



Dispensa MIUR 06 Il linguaggio Html (1° parte)

1.	Introduzione	3
1.1.	Come funziona un browser	3
	L'HTML e i browser	3
1.2.	Cos'è l'HTML.....	4
1.3.	Lo standard HTML	4
2.	Come è fatta una pagina HTML	5
2.1.	Le estensioni dei file e le impostazioni del browser	5
	L'estensione del file.....	5
2.2.	I TAG dell'HTML: come scriverli	6
	Struttura di un tag.....	6
	Annidamento e indentazione.....	6
2.3.	I commenti	7
2.4.	Maiuscolo o minuscolo?	7
2.5.	Struttura della pagina	8
3.	L'HTML e i fogli di stile (CSS).....	9
3.1.	Separare il layout dal contenuto.....	9
3.2.	Gli elementi HTML e i fogli di stile	10
4.	Lo sfondo di un documento HTML	10
4.1.	Impostare il colore di sfondo	10
4.2.	Inserire un'immagine di sfondo	12
4.3.	Eliminare i margini delle pagine.....	13
4.4.	Impostare la lingua del documento	14
4.5.	Approfondimenti: lo sfondo con i CSS.....	14
5.	Il testo di un documento HTML	15
5.1.	Il testo.....	15
5.2.	I link	15
5.3.	Titoli, paragrafi, blocchi di testo e contenitori.....	17
6.	titolo 1	17
6.1.	titolo 2	17
	titolo 3	17
	Allineare il testo.....	19
6.2.	Scegliere lo stile (grassetto, corsivo & C.).....	20
6.3.	Gli stili fisici	20
6.4.	Gli stili logici	21
6.5.	Scegliere il font del testo.....	24
6.6.	Scegliere il colore del testo	25
6.7.	Le dimensioni del testo	27
6.8.	NOTA BENE.....	29
6.9.	Gli elenchi nell'HTML.....	29
7.	Gli elenchi ordinati	29
8.	Gli elenchi non ordinati	31
9.	Elenchi di definizioni	32
9.1.	Approfondimenti.....	32

1. Introduzione

1.1. Come funziona un browser

L'HTML e i browser

L'HTML è il linguaggio con cui potete indicare come i vari elementi vanno disposti in una pagina Web. Un documento html non è nient'altro infatti che un file di testo con delle indicazioni sul colore delle scritte, sulla posizione delle immagini all'interno della pagina, su come far scorrere il testo, e altre cose di questo genere.

Il **Browser** è il programma che usate quando navigate nel Web e svolge principalmente due compiti:

- scarica i vari files che si trovano su un computer remoto (il server) e che fanno riferimento a un certo indirizzo
- legge i documenti scritti in html, e a seconda delle indicazioni ivi contenute, visualizza la pagina in un modo, piuttosto che in un altro; inoltre i vari files associati a quel documento (ad esempio le immagini, o i filmati in flash) vengono disposti secondo le indicazioni del codice html

Oltre ad Internet Explorer, il browser più diffuso, esistono altri browser: prima di tutto lo "storico" **Netscape Navigator**, con cui la Microsoft ha ingaggiato una vera e propria guerra (vincendola). Poi il browser open source **Mozilla**, che nasce da Netscape e ha la particolarità di essere a codice aperto, cioè con la possibilità per gli sviluppatori di vedere com'è fatto il programma. Una parte di utenti (si tratta sempre di una minoranza comunque rispetto allo strapotere di Internet Explorer) utilizza poi **Opera**, un browser norvegese celebre per la sua velocità di visualizzazione delle pagine. Ovviamente esistono anche molti altri browser. Per ciascuno di essi esistono poi differenti versioni a seconda del sistema operativo (Windows, Mac OS, Linux, o altri).

È importante sin dall'inizio acquisire una **mentalità multi-browser**, perché il mestiere del webmaster non consiste tanto nel conoscere nei minimi dettagli il codice HTML, quanto piuttosto nel sapere come il codice HTML verrà visualizzato sul computer dell'utente: infatti uno dei lavori più difficili è quello di riuscire a far vedere correttamente il proprio sito con i browser e le piattaforme più svariate.

I files scaricati dal web vengono memorizzati in una particolare cartella del computer che prende il nome di **cache**.

In Internet Explorer è possibile visualizzarla utilizzando i comandi:

Strumenti > Opzioni Internet > Generale > Impostazioni > Visualizza file

In Mozilla:

Modifica > Preferenze > Avanzate > Cache

In questo modo verrà mostrato il percorso della cartella in cui i documenti vengono temporaneamente memorizzati.

La visualizzazione di un file html da parte del browser prende il nome di **rendering** della pagina. **Motore di rendering** è dunque quella sezione del browser che si occupa di mostrare sul video la pagina.

Il compito del linguaggio HTML è dunque quello di spiegare al browser come i vari files relativi al documento in esame devono essere disposti all'interno della pagina che stiamo visualizzando.

In qualsiasi momento è possibile visualizzare il **codice HTML** delle pagine che stiamo visitando. Con Internet Explorer:

Visualizza > HTML

Studiodelta Srl - Via G. Amendola 162/1-70126 Bari - Italy tel.+39 080 5461860 - fax +39 080 5461878 P.IVA : 04366410720

Con Mozilla :

Visualizza > Codice Sorgente

oppure si può effettuare la stessa operazione, utilizzando il tasto destro del mouse per visualizzare il menu a tendina, e scegliendo poi la voce corrispondente.

1.2. Cos'è l'HTML

HTML è l'acronimo di **Hypertext Markup Language** ("Linguaggio di contrassegno per gli Iper testi") e non è un linguaggio di programmazione (sono linguaggi di programmazione il C, il C++, il Pascal, il Java, e sono linguaggi di scripting il PHP, l'ASP, il PERL, il JavaScript).

Si tratta invece di un **linguaggio di contrassegno** (o 'di marcatura'), che permette di indicare come disporre gli elementi all'interno di una pagina: le indicazioni vengono date attraverso degli appositi marcatori, detti "tag".

Ciò significa che l'HTML **non ha meccanismi che consentono di prendere delle decisioni** ("in questa situazione fai questo, in quest'altra fai quest'altro"), e non è in grado di compiere delle iterazioni ("ripeti questa cosa, finché non succede questo"), né ha altri costrutti propri della programmazione.

Il linguaggio HTML, pur essendo dotato di una sua sintassi, **non presuppone la logica ferrea e inappuntabile dei linguaggi di programmazione**: se vi dimenticate di chiudere un tag, non verranno prodotti dei messaggi di errore; se non rispettate la sintassi probabilmente non otterrete la visualizzazione della pagina che desiderate, ma nient'altro. A volte vi troverete persino a dover adottare dei "trucchetti", non proprio da manuale, pur di visualizzare la pagina correttamente con ogni browser.

Suggerimenti: Può succedere - soprattutto a chi è alle prime armi - di continuare a modificare un file, ma di non riuscire a vederne le modifiche. Questo succede perché la pagina visualizzata è sempre quella vecchia memorizzata nella cache. Quando state elaborando pagine per il web, ricordatevi di impostare la cache del vostro browser in modo che il file html venga ricaricato ogni volta che richiamate la pagina.

In Internet Explorer:

Strumenti > Opzioni Internet > Generale > Impostazioni > Ricerca versioni più recenti delle pagine memorizzate:

- all'apertura della pagina

In Mozilla:

Modifica > Preferenze > Avanzate > Cache > Confronta la pagina nella cache con la pagina in rete:

- ogni volta che vedo una pagina

1.3. Lo standard HTML

L'organizzazione che si occupa di standardizzare la **sintassi del linguaggio HTML** (il W3C: [Word Wide Web Consortium](#)) ha rilasciato diverse versioni di questo linguaggio (HTML 2.0, HTML 3.2, HTML 4.0); e - da un certo punto in poi - l'HTML si è evoluto in XHTML (si tratta dell'HTML riformulato come linguaggio XML - ne sono già state rilasciate due versioni).

La versione dell'HTML che esamineremo in questo corso è l'ultima rilasciata: si tratta dell'HTML 4.01 del 24 dicembre 1999.

Anche se abbiamo detto che l'HTML si è evoluto in XHTML ci sono delle ottime ragioni per incominciare a studiare l'HTML e non l'XHTML:

- di fatto l'HTML **verrà utilizzato ancora per diversi anni** come linguaggio principe delle pagine web

Studiodelta Srl - Via G. Amendola 162/1-70126 Bari - Italy tel.+39 080 5461860 - fax +39 080 5461878 P.IVA : 04366410720

- alcuni concetti dell'XHTML richiedono già **una certa comprensione dei problemi** che si acquisisce solo con l'esperienza. L'HTML è più immediato e consente di incominciare subito a produrre documenti web
- **chi conosce l'XHTML non può non conoscere l'HTML**. La conoscenza dell'HTML è infatti il prerequisito essenziale di ogni webmaster. Comunque le differenze tra i due linguaggi non sono così marcate e passare dall'uno all'altro non dovrebbe richiedere molta fatica.

Per gli approfondimenti sulle differenze tra i vari linguaggi vi rimando tuttavia all'appendice di questa guida.

Un'ultima avvertenza: in molte lezioni è presente una sezione denominata "approfondimenti". Chi inizia adesso a studiare HTML ed è alla sua prima lettura può tranquillamente ignorare quel paragrafo. Le indicazioni ivi contenute vi torneranno utili a una seconda lettura, o man mano che prendete confidenza con l'HTML e l'arte di sviluppare siti web

2. Come è fatta una pagina HTML

2.1. Le estensioni dei file e le impostazioni del browser

Per iniziare a scrivere pagine web avete bisogno di:

- uno o più **browser** per visualizzare le pagine
- un **editor testuale** per scrivere il codice HTML (potete usare il blocco note di Windows, o altri editor testuali come 1Page, che è gratuito è First Page)
- durante questo corso non utilizzeremo editor visuali: né FrontPage, né DreamWeaver, né GoLive, o altri.

L'estensione del file

Aperte una pagina con il blocco note, e salvate il file in qualche cartella del vostro computer. Il file dovrà avere estensione "html", ad esempio **miaPagina.html**.

Fino a qualche tempo fa si era soliti attribuire ai file l'estensione **htm**, ma questo avveniva perché il dos e poi Windows 3.1 non erano in grado di gestire i file con nomi di grandezza superiore a 8 caratteri ed estensione superiore alle 3 lettere. Dunque **.html** era diventato **.htm**, così come **.jpeg** era diventato **.jpg**.

Il problema delle estensioni è stato ampiamente superato sin dai tempi di Windows 95, e di conseguenza oggi il webmaster può decidere se attribuire ai files estensione .html o .htm. Siccome stiamo parlando di linguaggio HTML, personalmente preferisco l'estensione .html, ma è una questione di gusti (Nomesito.it, ad esempio, continua con il vecchio metodo).

Se avete dato alla pagina l'estensione .html o .htm, il browser dovrebbe essere in grado di aprire il file in automatico cliccandoci su due volte. Per modificare la pagina utilizzate i comandi **Visualizza > HTML**, cambiate il codice, salvate, utilizzate il pulsante "aggiorna" del browser... e dovrete visualizzare le modifiche.

Se invece il file non è associato al browser, ma continua ad apparire come documento di testo, evidentemente questo avviene perché l'estensione non è .html, ma **.html.txt**, alcuni sistemi operativi hanno infatti la cattiva abitudine di nascondere l'estensione dei file (con il pretesto di rendere più usabile il sistema operativo stesso).

Per visualizzare l'estensione del file in sistemi Windows andate in una cartella e quindi:

Strumenti > Opzioni cartella > Visualizzazione

E poi togliere la spunta da:
"Nascondi le estensioni dei file per i tipi di file conosciuti"

infine premere il pulsante:
"Come cartella corrente"

2.2. I TAG dell'HTML: come scriverli

Struttura di un tag

Abbiamo detto che all'interno di ogni pagina è presente una serie di marcatori (i **TAG**), a cui viene affidata la visualizzazione e che hanno differenti nomi a seconda della loro funzione. I tag vanno inseriti tra parentesi uncinate (<TAG>), la chiusura del tag viene indicata con una "/" (è il simbolo comunemente detto "slash". Quindi: </TAG>). Il contenuto va inserito tra l'apertura e la chiusura del tag medesimo, secondo questa forma:

```
<TAG attributi>contenuto</TAG>
```

Ecco un esempio, con una sintassi che serve a disporre un testo giustificato a destra:

```
<P align="right">testo</P>
```

dall'esempio è evidente che la struttura di un attributo è: **attributo="valore"**

Quindi in definitiva la struttura di un tag sarà:

```
<TAG attributo_1="valore1" attributo_2="valore2">contenuto</TAG>
```

Alcuni particolari tag non hanno contenuto - perché ad esempio indicano la posizione di alcuni elementi (come il tag delle immagini) -, conseguentemente questi tag non hanno neanche chiusura. La loro forma sarà dunque:

```
<TAG attributi>
```

Ecco un esempio di immagine:

```
<IMG width="20" height="20" SRC="miaImmagine.gif" ALT="alt">
```

come si vede il tag non viene chiuso. Questo tipo di tag viene detto "empty", cioè "vuoto".

Annidamento e indentazione

Una caratteristica importante del codice HTML è che i tag possono essere annidati l'uno dentro l'altro. Anzi molto spesso è necessario farlo.

Ad esempio:

```
<TAG1 attributi>
  contenuto 1
  <TAG2>
    contenuto 2
  </TAG2>
</TAG1>
```

Potremmo quindi avere ad esempio:

```
<P align="right">
  testo 1
  <P align="left">
    testo 2
  </P>
</P>
```

L'annidamento ci permette quindi di attribuire formattazioni successive al testo che stiamo inserendo.

Come si può vedere già nell'esempio, è una buona norma utilizzare dei **caratteri di tabulazione** (il tasto tab a sinistra della lettera Q) per far rientrare il testo ogni volta che ci troviamo in presenza di un annidamento e man mano che entriamo più in profondità nel documento.

In pratica apertura e chiusura del tag si trovano allo stesso livello, mentre il contenuto viene spostato verso destra di un tab: non si tratta soltanto di un fattore visivo, ma l'allineamento di apertura e chiusura tag viene

mantenuto anche se scorriamo in verticale il documento con il cursore.

Questa procedura si chiama **indentazione**, e grazie ad essa il codice HTML risulta più leggibile. Si confronti ad esempio:

```
<P align="right">testo 1<P align="left"> testo 2 </P></P>
```

con:

```
<P align="right">
    testo 1
    <P align="left">
        testo 2
    </P>
</P>
```

</P>

per il browser i due esempi sono equivalenti, ma per l'utente umano è evidente che la differenza è notevole: pensate ad una pagina complessa visualizzata in un unico blocco di testo: sarebbe del tutto illeggibile!

2.3. I commenti

Un'altra strategia importante, per rendere il nostro codice più leggibile è quella di inserire dei **"commenti"** nei punti più significativi: si tratta di indicazioni significative per il webmaster, ma invisibili al browser.

Inserendo i commenti in punti specifici del documento ci permette di mantenere l'orientamento anche in file molto complessi e lunghi. La sintassi è la seguente:

```
<!-- questo è un commento -->
```

e ci permette di "commentare" i vari punti della pagina. Ad esempio:

```
<!-- menu di sinistra -->
```

```
<!-- barra in alto -->
```

```
<!-- eccetera -->
```

2.4. Maiuscolo o minuscolo?

L'HTML è **"case insensitive"**, cioè indipendente dal formato. Questo significa che è del tutto indifferente se scrivere i tag in maiuscolo o in minuscolo.

```
<P ALIGN="RIGHT">
```

e

```
<p align="right">
```

vengono letti allo stesso modo dal browser.

Fino a qualche tempo fa, per aumentare la leggibilità del codice, era buona norma scrivere in maiuscolo il nome del tag (es: **<P>**) e in minuscolo gli attributi (es: **align="right"**). Quindi:

```
<P align="right">
```

Tuttavia oggi, per analogia con l'XHTML (che è figlio dell'XML e dell'HTML ed è **"case sensitive"**, sensibile al formato) è consigliabile scrivere tutto in minuscolo, per abituarsi già al linguaggio che verrà. Maiuscolo e minuscolo, in ogni caso non costituiscono errore.

Fino a questo momento - per rendere più chiare le differenze - abbiamo utilizzato la vecchia abitudine di alternare maiuscolo e minuscolo differenziando tag e attributi, d'ora in poi invece tutta la sintassi HTML della guida sarà in minuscolo.

2.5. Struttura della pagina

Basandoci sulle indicazioni precedenti, incominciamo a scrivere la nostra prima pagina html.

Per prima cosa inseriamo una riga che indica che stiamo utilizzando le specifiche del Word Wide Web Consortium che riguardano il codice HTML:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//IT">
```

esamineremo ulteriormente questa riga nell'appendice, per ora lasciamola così.

Poi apriamo il nostro primo tag, che indica che quanto è compreso tra apertura e chiusura è in codice HTML:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//IT"> <html>
... altri tag ...
</html>
```

Un documento HTML è normalmente diviso in due sezioni:

Testa (<head>)	Contiene informazioni non immediatamente percepibili, ma che riguardano il modo in cui il documento deve essere letto e interpretato. Questo è il luogo dove scrivere - ad esempio - i meta-tag (alcuni sono ad esclusivo beneficio dei motori di ricerca), script JavaScript o VbScript, fogli di stile, eccetera
Corpo (<body>)	Qui è racchiuso il contenuto vero e proprio del documento

Ci occuperemo in seguito della head (l'argomento verrà ripreso poi nella conclusione della guida. Per ora facciamo riferimento soltanto a due tag che devono essere presenti in questa sezione:

```
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
```

indica al browser che deve caricare il set di caratteri occidentale (e non - ad esempio - il set di caratteri giapponese).

```
<title>Nome del sito</title>
```

Il title è il titolo della pagina e compare in alto sulla barra del browser (se guardate in alto a sinistra del browser noterete la scritta "Struttura della pagina | Guida HTML | Nomesito.it"). È bene compilarlo da subito, onde evitare poi di avere pagine senza titolo. Da quanto abbiamo detto la nostra prima pagina sarà questa,

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//IT">
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
  <title>Nomesito.it</title>
</head>
<body>
  <!-- Scriveremo qui -->
  Qui il nostro contenuto
</body>
</html>
```

D'ora in poi i vari tag che impareremo all'interno della guida andranno scritti all'interno del body, quando non sia indicato diversamente.

3. L'HTML e i fogli di stile (CSS)

3.1. *Separare il layout dal contenuto*

L'HTML in origine è nato come linguaggio per formattare i documenti presenti sul Web. Proprio per questo motivo il contenuto (ad esempio `<p>qui il mio testo</p>`) e i tag che indicano uno stile o una colorazione del contenuto (ad esempio ``, che colora il testo di rosso) si trovavano mischiati allo stesso livello.

Tuttavia vari anni di Web hanno fatto nascere l'esigenza di separare il contenuto dalla presentazione del contenuto medesimo.

Se per esempio io avessi tutti i titoli del mio documento in rosso e in grassetto, e a un certo punto decidessi di trasformarli in verde e in corsivo, con l'HTML classico (cioè l'HTML 3.2) dovrei andare a modificarmi a mano ogni tag contenente le indicazioni della formattazione.

Quindi:

```
<p>
  <font color="red">
    <b>titolo 1</b>
  </font>
</p>
```

diventerebbe:

```
<p>
  <font color="green">
    <i>titolo 1</i>
  </font>
</p>
```

ma se questa operazione non comporta difficoltà su una singola pagina, questa operazione diventa insostenibile (o quantomeno difficoltosa, tanto che converrebbe scrivere un programma che effettuasse la conversione al posto nostro) su website molto grandi, a volte di centinaia di pagine.

Proprio per questo – come dicevamo - da un certo punto in poi è nata l'esigenza di separare il contenuto (la scritta "titolo 1"), dalla formattazione (il colore rosso e il grassetto). Per farlo è necessario utilizzare i fogli di stile, e il contenuto della pagina vista poc'anzi diventerebbe qualcosa di questo genere:

```
<p class="formattaTitoli">
  titolo 1
</p>
```

mentre la colorazione del testo verrebbe poi affidata alla classe "formattaTitoli", descritta in un'altra parte del documento, o anche in un file separato. Dunque basta editare la classe "formattaTitoli" per cambiare l'aspetto anche di centinaia di pagine.

È importante sapere da subito che alcune cose che stiamo imparando hanno la possibilità di essere espresse con una soluzione più elegante, e che consente al webmaster di gestire più agevolmente i propri siti. Alcuni elementi descritti nella guida corrente sono addirittura "**deprecati**" dal W3C, cioè destinati a cadere in disuso (come il tag ``): man mano che li incontreremo (perché allo stato attuale del Web è ancora importante conoscerli) vi avvertirò che esistono altre soluzioni applicabili tramite i fogli di stile. Tuttavia in questo contesto non esamineremo i fogli di stile (detti anche CSS: "Cascading Style Sheets"), perché è un argomento che presuppone già la conoscenza del linguaggio HTML

3.2. *Gli elementi HTML e i fogli di stile*

Un altro concetto importante è che gli elementi vengono classificati nella trattazione a fogli di stile secondo tre tipologie:

Elementi di blocco	Sono sostanzialmente gli elementi che costituiscono un blocco attorno a sé, e che di conseguenza vanno a capo, come i paragrafi, le tabelle, le form.
Elementi "inline"	Sono gli elementi che – non andando a capo - possono essere integrati nel testo, come i collegamenti o le immagini
Liste	Lista numerate, o non numerate

La guida che state leggendo, senza entrare minuziosamente in questa classificazione, ne tiene conto, in modo da rendere più agevole il passaggio da una formattazione inserita nel codice HTML, a una formattazione che utilizzi i fogli di stile. Infatti, man mano che comincerete a costruire siti web, sentirete l'esigenza di passare a una formattazione avanzata. Le due cose tuttavia non vanno sentite in contrapposizione: i fogli di stile sono semmai un arricchimento e un'espansione del codice HTML, viceversa non è possibile apprendere i fogli di stile senza conoscere il codice HTML.

4. Lo sfondo di un documento HTML

4.1. *Impostare il colore di sfondo*

Incominciamo a vedere come ottenere la nostra prima pagina HTML nel modo in cui desideriamo visualizzarla.

Se vogliamo impostare un colore di sfondo è necessario impostare il relativo attributo del tag body. Così:

```
<body bgcolor="blue">
```

bgcolor sta per "background color", cioè "colore di sfondo". Molti colori sono disponibili utilizzando le corrispondenti parole chiave in inglese.

Tuttavia non è consigliabile inserire la notazione del colore facendo riferimento a questo tipo di sintassi, dal momento che non possiamo sapere esattamente a quale tonalità di colore corrisponda il blu del computer dell'utente. È preferibile in molti casi utilizzare la corrispondente codifica esadecimale del colore, che ci permette – tra le altre cose – di scegliere anche tonalità di colore non standard. Con la notazione esadecimale il nostro esempio diventa:

```
<body bgcolor="#0000FF">
```

Ecco una tabella con la notazione di alcuni colori (molti di essi sono disponibili anche nelle varianti "dark" e "light", ad esempio: "darkblue", "lightblue"):

colore	parola chiave	notazione esadecimale
arancione	orange	#FFA500
blu	blue	#0000FF
bianco	white	#FFFFFF
giallo	yellow	#FFFF00
grigio	gray	#808080
marrone	brown	#A52A2A

nero	black	#000000
rosso	red	#FF0000
verde	green	#008000
viola	violet	#EE82EE

Il numero di colori che l'utente ha a disposizione dipende dalla scheda video. Oggi si va da una risoluzione minima di 256 colori a una risoluzione che prevede svariati milioni di colori.

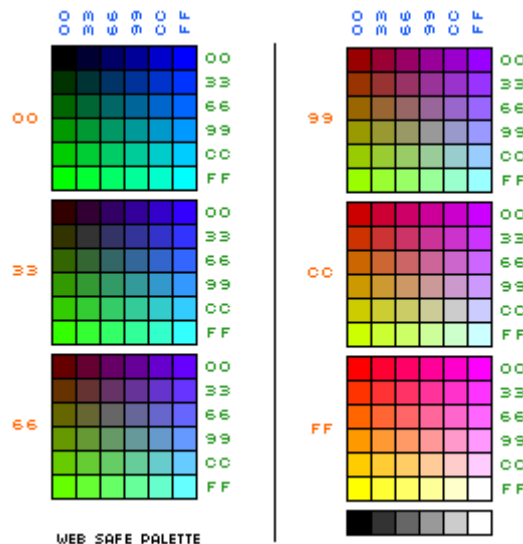
Per capire di cosa stiamo parlando, provate a visualizzare una pagina web cambiando il numero di colori visualizzati sul monitor. Per fare ciò, in Windows, andate in: **Pannello di controllo > Schermo > Impostazioni** e cambiate il numero dei colori, applicate i cambiamenti e tornate a visualizzare la pagina. Come si vede la visualizzazione della tonalità di colore è sensibilmente diversa passando da 256 a 65.536 colori (16 bit).

Poiché non c'è modo di sapere quale scheda video abbia l'utente (o come l'abbia impostata), i webdesigner per molto tempo hanno fatto riferimento alla "palette sicura" dei 256 colori che sicuramente l'utente è in grado di visualizzare. Si tratta della cosiddetta palette web safe.

Cosa sono i colori web safe?

I computer più vecchi hanno schede video e monitor in grado di visualizzare correttamente soltanto **256 colori**. Questi 256 colori vengono gestiti diversamente dalle piattaforme PC e Mac; a causa di queste differenze rimangono soltanto 216 colori comuni.

Se vogliamo essere sicuri che i nostri lavori vengano visti perfettamente da qualsiasi utente (anche quelli che hanno computer con settaggi video a 8 bit, ovvero che visualizzano solo 256 colori), dobbiamo utilizzare per i nostri lavori solo i 216 colori riportati nella tabella qui sotto, detti appunto colori "web safe".



Oggi, comunque, la stragrande maggioranza degli utenti ha computer che dispongono di una ampia quantità di ram video e di monitor moderni che consentono di visualizzare ben più di 256 colori.

Sintesi addittiva e sottrattiva

Parlando di web, è giusto considerare che i nostri lavori verranno nella maggior parte dei casi visualizzati su di un monitor e non letti su carta. Ci sono differenze fondamentali su come i vari media **visualizzano i colori** e vale la pena averne una minima conoscenza.

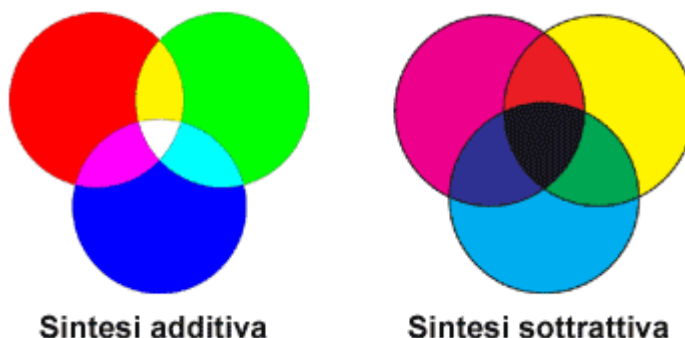
Studiodelta Srl - Via G. Amendola 162/1-70126 Bari - Italy tel.+39 080 5461860 - fax +39 080 5461878 P.IVA : 04366410720

I **monitor** (e in generale molte delle apparecchiature elettroniche) usano la sintesi additiva RGB (Red - Green - Blue), al contrario tutte le stampanti e le pubblicazioni su carta usano la sintesi sottrattiva CMYK (Cyan - Magenta - Yellow - black).

Senza entrare troppo in argomenti tecnici, lo "spazio colore" (così si chiama la gamma dei colori visualizzabili) dei monitor è dato dalla combinazione di fasci di luce di colore differente (appunto rosso, verde e blu) che colpiscono l'occhio; quest'ultimo, quando viene raggiunto da una luce costituita da due o più componenti, non è in grado di percepire separatamente le parti: ne elabora invece una sintesi globale, facendoci "vedere" un colore intermedio fra quelli iniziali.

Lo **spazio colore degli stampati** è costruito invece tramite i pigmenti ciano, magenta, giallo (e nero), che si sovrappongono e si combinano assorbendo ognuno la componente di luce del colore complementare al proprio e riflettendo invece la componente che hanno in comune.

È interessante notare che, mentre nella sintesi additiva il colore ottenuto dalla combinazione di rosso, verde e blu è il bianco, nella sintesi sottrattiva il colore risultante dalla somma di ciano, magenta e giallo è il nero. La ragione è che avendo ognuno dei colori primari della sintesi sottrattiva (ciano, magenta e giallo) il potere di assorbire una delle tre differenti parti della radiazione visibile, mescolandoli tutti e tre l'intero spettro visibile verrà assorbito e nessuna luce sarà riflessa verso l'osservatore.



È da notare inoltre che i colori CMYK risultano di solito meno brillanti di quelli RGB.

C'è però da dire che oramai la stragrande maggioranza dei computer è impostata per visualizzare almeno migliaia di colori, dunque l'utilizzo della palette "web safe" non è più così strettamente necessaria (lo era nei primi anni del web).

4.2. *Inserire un'immagine di sfondo*

Per inserire un'immagine come sfondo è sufficiente utilizzare la seguente sintassi:

```
<body background="imgSfondo.gif">
```

Per ora presupponiamo che l'immagine di sfondo si trovi nella stessa cartella della nostra pagina HTML, vedremo in seguito (quando parleremo delle immagini) come inserire immagini che si trovano in altre cartelle.

L'immagine di sfondo verrà ripetuta in orizzontale e in verticale.

È anche possibile combinare i due attributi, in modo che mentre l'immagine di sfondo viene caricata, venga comunque visualizzata una colorazione della pagina:

```
<body bgcolor="#0000ff" background="imgSfondo.gif">
```

È importante **assegnare sempre un colore alla pagina** anche quando lo sfondo della pagina è bianco (al massimo assegnare **bgcolor="#FFFFFF"**). Infatti, come impostazione predefinita, il browser assegna alla pagina il colore di sfondo che l'utente ha impostato nella finestra del sistema operativo: quindi se l'utente ha impostato uno sfondo nero e voi non avete assegnato nessun colore di sfondo alla pagina, la vostra pagina sarà nera.

4.3. *Eliminare i margini delle pagine*

Abbiamo detto all'inizio che il lavoro del webmaster consiste non soltanto nel conoscere alla perfezione il linguaggio HTML, ma soprattutto nell'essere un esperto del modo in cui i browser visualizzano le pagine.

Negli esempi precedenti avrete notate che il browser – secondo l'impostazione predefinita - lascia un po' di margine tra la pagina e il bordo della finestra. Questo in alcune situazioni (ad esempio se volete disporre un logo in alto a sinistra) può dare fastidio.

Per eliminare il bordo è sufficiente inserire i seguenti attributi del body:

```
<body leftmargin="0" topmargin="0">
```

Questa sintassi funziona correttamente con ogni browser moderno (Internet Explorer, Netscape 6 o superiore, Mozilla, Opera),

Tuttavia è bene sapere che i browser nel corso degli anni hanno introdotto dei tag e degli attributi "proprietary", con lo scopo di ottenere determinati effetti di visualizzazione, o indicare in qualche modo particolare il contenuto.

Questa situazione capitava soprattutto nei primi anni del web, quando Microsoft e Netscape lottavano per il predominio del mercato: in qualche misura la [guerra dei browser](#) è stata anche guerra di tag proprietari, con gravi difficoltà per gli sviluppatori che si trovavano continuamente di fronte a pagine che non venivano visualizzate allo stesso modo.

Per questo motivo fino a qualche anno fa per togliere il margine con Netscape 4.x dovevate inserire:

```
<body marginleft="0" margintop="0">
```

Mentre per togliere il margine con Internet Explorer:

```
<body leftmargin="0" topmargin="0">
```

Se avrete a che fare con pagine web di altri webmaster vi capiterà spesso di incontrare questo genere di sintassi:

```
<body leftmargin="0" topmargin="0" marginleft="0" margintop="0">
```

Questa sintassi serviva per eliminare il margine sia con Netscape 4.x, sia con Internet Explorer, specificando tutti e quattro gli attributi.

Al giorno d'oggi potete invece limitarvi a scrivere:

```
<body leftmargin="0" topmargin="0">
```

Fortunatamente negli ultimi anni l'ottica della guerra dei browser è cambiata, e i produttori di software sono passati dalla competizione per chi implementa nuove e fantastiche funzionalità proprietarie, al tentativo di rilasciare browser che aderiscano al meglio agli [standard del W3C](#) (non è un caso che sia la Netscape, sia la Microsoft facciano parte del consorzio), senza perdere di vista la velocità nell'effettuare il rendering della pagina.

L'adesione agli standard non può che essere un bene, dal momento che potenzialmente significa per noi sviluppatori la stesura di codice "universale", che funzioni correttamente a prescindere dal browser e dalla piattaforma (speriamo).

4.4. Impostare la lingua del documento

Tramite l'attributo **"lang"** è possibile specificare ai motori di ricerca e al browser dell'utente quale lingua stiamo utilizzando. La sintassi per la lingua italiana è:

```
<body lang="it">
```

Questo attributo non è solo una proprietà del tag body, ma può essere riferito alla maggior parte dei tag HTML che vedremo (come paragrafi, blocchi, tabelle, eccetera). È importante sottolineare che questo attributo non carica automaticamente il set di caratteri necessari alla visualizzazione della lingua, ma si limita a specificare che il documento (o parte del documento) è nella lingua indicata.

Si tratta di un attributo che vi sarà utile soprattutto se vi capiterà di sviluppare dei siti multilingua (e poi di doverli inserire nei motori di ricerca).

Ecco il codice che esemplifica gli argomenti appresi finora in questa lezione,

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//IT">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<title>Nomesito.it</title>
</head>
<body leftmargin="0" topmargin="0" background="imgs/sfondo00006.gif" bgcolor="#66CCFF"
lang="it">
Testo di prova
</body>
</html>
```

4.5. Approfondimenti: lo sfondo con i CSS

Tutti gli attributi del body che abbiamo visto finora (da eccezione dell'attributo **"lang"**) sono caratteristiche che riguardano il layout della nostra pagina HTML. Come si può vedere, con una sintassi di questo genere:

```
<body leftmargin="0" topmargin="0" background="imgs/sfondo00006.gif" bgcolor="#66CCFF"
lang="it">
    Il nostro testo.
</body>
```

il layout e il contenuto sono mischiati tra loro. Gli attributi "background" e "bgcolor" sono addirittura deprecati nelle specifiche del W3C: significa che andranno perduti.

In un approccio di impaginazione che utilizzi i fogli di stile, l'aspetto che riguarda la visualizzazione deve essere separato dal contenuto.

Il nostro body si ridurrà quindi a qualcosa di minimale, come:

```
<body lang="it">
```

mentre le regole che indicano come visualizzare lo sfondo saranno visualizzate in una locazione separata del documento.

I fogli di Stile sono estremamente potenti, è anche possibile fissare lo sfondo in modo che non si ripeta:

```
<body style="background-image: url(sfondo.gif); background-repeat: repeat;">
```

si tratta di una sintassi che funziona bene persino con Netscape 4.x,

Oppure è possibile "fissare lo sfondo" in modo da potervi fare scorrere sopra il contenuto della pagina. La sintassi è la seguente:

```
<body style="background-image: url(sfondo.gif); background-attachment: fixed;">
```

5. Il testo di un documento HTML

5.1. Il testo

Se non impostate nessun colore per il testo, di default il testo di una pagina è nero.

Tuttavia il nero non sempre è leggibile con tutti i colori di sfondo. Immaginate ad esempio di volere utilizzare come sfondo il colore nero: con una pagina nera e testo nero non leggeremmo nulla!

Abbiamo allora la possibilità di assegnare un colore per il testo di tutta la pagina, semplicemente utilizzando questo attributo del tag body:

```
<body text="red">
```

Quindi potremo avere, ad esempio:

```
<body bgcolor="#0000ff" text="#ffffff">
```

5.2. I link

Non c'è bisogno di spiegare che cosa siano i link: l'esperienza della navigazione nel web ci ha infatti insegnato che il link è un collegamento, un ponte tra una pagina e l'altra.

Non tutti però sanno che i link testuali hanno diversi stati:

Status	Codifica in HTML	Descrizione
Collegamento normale	link	Normalmente il link quando si trova "a riposo" viene

		evidenziato in qualche maniera all'interno della pagina HTML, in modo che sia facile per l'utente individuarlo. Nell'HTML tradizionale il link è sempre sottolineato (è possibile eliminare la sottolineatura soltanto usando i CSS). Di default i link sono blu (#0000FF).
Collegamento visitato	visited	Un link è visitato, quando l'URL della pagina compare nella cronologia dell'utente. Di default i link visitati sono di color violetto (più esattamente: #800080).
Collegamento attivo	active	<p>Il collegamento è attivo nel momento in cui il link è stato cliccato e sta avvenendo il passaggio da una pagina all'altra.</p> <p>Non si tratta di una caratteristica particolarmente utile oggi, ma quando i modem avevano una velocità molto inferiore a quella odierna, vedere un link "attivo" era comunque un'indicazione sul fatto che qualcosa stava avvenendo.</p> <p>Con Internet Explorer è possibile vedere anche una linea tratteggiata attorno al collegamento attivo.</p> <p>Un ulteriore condizione in cui un link si rileva "attivo" è quando si utilizza il tasto destro del mouse su di lui. Insomma un link è attivo quando "ha il focus".</p>
Collegamento al passaggio del mouse	non presente ("hover" nei CSS)	Con l'HTML 4.01 al passaggio del mouse sul link si può fare ben poco, coi fogli di stile invece è possibile creare qualche effetto di visualizzazione.

Abbiamo dunque tre stati canonici dei link (link a riposo, link attivo e link visitato) e una condizione aggiuntiva introdotta dai fogli di stile (status del link al passaggio del mouse):

Anche il colore dei link di tutta la pagina può essere tramite gli attributi del body:

I link secondo le impostazioni predefinite sono blu, per cambiare colore:

```
<body link="red">
```

Per cambiare colore ai link visitati (di default viola):

```
<body vlink="green">
```

i link visitati vengono memorizzate nella cronologia del browser, quindi se volete ripristinare il colore originario dei link, è sufficiente cancellare la cronologia.

Per cambiare colore ai link attivi:

```
<body alink="yellow">
```

La sintassi completa per impostare i link è quindi:

```
<body link="red" alink="yellow" vlink="green">
```

5.3. Titoli, paragrafi, blocchi di testo e contenitori

Nulla ci vieta di scrivere direttamente all'interno del tag body, come già abbiamo visto negli esempi precedenti, senza utilizzare nessun tag.

A dire la verità è però più pratico racchiudere il testo in appositi tag a seconda della funzione che il testo sta svolgendo. La nostra pagina risulterà più semplice da leggere, quando dovremo modificarla, e inoltre potremo ottenere la formattazione che desideriamo.

Come abbiamo detto dall'inizio, i tag sono infatti dei marcatori che ci permettono di mantenere ordine nella pagina e ottenere il layout che desideriamo.

I principali tag-contenitori da utilizzare per "racchiudere" il testo sono:

Nome tag	Visualizzazione codice	Descrizione
<code><h1>titolo 1 </h1></code>	6. titolo 1 6.1. titolo 2 titolo 3 titolo 4 titolo 5 titolo 6	"H" sta per "heading", cioè titolo: le grandezze previste sono sei. Dall' <code><h1></code> , che è il più importante, si va via via degradando fino all' <code><h6></code> .
<code><h2>titolo 2 </h2></code>		
<code><h3>titolo 3 </h3></code>		
<code><h4>titolo 4 </h4></code>		
<code><h5>titolo 5 </h5></code>		Il tag <code><hx></code> (sia esso <code>h1</code> o <code>h6</code>) risulta formattato in grassetto e lascia una riga vuota prima e dopo di sé. Si tratta dunque di un elemento di blocco.
<code><h6>titolo 6 </h6></code>		
<code><p>paragrafo </p></code>	paragrafo 1 paragrafo 2	Il paragrafo è l'unità di base entro cui suddividere un testo. Il tag <code><P></code> lascia una riga vuota prima della sua apertura e dopo la sua

Esempio:		chiusura.
<code><p>paragrafo 1</p></code>		
<code><p>paragrafo 2</p></code>		
<code><div>Blocco di testo</div></code>	blocco 1 blocco 2	Il blocco di testo va a capo, ma - a differenza del paragrafo – non lascia spazi prima e dopo la sua apertura.
Esempio:		
<code><div>blocco 1</div></code>		
<code><div>blocco 2</div></code>		
<code>contenitore</code>	contenitore 1 contenitore 2 contenitore 3	Lo span è un contenitore generico che può essere annidato (ad esempio) all'interno dei DIV.
Esempio:		
<code>contenitore 1</code>		Si tratta di un elemento inline, che cioè non va a capo e continua sulla stessa linea del tag che lo include.
<code>contenitore 2</code>		
<code>contenitore 3</code>		Avrete modo di utilizzare lo <code></code> soprattutto quando incomincerete ad usare i fogli di stile.

Le differenze tra `<P>`, `<DIV>` e `` sono quindi che:

- **<P>** lascia spazio prima e dopo la propria chiusura
- **<DIV>** non lascia spazio prima e dopo la propria chiusura, ma - essendo un elemento di blocco - va a capo
- **** -essendo un elemento inline - non va a capo

Per quel che riguarda il tag heading (**<h1>**, ..., **</h6>**) è da notare che la grandezza del carattere varia a seconda delle impostazioni che l'utente ha sul proprio computer.

Con Internet Explorer, ad esempio, basta andare in: Visualizza > Carattere

Per vedere il titolo crescere o decrescere.

Allineare il testo

Tutti i "tag-contenitori" che abbiamo appena visto (e molti altri tag di quelli che vedremo) permettono di allineare il testo utilizzando semplicemente l'attributo **align**.

Se avete seguito finora la presente guida, avrete anche indovinato che l'attributo **"align"** è **disapprovato dal W3C**, dal momento che per allineare il testo bisognerebbe invece utilizzare i fogli di stile.

In ogni caso, vediamo come potremmo ad esempio allineare il testo di un paragrafo:

Allineamento	Sintassi	Visualizzazione codice HTML
Testo allineato a sinistra	<code><p align="left">testo</p></code>	<code><p align="left">Nel mezzo del cammin di nostra vita mi ritrovai per una selva oscura ché la diritta via era smarrita</p></code>
Testo allineato a destra	<code><p align="right">testo</p></code>	<code><p align="right">Nel mezzo del cammin di nostra vita mi ritrovai per una selva oscura ché la diritta via era smarrita</p></code>
Testo giustificato	<code><p align="justify">testo</p></code>	<code><p align="justify">Nel mezzo del cammin di nostra vita mi ritrovai per una selva oscura ché la diritta via era smarrita</p></code>

Andare a capo

Per andare a capo molti webmaster utilizzando l'apertura arbitraria di paragrafi che non contengono nulla e che vengono lasciati aperti. Ad esempio:

```
<p>
```

```
<p>
```

```
<p>
```

Si tratta in buona sostanza di un errore, visto che per andare a capo esiste il tag `
` ("break", cioè "interruzione").

Per andare a capo è quindi sufficiente scrivere un `
`. Per saltare una riga ne occorrono due:

```
<br><br>
```

Un altro valido tag per dividere la pagina in parti è il tag `<hr>` ("horizontal rule"), che serve per tracciare una linea orizzontale. Ecco il tag in azione:

Questo tag ha anche alcuni attributi (deprecati, perché la formattazione andrebbe fatta con i CSS):

Studiodelta Srl - Via G. Amendola 162/1-70126 Bari - Italy tel.+39 080 5461860 - fax +39 080 5461878 P.IVA : 04366410720

L'attributo "**noshade**" evita di sfumare la linea, "**size**" indica l'altezza in pixel, "**width**" è la larghezza in pixel o in percentuale, "**align**" l'allineamento. Con Internet Explorer si riesce persino a impostare il colore:

```
<hr noshade size="5" width="50%" align="center" color="red">
```

Risultato:



6.2. Scegliere lo stile (grassetto, corsivo & C.)

Nella grafica cartacea con "stile di un testo" si intende la variante del "tondo", del "corsivo", o del "grassetto" di un carattere tipografico.

Nel parlare di stili del testo in HTML solitamente si suddividono i tag in grado di attribuire lo stile al testo in **stili fisici** e **stili logici**:

- vengono definiti come **fisici** quei tag che definiscono graficamente lo stile del carattere, indipendentemente dalla funzione del contenuto del tag
- vengono definiti come **logici** quei tag che forniscono anche informazioni sul ruolo svolto dal contenuto del tag, e in base a questo adottano uno stile grafico

6.3. Gli stili fisici

I principali stili fisici sono:

Codice HTML	Visualizzazione	Descrizione
<p><code>testo in grassetto</code></p> <p>Esempio:</p> <p>Questo <code>testo</code> è in grassetto</p>	<p>Questo testo è in grassetto</p>	<p>Formatta il testo in grassetto.</p>
<p><code><i>testo in corsivo</i></code></p> <p>Esempio:</p> <p>Questo <code><i>testo</i></code> è in corsivo</p>	<p>Questo <i>testo</i> è in corsivo</p>	<p>Formatta il testo in corsivo. Tuttavia bisogna evitare di evidenziare in corsivo dei blocchi di lunghezza considerevole, perché la leggibilità del corsivo nel web lascia a desiderare.</p> <p>Meglio limitarsi a poche parole.</p>
<p><code><pre>testo preformattato</pre></code></p> <p>Esempio:</p>	<pre>PHP_FUNCTION { zval **parameters; zval *value; char* str;</pre>	<p>Il motore di rendering del browser restituisce il testo così come è stato inserito nel file html dall'autore stesso (preformattato quindi), senza</p>

<pre><pre></pre>		riformattarlo.
<pre>PHP_FUNCTION { zval **parameters; zval *value; char* str; } </pre></pre>		È un tag che si usa soprattutto nella rappresentazione di codice di programmazione.
<pre><u>testo sottolineato</u></pre>	Questo <u>testo</u> è sottolineato	Sottolinea il testo presente nel tag.
Questo <u>testo</u> è sottolineato		
Esempio:		
Questo <pre><u>testo</u></pre> è sottolineato		Nel web le sottolineature del testo sono da evitare, per non confondere il lettore con i link.
<pre><strike>testo barrato</strike></pre>	Questo testo è barrato	Con il testo barrato, vengono indicate (ad esempio) le correzioni.
Esempio:		
Questo <pre><strike>testo</strike></pre> è barrato		
<pre><sup>testo in apice</sup></pre>	$E=mc^2$	"Superscript": indica al browser di portare il testo al di sopra della linea di scrittura. Utile per formule matematiche (ad esempio le potenze)
Esempio:		
<pre>E=mc<sup>2</sup></pre>		
<pre><sub>testo in pedice</sub></pre>	H_2O	"Subscript": indica al browser di portare il testo al di sotto della linea di scrittura (utile ad esempio per i simboli chimici)
Esempio:		
<pre>H<sub>2</sub>O</pre>		

Di fatto i tag `` e `<i>` sono molto utilizzati, perché consentono di cambiare lo stile del testo al volo.

6.4. Gli stili logici

Come abbiamo visto gli **stili logici** forniscono anche informazioni sul contenuto e la loro formattazione è spesso lasciata al browser con risultati a volte deludenti. Proprio per questo gli stili logici sono entrati in disuso e sono poco usati.

Riportiamo di seguito i principali stili logici, per completezza, ma non sarà necessario ricordarseli.

Codice HTML	Visualizzazione	Descrizione
<pre><abbr>abbreviazione</abbr></pre>	C/A Nomesito.it	Indica un'abbreviazione. Nessun rendering del testo

Esempio:		particolare.
<code><abbr>C/A</abbr> Nomesito.it</code>		
<code><acronym>acronimo</acronym></code>	HTML	Indica un acronimo. Nessun rendering del testo particolare.
Esempio:		
<code><acronym>HTML</acronym></code>		
<code><address>indirizzo</address></code>	<i>Nomesito.it - via dei Castani 183/185 – 00172 Roma</i>	Serve per indicare gli indirizzi: siano essi e-mail, o indirizzi fisici. Il testo viene visualizzato in corsivo.
Esempio:		
<code><address>Nomesito.it - via dei Castani 183/185 – 00172 Roma</address></code>		
<code><blockquote>blocco di citazione</blockquote></code>	Nel mezzo del cammin di nostra vita mi ritrovai per una selva oscura ché la diritta via era smarrita	Sono blocchi di citazione. Il testo viene rientrato verso destra.
Esempio:		
<code><blockquote> Nel mezzo del cammin di nostra vita mi ritrovai per una selva oscura ché la diritta via era smarrita </blockquote></code>		
<code><cite>citazione</cite></code>	<i>Nel mezzo del cammin di nostra vita mi ritrovai per una selva oscura ché la diritta via era smarrita</i>	Per citazioni brevi: il testo è visualizzato in corsivo.
Esempio:		
<code><cite> Nel mezzo del cammin di nostra vita mi ritrovai per una selva oscura ché la diritta via era smarrita </cite></code>		
<code><code>codice</code></code>	if (document.all) alert ("ciao");	Indica un blocco di codice in linguaggio di programmazione. Nessun rendering del testo particolare.
Esempio:		
<code><code>if (document.all) alert ("ciao"); </code></code>		
<code><dfn>definizione</dfn></code>	<i>L'HTML è un linguaggio di contrassegno</i>	Indica una definizione: il testo è visualizzato in corsivo.
Esempio:		
<code><dfn>L'HTML è un linguaggio di contrassegno</dfn></code>		
<code>enfasi</code>	Ti ho detto <i>questo!</i>	Serve per porre l'enfasi su un'espressione: il

Esempio: Ti ho detto <code>questo!</code> <code><kbd>keyboard</kbd></code>		testo è visualizzato in corsivo.
Esempio: <code><kbd>digitazione da tastiera</kbd></code> <code><q>citazione all'interno della frase</q></code>	digitazione da tastiera	Indica una digitazione da tastiera: il testo viene visualizzato a spaziatura fissa.
Esempio: Come diceva Don Abbondio: <code><q>&quot; Il coraggio, uno non se lo può dare&quot;</q></code> . <code><samp>esempio</samp></code>	Come diceva Don Abbondio: "Il coraggio, uno non se lo può dare"	Indica una citazione breve all'interno del testo. Nessun rendering del testo particolare.
Esempio: <code><samp>ecco un esempio di &quot;samp&quot;</samp></code> . <code>rafforzamento</code>	ecco un esempio di "samp"	Indica un esempio. Il testo viene visualizzato a spaziatura fissa.
Esempio: Ecco un <code>testo rafforzato</code> <code><var>variabile</var></code>	Ecco un testo rafforzato	Evidenzia una parola. Il testo viene reso in grassetto
Esempio: Inseriamo i dati nella variabile temporanea <code><var>temp</var></code> ...	Inseriamo i dati nella variabile temporanea <i>temp</i> ...	La variabile viene visualizzata in corsivo.

Approfondimenti

Come si può vedere molti tag (logici e fisici) tradiscono l'origine scientifica e informatica del Web (sono presenti tag per blocchi di codice di programmazione, per definizioni, per l'indicazione delle variabili...).

Sorprendentemente nessuno dei tag fisici o logici è stato dichiarato "deprecato" dal W3C, ma anzi tutti questi tag sono passati dall'HTML 3.2 originario fino all'XHTML (passando illesi attraverso l'HTML 4).

Per quel che riguarda i tag fisici: a rigor di logica lo stile "grassetto" dovrebbe essere ottenuto con i fogli di stile (così come tutte le formattazioni), ma evidentemente la possibilità di ottenere un testo in grassetto semplicemente scrivendo "**testo**" è troppo comoda per poter essere considerata obsoleta.

Studiodelta Srl - Via G. Amendola 162/1-70126 Bari - Italy tel.+39 080 5461860 - fax +39 080 5461878 P.IVA : 04366410720

Per quel che riguarda i tag logici: in realtà questo tipo di tag offrono un ulteriore aiuto al webmaster anche in un approccio a fogli di stile. Se infatti si ha l'accortezza di ridefinire i tag all'interno della definizione degli stili, si hanno molte occasioni di utilizzare una formattazione mirata a seconda della funzione del contenuto: in quest'ottica, il fatto che alcuni tag logici non restituiscano nessun rendering particolare è addirittura un invito a ri-definire lo stile del tag.

6.5. Scegliere il font del testo.

La presente lezione tratta la scelta del colore, delle dimensioni e del tipo di carattere del testo attraverso l'utilizzo del tag "font". Si tratta di un **argomento obsoleto**, perché la formattazione del testo in tutti i siti moderni viene attribuita attraverso i fogli di stile. L'utilizzo del tag **** inoltre è disapprovato dal W3C, e dunque sta cadendo in disuso. In ogni caso si tratta di un argomento che un buon webmaster non può ignorare: come già detto per studiare i fogli di stile ci sarà tempo, e comunque è un passo che viene dopo la conoscenza dell'HTML.

Il tipo di carattere (cioè il "font") che il browser visualizza di default è il "Times".

Purtroppo questo carattere (ottimo per la carta stampata) non è adatto a essere visualizzato sul monitor di un computer: è una questione di "grazie" (le grazie sono quegli abbellimenti tipografici delle lettere, che dovrebbero servire per rendere più leggibile il carattere).

Dal momento che i caratteri con grazie non ottengono il risultato voluto sul monitor (quello cioè di rendere le lettere maggiormente riconoscibili e di conseguenza il testo più leggibile), ma anzi ottengono l'effetto contrario, si preferisce di solito utilizzare dei caratteri senza grazie come il "Verdana", l'"Arial" o l'"Helvetica"

Per scegliere il tipo di carattere con cui un font deve essere visualizzato è sufficiente usare la sintassi:

<code>testo in Arial</code>	testo in Arial
<code>testo in Verdana</code>	testo in Verdana
<code>testo in Geneva</code>	testo in Geneva

Tuttavia è bene sottolineare da subito che non è possibile far sì che l'utente visualizzi un testo in un carattere fantasioso scelto da noi. Allo stato attuale dell'arte l'utente che naviga in internet può visualizzare solo i caratteri che sono installati nel suo sistema: in Windows si tratta dei caratteri presenti in: **Pannello di controllo > Tipi di caratteri**.

Per questo motivo è bene tener conto di due accorgimenti:

- scegliere caratteri "sicuri", che siano cioè senz'altro presenti sul pc dell'utente
- non indicare un solo carattere, ma una serie di caratteri che gradualmente si allontanano dal risultato che vorremmo ottenere, ma non di molto, fino ad indicare la famiglia a cui il nostro carattere appartiene. In questo modo il browser dell'utente cercherà di trovare nella propria cartella dei fonts il primo carattere indicato, se non lo trova passerà al secondo, e solo come ultima spiaggia sceglierà di utilizzare il carattere predefinito (il famigerato "Times")

Vediamo alcuni esempi di famiglie "sicure" di caratteri:

<code>Verdana e caratteri simili
</code>	Verdana e caratteri simili
	Arial e caratteri simili
<code>Arial e caratteri simili
</code>	Times e caratteri simili
	Curier e caratteri simili
<code>Times e caratteri simili
</code>	Georgia e caratteri simili
	Geneva e caratteri simili
<code>Curier e caratteri simili
</code>	
<code>Georgia e caratteri simili
</code>	
<code>Geneva e caratteri simili</code>	

È vero: l'impossibilità di scegliere i caratteri che preferiamo limita terribilmente le nostre possibilità espressive, ma il bello di sviluppare per il web è proprio accettare di creare con delle regole ben definite, e a volte anche molto vincolanti.

Per i titoli delle pagine, i menu, e quant'altro potremmo poi sempre utilizzare delle immagini con il nostro carattere tipografico preferito (ad esempio delle "gif").

6.6. Scegliere il colore del testo

Adesso che abbiamo scelto il carattere con cui scrivere il nostro testo possiamo scegliere il colore, con la sintassi:

<code>testo blu</code>	testo blu
ovvero:	ovvero
<code>testo blu</code>	testo blu

La scelta del colore può essere effettuata nello stesso momento in cui si sceglie il tipo di carattere (dal momento che "face" e "color" sono entrambi attributi del tag "font"). La sintassi è:

```
<font face="Verdana, Arial, Helvetica, sans-serif" color="blue"> testo blu in Verdana
</font>
```

Una volta scelto il colore possiamo sempre decidere di cambiarlo:

```
<font face="Verdana, Arial, Helvetica, sans-serif" color="blue"> testo blu in Verdana
</font>
testo blu in Verdana</font><br>
o meglio ancora:
<font face="Verdana, Arial, Helvetica, sans-serif" color="red"> testo blu in Verdana
</font>
testo rosso
o meglio ancora:
<font face="Verdana, Arial, Helvetica, sans-serif" color="blue">
testo blu in Verdana<br>
<font color="red">
testo rosso
</font>
```

La seconda sintassi è preferibile alla precedente, perché la scelta del tipo di carattere viene effettuata una sola volta, evitando così di scrivere del codice inutile. Da notare che per evitare la ripetizione i due tag sono annidati l'uno dentro l'altro.

6.7. Le dimensioni del testo

Le dimensioni del testo si attribuiscono mediante l'attributo "size" del tag font.

Ci sono due modi per dare attribuire le dimensioni al testo tramite il tag :

- **valori interi da 1 a 7**
- **valori relativi** alla dimensione di base del tag font (di default "3")

Nel caso dei **valori interi**, ecco la scala di grandezza:

testo di grandezza 1 	testo di grandezza 1
testo di grandezza 2 	testo di grandezza 2
testo di grandezza 3 	testo di grandezza 3
testo di grandezza 4 	testo di grandezza 4
testo di grandezza 5 	testo di grandezza 5
testo di grandezza 6 	testo di grandezza 6
testo di grandezza 7 	testo di grandezza 7

Nel caso dei valori relativi alla dimensione di base è possibile "spostarsi" nella scala di grandezza del utilizzando i segni "+" e "-".

Abbiamo detto che la grandezza del font di base di default nel browser è 3.

Dunque se utilizziamo un size="+2", vuol dire che la dimensione del font deve essere di 2 misure più grande della dimensione del font di base, quindi avremo un font di grandezza 5. Vediamo l'esempio:

 Testo di grandezza +2 rispetto al font di base (3). Cioè font di grandezza 5. 	Testo di grandezza +2 rispetto al font di base (3).
--	---

```
Testo di grandezza 5.  
</font>
```

Cioè font di
grandezza 5.

Testo di grandezza
5.

Come si può vedere le due sintassi sono equivalenti.

La grandezza del font di base può anche esser cambiata:

```
<basefont size="1">  
<font size="+2">  
Testo di 2 grandezze superiore al font di base, sopra definito.  
</font>  
<br>  
<font size="3">  
Testo di grandezza 3.  
</font>  
<br><br>  
  
<basefont size="2">  
  
<font size="+2">  
Testo di 2 grandezze superiore al font di base, sopra ridefinito.  
</font>  
<br>  
<font size="3">  
Testo di grandezza 3.  
</font>
```

È importante evitare di cadere nell'errore di pensare che la dimensione relativa faccia riferimento al precedente tag font. La dimensione relativa fa sempre riferimento alla dimensione del font di base:

**Ecco un esempio
corretto, ma che non
darà il risultato
desiderato, perché la
dimensione relativa fa
sempre riferimento al
<basefont>:**

```
<font size="7">  
Testo di grandezza 7  
<font size="-1">  
testo di grandezza inferiore  
di 1 al font di base (che di  
default è 3),  
NON al tag precedente  
</font>  
  
</font>
```

Testo di
grandezza

7

testo di grandezza inferiore di 1
al font di base (che di default è 3), NON
al tag precedente

6.8. **NOTA BENE**

Quando state utilizzando il tag - sia che utilizziate il size i valori interi, sia che utilizziate le i valori relativi al tag di base -, in realtà la grandezza del carattere **dipende dalle impostazioni del browser dell'utente** (come già abbiamo visto per i tag "heading").

Con Internet Explorer ad esempio andando in: **Visualizza > Carattere**.

Se cambiate le dimensioni del carattere, vedrete cambiare le dimensioni dei font.

Questo appunto per le grandezze da 1 a 7 sono grandezze anch'esse relative.

Questa caratteristica da un lato è positiva (permette di ingrandire testi piccoli), dall'altra può risultare molto fastidiosa per il webmaster.

L'unico modo per fissare il carattere è (ancora una volta) quello di utilizzare i fogli di stile.

6.9. **Gli elenchi nell'HTML**

Se abbiamo la necessità di inserire un elenco di termini, possiamo utilizzare le "liste", che sono sostanzialmente di tre tipi:

- **Elenchi ordinati**
- **Elenchi non ordinati**
- **Elenchi di definizioni**

Tutti e tre i tipi di elenchi funzionano nel medesimo modo: si apre il tag, si elencano i vari elementi della lista (ciascuno con il proprio tag), si chiude il tag dell'elenco. La sintassi ha quindi questa forma:

```
<elenco>  
  <elemento>nome del primo elemento  
  <elemento>nome del secondo elemento  
</elenco>
```

come si può vedere, il tag che individua l'elemento della lista non ha bisogno di chiusura (la sua chiusura, in questo caso, è opzionale).

Le liste di definizioni hanno una struttura leggermente diversa che vedremo a breve.

7. **Gli elenchi ordinati**

Gli elenchi ordinati sono contraddistinti dall'enumerazione degli elementi che compongono la lista. Avremo quindi una serie progressiva ordinata e individuata da lettere o numeri (se utilizzate un programma di videoscrittura, siete abituati a chiamarli **elenchi numerati**).

Il tag da utilizzare per aprire un elenco ordinato è **** ("ordered list") e gli elementi sono individuati dal tag **** ("list item"):

<pre><div>Testo che precede la lista primo elemento</pre>	<pre>Testo che precede la lista 1. primo elemento</pre>
--	---

```

    <li>secondo elemento          2. secondo elemento
    <li>terzo elemento          3. terzo elemento
  </ol>testo che segue la lista  testo che segue la lista
</div>

```

Da notare che il tag che individua l'elenco lascia una riga di spazio prima e dopo il testo che eventualmente lo circonda (come avviene per il **<p>**); fa eccezione però l'inclusione di un nuovo elenco all'interno di un elenco preesistente: in questo caso non viene lasciato spazio, né prima, né dopo.

Gli elementi dell'elenco sono sempre rientrati di uno spazio verso destra: tutto questo serve a individuare in modo inequivocabile l'elenco.

Lo stile di enumerazione visualizzata di default dal browser è quello numerica, ma è possibile indicare uno stile differente specificandolo per mezzo dell'**attributo type**. Ad esempio:

```

<ol type="a">
  <li>primo elemento
  <li>secondo elemento
  <li>terzo elemento
</ol>

```

Gli stili consentiti sono:

Valore dell'attributo type		Stile di enumerazione	
type="1" (è così di default)	numeri arabi	<pre> <ol type="1"> primo secondo terzo </pre>	1. primo 2. secondo 3. terzo
type="a"	alfabeto minuscolo	<pre> <ol type="a"> primo secondo terzo </pre>	a. primo b. secondo c. terzo
type="A"	alfabeto maiuscolo	<pre> <ol type="A"> primo secondo terzo </pre>	A. primo B. secondo C. terzo
type="i"	numeri romani minuscoli	<pre> <ol type="i"> primo secondo terzo </pre>	i. primo ii. secondo iii. terzo
type="I"	numeri romani maiuscoli	<pre> <ol type="I"> primo secondo terzo </pre>	I. primo II. secondo III. terzo

8. Gli elenchi non ordinati

Gli elenchi non ordinati sono individuati dal tag `` ("unordered list"), e gli elementi dell'elenco sono contraddistinti anch'essi dal tag `` (in buona sostanza si tratta di quello che i programmi di videoscrittura chiamano **elenchi puntati**):

```
<ul>
  <li>primo elemento
  <li>secondo elemento
  <li>terzo elemento
</ul>
```

il tipo di segno grafico utilizzato per individuare gli elementi dell'elenco di default dipende dal browser, ma di solito è un "pallino pieno". È possibile comunque scegliere un altro tipo di segno:

Valore dell'attributo type	Stile di enumerazione
type="disc" (è così di default)	visualizza un "pallino pieno" . È la visualizzazione di default <pre><ul type="disc"> primo secondo terzo </pre> <ul style="list-style-type: none"> • primo • secondo • terzo
type="circle"	visualizza un cerchio vuoto al proprio interno <pre><ul type="circle"> primo secondo terzo </pre> <ul style="list-style-type: none"> ○ primo ○ secondo ○ terzo
type="square"	Visualizza un quadrato pieno al proprio interno <pre><ul type="square"> primo secondo terzo </pre> <ul style="list-style-type: none"> ▪ primo ▪ secondo ▪ terzo

Da notare inoltre che il tipo di segno grafico, varia in automatico al variare dell'annidamento della lista. Ad esempio:

```
<ul>
<li>primo della 1a lista
<li>secondo della 1a lista
  <ul>
    <li>primo della 2a lista
      <ul>
        <li>primo della 3a lista
      </ul>
    <li>secondo della 2a lista
      <ul>
        <li>primo della 3a lista
      </ul>
    <li>terzo della 2a lista
      <ul>
        <li>primo della 3a lista
      </ul>
  </ul>
</ul>
```

- primo della 1a lista
- secondo della 1a lista
 - primo della 2a lista
 - primo della 3a lista
 - secondo della 2a lista
 - primo della 3a lista
 - terzo della 2a lista
 - primo della 3a lista

```
<li>terzo della 1a
lista
</ul>
```

9. Elenchi di definizioni

Gli elenchi di definizioni sono individuati dal tag **<dl>**. Gli elementi dell'elenco (a differenza delle liste ordinate, e delle liste non ordinate) questa volta sono formati da due parti:

```
<dt> definition term: indica il termine da definire. A differenza
dell'elemento <li> in questo caso non c'è rientro.
<dd> definition description: è la definizione vera e propria del
termine. L'elemento è rientrato.
```

Vediamo un esempio:

```
<p>Ecco i principali tag
per delimitare il testo:
<dl>
<dt><p>
<dd>individua l'apertura di un nuovo paragrafo
Ecco i principali tag per delimitare il
testo:

<dt><div>
<dd>individua l'apertura di un nuovo blocco di
testo
individua l'apertura di un nuovo
paragrafo

<div>
individua l'apertura di un nuovo
blocco di testo

<dt><span>
<dd>individua l'apertura di un elemento inline,
cui attribuire una
formattazione attraverso
gli stili
individua l'apertura di un
elemento inline, cui attribuire
una formattazione attraverso gli
stili
ci sono poi altri tag che...
</dl>
ci sono poi altri tag
che...
</p>
```

9.1. Approfondimenti

Ovviamente la scelta del tipo di elenco attraverso l'attributo **type** è deprecato dal W3C, perché si tratta di una caratteristica che riguarda la formattazione, e dunque andrebbe effettuata utilizzando i CSS. Con i fogli di stile c'è anche la possibilità di scegliere un'immagine (ad esempio una GIF) come segno distintivo per l'elenco puntato.