HTML

"Linguaggio **HTML** (HyperText Markup Language): puro testo ASCII con uso di TAG o marcatori"

Si basa sul linguaggio SGML (Standard Generalized Markup Language) per la elaborazione di documenti che consente di descrivere la **struttura** generale dei contenuti del documento e non l'aspetto esteriore su una pagina o sul lo schermo. Il linguaggio HTML, dunque, descrive la struttura di un documento definendo una serie di stili fissi da usare nelle pagine Web : intestazioni, paragrafi, elenchi e tabelle. Consente, inoltre, di definire lo stile dei caratteri come il grassetto e gli esempi di codice. Ogni elemento ha un nome ed e' contenuto in un oggetto chiamato tag.

I tag indicano semplicemente che un elemento e' un'intestazione o un elenco ma non dicono nulla sul modo in cui devono essere formattati tale inte stazione o elenco. I **browser Web**, oltre a fornire le funzioni di navigazione che consentono di ricercare i documenti in rete, sono anche formattatori HTML: **leggono** e **interpretano** le informazioni al caricamento di una pagina e fanno corrispondere ai vari elementi stili di visualizzazione del testo e delle immagini sullo schermo spesso **differenti** rispetto ad altri browser. Un progettista di siti Web deve sempre ricordare che le pagine create con linguaggio HTML possono avere un aspetto radicalmente diverso da sistema a sistema e da browser a browser.

Poiche' lo scopo finale del Web e' presentare pagine leggibili da chiunque si progetteranno pagine visibili nel maggior numero possibile di browser e la scelta del tipo di tag da usare nascera' da un compromesso ottimale tra il piu' ampio supporto della pagina (HTML 2.0) e la massima flessibilita' nel posizionamento degli elementi nella pagina (HTML 4.0) ma con problemi di compatibilita' con altri browser.

La prima regola sara' concentrarsi nella realizzazione di ***contenuti chiari ben strutturati, facili da leggere e da comprendere*** (ancora oggi si usano browser per terminali in grado di visualizzare solo testo).

Le pagine HTML contengono due tipi di oggetti:

- il testo del documento
- i tag HTML che indicano gli ELEMENTI, la STRUTTURA, la formattazione e i LINK ipertestuali con altre pagine o che consentono di includere altri file. La maggior parte dei tag HTML ha il seguente aspetto:

  <NomeTag> testo influenzato dal tag </NomeTag>

I tag HTML sono quelle parti racchiuse fra parentesi angolari (<>) che indicano gli elementi di una pagina. Sono codici di formattazione.

Il tag iniziale attiva una funzionalita' (un titolo, l'enfasi da porre su un concetto, etc..) ed il tag di chiusura preceduto da una barra (/) disattiva la stessa funzionalità. Alcuni tag vengono usati singolarmente mentre altri vengono usati come "contenitori" per informazioni. Nei nomi dei tag non si fa differenza tra lettere maiuscole e minuscole; era buono stile usare il maiuscolo per distinguerli meglio dal testo mentre oggi , per compatibilità con l'XHTML[1], si preferisce il minuscolo.

[1] Extensible HTML, che sarà facilmente accessibile non soltanto per gli odierni browsers desktop, ma da altre apparecchiature – quali i telefoni cellulari.

HTML - Riferimenti

L'elenco di riferimento dei tag HTML raggruppa i tag di uso comune (specifiche standard) e le estensioni di Netscape o Internet Explorer non sempre supportate da altri browser:

- tag per la struttura globale di un documento: <HTML>,<HEAD>,<BODY>
- tag per titoli, intestazioni: <TITLE>, <H1>...<H6>
- tag per suddivisione in paragrafi <P>, sottoparagrafi <DIV>, ritorni a capo
, inserimento linee di separazione <HR> e blocchi di testo
- tag per commenti
- contenitori in-line <SPAN> per influenzare piccole porzioni di testo
- tag per le liste
- tag per collegamenti, immagini, schede : <A>,<IMG>,<FORM>
- tag per tabelle, cornici: <TABLE>, <FRAMESET>
- tag *datati* per testo scorrevole, colonna sonora : <MARQUEE> e <BGSOUND>[1], <EMBED>[2]
- tag per la multimedialità e l'inserimento di Applet : <OBJECT>

I tag per la **struttura** di una *pagina*[3] HTML:

<HTML> </HTML> racchiude l'intero documento HTML

<HEAD> </HEAD> racchiude la parte iniziale del documento HTML
puo' includere: <TITLE>, <META>, <SCRIPT> etc..
<TITLE> </TITLE> esempio di meta istruzione (informazione di servizio al browser)

<BODY> </BODY> con ATTRIBUTI[4] BGCOLOR per fissare il colore di sfondo (*obsoleto*)
BACKGROUND per immagini di sfondo (*obsoleto*)
TEXT per fissare il colore del testo (*obsoleto*)
LINK, ALINK, VLINK per fissare il colore dei link, dei link attivi e dei link visitati (*obsoleti*)

per le **intestazioni**:
<Hn> </Hn> con n da 1 a 6 (livelli)

per **interrompere linea o creare paragrafi**:

 codice di "a capo" (senza inserire linee bianche)
<P> </P> un normale paragrafo (inserisce linea bianca prima e dopo).
Il tag di chiusura e' opzionale

per **inserire riga orizzontale**:
<HR> con possibili ATTRIBUTI per stile

per **blocco di testo**:
<BLOCKQUOTE> </BLOCKQUOTE> citazioni piuttosto estese
<ADDRESS> </ADDRESS> per blocchetti di firma o informazioni su autore

[1] Correttamente interpretati solo da IE
[2] Per vecchie versioni di Netscape
[3] Documento di qualsiasi dimensione, da poche righe ad un libro intero
[4] E' buono stile racchiudere il valore degli attributi tra virgolette (per compatibilità con l'XHTML)

per **commenti**:

<!-- commento -->

per elementi di **presentazione del testo o stili** (formattazione del carattere):

- *stile logico* (descrive il ruolo cioe' scopo e funzione di un testo nel documento):

<EM> </EM>	enfasi
<STRONG></STRONG>	enfasi maggiore
<CITE> </CITE>	una citazione
<BIG> </BIG>	Più GRANDE
<SMALL> </SMALL>	Più PICCOLO
<Q> </Q>	breve citazione (tra virgolette nel testo; paragrafi senza breaks)
<ACRONYM> </ACRONYM>	per acronimi (es: HTML, HTTP, WWW)
<ABSTRACT> </ABSTRACT>	riassunto (font più piccolo, centrato e giustificato)
<MARGIN> </MARGIN>	nota a margine
<FOOTNOTE> </FOOTNOTE>	nota a piè di pagina
<CODE> </CODE>	codice (font monospazio, di solito Courier)
<SAMPLE></SAMPLE>	testo di esempio (anche SAMP)
<KBD> </KBD>	testo da immettere da tastiera
<VAR> </VAR>	variabile o indicatore per altro valore (corsivo)
<DFN></DFN>	definizione di un termine (corsivo)

- per *stile fisico* (forza l'aspetto estetico):

<TT> </TT>	font da macchina per scrivere
<I> </I>	corsivo
<B> </B>	grassetto
<U> </U>	sottolineato
<STRIKE> </STRIKE>	scrive il testo sbarrato
<BLINK> </BLINK>	visualizza il testo lampeggiante
	a pedice
	ad apice
<PRE> </PRE>	per testo formattato (interpretando i tag interni)
<XMP> </XMP>	APPLICATO a testo formattato per visualizzare i tag interni

nb: FONT tag considered harmful! da http://www.w3.org/MarkUp/#tutorials

Per indice degli elementi:

inglese http://www.w3.org/TR/REC-html40/index/elements.html
italiano **http://www.diodati.org/w3c/html401/index/elements.html**

sommario

inglese http://www.w3.org/TR/html4/#minitoc
italiano **http://www.diodati.org/w3c/html401/cover.html#minitoc**

Caratteri Speciali[5]:

Symbol	Entity	Example
Less than	<	<
Greater than	>	>
Ampersand	&	&
nonbreaking space		
em dash	—	—
quotation mark	"	"

per inserire ad esempio la parentesi angolare < digitare **<** oppure per > digitare **>**

Entità per lettere accentate

à	à	À	À
á	á	Á	Á
â	â	Â	Â
ã	ã	Ã	Ã
ä	ä	Ä	Ä
å	å	Å	Å

per creare **collegamenti**:

<A> </A> con OBBLIGATORIO **ATTRIBUTO HREF= *"url"*** crea un link
es. <A HREF="file:///drive:/path/nome.html#UNO">..</A> indirizzo assoluto
es. <A HREF="../nome.html#UNO">..</A> indirizzo relativo salendo di un livello
es. <A HREF=#UNO>..</A>
o **NAME** (anche ID) crea aggancio usato per punto di ancoraggio del link
es.<A NAME=UNO>..</A>

per **creare liste**:

<DL> per liste di definizioni o a glossario
<DT> <DD> termine da definire (DT) e definizione (DD)
</DL>

<UL> per liste non ordinate con ATTRIBUTI COMPACT, PLAIN, WRAP
<LI> elemento di lista, accetta come attributo SRC (chiusura opzionale)
</UL>

<OL> per liste ordinate (numerate) con ATTRIBUTO COMPACT
<LI> elemento di lista, accetta come attributo SRC (chiusura opzionale)
</OL>

esempio <UL>
<LI> elemento di lista preceduto dal simbolo punto (*default*)
<DD> sottoparagrafo rientrato
</UL>

- Capitolo 1
  Paragrafo 1

[5] da http://www.w3.org/MarkUp/Guide/Advanced.html

per inserire **immagini**:

<IMG> con ATTRIBUTI **SRC** per indicare da dove prelevarla (URL) OBBLIGATORIO
ALT per testo alternativo: compatibilità con browser solo testuali
ALIGN per posizionarla rispetto al testo (right|left|top| middle|bottom)
USEMAP per creare mappe sensibili tipo *client-side* (ISMAP se *server-side*)

per creare **tabelle (griglie dinamiche)**:

<TABLE> con ATTRIBUTI CELLSPACING spaziatura tra celle
CELLPADDING spaziatura interna alle celle
BORDER anche per spessore del bordo
WIDTH larghezza della cella
HEIGHT altezza della cella
BGCOLOR per specifico colore in ogni cella (*obsoleto*)

<TD> </TD> o <TH> </TH> con ATTRIBUTI ROWSPAN
COLSPAN
ALIGN
VALIGN

<TR> </TR> con ATTRIBUTI ALIGN
VALIGN
NOWRAP per forzare non a capo

<COLGROUP> con ATTRIBUTI SPAN[6] , WIDTH (possibile valore "0*" : minimo necessario)
<COL> con ATTRIBUTI SPAN , WIDTH
<CAPTION> il titolo di tabella con ATTRIBUTO ALIGN
<CENTER> </CENTER> nidificando la tabella che si vuole centrare

per creare **FRAME** (riquadri o cornici):

<FRAMESET> </FRAMESET> alternativo a <BODY> [Netscape 2.0 e successivi]
con ATTRIBUTO COLS, ROWS, BORDER, FRAMEBORDER
e BORDERCOLOR

<FRAME> </FRAME> con ATTRIBUTI SRC
NAME
MARGINWIDTH, MARGINHEIGHT
SCROLLING, NORESIZE
TARGET con possibili[7] **_top / _self / _blank / _parent**
FRAMEBORDER e BORDERCOLOR

<NOFRAMES> </NOFRAMES> per pagine alternative

---

[6] Alternativo all'uso di <COL REPEAT=n>

[7] Per aprire la pagina **nel riquadro** con nome (NAME) "WINDOW-1":
<A HREF = "nome_file.html" **TARGET** = "WINDOW-1" **>** collegamento </A>
oppure <A HREF = "http://percorso_completo" TARGET = "WINDOW-1" >collegamento </A>
oppure frame di destinazione con **un link a tutto schermo** "TARGET = **_top**" (minuscolo)
Se tutti i collegamenti aprono la pagina **nel riquadro** con lo stesso nome in **Head**: <BASE TARGET = nome >

per creare **mappe sensibili (lato client)** :

<MAP> </MAP> con ATTRIBUTO NAME
<AREA> con ATTRIBUTO SHAPE, COORDS, HREF

Vedi anche http://xhtml.html.it/guide/lezione/1680/le-mappe-di-immagine/

Programma per *creare* mappe d'immagini cliccabili:

- guida all'uso di GeoHTML http://stclassi.altervista.org/HTML_CSS/GeoHTML.pdf

INTERATTIVITÀ

per creare **MODULI** (schede) o FORMS nell'interazione col WEB:

<FORM> </FORM> indica una scheda
con ATTRIBUTI ACTION per scegliere il modo di elaborare dati su server
METHOD per indicare l'operazione eseguita dal server
ENCTYPE per accettare file come input

<INPUT> </INPUT> un'area di input in una scheda
con ATTRIBUTI TYPE ad es. checkbox, hidden, password, radio, reset, submit, text o image
ALIGN, CHECKED
MAXLENGHT, NAME, SIZE, SRC, VALUE

<TEXTAREA> </TEXTAREA> indica un'area per l'input di testo (piu' righe)
con ATTRIBUTI COLS | ROWS | NAME

<SELECT></SELECT> crea un menu o lista a scorrimento
con ATTRIBUTI ERROR
MULTIPLE
NAME
SIZE
<OPTION> </OPTION> indica un elemento nell'oggetto <SELECT>
con ATTRIBUTI DISABLED, SELECTED, VALUE

MULTIMEDIA:

vecchie modalità

per inserire **musica** di sottofondo o a richiesta o **video**[8] (Netscape *vecchie versioni*):

<EMBED> con ATTRIBUTO SRC per iserimento di file audio(.wav) o video (.avi)
WIDTH, HEIGHT perche' alcuni browser vedano i controlli
Start e Stop
AUTOSTART,
LOOP

I tag specifici di **Internet Esplorer (sconsigliati)**:

<BODY> aggiungendo l'attributo BGPROPERTIES=FIXED
si ottiene uno sfondo fisso che non scorre col contenuto della pagina
<BODY BACKGROUND="nomesfondo.gif" BGPROPERTIES=FIXED>

<IMG> con nuovo attributo DYNSRC=indirizzo_URL per inserire filmati .AVI ed
inserimento dell'immagine al posto del video per browser che non consentono la
visualizzazione in linea del filmato.
<IMG SRC="nome.gif" DYNSRC="filmato.avi">
altri ATTRIBUTI: START=FILEOPEN o START=MOUSEOVER
per determinare il movimento d'avvio del filmato;
LOOP quante volte,
LOOPDELAY msec tra riproduzioni
nuovo attributo VRML=indirizzo_URL per inserire mondi virtuali (3 dimensioni)

<BGSOUND> in HEAD per aggiungere colonna sonora alla pagina Web
(formato campionato o MIDI)
<BGSOUND SRC="file.wav">

<MARQUEE> per visualizzare una porzione di testo scorrevole nella pagina.
Molti attributi per definire direzione, velocita', fluidita' del movimento.
<MARQUEE ALIGN=MIDDLE> testo scorrevole </MARQUEE>

[8] Netscape usa **plug-in**, IE controlli **ActiveX**; HTML 4.0 **<OBJECT></OBJECT>** rende inutili TAG <EMBED> e serve anche per inserire Applet (in linguaggio Java) che dinamizzano le pagine.

nuove[9] modalità

L'elemento OBJECT offre una soluzione globale per l'inclusione generica di oggetti (termine "oggetto" usato per descrivere le cose che le persone desiderano inserire nei documenti HTML). Sono oggetti: applet, plug-in, gestori di media, ecc.

Il nuovo elemento OBJECT comprende pertanto alcune delle funzioni svolte dagli elementi esistenti. Si consideri il seguente grafico di funzionalità:

Tipo di inclusione	Elemento specifico	Elemento generico
Immagine	IMG	OBJECT
Applet	APPLET (Disapprovato.)	OBJECT
Un altro documento HTML	IFRAME	OBJECT

In tabella si sintetizza che ciascun tipo di inclusione ha una soluzione specifica ed una generale.

L'elemento generico OBJECT servirà come soluzione per implementare anche futuri tipi di media.

Per includere immagini, gli autori possono usare l'elemento OBJECT o l'elemento IMG.
Per includere applet, gli autori dovrebbero usare l'elemento OBJECT dal momento che l'elemento APPLET è disapprovato.

Per includere un documento HTML all'interno di un altro, gli autori possono usare sia il nuovo elemento IFRAME sia l'elemento OBJECT. In entrambi i casi il documento incorporato rimane indipendente dal documento principale. I programmi utente di tipo visuale possono presentare il documento incorporato in una finestra distinta all'interno del documento principale. Si consultino le note sui documenti incorporati per un confronto tra OBJECT e IFRAME per quanto riguarda l'inclusione di documenti.

Immagini ed altri oggetti inclusi possono avere collegamenti ipertestuali associati con essi, sia attraverso il meccanismo di collegamento standard sia attraverso le mappe immagine.

[9] Vedi http://www.diodati.org/w3c/html401/struct/objects.html

Uso di OBJECT (IE) ed EMBED (NN[10]): per incorporare multimedia nel browser

Per riprodurre con **Plug In** suoni con estensione .WAV o MIDI, video con estensione .AVI, immagini (anche .PNG se installato QuickTime[11]):

```
<embed src="nome" width="342" height="174"> </embed>
```

Per visualizzare un' animazione Flash (con richiamo al plug-in) :

```
<embed src="nome.swf"  type="application/x-shockwave-flash">
```

Il tag **OBJECT** per includere **video** e **immagini in movimento**:

Un primo esempio :
```
<OBJECT width="32" height="32">
 <embed src="nome.avi" width="231" height="188"> </embed>
 </OBJECT>
```
NB: embed forza le dimensioni anche dell'object (in IE necessario visualizzatore ActiveX)

Per **immagini**[12] (.gif anche animate, .jpg, .png) :

```
<OBJECT   type="image/gif"                          <!-- type e' opzionale ma raccomandato  -->
          data="nome.gif"
          width="n" height="n">                          <!-- dimensioni non opzionali  in IE -->
           <embed src="nome.gif" width="n" height="n" > </embed>
testo alternativo                          <!-- il testo e' visualizzato solo nel caso di browser  testuali-->
</OBJECT>
```

Per riprodurre come **oggetti ActiveX** suoni (con estensione .au) :

```
<OBJECT classid="clsid:663C8FEF-1EF9-11CF-A3DB-080036F12502"
          data="nome.au">
          <embed src="nome.au"></embed>
</OBJECT>
```

Per introdurre filmati con estensione .AVI, .WMV, utilizza il tag OBJECT con richiamo all'ActiveX specifico CLSID:22d6f312-b0f6-11d0-94ab-0080c74c7e95 (che sostituisce il richiamo al plug-in in NN con embed type"application/x-mplayer2)
Per introdurre filmati con estensione .WMV anche CLASSID=CLSID:6BF52A52-394A-11D3-B153-00C04F79FAA6 per grafica piu' elegante.
Per filmati Quicktime (estensione .MOV) clsid:02BF25D5-8C17-4B23-BC80-D3488ABDDC6B

Per IExplorer (dalla versione 3.x in poi) è stato realizzato un apposito ActiveX in grado di poter governare **ogni singolo evento mediale**: clsid:CFCDAA03-8BE4-11cf-B84B-0020AFBBCCFA

[10] Dalla versione 6 anche Nescape Navigator interpreta correttamente il tag OBJECT

[11] QuickTime, in fatto d'incorporamento del plug-in player, fornisce solo la tecnica di controllo via <EMBED> con la quale garantisce il pieno supporto, dalle versioni 3.0 in poi, sia di IExplorer che di Navigator.

[12] In NN (ver 7.0) possibile uso dell'attributo SRC: <object SRC = "nome.gif"> </object>

Per introdurre **animazioni FLASH** (con estensione .swf):

```
<object classid="clsid:D27CDB6E-AE6D-11CF-96B8-444553540000">
 <PARAM name="movie" value="nome.swf">
</object>
```

Per entrambi i browser (NN ed IE) un'**animazione flash** si puo' introdurre con la soluzione seguente:

```
<object width="400" height="300"
                                                    classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000"
codebase="http://download.macromedia.com/pub/shockwave/cabs/flash/swflash.cab#version=5,0,0,0">
<param name="movie" value="nome.swf">
<param name="quality" value="high">
   <embed src="nome.swf" quality="high"
                                  pluginspage="http://www.macromedia.com/shockwave/download/index.cgi?
P1_Prod_Version=ShockwaveFlash"
            type="application/x-shockwave-flash" width="100%"
            height="100%">
  </embed>
</object>
```

(vedi anche http://realgar.mcli.dist.maricopa.edu/writinghtml_it/tut/tut29d.html per swmovie)

Si usano tag object nidificati per visualizzare in alternativa (in NN non IE).

L'attributo **codebase** serve per il download del Plug-in adatto.

NB: per impostare un percorso: con value="c:\percorso\nome.swf"; (anche: "directory/nome")
con data="///c:/percorso/nome.gif"
("../dir/nome" salendo di un livello es: data = "../css/nome.gif")

Per introdurre **Applet (programmi in Java)** :

Inserimento di Applet con Java Plug-in "***versione dinamica***":

```
<object classid="clsid:8AD9C840-044E-11D1-B3E9-00805F499D93" width="200" height="100">
<param name="code" value="SecondoApplet.class">
<param name="type" value="application/x-java-applet">
<param name="Nome" value="Paola">
<!-- Il tag COMMENT è una estensione di IE e fa sì che venga ignorato il contenuto. Invece Netscape lo
vede -->
        <comment>
                <embed type="application/x-java-applet" width="200" height="100"
                        code="SecondoApplet.class"
                           Nome = Paola>
                 </embed>
        </comment>
</object>
```

Nota: l'attributo **classid** del tag OBJECT e' unico per la versione dinamica con Plug-in (versione almeno 1.3.1) e particolare se la versione e' "***statica***": ad esempio per Plug-in 1.4.0 classid="clsid:CAFEEFAC-0014-0000-0000-ABCDEFFEDCBA" invece per Plug-in 1.4.1_01 classid="clsid:CAFEEFAC-0014-0001-0001-ABCDEFFEDCBA"
oppure per Plug-in 1.4.1_02 classid="clsid:CAFEEFAC-0014-0001-0002-ABCDEFFEDCBA"

L'attributo **codebase** che serve per il download del Plug-in adatto sarà ad esempio:
codebase="http://java.sun.com/products/plugin/autodl/jinstall-1_4_0-win.cab" per versione statica 1.4.0
codebase = "http://java.sun.com/products/plugin/autodl/jinstall-1_4_1_02-windows-i586.cab#Version=1,4,1,20" per versione statica 1.4.1_02

Mentre per Netscape (versioni precedenti alla 6.0) si usa il tag <EMBED> con attributo type = "application/x-java-applet;jpi-version=1.4.1_02"
ed attributo pluginspage = "http://java.sun.com/products/plugin/index.

Per facilitare conversioni di applet già scritte, Sun mette a disposizione un convertitore liberamente scaricabile dalla rete: esso si impegna a trovare i tag applet nei file HTML e a convertirli in tag object (ver 1.3).

Per saperne di più sul Java Plug-in e per scaricare il convertitore, fate riferimento al collegamento all'URL http://java.sun.com/products/plugin

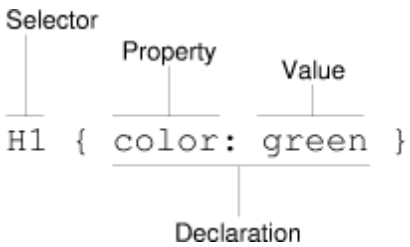
Per ripassare

I CASCADING STYLE SHEETS

Attualmente si consiglia di pensare prima ai contenuti e solo in un momento successivo allo **stile di presentazione**. Questa e' la filosofia dei fogli di stile che sostituiscono i tag di formattazione (come <FONT FACE> inserito direttamente nel codice) mantenendo un aspetto uniforme in tutte le pagine associate e si aggiungono alle pagine web in uno dei quattro modi seguenti:

- Con **inclusione** nel documento HTML (uso della metaistruzione <STYLE TYPE = "text/css"><!--…--></STYLE > di solito all'interno della sezione HEAD dopo TITLE)
- Con **collegamento** ad un **foglio di stile esterno** (uso della metaistruzione <LINK href = "*nome*.css" rel = "stylesheet" type = "text/css"> all'interno della sezione HEAD). Se si vogliono combinare più' CSS separati ed applicarli alla stessa pagina basta assegnare lo stesso nome (TITLE) a tutti gli stili (ad es: <LINK href = "*nome1*.css" rel = "stylesheet" type = "text/css" title = "Stili"> <LINK href = "*nome2*.css" rel = "stylesheet" type = "text/css" title = "Stili">).
- **Importando** uno o più **fogli di stile esterni** (con uso di @import url(*nome*.css); all'interno di altro foglio di stile in modo da creare una "cascata"). Le dichiarazioni del successivo hanno la precedenza su quelle del primo e lo stesso Style Sheet che importa ha precedenza sugli altri.
- Con aggiunta "**inline**" (con uso di modifica del singolo tag mediante attributo STYLE)

In un foglio di stile, ogni istruzione si suddivide in *selettore* (il tag) e *dichiarazione* che suggerisce come presentarlo: le dichiarazioni delle *proprietà* hanno la forma "name : value".



Si possono impostare più proprietà tutte applicate allo stesso tag separandole col carattere "**;**"

H1 { color: green; text-align: center; }

Possono essere impostati più valori di ogni proprietà, separandoli con una virgola: il precedente ha priorità sul successivo (valori consigliati in alternativa al browser che può non implementarli).
Ad esempio : I { font-family: courier, arial, times }

Si possono raggruppare sia selettori sia dichiarazioni per rendere più compatto il codice, ad esempio:
BLOCKQUOTE, EM, I { color : marron; font-family: courier }
e' equivalente a:
BLOCKQUOTE { color : marron; font-family: courier }
EM { color : marron; font-family: courier }
I { color : marron; font-family: courier }

I selettori possono essere concatenati. Ad esempio H1 STRONG { color : blue } agisce sul testo in ogni tag STRONG all'interno di tag H1.

L'**ordine di precedenza**[1] con cui IE e Netscape supportano l'interpretazione dei fogli di stile e' diverso cambiando tra una versione e quella più recente con la seguente tendenza :

1. Inline (livello locale)
2. Embedded (inclusi o livello globale)
3. Linked (collegati)
4. Imported (importati)
5. Reader
6. Browser default

Nelle versioni Explorer 3 o 4 e' supportato l'elemento *important* per forzare l'importanza di una scelta di apparenza nel caso di collisione di stili all'interno dello stesso foglio ad esempio:

```
<HTML>
<STYLE TYPE= "text/css" MEDIA = screen >
<! --
BODY {  color : yellow;
          font-family: impact  }
P  {  color : red ! important;
      font-family: arial ! important  }
CODE { color : marron;
          font-family: courier  }
-->
</STYLE>
<HEAD><TITLE>Quando gli stili collidono</TITLE></HEAD>
<BODY><P><CODE>Testo </CODE></P>
</BODY>
</HTML>
```

EFFETTO: il testo sarà scritto in Arial ed in colore rosso; si noti che le istruzioni sono inserite come commento per compatibilità con browser che non supportano i CSS; se si aggiunge <I STYLE="color: green; font_family: times"></I> la forzatura inline prevale ed il testo appare in corsivo di stile times in verde.

Il **CSS2** "eredita" il valore **!important** per tutte le proprietà degli stili che governano parte o tutto il documento. Per esempio, lo stile seguente forza tutti gli sfondi in bianco e tutto il testo in nero:

```
 /* Configura il colore del testo in nero ed il colore di sfondo del documento in bianco */

        body {  color: black ! important ;
                  background: white ! important
                }              /* Fa in modo che i valori di "color" e "background" debbano essere ereditati da tutti gli altri elementi,
                                  rafforzandoli con "!important".
                                    Notare che questo può essere annullato da un altro più specifico operatore di stile. */
        * {color: inherit ! important ;
           background: inherit ! important
        }                  /* Uso di selettore UNIVERSALE: il simbolo asterisco significa "ogni elemento" */
```

[1] In realtà i browser applicano lo stile definito per ultimo (sono nate dunque regole di stile per definire prima a livello più esterno ad esempio @import deve precedere ogni definizione di stile inclusa)

I CSS2 includono anche queste **caratteristiche di controllo**:

- Colori (**color**, **background-color**, **border-color**, **outline-color**) e caratteri (**font**): l'utente può applicare le sue preferenze al colore e ai font del documento Web .

- I contorni dinamici (la proprietà **outline**) permette gli utenti (ad esempio gli ipovedenti) di creare dei profili attorno al contenuto che non cambiano il layout ma che forniscono informazioni importanti.

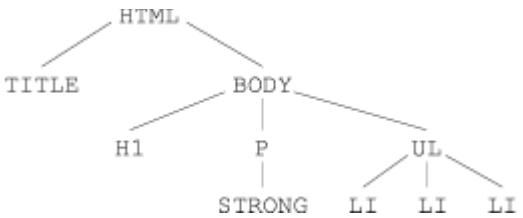
Per esempio, per disegnare uno spessa linea nera attorno ad un elemento quando su di esso c'è il focus, ed una spessa linea rossa quando è attivo, si possono usare queste righe:

```
:focus { outline: thick solid black }
:active { outline: thick solid red }
```

Il concetto di inheritance (ereditarietà)

```
<HTML>
 <TITLE>Bach's home page</TITLE>
 <STYLE TYPE="text/css">
          BODY { color: green }
  </STYLE>
 <BODY>
   <H1>Bach's home page</H1>
   <P>Johann Sebastian Bach was a <STRONG>prolific</STRONG> composer. Among his  works are:
   <UL>
     <LI>the Goldberg Variations
     <LI>the Brandenburg Concertos
     <LI>the Christmas Oratorio
   </UL>
 </BODY>
</HTML>
```

La struttura di questo documento e':



Una proprietà CSS che setta il valore di un elemento sarà trasferita attraverso l'albero a tutti i discendenti. Nell'esempio se si vuole che il tag H1 sia associato ad altro colore basta aggiungere un istruzione che scritta prima o dopo ha comunque precedenza nel conflitto:

```
<STYLE TYPE="text/css">
 BODY { color: green }
 H1 { color: navy }
</STYLE>
```

NB: Il tag background non ammette inheritance.

Le classi

Le **classi** permettono di impostare apparenza diversa per lo stesso tag o per diversi tag. Un selettore di classe che permette di stabilire più di uno stile per un tag/elemento usa la sintassi punto nel CSS : "**. nomeclasse**" e si applica con l'attributo CLASS = "nomeclasse" nel codice HTML.
Ad esempio:

```
<STYLE TYPE= "text/css">
P.bluetext {  color : blue }
</STYLE>
```

imposterà il colore blu usando nel documento HTML <P CLASS= bluetext>
Se si usa invece:

```
<STYLE TYPE= "text/css">
.bluetext {  color : blue }
</STYLE>
```

imposterà il colore blu sia usando <H3 CLASS= "bluetext">…</H3> sia <EM CLASS= "bluetext"> ..</EM> nel documento HTML.

E' prevista la dichiarazione di **classi multiple**. Ad esempio la regola seguente applicherà gli stili impostati a tutti gli elementi in cui siano presenti (in qualunque ordine) almeno tutti i nomi delle classi definiti nel selettore:

h1**.testorosso.grassetto** {color: red; font-weight: bold;}

L'attributo id

Per limitare lo scope delle informazioni di stile ad una singola istanza di un elemento, si setta l'**attributo "id"** che usa la sintassi cancelletto nel CSS: "**#nomeid**" e si applica con l'attributo ID = "nomeid" nel codice HTML.

```
<HTML> <HEAD> <STYLE type="text/css">
                #myid {border-width: 1; border: solid; text-align: center}
               </STYLE>
    </HEAD>
   <BODY>  <H1 id="myid"> Questo H1 e' influenzato dallo style </H1>  </BODY>
</HTML>
```

Un **id** è simile ad una classe, ma con una fondamentale differenza: una classe può essere applicata a più elementi nella stessa pagina, mentre si consiglia di usare un id per riferirsi ad un solo elemento definendone un preciso stile. In pratica, posso avere nello stesso documento: <p class="testo"> e <table class="testo">, ma si consiglia di usare ad esempio <table id="sfondo"> senza più usarlo per altri elementi.

Gli attributi di stile inline hanno precedenza su ID, ID sulla class, e class sugli elementi HTML definiti come stylesheet.

***Regole**: Usare un numero minimo di CSS per ogni sito*

Usare i CSS esterni piuttosto di stili interni, ed evitare i CSS in riga (inline).

Se vi sono più CSS, usare lo stesso nome di classe per lo stesso concetto in tutti.

Le pseudo-classi

Il concetto di pseudo-classe ha qualcosa di "filosofico". Una pseudo-classe non definisce infatti un elemento ma un particolare stato di quest'ultimo. Imposta uno stile per un elemento al verificarsi di certe condizioni.

A livello sintattico le pseuso-classi non possono essere mai dichiarate da sole, ma per la loro stessa natura devono sempre appoggiarsi ad un selettore. Il primo costrutto che esaminiamo è quello con un elemento semplice:

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

La regola vuol dire: i collegamenti ipertestuali (**<A>**) che non siano stati visitati (**:link**) avranno il colore blue. Da qui risulta più chiaro il concetto espresso all'inizio: la pseudo-classe **:link** definisce lo stile (colore blue) solo in una determinata situazione, ovvero quando il link non è stato attivato. Appena ciò dovesse avvenire, il testo non sarà più blue, perché la condizione originaria è venuta meno.

Torniamo alla sintassi. La pseudo-classe (tutte iniziano con i **due punti**) segue senza spazi l'elemento. Subito dopo si crea nel modo consueto il blocco delle dichiarazioni.

Una pseudo-classe può anche essere associata a selettori di tipo classe. I costrutti possibili sono due. Il primo è quello sancito nella specifica **CSS1**. La pseudo-classe doveva seguire la dichiarazione della classe:

```
a.collegamento:link {color: green;}
```

Questa regola fissa il testo verde (green) solo per i link non visitati che abbiano come attributo **class="collegamento"**. Sarà verde questo collegamento:

```
<a href="pagina.htm" class="collegamento">
```

ma non questo:

```
<a href="pagina2.htm">
```

A partire dalla specifica **CSS2** è consentita anche questa sintassi:

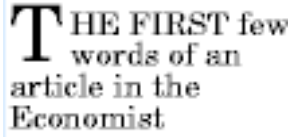
```
a:link.collegamento
```

in cui la classe segue la pseudo-classe. Significato e risultati sono comunque identici al primo esempio. Il primo tipo di sintassi garantisce una maggiore compatibilità con i browser più datati. Gli esempi e la sintassi presentati valgono per tutte le pseudo-classi.

Pseudoelementi: **:first-letter, :first-line, :before, :after**

nb: dalla CSS3 si introduce la notazione : : per distinguere pseudoelementi da pseudoclassi
ad esempio con p::first-line { text-transform: uppercase } si cambiano le lettere della prima linea di ogni paragrafo in maiuscolo

L'effetto in figura può essere realizzato col segmento
<P><SPAN><**P::first-letter**>T</**P::first-letter**>he first</SPAN>
few words of an article in the Economist.
</P>
con SPAN { text-transform: uppercase }



Come esempio di **pseudo-classe** si usa **:link** per impostare diversi **effetti sui Link**:

- **Link senza sottolineatura:**
```
<style type=text/css>
 a:link, a:visited { text-decoration: none}
</style>
```

- **Link sottolineato solo al passaggio del mouse:**
```
<style type=text/css>
 a:link, a:visited { text-decoration: none}
 a:hover { text-decoration: underline }
</style>
```
- **Link che cambia colore al passaggio del mouse:**
```
<style type=text/css>
 a:link, a:visited {color: #00ff00}
 a:hover {color: #ff0000 }
</style>
```
- **Link che s'ingrandisce al passaggio del mouse:**
```
<style type=text/css>
 a:link { font-size: 15px }
 a:visited { font-size: 15px }
 a:hover { font-size: 18px }
</style>
```

Si applica solo all'elemento (X)HTML **<A>** che abbia anche l'attributo **href**. Quindi, non alle cosiddette ancore invisibili ma solo ai link ipertestuali. Definisce lo stile per questo elemento quando il collegamento punta ad un sito o ad una pagina non ancora visitati.

:active quando si clicca su un collegamento ipertestuale

:visited quando si ritorna dal collegamento ipertestuale andato a buon fine

:hover Azione al passaggio del mouse

:focus Su Explorer Win non noterete nulla, ma e' attivo quando il campo di testo riceve il **focus**

:lang stile attivo per data lingua

CSS per specificare il posizionamento

Oltre a rimpiazzare i tag di formattazione, gli Style Sheet si propongono come **sostituti di tabelle e gif trasparenti** per specificare il posizionamento degli elementi della pagina HTML. E' possibile specificare che ogni oggetto sia contenuto in un rettangolo (*bounding box*) del quale si possono indicare posizione (sia come distanza dai bordi della finestra sia relativamente ad altri oggetti), dimensione, sovrapporre ad altri oggetti e rendere invisibile.

La posizione di un elemento si specifica nello style sheet con la *proprietà **position***, ad esempio si usa :
UL { position : absolute; top: 100px; left: 50px; width: 400px}
per ottenere tutte le liste larghe 400 pixel, a 50 pixel dal bordo sinistro e 100 da quello superiore della finestra del browser.

Per ottenere il posizionamento separando il contenuto nella pagina si può ricorrere ai **tag DIV** e agli **stili inline**:
<DIV STYLE= "position : absolute; top: 150px; left: 150px; background-color: blue">Questo testo e' contenuto in rettangolo blu a 150 pixel dai bordi superiore e sinistro della finestra del browser.
</DIV>

Float per avvolgere altro elemento

Il floating è un'operazione che veniva fatta in HTML sulle immagini. Bastava usare nel tag IMG l'attributo align e impostare come valore left, right, middle, etc.

Sintassi: <selettore> {float: <valore>}

Valori **left.** L'elemento viene spostato sul lato sinistro del box contenitore, il contenuto scorre a destra.
right. L'elemento viene spostato sul lato destro, il contenuto scorre a sinistra.
none. Valore iniziale e di default in mancanza di una dichiarazione esplicita. L'elemento mantiene la sua posizione normale.

Se usate il float con le immagini non avete problemi perché esse hanno una dimensione intrinseca che il browser riconosce. Ma se lo applicate ad altri elementi dovete esplicitamente impostare una dimensione orizzontale con la proprietà width.
Esempi

```
div {width: 200px; float:right;} // obbligatoria dimensione orizzontale
img {float: left;}
```

Vedi anche **clear** ed esempi http://www.html.it/css/test/float_clear.html

Per ottenere **effetti di sovrapposizione** tra elementi si usa la *proprietà **z-index***, che specifica il livello in cui si trova l'elemento cui e' applicata (più il valore e' alto, più vicino all'osservatore e' l'oggetto) ad esempio con:

```
.sotto{ position : absolute; top: 100px; left: 20px; z-index: 1; background-color: yellow}
.sopra { position : absolute; top: 115px; left: 40px; z-index: 2; background-color: red}
```

si potranno impostare livelli <P CLASS= sotto> e <P CLASS=sopra>.

Per effetti di sovrapposizione e **per effetti nascosti** (rettangoli in IE sono zone in Netscape) esempi

Le proprietà singole sono la maggior parte: impostano per un dato elemento o selettore un singolo aspetto.

Con le **shorthand properties** (*proprietà abbreviate*), è possibile invece definire con una sola dichiarazione un insieme di proprietà.

Chiariamo con un esempio.

Ogni elemento presenta sui suoi quattro lati un certo margine rispetto a quelli adiacenti. È possibile definire per ciascuno di essi un valore usando quattro proprietà singole separate:

- **margin-top**
- **margin-right**
- **margin-bottom**
- **margin-left**

La regola sarebbe questa:

```
div {
        margin-top: 10px;
        margin-right: 5px;
        margin-bottom: 10px;
        margin-left: 5px;
}
```

Lo stesso risultato si può ottenere usando la *proprietà a sintassi abbreviata* **margin**:

```
div {margin: 10px 5px 10px 5px;}
```

Esercizio[2]**:**
Ricerca le proprietà (usi e costrutti sintattici di ciascuna) dell'elenco:
background | border | border-top | border-right | border-bottom | border-left | cursor | font | list-style | margin | padding.

Vedi anche at_media.html in test.rar che contiene anche esempi d'uso di Background-attachment: fixed, Background-attachment:scroll, Background-color, Background-image, Background-position, Background-repeat, Color, Cursor, proprietà singole ed abbreviate per bordi, margini, padding, liste e posizione, chiarimenti sulla struttura di box, Display, Ereditarietà, Float e clear, Overflow per compensare superamento dei limiti di height, pseudoclassi e pseudoelementi …

nb:
gli esempi d'uso di proprietà CSS, memorizzati in forma compressa sono anche consultabili con navigazione nella guida on-line "CSS di base" di HTML.it

Ad esempio, all'indirizzo http://www.html.it/guide/esempi/css/test/zindex.html si può vedere l'effetto nell'uso della proprietà z-index

[2] Reference anche http://www.morpheusweb.it/html/manuali/css.asp con esempi

Aspetto diverso a seconda del “supporto”

L'attributo **media** può accompagnare sia l'elemento <LINK> che l'elemento <STYLE>. Ecco due esempi di sintassi (funzionanti con Netscape 6 e Internet Eplorer a partire dalla versione 5):

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

Una versione per il video e una per la carta (http://www.fucinaweb.com/design/stampacss.asp)
Esempio che usa la creazione i due css: **video.css** e **printer.css.** L'unica differenza tra i due css è la presenza della regola **display:none** per la versione di stampa. La regola display posta a none nasconde i tag che la usano. Si crea dunque una diversa classe .screen {display: none} e la si associa a tutte le parti del documento Html che non si vogliono stampare.

Uso di **id** per diversificare la visualizzazione su dispositivi diversi

(http://professoressa.altervista.org/style/esempio01.html e http://css.html.it/guide/lezione/89/ridefinire-il-foglio-distile-per-lo-schermo/)

NB:
Con i browser più recenti è possibile definire all'interno della direttiva @import anche il supporto cui applicare il CSS, in modo simile a quanto abbiamo visto a proposito dell'attributo **media**. Per fare ciò basta far seguire all'url del CSS l'indicazione di uno dei media previsti nella specifica:

```
<style type="text/css">
@import url(foglio_stampa.css) print, tv, aural;
@import url(foglio_schermo.css) screen;
</style>
```

Le precedenti regole hanno lo stesso effetto dell’uso della notazione @media ma sono preferiti nei download. Anche in questo modo, invece di costruire due fogli di stile, si creano in un unico CSS esterno queste regole per diversificare l'aspetto dell'elemento:

```
@media print,tv, aural {
  h1 {color: black;}
  }
@media screen {
h1 {color: red;}
}
```

Per approfondimenti http://www.webaccessibile.org/argomenti/documento.asp?DocID=133 (suoni)

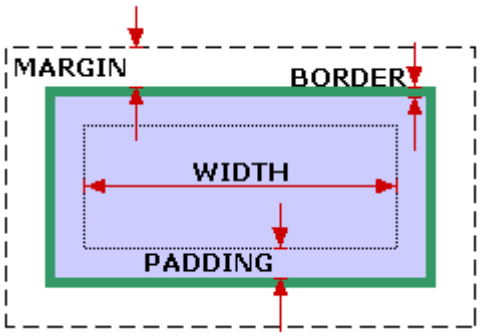
Un altro modo per impostare stili alternativi è ricorrere all’uso di JavaScript.

Il box model dei CSS

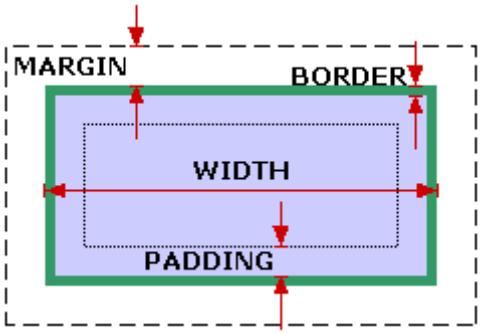
Il box model[3] dei CSS permette di definire dei blocchi rettangolari con specifici valori per la larghezza e l'altezza della sezione contenuti, del padding, dei bordi, dei margini. Purtroppo, IE5 per Windows non ha una interpretazione corretta del box model. E' molto importante conoscere il problema, essendo il box model uno dei migliori strumenti offerti dai CSS ed essendo IE5 per Windows ancora il browser più diffuso.

Il box model viene specificato impostando la larghezza e l'altezza dei contenuti, il padding, dimensioni e stile del bordo e il margine. Il padding è lo spazio fra i contenuti e il bordo, mentre il margine è lo spazio fra il bordo e gli altri contenuti della pagina.

Quando, con la proprietà 'width', si specifica la larghezza del box, si specifica la larghezza dell'area che conterrà il testo del box. La larghezza del box fino al bordo è la somma della larghezza specificata con width + ampiezza del padding + spessore del bordo (vedi immagine successiva).



Questo secondo gli standard CSS



IE5 per Windows (non IE5 per Mac) considera la larghezza specificata con width, come la larghezza del box fino al bordo.
Bordo e padding vengono scalati dalla larghezza specificata

nb: con i CSS3 si possono realizzare in modo facile bordi arrotondati senza ricorrere ad immagini. Si vedano esempi e compatibilità

Bordi					
border-radius	9.0+	4.0+	5.0+	4.0+	10.5+

[3] Vedi anche http://css.html.it/guide/lezione/28/il-box-model/

Link utili

Word Wide Web Consortium http://www.w3.org/ in particolare la sezione Style Sheets (W3C)
o più in dettaglio la sottosezione con gli esempi http://www.raggiorosso.com/Style/Examples/007/ (**italiano**)
oppure un chiaro capitolo introduttivo ai CSS http://www.w3.org/Style/LieBos2e/enter/ (inglese)

Analog http://www.analog.cx/ per rilevare quanti visitatori in percentuale usano browser recenti

Errori nella costruzione di un sito:
importante è fare tutto il contrario rispetto all'autore del **sito più brutto del mondo**... **Il sito internet più brutto del mondo**

AskTog http://www.asktog.com/menus/designMenu.html con l'illustrazione degli errori più comuni nel progetto di web design "Top 10 Mistakes of Web Design" (inglese)

In lingua italiana:
http://web-link.it/css/css.htm (guida semplice) ad esempio per colori http://web-link.it/html/colori.htm
http://www.html.it/css/ o http://www.risorse.net/css/ o http://www.fauser.edu/fau/css/tuttocss.htm
http://pro.html.it/lista_articoli.asp/idcat_8/ (CSS3)
http://www.lukeonweb.net/linguaggi.asp?lang=Css&cat=Trucchi anche per trucchi
http://www.graficagratis.com/tecnica/guida_css.asp (esempi di modifica grafica dei link)
http://www.usabile.it/cssdesign.htm usabilità ed accessibilità http://lau.csi.it/risorse/CSS2/index.shtml
http://www.fucinaweb.com/design/stampacss.asp (stampare con i CSS)

con link di approfondimento:

> Per approfondire l'argomento CSS:
>
> W3C-CSS2: Raccomandazioni del W3C sui CSS2
>
> W3-School: a scuola di CSS direttamente dal W3C
> http://www.w3schools.com/css3/default.asp (in costruzione Raccomandazioni del W3C sui CSS3)
> http://www.w3.org/Style/CSS/current-work (in costruzione Raccomandazioni del W3C)
>
> Web Page Design for Designers: Joe Gillespie Con Stile: risorsa italiana ricca di tutorial ed esempi originali Glish.com: tutto sulle tecniche di Layout tramite CSS

Uso di CSS per introdurre **creatività** http://www.meyerweb.com/eric/css/edge/index.html
in particolare esempi d'uso effetto float (Eric meyer) : http://www.meyerweb.com/eric/css/edge/gallery.html
con reference in http://www.meyerweb.com/eric/css/references/css2ref.html
Esempi di template http://www.intensivstation.ch/en/templates/
CSS per emulare immagini mappate
http://stclassi.altervista.org/HTML_CSS/Emulare_le_immagini_mappate_con_i_CSS.pdf
CSS per modificare barre a scorrimento
http://stclassi.altervista.org/HTML_CSS/MODIFICARE_LA_BARRA_DI_SCORRIMENTO_CON_I_CSS.pdf

Uso di JavaScript per stili alternativi

```
<html>
<head>
        <title>Stili alternativi</title>
        <script type="text/javascript" src="styleswitcher.js"></script>
        <link rel="stylesheet" type="text/css"  title="principale" href="screen.css" />
        <link rel="alternate stylesheet" type="text/css"  title="alternativo" href="print.css" />
</head>
<body>
<h1>Stili alternativi</h1>
<p>A questo documento abbiamo applicato due fogli di stile, uno principale e uno
 alternativo. Questo il codice HTML: </p>
<p>&lt;link rel=&quot;stylesheet&quot; type=&quot;text/css&quot; title=&quot;principale&quot;
 href=&quot;screen.css&quot; /&gt;<br />  &lt;link rel=&quot;alternate stylesheet&quot;
type=&quot;text/css&quot; title=&quot;alternativo&quot;   href=&quot;print.css&quot; /&gt;<br />
</p>
<p>Al documento abbiamo collegato uno script tramite il quale possibile cambiare
 il foglio di stile. Verificate cliccando sul link qui sotto.</p>
<p><a onclick="setActiveStyleSheet('alternativo'); return false;" href="#">Alternativo</a></p>
</body>
</html>
```

Con file "**styleswitcher.js**":

```
function setActiveStyleSheet(title) {
 var i, a, main;
 for(i=0; (a = document.getElementsByTagName("link")[i]); i++)
  {      if(a.getAttribute("rel").indexOf("style") != -1 && a.getAttribute("title"))
              {  a.disabled = true;
                 if(a.getAttribute("title") == title) a.disabled = false;
              }
    }
  }
function getActiveStyleSheet() {
 var i, a;
 for(i=0; (a = document.getElementsByTagName("link")[i]); i++)
  {    if(a.getAttribute("rel").indexOf("style") != -1 && a.getAttribute("title") && !a.disabled)
            return a.getAttribute("title");
    }
 return null;
}
```

Con file "**print.css**":

```
body {          font-family: "Times New Roman", serif; font-size: 12pt; background:
White;}
                H1 {background: White; color: Black;}
```

Con file "**screen.css**":

```
body { font-family: "Verdana",  sans-serif; font-size: 12px; background: White;}
                H1 {background: Black; color: Red;}
```

APPENDICE

Il 26 gennaio 2000 sono state pubblicate le specifiche di XHTML (Extensible HyperText Matkup Language). Si tratta di una riformulazione di HTML 4 in termini di XML e può già funzionare con i browser esistenti.

XHTML è il primo passo verso un linguaggio per il web estensibile e modulare.
E' molto simile a HTML 4 con qualche eccezione di rilievo, tra cui:

- tag e attributi devono essere scritti in **minuscolo** (<td bgcolor="#ffcc33">)
- gli elementi devono **nidificarsi**, non sovrapporsi (Be bold!)
- tutti gli elementi non vuoti devono essere **chiusi** (First paragraph)
- gli elementi vuoti devono essere **terminati** (
)
- il valore di un **attributo va tra virgolette** (
- le **coppie attributo-valore non** possono essere **minimizzate** (<option ... selected="selected">)

La **struttura di base** di un documento XHTML è

```
<!DOCTYPE ...>
<html ... >
<head> ... </head>
<body> ... </body>
</html>
```

e un documento minimale è questo:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "DTD/xhtml1-
  transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>   <title>Minimal document</title>   </head>
<body>
<p> <a href="http://validator.w3.org/check/referer"> validate</a> </p>
</body>
</html>
```