggiunta di contenuti, etc.
- **CSS3:** Cascading Style Sheets, level 3
 - <http://www.w3.org/TR/CSS/>
 - <http://www.w3.org/TR/css3-roadmap/>
 - modularizzazione di CSS → maggior flessibilità
- **CSS Profiles:** Mobile 2.0, Print, TV (WGNote 2014)

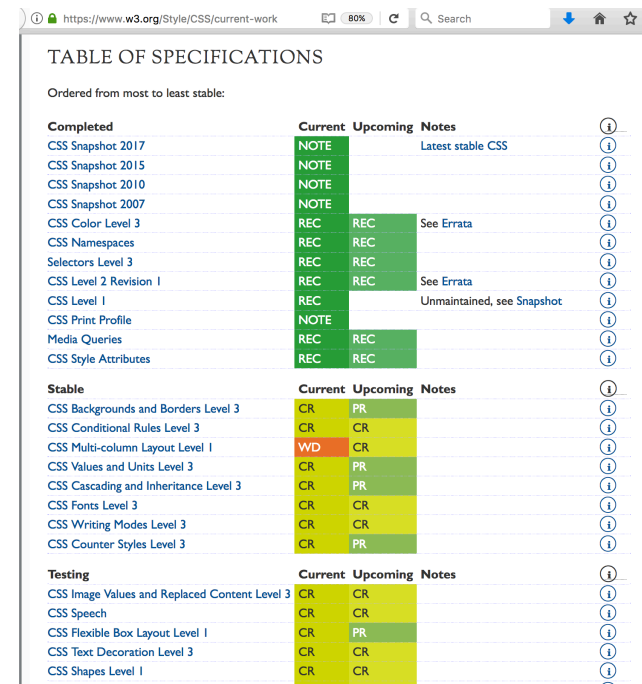


TABLE OF SPECIFICATIONS

Ordered from most to least stable:

	Current	Upcoming	Notes	
Completed				
CSS Snapshot 2017	NOTE		Latest stable CSS	(i)
CSS Snapshot 2015	NOTE			(i)
CSS Snapshot 2010	NOTE			(i)
CSS Snapshot 2007	NOTE			(i)
CSS Color Level 3	REC	REC	See Errata	(i)
CSS Namespaces	REC	REC		(i)
Selectors Level 3	REC	REC		(i)
CSS Level 2 Revision 1	REC	REC	See Errata	(i)
CSS Level 1	REC		Unmaintained, see Snapshot	(i)
CSS Print Profile	NOTE			(i)
Media Queries	REC	REC		(i)
CSS Style Attributes	REC	REC		(i)
Stable				
CSS Backgrounds and Borders Level 3	CR	PR		(i)
CSS Conditional Rules Level 3	CR	CR		(i)
CSS Multi-column Layout Level 1	WD	CR		(i)
CSS Values and Units Level 3	CR	PR		(i)
CSS Cascading and Inheritance Level 3	CR	PR		(i)
CSS Fonts Level 3	CR	CR		(i)
CSS Writing Modes Level 3	CR	CR		(i)
CSS Counter Styles Level 3	CR	PR		(i)
Testing				
CSS Image Values and Replaced Content Level 3	CR	CR		(i)
CSS Speech	CR	CR		(i)
CSS Flexible Box Layout Level 1	CR	PR		(i)
CSS Text Decoration Level 3	CR	CR		(i)
CSS Shapes Level 1	CR	CR		(i)

Come sono fatti i fogli di stile CSS

Come sono fatti i CSS

- CSS = sequenza di regole da applicare ad un documento (X)HTML

```
body    {background: lightyellow }  
  
h1      {font-size: 18pt;  
        font-weight: bold;  
        color: maroon}  
  
P       {font-size: 12pt;  
        font-family: Arial;  
        color:darkblue}
```

- Una regola serve per associare certe proprietà ad uno o più elementi

Una regola CSS

selettore

h1

blocco della dichiarazione

{color: red}

proprietà

valore

- Il **selettore** serve a definire la parte del documento a cui verrà applicata la regola
- Il **blocco della dichiarazione** è composto da coppie *proprietà: valore*

CSS3 Selectors

Selectors Level 3, W3C Recommendation 29 September 2011

Ref.: <http://www.w3.org/TR/css3-selectors/>

Pattern	Meaning	Described in section	First defined in CSS level
*	any element	Universal selector	2
E	an element of type E	Type selector	1
E[foo]	an E element with a "foo" attribute	Attribute selectors	2
E[foo="bar"]	an E element whose "foo" attribute value is exactly equal to "bar"	Attribute selectors	2
E[foo~="bar"]	an E element whose "foo" attribute value is a list of whitespace-separated values, one of which is exactly equal to "bar"	Attribute selectors	2
E[foo^="bar"]	an E element whose "foo" attribute value begins exactly with the string "bar"	Attribute selectors	3
E[foo\$="bar"]	an E element whose "foo" attribute value ends exactly with the string "bar"	Attribute selectors	3
E[foo*="bar"]	an E element whose "foo" attribute value contains the substring "bar"	Attribute selectors	3
E[foo="en"]	an E element whose "foo" attribute has a hyphen-separated list of values beginning (from the left) with "en"	Attribute selectors	2
E:root	an E element, root of the document	Structural pseudo-classes	3
E:nth-child(n)	an E element, the n-th child of its parent	Structural pseudo-classes	3
E:nth-last-child(n)	an E element, the n-th child of its parent, counting from the last one	Structural pseudo-classes	3
E:nth-of-type(n)	an E element, the n-th sibling of its type	Structural pseudo-classes	3
E:nth-last-of-type(n)	an E element, the n-th sibling of its type, counting from the last one	Structural pseudo-classes	3
E:first-child	an E element, first child of its parent	Structural pseudo-classes	2
E:last-child	an E element, last child of its parent	Structural pseudo-classes	3
E:first-of-type	an E element, first sibling of its type	Structural pseudo-classes	3
E:last-of-type	an E element, last sibling of its type	Structural pseudo-classes	3
E:only-child	an E element, only child of its parent	Structural pseudo-classes	3
E:only-of-type	an E element, only sibling of its type	Structural pseudo-classes	3
E:empty	an E element that has no children (including text nodes)	Structural pseudo-classes	3
E:link	an E element being the source anchor of a hyperlink of which the target is not yet visited (:link) or already visited (:visited)	The link pseudo-classes	1
E:active	an E element during certain user actions	The user action pseudo-classes	1 and 2
E:hover			
E:focus			
E:target	an E element being the target of the referring URI	The target pseudo-class	3
E:lang(fr)	an element of type E in language "fr" (the document language specifies how language is determined)	The :lang() pseudo-class	2
E:enabled	a user interface element E which is enabled or disabled	The UI element states pseudo-classes	3
E:disabled			
E:checked	a user interface element E which is checked (for instance a radio-button or checkbox)	The UI element states pseudo-classes	3
E::first-line	the first formatted line of an E element	The ::first-line pseudo-element	1
E::first-letter	the first formatted letter of an E element	The ::first-letter pseudo-element	1
E::before	generated content before an E element	The ::before pseudo-element	2
E::after	generated content after an E element	The ::after pseudo-element	2
E.warning	an E element whose class is "warning" (the document language specifies how class is determined).	Class selectors	1
E#myid	an E element with ID equal to "myid".	ID selectors	1
E:not(s)	an E element that does not match simple selector s	Negation pseudo-class	3
E F	an F element descendant of an E element	Descendant combinator	1
E > F	an F element child of an E element	Child combinator	2
E + F	an F element immediately preceded by an E element	Adjacent sibling combinator	2
E ~ F	an F element preceded by an E element	General sibling combinator	3

Esempi di regole

Elementi come selettori

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



applica il colore rosso e allinea al centro tutti gli elementi *h1*

```
h2, h3 {font-size: 18pt;  
font-family: Times;  
font-weight: bold}
```



applica il font “Times” 18 punti e grassetto a tutti gli elementi *h2* e *h3* (*raggruppamento di elementi*)

Esempi di regole

Elementi come selettori: selettore universale

* {color: red}



applica il colore rosso a tutti gli elementi

```
div p {color: blue}
```

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

Esempi di regole

Attributi come selettori

```
input [ id ] {background: red;}
```



- applica uno sfondo rosso a tutti gli elementi *input* con un attributo *id*

```
input [ id = "text" ] {background: red;}
```



- applica uno sfondo rosso a tutti gli elementi *input* con un attributo *id* con valore "text"

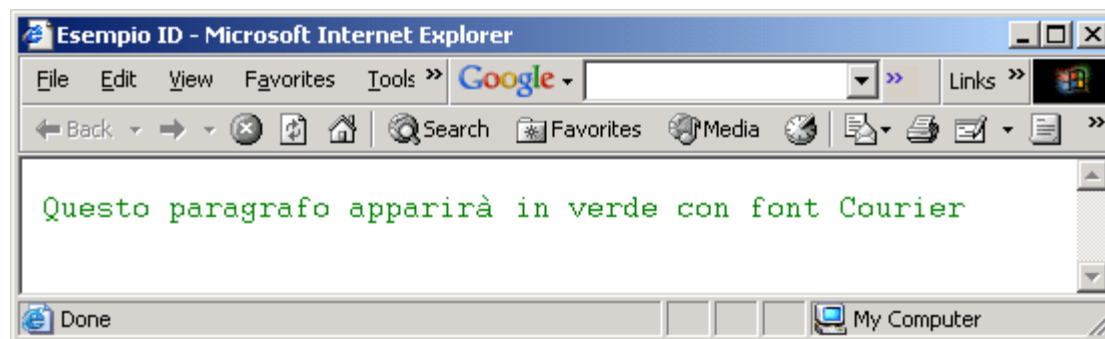
Esempi di regole

Attributo *id* come selettore

[esempiID.htm](#)

```
<p id="primo">  
Questo paragrafo apparirà in verde con font Courier</p>
```

```
#primo {color: green; font-family: Courier}
```



- l'elemento con attributo *id* uguale a *primo* sarà mostrato in verde e con font "Courier"
- essendo l'id univoco non serve specificare il nome dell'elemento

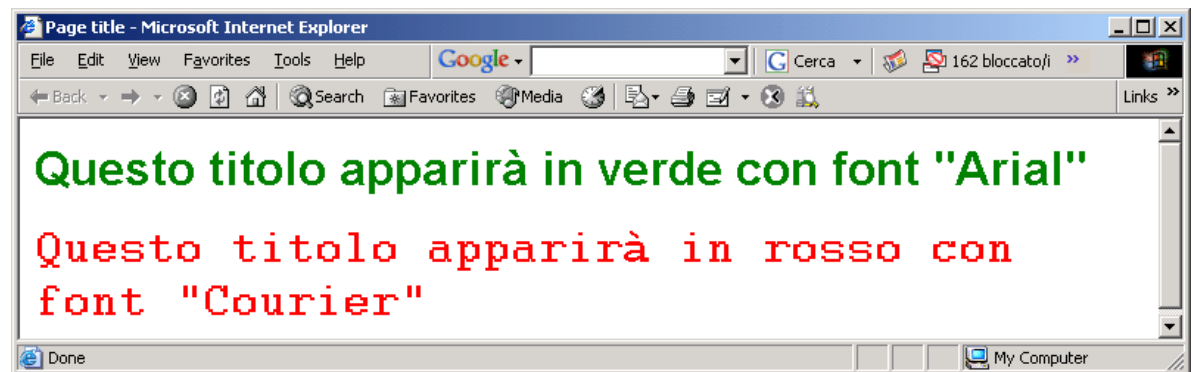
Esempi di regole

Attributo **class** come selettore

[esempioClass.html](#)

```
<h1 class="pippo">Questo ... in verde con font "Arial"</h1>  
<h1 class="pluto">Questo ... in rosso con font "Courier"</h1>
```

```
h1.pippo    {color: green; font-family: Arial}  
h1.pluto    {color: red; font-family: "Courier"}
```



```
.mirtillo {color: red}
```

tutti gli elementi con attributo **class** uguale a *mirtillo* saranno mostrati in rosso

Esempi di regole

pseudo-classe :link

- Una pseudo-classe definisce un particolare stato di un elemento
 - *:link: si applica solo all'elemento <a> che abbia l'attributo href*
- i link non visitati saranno di colore blu, quelli visitati rossi

a:link	{color: blue}
a:visited	{color: red}

- solo i link non visitati con attributo class="collegamento" saranno di colore verde

a.collegamento:link	{color: green}
---------------------	----------------

Commenti

- Oltre alle regole, i fogli di stile CSS possono contenere commenti, racchiusi tra `/* ... */`

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

```
/* applica il colore rosso e allinea al centro tutti gli elementi h1 */
```

CSS e (X)HTML

Come associare fogli di stile a documenti (X)HTML

- Si incorporano nella pagina (X)HTML (fogli di stile interni)
 - in un unico blocco (embedded o incorporati)
 - come istruzioni di stile dei singoli elementi (inline o inlinea)
- Si mettono in un file esterno collegato alla pagina (X)HTML (fogli di stile esterni)

Esempio foglio di stile interno in un unico blocco (embedded)

[esempioCSSinternoEmbedded.html](#)

```
<html xmlns="http://www.w3.org/1999/xhtml" lang="it">
  <head>
    <meta content="text/html; charset=iso-8859-1" http-equiv="Content-Type" />
    <style type="text/css">
      body {background: lightyellow }
      h1 {font-size: 18pt;
        font-weight: bold;
        color: maroon}
      p {font-size: 12pt;
        font-family: Arial;
        color: darkblue}
    </style>
  </head>
  <body>
    <h1>Lezione su CSS</h1>
    <p>Un foglio di stile indica come deve apparire una pagina HTML</p>
    <p>CSS è un linguaggio specifico per definire fogli di stile</p>
  </body>
</html>
```

Foglio di stile interno (embedded)

attributi *type* e *media*

- Elemento *style* dentro *head* e gli attributi *type* e *media*
 - *type*: obbligatorio, indica il linguaggio usato (“text/css”)
 - *media*: opzionale, indica a quale media (screen, print, etc.) applicare il foglio di stile

Esempio foglio di stile interno

istruzioni di stile associati ai singoli elementi (inline)

[esempioCSSinternoInline.html](#)

```
<html xmlns="http://www.w3.org/1999/xhtml" lang="it">
  <head>
    <meta content="text/html; charset=iso-8859-1" http-equiv="Content-Type" />
    <title>CSS interno inline</title>
  </head>
  <body style="background: lightyellow">
    <h1 style="font-size:18pt; font-weight: bold; color: maroon">
      Lezione su CSS
    </h1>
    <p style="font-size:12pt; font-family:Arial; color: darkblue">
      Un foglio di stile indica come deve apparire una pagina HTML</p>
    <p style="font-size:12pt; font-family:Arial; color: darkblue">
      CSS è un linguaggio specifico per definire fogli di stile</p>
  </body>
</html>
```

Esempio foglio di stile esterno

esempioCSSesterno.html

```
<html xmlns="http://www.w3.org/1999/xhtml" lang="it">
  <head>
    <meta content="text/html; charset=iso-8859-1" http-equiv="Content-Type" />
    <link rel="stylesheet" type="text/css" href="esempio.css"/>
    <title>CSS esterno</title>
  </head>
  <body>
    <h1>Lezione su CSS</h1>
    <p>Un foglio di stile indica come deve apparire una pagina HTML</p>
    <p>CSS è un linguaggio specifico per definire fogli di stile</p>
  </body>
</html>
```

esempio.css

```
body    {background: lightyellow }

h1      {font-size: 18pt; font-weight: bold; color: maroon}

P       {font-size: 12pt; font-family: Arial; color:darkblue}
```

Foglio di stile esterno

- Elemento *link* dentro *head* e attributi *href*, *type*, *rel*, *title* e *media*
 - *href*: obbligatorio, indica l'URL del foglio di stile
 - *type*: obbligatorio, indica il linguaggio usato ("text/css")
 - *rel*: obbligatorio, descrive il tipo di relazione tra documento e file collegato; valori possibili: *stylesheet* o *alternate stylesheet*
 - *media*: opzionale, indica a quale media (screen, print, etc.) il foglio di stile deve essere applicato
- Soluzione migliore ai fini dell'accessibilità
- In (X)HTML è possibile associare molti fogli di stile ad uno stesso documento

Come associare fogli di stile: soluzioni a confronto

- Se uno stesso foglio di stile deve essere applicato a tutte le pagine di un sito, conviene definire un foglio di stile esterno e collegarlo a tutte le pagine
- Se in una pagina si richiede la modifica di qualche regola del foglio di stile esterno, si può definire un foglio incorporato che sovrascriverà quello esterno

```
<style type="text/css">  
    h2 {color: green; }  
</style>
```

- Evitare l'uso estensivo di fogli inline: le pagine risulterebbero pesanti e difficili da gestire

Esercizio: assegnare un foglio di stile

- Dato un documento X(HTML) semplice definire lo stesso foglio di stile nelle tre modalità indicate (esterno, incorporato, inline)

Fogli di stile per diversi media

- Grazie all'attributo *media* è possibile associare un foglio di stile diverso per ogni supporto su cui la pagina verrà distribuita (palmari, cellulari, browser vocali e altri dispositivi)

```
<link rel="stylesheet" type="text/css" href="screen.css" media="screen" />
<link rel="stylesheet" type="text/css" href="print.css" media="print" />
<style type="text/css" media="screen, aural">
...
</style>
```

- Nel CSS si può dichiarare quali sezioni applicare a quali media

```
@media screen {
  * { font-family: sans-serif }
}
```

Esempi: diverso layout su schermo e in stampa
stesso layout su schermo e in stampa

Valori dell'attributo media esempi

- screen:
 - screen di computer
- tty
 - dispositivi a carattere fisso come terminali o device portatili con capacità di display limitate
- tv
 - per device tipo televisione
- projection
 - proiettori
- handheld
 - device palmari
- print
 - per stampa e per documenti visti in modalità “print preview”
- braille
 - per device braille
- aural
 - per sintetizzatori vocali
- all
 - per tutti i dispositivi (valore di default per attributo *media*)

Problemi di CSS nel supporto ai diversi tipi di media

- problemi esterni

- i browser offrono un supporto non uniforme né universale al cambio automatico di foglio di stile a seconda dei tipi di media (e.g. aural poco supportato)
- gli sviluppatori non sanno come progettare fogli di stile per periferiche particolari (e.g. braille)

- problemi del linguaggio

- limiti di concezione dei CSS2, molte limitazioni relative a periferiche particolari

Fogli di stile alternativi

- Tre tipi di stylesheets: *persistent*, *preferred*, e *alternate*
- L'attributo *rel* può assumere due valori:
 - *stylesheet*
 - *alternate stylesheet*

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

```
<link rel="alternate stylesheet" type="text/css" href="stile_alternativo.css" />
```

- Generalmente non è automatica nei browser la presentazione all'utente dei diversi fogli di stile. L'autore deve fornire un meccanismo per consentire l'applicazione del foglio alternativo, per esempio con un codice Javascript
- Esempio ([stili_alternativiLS.html](#)) di fogli di stile alternativi e dipendenti dai media

Problemi di compatibilità tra browser

- Storicamente si è assistito ad uno scarso supporto per CSS (e le specifiche del W3C) nei browser
 - Explorer 5: primo ad offrire un supporto adeguato
 - Mozilla: ottimo supporto per CSS
- Principali difficoltà:
 - compatibilità con i vecchi browser
 - soluzioni:
 - dimenticare i vecchi browser
 - soluzioni cross-browser
 - diverse modalità di rendering di certe proprietà
 - non esiste una strategia univoca
 - è importante TESTARE con più browser possibili

Testing supporto

- Risorse:
- W3C Test Suite
<http://www.w3.org/Style/CSS/Test/>
- Media queries (W3C Recommendation 19 June 2012)
<http://www.w3.org/TR/css3-mediaqueries/>
- Can I Use <http://caniuse.com/>

Riferimenti

Specifiche

- **W3C** <http://www.w3.org/Style/CSS/current-work>
- **Selector Patterns**
<https://www.w3.org/TR/css3-selectors/#selectors>

Tutorial

- **W3C** <http://www.w3.org/Style/CSS/learning>
- **W3schools** <https://www.w3schools.com/css/>