GUIDA HTML

Di Wolfgang Cecchin

Versione originale: http://xhtml.html.it/guide/lezione/1654/

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

Con Mozilla :

Visualizza > Codice Sorgente

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

Come funziona un browser

HTML è l'acronimo di **Hypertext Markup Language** ("Linguaggio di contrassegno per gli Ipertesti") 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 visualizzare 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

Prima di cominciare davvero: lo standard HTML

L'organizzazione che si occupa di standardizzare la **sintassi del linguaggio HTML** (il W3C: World Wide Web Consortium (http://www.w3.org/)) 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
- 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.

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 Ultra Edit oppure Html Kit, che è gratuito.
- durante questo corso non utilizzeremo editor visuali: né FrontPage, né DreamWeaver, né GoLive, o altri. Su HTML.it troverete delle guide appositamente scritte per loro.

L'estensione del file

Aprite 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 (HTML.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 dovreste 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).

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 definita 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 widht="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>
```

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!

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 -->
```

Maiuscolo o minuscolo?

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

<P ALIGN="RIGHT">

e

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 a maiuscole/minuscole - cfr. guida XHTML) 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.

Struttura della pagina

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:

<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"). È bene compilarlo da subito, onde evitare poi di avere pagine senza titolo.

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

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 ** qui il mio testo **) e i tag che indicano uno stile o una

colorazione del contenuto (ad esempio **<font color="red">**, 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 contente 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, 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 pocanzi diventerebbe qualcosa di questo genere:

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

la colorazione del testo verrebbe affidata alla classe "formattaTitoli", descritta altrove del documento, o su 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 **<font>**): 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. Per questo vi rimandiamo all'apposita guida ai CSS di HTML.it, che se vorrete potrete consultare dopo aver letto la guida all'HTML.

Impostare il colore di sfondo

Incominciamo col 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.

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.

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 "proprietari", 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).

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, risulta 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'inzio, i tag sono infatti dei marcatori che ci permettono di mantenere ordine nella pagina e ottenere il layout che desideriamo.

Vediamo i principali tag-contenitori da utilizzare per "racchiudere" il testo.

I titoli: h1, h2, ..., h6

I tag h1, h2 ... h6

```
<h1>titolo 1 </h1>
<h2>titolo 2 </h2>
<h3>titolo 3 </h3>
<h4>titolo 4 </h4>
<h5>titolo 5 </h5>
<h6>titolo 6 </h6>
```

La "h" sta per "heading", cioè **titolo** e le grandezze previste sono sei. Dall'<h1>, che è il più importante, si va via via degradando fino all' <h6>. Il tag <hx> (sia esso h1 o h6) risulta formattato in grassetto e lascia una riga vuota prima e dopo di sé.

Il paragrafo <p>

Il paragrafo è l'unità di base entro cui suddividere un testo. Il tag <p> lascia una riga vuota prima della sua apertura e dopo la sua chiusura.

Esempio: due paragrafi

```
<p>paragrafo 1</p>
<p>paragrafo 2</p>
```

Il

Il blocco di testo va a capo, ma - a differenza del paragrafo - non lascia spazi prima e dopo la sua apertura.

Esempio: due

```
<div>Blocco di testo 1</div>
<div>Blocco di testo 2</div>
```

Lo

Lo è un contenitore generico che può essere annidato (ad esempio) all'interno dei . Si tratta di un **elemento inline**, che cioè non va a capo e continua sulla stessa linea del tag che lo include.

Esempio: due

```
<span>contenitore 1</span><span>contenitore 2</span>
```

Lo è un elemento molto utilizzato soprattutto insieme ai fogli di stile, ad esempio per definire delle aree di testo particolari. Se non viene associato ad uno stile risulta praticamente invisibile.

Montare gli elementi insieme

Le caratteristiche più evidenti di , e sono quindi:

- ** ** lascia spazio prima e dopo la propria chiusura
- ** ** 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 i tag heading (<h1>, …, </h6>) è da notare che la grandezza del carattere varia a seconda delle impostazioni che l'utente ha sul proprio browser.

Con Internet Explorer, ad esempio, basta andare in Visualizza > Carattere per vedere il titolo crescere o decrescere.

Allineare il testo

I "tag-contenitori" che abbiamo appena visto (e molti altri) 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 (http://css.html.it/guide/lezione/31/gestione-del-testo-proprieta-di-base/).

In ogni caso, vediamo cosa ci è concesso fare per l'allineamento con HTML 4: consideriamo il testo di un paragrafo:

Allineamento	Sintassi	Visualizzazione codice HTML
Testo allineato a sinistra	testo	Nel mezzo del cammin di nostra vita mi ritrovai per una selva oscura ché la diritta via era smarrita
Testo allineato a destra	testo	Nel mezzo del cammin di nostra vita mi ritrovai per una selva oscura ché la diritta via era smarrita
Testo giustificato	testo	Nel mezzo del cammin di nostra vita mi ritrovai per una selva oscura ché la diritta via era smarrita

Andare a capo

Nell'andare a capo si può commettere l'errore, per fortuna sempre meno diffuso, di lasciare paragrafi vuoti o aperti. Ad esempio:

```
<p>
<p>
<p>
```

Quando per andare a capo all'interno di un paragrafo possiamo utilizzare il tag
 ("break", cioè "interruzione di riga").

Se per andare a capo è sufficiente un <br>, per saltare una riga ne occorrono due:

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

Un altro valido tag per suddividere la pagina in più parti è il tag **<hr />** ("horizontal rule"), che serve per tracciare una linea orizzontale. Eccone un esempio:

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

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" />

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

Gli stili fisici

I principali stili fisici sono:

Codice HTML	Visualizzazione	Descrizione
testo in grassetto **Esempio:** Questo testo è in grassetto	Questo **testo** è in grassetto	Formatta il testo in grassetto.
testo in corsivo **Esempio:** Questo testo è in corsivo	Questo *testo* è in corsivo	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. Meglio limitarsi a poche parole.

<pre>testo preformattato</pre>
 	
Esempio:
 	
<pre>
 	
PHP_FUNCTION
{
  zval **parameters;
  zval *value;
  char* str;
 	
</pre>
	PHP_FUNCTION
	{
	  zval **parameters;
	  zval *value;
	  char* str;
		Il motore di rendering del browser restituisce il testo così come è stato inserito nel file html dall'autore stesso (preformattato quindi), senza riformattarlo.
		
		È un tag che si usa soprattutto nella rappresentazione di codice di programmazione.

<u>testo sottolineato</u> Questo <u>testo</u> è sottolineato	Questo <u>testo</u> è sottolineato	Sottolinea il testo presente nel tag. Nel web le sottolineature del testo sono da evitare, per non confondere il lettore con i link.

Esempio: Questo <u>testo</u> è sottolineato		
<strike>testo barrato</strike> **Esempio:** Questo <strike>testo</strike> è barrato	Questo ~~testo~~ è barrato	Con il testo barrato, vengono indicate (ad esempio) le correzioni.
^{testo in apice} **Esempio:** E=mc²	$E=mc^2$	"Superscript": indica al browser di portare il testo al di sopra della linea di scrittura. Utile per formule matematiche (ad esempio le potenze)
_{testo in pedice} **Esempio:** H₂O	H_2O	"Subscript": indica al browser di portare il testo al di sotto della linea di scrittura (utile ad esempio per i simboli chimici)

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

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 eseguito i principali stili logici, per completezza, ma non sarà necessario ricordarseli.

Codice HTML	Visualizzazione	Descrizione
<abbr>abbreviazione</<abbr> **Esempio:** <abbr>C/A</abbr> HTML.it	C/A HTML.it	Indica un abbreviazione. Nessun rendering del testo particolare.
<acronym>acronimo</acronym> **Esempio:** <acronym>HTML</acronym>	HTML	Indica un acronimo. Nessun rendering del testo particolare.
<address>indirizzo</address> **Esempio:** <address>HTML.it - via dei Castani 183/185 – 00172 Roma</address>	*HTML.it - via dei Castani 183/185 - 00172 Roma*	Serve per indicare gli indirizzi: siano essi e-mail, o indirizzi fisici. Il testo viene visualizzato in corsivo.
<blockquote>blocco di citazione</blockquote> **Esempio:** <blockquote> Nel mezzo del cammin di	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.

nostra vita mi ritrovai per una selva oscura ché la diritta via era smarrita </blockquote>		
<cite>citazione</cite> **Esempio:** <cite>Nel mezzo del cammin di nostra vita mi ritrovai per una selva oscura ché la diritta via era smarrita</cite>	*Nel mezzo del cammin di nostra vita mi ritrovai per una selva oscura ché la diritta via era smarrita*	Per citazioni brevi: il testo è visualizzato in corsivo.
<code>codice</code> **Esempio:** <code>if (document.all) alert ("ciao");</code>	if (document.all) alert ("ciao");	Indica un blocco di codice in linguaggio di programmazione. Nessun rendering del testo particolare.
<dfn>definizione</dfn> **Esempio:** <dfn>L'HTML è un linguaggio di contrassegno</dfn>	*L'HTML è un linguaggio di contrassegno*	Indica una definizione: il testo è visualizzato in corsivo.
enfasi **Esempio:** Ti ho detto questo!	Ti ho detto *questo!*	Serve per porre l'enfasi su un'espressione: il testo è visualizzato in corsivo.
<kbd>keyboard</kbd> **Esempio:** <kbd>digitazione da tastiera</kbd>	digitazione da tastiera	Indica una digitazione da tastiera: il testo viene visualizzato a spaziatura fissa.
<q>citazione all'interno della frase</q> **Esempio:** Come diceva Don Abbondio: <q>***&quot***;Il coraggio, uno non se lo può dare***&quot***;</q>.	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.
<samp>esempio</samp> **Esempio:** <samp>ecco un esempio di "samp"</samp>.	ecco un esempio di "samp"	Indica un esempio. Il testo viene visualizzato a spaziatura fissa.
rafforzamento **Esempio:** Ecco un testo rafforzato	Ecco un **testo rafforzato**	Evidenzia una parola. Il testo viene reso in grassetto
<var>variabile</var> **Esempio:** Inseriamo i dati nella variabile temporanea <var>temp</var> …	Inseriamo i dati nella variabile temporanea *temp* ...	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.

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 mirate 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.

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" (si veda l'articolo «I font e la tipografia del testo» in questo sito).

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

testo in Arial	testo in Arial
testo in Verdana	testo in Verdana
testo in Geneva	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**.

Se ad esempio scarichiamo dal nostro archivio preferito di font il carattere «Hackers» e lo inseriamo nella cartella dei caratteri, saremo poi in grado di visualizzare sul nostro computer il testo in Hackers.

Ma quando metteremo il nostro sito nel web gli utenti visualizzeranno un semplicissimo Times. Come nell'esempio sotto indicato:

testo in hackers	**testo in hackers**

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 nostra 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 si utilizzare il carattere predefinito (il famigerato "Times")

Vediamo alcuni esempi di famiglie "sicure" di caratteri:

 Verdana e caratteri simili	Verdana e caratteri simili
Arial e caratteri simili	Arial e caratteri simili
Times e caratteri simili	Times e caratteri simili
Curier e caratteri simili	Curier e caratteri simili
 Georgia e caratteri simili 	Georgia e caratteri simili
 Geneva e caratteri simili	Geneva e caratteri simili

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").

Scegliere il colore del testo

Una volta scelto il carattere con cui scrivere il nostro testo possiamo scegliere il colore, ecco il codice:

Codice

```
<font color="blue">testo blu</font>
<br/>
ovvero:
<br/>
<font color="#0000FF">testo blu</font>
```

Effetto

testo blu
ovvero:
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>):

Codice

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

testo blu in Verdana

</font>
```

Effetto

testo blu in Verdana

Una volta scelto il colore possiamo sempre decidere di cambiarlo:

Codice

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

Effetto

testo blu in Verdana
testo rosso
o meglio ancora:
testo blu in Verdana
testo rosso

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

Le dimensioni del testo

Le dimensioni del testo si attribuisco mediante **l'attributo 'size'** del tag <font>. Ci sono due modi per dare attribuire le dimensioni al testo tramite il tag <font>:

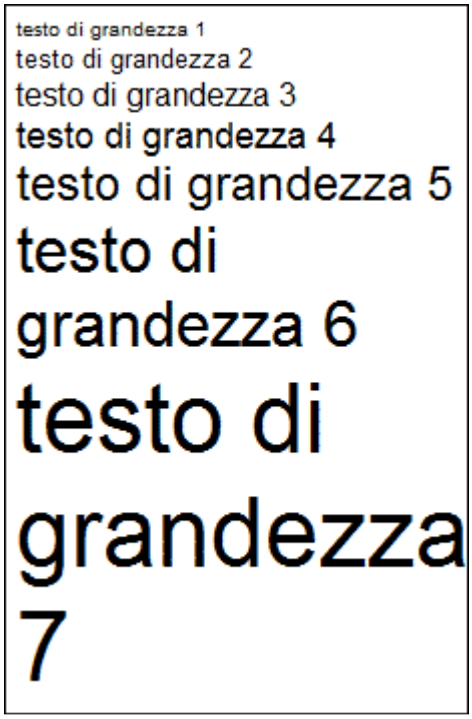
- **valori interi da 1 a 7**
- **valori relativi** alla dimensione di base del tag font (di default "3")

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

Codice

```
<font size="1">testo di grandezza 1</font>
<br />
<font size="2">testo di grandezza 2</font>
<br />
<font size="3">testo di grandezza 3</font>
<br />
<font size="4">testo di grandezza 4</font>
<br />
<font size="5">testo di grandezza 5</font>
<br />
<font size="6">testo di grandezza 6</font>
<br />
<font size="7">testo di grandezza 7</font>
```

Effetto



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

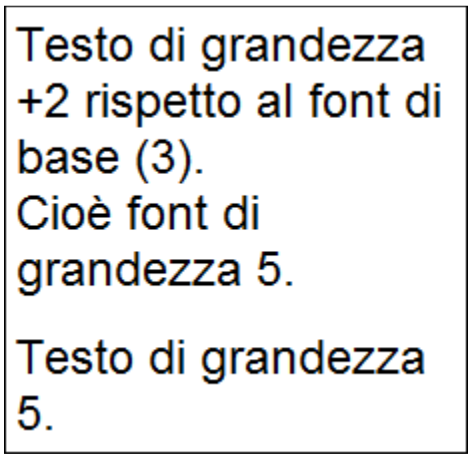
Abbiamo detto che la grandezza del font di base di default nel browser è 3.

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:

Codice

```
<font size="+2">
Testo di grandezza +2 rispetto al font di base (3).<br />
Cioè font di grandezza 5.
</font>
<br /><br />
<font size="5">
Testo di grandezza 5.
</font>
```

Effetto



Come si può vedere le due sintassi sono equivalenti.

La grandezza del font di base può anche essere 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:

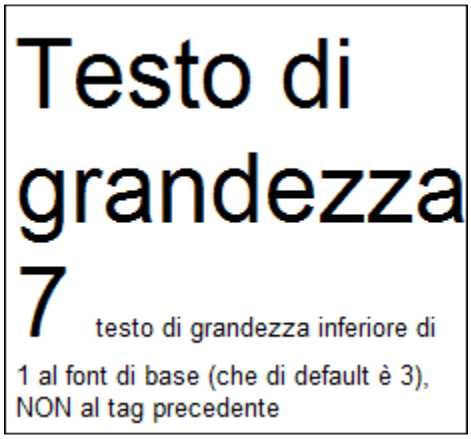
Codice

```
<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>
```

Effetto



In questo caso, ad esempio, sarebbe sbagliato aspettarsi una dimensione 6 dalla seconda istanza di <font>, perché la dimensione relativa fa sempre riferimento al <basefont>:

Anche se non è corretto farlo, Internet Explorer consente di utilizzare il tag <basefont> per impostare in una sola volta il tipo di carattere del testo e il suo colore.

Tuttavia questo tipo di trucco non funziona correttamente né con Mozilla (e quindi neanche con Netscape 6 o superiore, dal momento che eredita il motore di rendering di Mozilla), né con Opera.

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.

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 **<ol>** ("ordered list") e gli elementi sono individuati dal tag **<li>**("list item"):

Codice	Resa
Testo che precede la lista - primo elemento - secondo elemento - terzo elemento Testo che segue la lista	Testo che precede la lista 1. primo elemento 2. secondo elemento 3. terzo elemento Testo che segue la lista

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, néprima, 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	Descrizione	Codice	Resa
type="1" (è così di default)	**numeri arabi**	<ol type="1"> - primo - secondo - terzo 	1. primo 2. secondo 3. terzo
type="a"	**alfabeto minuscolo**	<ol type="a"> - primo - secondo - terzo 	a. primo b. secondo c. terzo
type="A"	**alfabeto maiuscolo**	<ol type="A"> - primo - secondo - terzo 	A. primo B. secondo C. terzo
type="i"	**numeri romani minuscoli**	<ol type="i"> - primo - secondo - terzo 	i. primo ii. secondo iii. terzo
type="I"	**numeri romani maiuscoli**	<ol type="I"> - primo - secondo - terzo 	I. primo II. secondo III. terzo

Gli elenchi non ordinati

Gli elenchi non ordinati sono individuati dal tag **<ul>** ("unordered list"), e gli elementi dell'elenco sono contraddistinti anch'essi dal tag <li> (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	Descrizione	Codice	Resa
type="disc" (è così di default)	visualizza un **"pallino" pieno**. È la visualizzazione di default	<ul type="disc"> - primo - secondo - terzo 	• primo • secondo • terzo
type="circle"	visualizza un **cerchio vuoto** al proprio interno	<ul type="circle"> - primo - secondo - terzo 	o primo o secondo o terzo
type="square"	Visualizza un **quadrato pieno** al proprio interno	<ul type="square"> primo secondo terzo 	▪ primo ▪ secondo ▪ terzo

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

Codice

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

Resa

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

Liste di definizione

Gli liste di definizione 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:

Tag	Descrizione
<dt>	**definition term**: indica il termine da definire. A differenza dell'elemento ** ** in questo caso non c'è rientro
<dd>	**definition description**: è la definizione vera e propria del termine. In genere questo elemento è reso con un rientro

Vediamo un esempio:

Codice

```
<p>Ecco i principali tag per delimitare il
testo:</p>

<dl>
  <dt>&lt;p&gt;</dt>
  <dd>individua l'apertura di un
     nuovo paragrafo</dd>

  <dt>&lt;div&gt;</dt>
  <dd>individua l'apertura di un
     nuovo blocco di testo</dd>

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

Resa

Ecco i principali tag per delimitare il testo:

<p>
    individua l'apertura di un nuovo paragrafo

    individua l'apertura di un nuovo blocco di testo

    individua l'apertura di un elemento inline, cui attribuire una formattazione atraverso gli stili
ci sono poi altri tag che...

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.

I link e l'ipertestualità

Una delle caratteristiche che ha fatto la fortuna del web è l'essere costituito non da **testi** ma da **ipertesti** (un'altra delle caratteristiche che hanno fatto grande il web è senz'altro la possibilità di interagire, ma questo è un altro discorso).

I link sono "il ponte" che consente di passare da un testo all'altro. In quanto tali, i link sono formati da due componenti:

il contenuto che "nasconde" il collegamento (non importa se si tratta di testo o di immagine)	È la parte visibile del link, e proprio per questo l'utente deve essere sempre in grado di capire quali sono i collegamenti da cliccare all'interno della pagina
la risorsa verso cui il collegamento punta	Si tratta di un'altra pagina (sullo stesso server o su un server diverso), oppure è un collegamento interno a un punto della pagina stessa

Di solito per spiegare che cosa sono i link si utilizza la metafora dell'ancora con "la testa" all'interno del documento stesso, e la "coda" in un altro documento (o all'interno di un altro punto del documento stesso).

Link che puntano ad altri documenti

Ecco la sintassi per creare un link con riferimento a un sito web:

Le risorse per webmaster sono su ****HTML.IT****.

Che dà come risultato: 'Le risorse per webmaster sono su HTML.IT (http://www.html.it/)'.

Come si può intuire la testa della nostra àncora è il testo "HTML.IT", mentre la coda, cioè la destinazione (specificata dall'attributo **href**) è il sito web verso cui il link punta, cioè http://www.html.it.

È indifferente che la destinazione dell'ancora sia una pagina HTML di un sito, un'immagine, un file pdf , un file zip, o un file exe: il meccanismo del link funziona allo stesso modo indipendentemente dal tipo di risorsa; poi il browser si comporterà in modo differente a seconda della risorsa. Ad esempio:

Immagine .gif, .jpg, .png	Viene visualizzata nel browser
Documento .html, .pdf, .doc	La pagina è visualizzata nel browser. Nel caso dei documenti **.doc** e **.pdf** l'utente deve avere installato sul proprio pc l'apposito plugin (nella maggior parte dei casi è sufficiente che abbia installato rispettivamente Microsoft Word e Adobe Acrobat Reader). Se non è installato il plugin il sistema chiederà all'utente se salvare il file.
File .zip, file .exe	Viene chiesto all'utente di scaricare il file **NOTA bene: per motivi di sicurezza non è possibile eseguire un file ".exe" direttamente dal web; l'utente dovrà sempre prima scaricarlo sul proprio PC.**

Potete anche specificare un indirizzo e-mail. In questo caso si aprirà direttamente il client di posta dell'utente con l'indirizzo e-mail pre-impostato. La sintassi è la seguente:

```
<a href="mailto:tuaMail@nomeTuoSito.it">
 Mandami una e-mail
</a>.
```

Che dà come risultato: Mandami una e-mail (mailto:tuamail@nomeTuoSito.it).

I percorsi assoluti e relativi

Percorsi assoluti

Fino a quando ci troviamo nella condizione di creare un sito web di dimensioni ridotte (poche pagine) non avremo problemi di complessità, e possiamo anche ipotizzare di lasciare tutti i nostri file in una medesima cartella. È evidente però che – man mano che il nostro sito web cresce – avremo bisogno di un maggior ordine.

Si presenterà allora l'esigenza di inserire le immagini del sito in una cartelle diverse (in modo da averle tutte nella medesima locazione), e magari sarà opportuno dividere il sito in varie sezioni, in modo da avere tutti i documenti dello stesso tipo all'interno di un contesto omogeneo.

I siti web sono dunque organizzati in strutture ordinate: non a caso si parla di **albero di un sito**, per indicare la visualizzazione della struttura alla base del sito.

Poiché l'organizzazione di un sito in directory e sottodirectory è una cosa normalissima, dobbiamo imparare a muoverci tra i vari file che costituiscono il sito stesso, in modo da essere in grado di creare collegamenti verso i documenti più reconditi, destreggiandoci tra le strutture più ramificate.

Per farlo esistono due tecniche:

- **indicare un percorso assoluto**
- **indicare un percorso relativo**

Nel caso in cui il documento a cui vogliamo puntare si trovi in una particolare directory del sito di destinazione, con i percorsi assoluti non abbiamo che da indicare il percorso per esteso.

Se esaminiamo:

Leggi le risorse sui fogli di stile

Possiamo vedere chiaramente che il link indica un percorso assoluto e fa riferimento ad una particolare directory. Nella fattispecie:

http://	Indica al browser di utilizzare il protocollo per navigare nel web (l'http)
www.html.it/	Indica di fare riferimento al sito www.html.it
css/	Indica che la risorsa indicata si trova all'interno della cartella "css"
index.html	Indica che il file da collegare è quello chiamato "index.html"

Insomma, per creare un collegamento assoluto è sufficiente fare riferimento all'url che normalmente vedete scritto nella barra degli indirizzi. I percorsi assoluti si usano per lo più, quando si ha la necessità di fare riferimento a risorse situate nei siti di terze persone.

Percorsi relativi

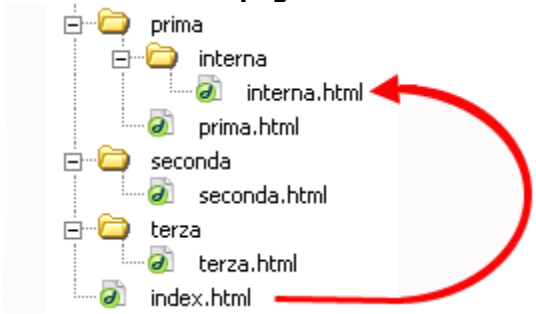
Spesso vi troverete tuttavia a fare riferimento a documenti situati nel vostro stesso sito, e – se state sviluppando il sito sul vostro computer di casa (cioè "in locale") – magari non avete ancora un indirizzo web, e non sapete di conseguenza come impostare i percorsi. È utile allora capire come funzionano i percorsi relativi.

I percorsi relativi fanno riferimento alla posizione degli altri file rispetto al documento in cui ci si trova in quel momento.
Per linkare due pagine che si trovano all'interno della stessa directory è sufficiente scrivere:

collegamento alla pagina da linkare nella stessa directory della pagina presente

Poniamo ora di trovarci in una situazione di questo genere:

Figura 1. Riferimento a pagina di una sottodirectory



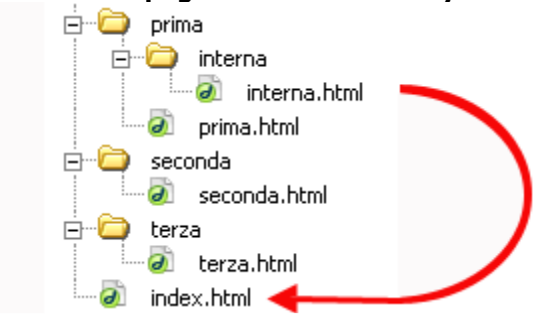
Dalla pagina "index.html" vogliamo cioè far riferimento al file "interna.html", che si trova all'interno della directory "interna", che a sua volta si trova all'interno della directory "prima".

La sintassi è la seguente:

Visita la pagina interna

Vediamo adesso l'esempio opposto: dalla pagina interna vogliamo far riferimento a una pagina ("index.html") che si trova più in alto di due livelli:

Figura 2. Riferimento a pagina in una directory di livello superiore



La sintassi è la seguente:

Visita la pagina interna

Come si vede, con i percorsi relativi valgono le seguenti regole generali:

Per far riferimento a un file che si trovi all'interno della stessa directory basta linkare il nome del file	collegamento alla pagina
Per far riferimento a un file contenuto in una cartella di livello inferiore alla posizione corrente, basta nominare la cartella seguita dallo "slash", e poi il nome del file. Secondo la formula: **cartella/nomeFile.html**	Visita la pagina interna
Per tornare su di un livello, è sufficiente utilizzare la notazione: **../nomeFile.html**	Visita la pagina interna

Grazie a questi accorgimenti potete agevolmente navigare all'interno delle directory del vostro sito: se ce ne fosse bisogno potrete per esempio tornare su di un livello rispetto alla posizione del file, scegliere un'altra cartella, e poi scegliere un altro file:

../altraCartella/nuovoFile.html

Per approfondimenti potete consultare la pagina d'esempio (http://html.it/guide/esempi/guida_html/esempi/link/index.html).

Approfondimenti

A volte potrete incontrare la notazione:

Leggi le risorse sui fogli di stile

Se il vostro sito è all'interno di un server Unix (ma la sintassi funziona anche in sistemi Windows, basta che non siano in locale), questa notazione non deve stupirvi: il carattere '/' indica la directory principale del sito, altrimenti detta **"root"**. Dunque <a href="/css/index.html"> è un altro modo di esprimere i percorsi assoluti all'interno del proprio sito.

Un'altra cosa importante da sapere è che quando metterete il vostro sito all'interno dello spazio web, l'indicazione della index all'interno di una directory è facoltativa. Al posto di questo:

http://www.html.it/css/index.html

è sufficiente indicare la directory:

http://www.html.it/css/

Verificate solo con il vostro gestore dello spazio web (cioè "hosting"), se le pagine index della directory devono avere forma **index.html, index.htm, index.asp, index.php, home.asp**, o altro.

Consigli per i nomi dei file

Quando mettere nel web il vostro sito internet, vi accorgerete che esistono due famiglie di sistemi operativi: **Windows** e **Unix**. Questi due sistemi operativi utilizzano differenti modi per gestire i file, dunque alcuni accorgimenti sono necessari:

- è consigliabile non lasciare spazi vuoti nei nomi dei file (gli spazi vuoti non sempre vengono interpretati correttamente), meglio ovviare a questa necessità con un "trattino basso" (cioè "_"). Ad esempio: **mio_file.html**
- maiuscole e minuscole possono fare la differenza (in ambiente Unix spesso la fanno), quindi controllate il modo in cui avete scritto i file

Inoltre quando create un collegamento state attenti a non avere una notazione simile a questa:

<a href="file:///C|percorsonomeFile.html">testo</A>

significa che state facendo un riferimento (assoluto) al vostro stesso computer: chiaro che quando metterete i file nel vostro spazio web, le cose non funzioneranno più.

Colorare i link

Abbiamo già visto (http://xhtml.html.it/guide/lezione/1668/impostare-il-colore-del-testo-e-dei-link-per-tutta/) come colorare i link in tutta la pagina. Possiamo però aver bisogno di colorare alcuni link della pagina in modo diverso. Per farlo è sufficiente annidare il tag <font> all'interno del link:

```
<a href="http://www.html.it/" target="_blank" ><font color="red" size="2" face="Verdana, Arial, Helvetica, sans-serif">Torna all'home page di HTML.it</font></a>
```

cioè:

Torna all'home page di HTML.it (http://www.html.it/)

ovviamente il modo giusto per colorare i link non è quello di utilizzare il tag font, ma quello di utilizzare i fogli di stile, come spiegato nella Guida CSS (http://css.html.it/guide/lezione/29/gestione-del-colore/).

Inserire le immagini

Finora abbiamo visto come inserire e formattare il testo all'interno delle nostre pagine Web. Naturalmente possiamo inserire anche delle immagini: diagrammi e grafici, fotografie, e in genere immagini create con un programma di elaborazione grafica (come GIMP (http://grafica.html.it/guide/leggi/15/guida-gimp/) o Photoshop (http://grafica.html.it/guide/leggi/189/guida-photoshop-cs5/)).

I formati di immagine per il Web

I formati ammessi nel Web sono sostanzialmente tre:

Formato	Descrizione
GIF	(**Graphic Interchange Format**): le immagini GIF hanno una tavolozza che non supera i **256 colori**, per questo vengono utilizzate spesso per grafici o icone. **È possibile ottimizzare il peso** questo tipo di file riducendo ancora il numero di colori. Oltre a questo GIF ci consente di **impostare trasparenze** nelle immagini e di **creare piccoli banner**, questo formato consente infatti di rappresentare anche semplici animazioni (GIF animate)
JPG	(o JPEG) È l'acronimo del gruppo di ricerca che ha ideato questo formato (il **Joint Photographic Experts Group**), idoneo per le immagini di qualità fotografica
PNG	(**Portable Network Graphic**). Il PNG è un tipo di immagine introdotto più recentemente, elaborato dal W3C (http://www.w3c.org) per risolvere i problemi di copyright del formato GIF (che è invece proprietario); tuttavia oggi il PNG è letto oramai da tutti i browser e offre alcune caratteristiche che gli altri formati non hanno (come il supporto al canale alfa, caratteristica questa non ancora perfettamente supportata da ogni browser). PNG permette sia di rappresentare immagini di qualità fotografica (PNG24), sia di ottimizzare colori in modo simile a GIF (PNG8 - 256 colori)

Inutile provare dunque a inserire un file ".psd" (formato nativo di Photoshop) all'interno della vostra pagina HTML: con grande probabilità il browser non caricherà il file (dovete infatti prima convertire il file in uno dei formati sopra-indicati).

È importante ricordare che il codice HTML fornisce delle indicazioni al browser su come visualizzare il testo e le immagini - ed eventualmente i video e i suoni - all'interno della pagina: il testo (come abbiamo visto) è scritto direttamente nel file HTML, le immagini invece sono caricate insieme alla pagina.

Attenzione dunque a non inserire immagini troppo pesanti (ricordatevi di ottimizzare sempre i file); bisogna evitare inoltre di sovraccaricare la pagina con troppe immagini. Un peso eccessivo rende le pagine lente da caricare e questo può diventare un problema sia per gli utenti, sia per i motori di ricerca.

Per ottenere un sito web dalla grafica accattivante, spesso è sufficiente giocare con i colori dello sfondo e delle scritte.

Il codice

La sintassi per inserire una immagine è:

Esaminiamo il significato delle keyword principali:

Keyword	Descrizione
img	Abbreviazione di "image" (immagine), è **il nome del tag** per inserire le immagini
src	Sta per "source" (origine), è il percorso (URL) in cui il browser troverà il file da mostrare
alt	È il "testo alternativo", ovvero il testo che appare se, per qualche motivo, il browser non riesce a mostrare l'immagine. Possiamo anche omettere questo attributo, ma risulta utile per l'accessibilità e per i motori di ricerca

Il tag <img /> è un tag senza un contenuto, per questo non ha un elemento </img> di chiusura: lo chiudiamo utilizzando lo slash ("/") prima della parentesi angolare.

Ecco ad esempio come inserire il logo di HTML.it in una pagina dallo sfondo blu (si presuppone che il logo si trovi nella stessa cartella del file HTML):

Codice	Risultato
	

Resta valido il discorso sui percorsi relativi ed assoluti visto in precedenza. Avremo ad esempio:

Dal momento che il browser normalmente non sa quali siano le dimensioni dell'immagine, finché questa non sia caricata completamente, è un'ottima abitudine quella di indicare già nel codice la larghezza (**width**) e l'altezza (**height**) dell'immagine: in questo modo si evita di vedere la pagina costruirsi man mano che viene caricata, poiché stiamo dando al browser un'idea dell'ingombro. Ad esempio:

L'attributo **alt** è utile per specificare il **testo alternativo** (**alternative text**), fintanto che l'immagine non viene caricata o nel caso in cui non lo sia affatto:

Codice	Risultato
	

L'attributo **alt** è di estrema utilità per rendere il **sito accessibile** a tutti gli utenti: i disabili che non sono in grado di vedere nitidamente le immagini sullo schermo potrebbero avere delle difficoltà, nel caso in cui l'attributo alt non sia specificato.

Gli ipo-vedenti e i non-vedenti sono infatti in grado di comprendere il contenuto delle immagini grazie a dei software appositi (gli **screen reader**) che "leggono" lo schermo tramite un programma di sintesi vocale. Non specificare il testo alternativo significa rendere impossibile la navigazione.

Nel caso in cui la spiegazione dell'immagine sia particolarmente lunga, è possibile espandere la descrizione sintetica - fornita tramite l'attributo "alt" - grazie ad un altro attributo: si tratta di **longdesc**

(**long description**), che permette di specificare un file con una spiegazione estesa dell'immagine. Ecco la sintassi:

Nell'esempio allegato (http://html.it/guide/esempi/guida_html/esempi/esLongDesc.htm) è possibile visualizzare il codice di una pagina con la descrizione estesa dell'immagine (http://html.it/guide/esempi/guida_html/esempi/descrizione.htm). Nel caso in cui si utilizzi questo attributo è anche buona norma utilizzare un link esplicito alla pagina della descrizione.

longdesc dovrebbe essere utilizzato soprattutto nel caso in cui si usino delle immagini mappate (argomento che analizzeremo in seguito), in modo da fornirne una spiegazione esauriente in ogni contesto.

In realtà l'attributo **alt** non serve, come molti credono, a visualizzare un etichetta esplicativa dell'immagine nel caso in cui il cursore del mouse si soffermi sopra essa: questo semmai è un effetto collaterale che si verifica con Internet Explorer. L'attributo corretto per far visualizzare un testo che commenti l'immagine è infatti **title**:

Codice	Risultato
	

È inoltre possibile specificare la grandezza (in pixel) del bordo attorno all'immagine:

Codice	Risultato
	

Immagini con i link

Si noti che **i link per default lasciano sempre un bordo** di un pixel attorno all'immagine (il colore sarà quello espresso nel body dall'attributo **link**, oppure quello default - quindi blu - se non specificato altrimenti):

Codice	Risultato
	

Dunque, nel caso dei link se non si desidera avere i bordi, sarà necessario impostarli a **"0"**:

Codice	Risultato
	

Disporre le immagini in un contesto

Se inserita in un testo, normalmente una immagine è inserita nel testo. Così:

Esempio di immagine nel testo

HTML.it è il primo sito italiano sul web publishing <img src="logo.gif" width="224" height="69" border="0"> con centinaia di esempi e guide esplicative

HTML.it è il primo sito italiano sul web publishing HTML.it è il primo sito italiano sul web publishing HTML.it è il primo sito italiano sul web publishing HTML.it è il primo sito italiano sul web publishing

con centinaia di esempi e guide esplicative HTML.it è il primo sito italiano sul web publishing HTML.it è il primo sito italiano sul web publishing

Abbiamo tuttavia la possibilità di allineare l'mmagine e il testo come preferiamo, utilizzando l'attributo **align**. Vediamo di seguito come vengono visualizzati **align="left"** e **align="right"**:

Esempio di immagine allineata a sinistra

<img src="logo.gif" width="224"height="69"border="1"**align="left"**>
HTML.it èil primo sito italianosul web publishing, con centinaia di esempi e guide esplicative

HTML.it è il primo sito italiano sul web publishing, con centinaia di esempi e guide esplicative HTML.it è il primo sito italiano sul web publishing HTML.it è il primo sito italiano sul web publishing HTML.it è il primo sito italiano sul web publishing HTML.it è il primo sito italiano sul web publishing HTML.it è il primo sito italiano sul web publishing HTML.it è il primo sito italiano sul web publishing

Esempio con immagine allineata a destra

 <img src="logo.gif" width="224" height="69" border="1" **align="right"**>
HTML.it è il primo sito italiano sul web publishing, con centinaia di esempi e guide esplicative

HTML.it è il primo sito italiano sul web publishing, con centinaiadi esempi e guide esplicative HTML.it è il primo sito italiano sul web publishing HTML.it è il primo sito italiano sul web publishing HTML.it è il primo sito italiano sul web publishing HTML.it è il primo sito italiano sul web publishing HTML.it è il primo sito italiano sul web publishing HTML.it è il primo sito italiano sul web publishing HTML.it è il primo sito italiano sul web publishing

Altri valori possibili sono:

Valore di align	Visualizzazione
bottom	allinea la prima riga di testo sulla sinistra nella parte bassa dell'immagine (è così di default).
middle	allinea la prima riga di testo sulla sinistra al centro dell'immagine.
top	allinea la prima riga di testo sulla sinistra nel lato superiore dell'immagine.

Da notare che, mentre **align="left"** e **align="right"**, sono utili per spostare l'immagine a sinistra o a destra, gli altri valori servono piuttosto per disporre le posizioni verticali di testo e immagini.

Infine con **hspace** (**horizontal space**, cioè "spazio orizzontale") e vspace (**vertical space**, cioè "spazio veritcale") possiamo impostare lo spazio (in pixel) che deve essere lasciata tra l'immagine e cioè che la circonda.

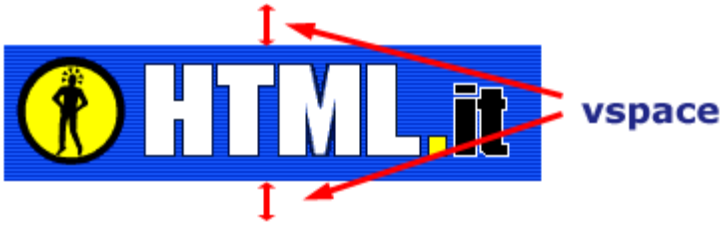
Nel caso di hspace impostiamo uno spazio orizzontale da ambo i lati, come in questo caso:

<img src="logo.gif" width="224" height="69" **hspace="20"**>

Nel caso di vspace lo spazio è verticale, ma sempre da ambo i lati:

<img src="logo.gif" width="224" height="69" **vspace="20"**>

cioè:



Un attributo importante - di cui non vedrete nessun effetto "pratico" di formattazione, ma che vi servirà ad esempio per creare un effetto di "scambio immagine" grazie a JavaScript - è quello che permette di specificare il nome dell'immagine:

<img src="miaImmagine.gif" ***name="logo"***>

Approfondimenti

Ovviamente sarebbe meglio impostare lo spessore e il colore dei bordi, gli spazi e la disposizione del testo attorno alle immagini attraverso i fogli di stile.

Tabella: struttura di base

Le tabelle sono una delle parti più importanti di tutto il codice HTML: nate sin dagli inizi del Web per impaginare dati aggregati, si sono poi trasformate in uno strumento indispensabile per gestire i layout grafici.

Il loro ampio utilizzo all'interno dei documenti ha fatto sì che – nel passaggio dall'HTML 3.2 all'HTML 4 - le specifiche delle tabelle venissero estese con una serie di notazioni destinate a "far ordine" all'interno di un codice che rischiava di diventare troppo vasto.

Immaginiamo la nostra prima tabella come una griglia formata da righe e colonne. I tag necessari per creare una tabella sono:

<table> apre la tabella
<tr> **"table row"**: indica l'apertura di una riga
<td> **"table data"**: indica una cella all'interno di una riga

In questi nostri primi esempi presupponiamo che il numero delle celle all'interno di ciascuna riga sia costante: ogni riga avrà cioè lo stesso numero di celle. Ci sono dei metodi per variare il numero delle celle all'interno di una riga, ma li vedremo in seguito.

L'attributo **border** permette di specificare di quanti pixel deve essere il bordo delle tabelle. Ad esempio:

<table border="2">

Lo useremo in questi esempi, altrimenti non percepiremmo la struttura di quanto stiamo costruendo. Ecco un primo esempio di tabella:

```
<table border="1">
   <tr>
     <td>prima cella</td>
     <td>seconda cella</td>
  </tr>

  <tr>
    <td>terza cella</td>
    <td>quarta cella</td>
  </tr>
</table>
```

Che viene visualizzato così:

prima cella	seconda cella
terza cella	quarta cella

Possiamo specificare la larghezza e l'altezza delle tabelle tramite gli attributi **width** e **height** che possono essere riferiti a tutti e tre i tag (**<table>**, **<tr>**, **<td>**). Il valore di questi attributi può essere specificato con una larghezza fissa (in pixel: in questo caso basta indicare un numero intero), oppure in percentuale (il numero deve essere allora seguito dal simbolo "%"): in questo caso la tabella si adatta secondo lo spazio a disposizione.

```
<table width="300" height="200" border="1">
   <tr>
     <td>prima cella</td>
     <td>seconda cella</td>
  </tr>
```

```
  <tr>
    <td>terza cella</td>
    <td>quarta cella</td>
  </tr>
</table>
```

Che viene visualizzato così:

prima cella	seconda cella
terza cella	quarta cella

Oppure:

```
<table width="75%" border="1">
   <tr>
     <td width="25%">prima cella</td>
     <td width="75%">seconda cella</td>
  </tr>

  <tr>
    <td width="25%">terza cella</td>
    <td width="75%">quarta cella</td>
  </tr>
</table>
```

Che viene visualizzato così:

prima cella	seconda cella
terza cella	quarta cella

Di solito la larghezza e l'altezza globali della tabella sono espresse nel tag **<table>**, mentre la larghezza delle varie celle viene espressa nei **<td>** della prima riga. L'altezza in percentuale non sempre è visualizzata correttamente da tutti i browser.

Come detto inizialmente le tabelle vanno immaginate come delle griglie, tutto sommato abbastanza rigide: l'eventuale larghezza specificata nelle celle della prima riga avrà effetto dunque anche sulle celle delle righe sottostanti.

Viceversa non è possibile variare arbitrariamente le dimensioni delle celle: le misure specificate nelle righe sottostanti non avranno infatti effetto.

Le dimensioni espresse non devono tuttavia essere in contraddizione ma mano che si procede verso l'interno della tabella: in un caso simile infatti "vincerebbe" il valore specificato nel tag genitore.

Inoltre (come si evince dagli esempi) la visualizzazione dei layout con indicazioni non corrette è a discrezione del browser, quindi si rischia di ottenere risultati diversi da quelli voluti.

Raggruppare celle con rowspan e colspan

Finora abbiamo immaginato le tabelle come griglie rigide, in cui il numero delle colonne era dato come costante e non modificabile. I raggruppamenti di righe e colonne che abbiamo esaminato finora non hanno modificato minimamente questa struttura.

In realtà è possibile raggruppare le delle all'interno delle colonne in modo da avere ad esempio una riga da 2 colonne e un'altra da 3. Per ottenere questo risultato è necessario specificare che una cella deve occupare il posto di 2 (o più colonne). In questo caso si utilizza l'attributo **colspan** sul **<td>**, specificando come valore il numero di celle che devono essere occupate.

Ad esempio:

	<td colspan="2">	

Il cui codice corrispondente è:

```
<table width="430" border="1" bordercolor="#000000">
 <tr>
  <td width="30%"> </td>
  <td width="30%"> </td>
  <td width="30%"> </td>
 </tr>
 <tr>
  <td> </td>
  <td colspan="2" align="center" valign="middle">
   <b>&lt;td colspan=&quot;2&quot;&gt;</b>
  </td>
 </tr>
</table>
```

Tramite l'attributo **rowspan** (da riferirsi sempre a **<td>**) è invece possibile creare delle celle che occupino più di una riga. Ad esempio:

	<tdrowspan="2">	

il cui codice corrispondente è:

```
<table width="430" border="1" bordercolor="#000000">
 <tr>
  <td width="30%"> </td>
  <td width="30%" rowspan="2" align="center" valign="middle">
   <b>&lt;tdrowspan=&quot;2&quot;&gt;</b>
  </td>
  <td width="30%"> </td>
 </tr>
 <tr>
  <td> </td>
  <td> </td>
 </tr>
</table>
```

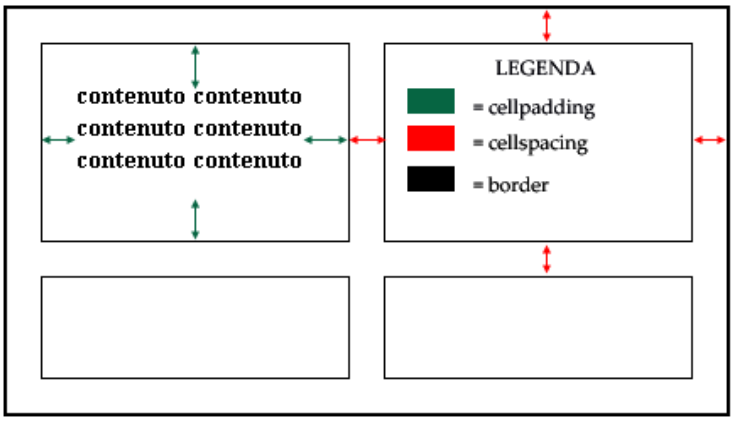
Attributi del tag table

Per quel che riguarda il tag **<table>**, i seguenti attributi che ci permettono di regolare le distanze tra i margini della tabella (o della cella) e il contenuto:

border	(che abbiamo già visto) specifica la larghezza dei bordi di una tabella (in pixel)
cellspacing	specifica la distanza (in pixel) tra una cella e l'altra, oppure tra una cella e il bordo. Di default è un pixel, dunque occorrerà sempre azzerarlo esplicitamente, quando non lo si desideri
cellpadding	indica la distanza tra il contenuto della cella e il bordo. Se il valore viene indicato con un numero intero, la distanza è espressa in pixel; il cellpadding tuttavia può anche essere espresso in percentuale. Di default la distanza è nulla

La dimensione indicata nel **cellpadding** e dal **cellspacing** - una volta specificata – ha effetto su tutti i lati della cella.

I rapporti tra gli attributi che abbiamo appena esaminato sono regolati come segue:



Con questa sintassi ad esempio si imposta una tabella con bordo di 1 pixel, senza spazio tra le celle e con il contenuto che è distanziato dai bordi della cella di 10 pixel:

<table width="75%" border="1" cellpadding="10" cellspacing="0">

Attributi di <table>, <tr>, <td>

I seguenti attributi invece hanno invece valore per tutti gli elementi della tabella (**<table>**, **<tr>**, **<td>**), li presenteremo quindi in un medesimo contesto.

Dimensioni

Abbiamo già esaminato gli attributi **width** e **height** che determinano la larghezza e l'altezza (in pixel o in percentuale) di tabelle, righe o celle.

Lo sfondo

Possiamo assegnare un colore di sfondo tramite l'attributo **bgcolor**, oppure un'immagine tramite **background**, come abbiamo già visto a proposito del tag <body>

Vediamo un esempio:

bgcolor	codice: <table width="75%" border="1" align="center" bgcolor="#00FF00"> <tr> <td width="50%" bgcolor="#FF0000"> prova </td> <td width="50%"> </td> </tr> </table> visualizzazione: prova

background	codice: <table width="75%" border="1" align="center" bgcolor="#00FF00"> <tr> <td width="50%" background="sfondo.gif"> prova </td> <td width="50%"> </td> </tr> </table> visualizzazione: prova

Come già nel **<body>** l'immagine di sfondo viene ripetuta, ed è possibile specificare entrambi gli attributi (**bgcolor** e **background**) all'interno dello stesso tag:

<td width="50%" bgcolor="#0000FF" background="tabelle/sfondo.gif">

L'allineamento

L'attributo **align**, se riferito al tag **<table>**, sposta la tabella rispettivamente a sinistra (**align="left"** – è così di default), a destra (**align="right"**), o al centro (**align="center"**) del documento. Es:

<table align="right">

Se riferito a **<tr>** o a **<td>** è invece il contenuto delle celle ad essere allineato a sinistra, al centro oppure a destra. Es:

<td align="right">
contenuto
</td>

Allo stesso modo **valign** è utile per l'allineamento verticale delle celle. I valori possibili sono **top** (alto), **middle** (in mezzo – è il valore di default), **bottom** (in basso), **baseline** (alla linea di base). Es:

<td height="100" valign="middle">
contenuto
</td>

Colori dei bordi

Per i bordi esistono gli attributi **bordercolor**, **bordercolorlight**, **bordercolordark**. Ad esempio:

<table border="2" bordercolor="blue" bordercolorlight="#00CCFF" bordercolordark="#000099">

Questi attributi - che consentono di creare degli effetti bellissimi - sono visualizzati correttamente soltanto da Internet Explorer, mentre con gli altri browser (Mozilla, Opera) verranno visualizzati in modo parziale se non scorretto.

In realtà il modo corretto per attribuire un colore al bordo è quello di usare i CSS.

Ci sono tuttavia delle soluzioni - utilizzate dagli sviluppatori sin dall'HTML 3 – che permettono di mostrare un filetto colorato attorno alle tabelle. La tecnica di solito è quella di lasciar trasparire il colore di sfondo attraverso lo spazio fra le celle. Vediamo un esempio:

```
<table width="450" bgcolor="#00CCFF" cellpadding="10" cellspacing="1">

  <tr bgcolor="FFFFFF">
    <td width="50%"><b>contenuto</b></td>
    <td width="50%"> </td>
  </tr>

</table>
```

che dà:

contenuto	

Grazie all'attributo si può far sì che il contenuto di una cella non vada a capo, a meno che non lo forziamo espressamente con un **
** (è un **"break**", cioè un' "interruzione"):

```
<table width="100" border="1">
 <tr>
  <td >
   Se non lo vogliamo non va a capo.<br>
   Qui va a capo.
  </td>
 </tr>
</table>
```

cioè:

Se non lo vogliamo non va a capo. Qui va a capo.

Approfondimenti

Da notare che quando una cella non viene riempita con un qualsiasi elemento non tutti i browser visualizzeranno i bordi allo stesso modo:

```
<table width="200" border="1">
 <tr>
  <td width="50%">
  </td>
  <td width="50%">contenuto
  </td>
 </tr>
</table>
```

cioè:

	contenuto

Dunque è opportuno riempire sempre le celle con qualcosa, sia pure un ** ** (è la notazione per indicare un **"non-breaking space"**, cioè uno "spazio che non va a capo"), o un **
. Attenzione che questi caratteri speciali prendono le dimensioni del tag **<font> all'interno di cui sono contenuti.

Con Netscape 4 per ottenere la visualizzazione desiderata è spesso necessario introdurre una **gif trasparente di 1 pixel x 1 pixel** (detta "**shim**") come sfondo della cella.

Ovviamente per ottenere il layout desiderato di bordi e tabelle sarebbe più opportuno utilizzare i fogli di stile su caratteristiche come i margini, il padding, i bordi, lo sfondo.