HTML

(http://xhtml.html.it/guide/leggi/51/guida-html/)

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) 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).

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"

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

<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 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

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 World Wide Web Consortium che riguardano il codice HTML:

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

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

Da quanto abbiamo detto la nostra prima pagina sarà questa, che è consultabile anche nell'esempio allegato:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"> <html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
  <title>HTML.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.

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.

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.

Approfondimenti

Nella lezione della **guida CSS** dedicata alla Classificazione degli elementi viene approfondito l'argomento da noi trattato.

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.

Qui potete trovare un esempio della pagina con lo sfondo blu

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 questa pagina: 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 cambiameni 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 (cfr. articoli su "palette WEB SAFE" nella sezione PRO).

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).

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">

Ecco subito un esempio di pagina impostata con lo sfondo.

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.

Se usate Windows, per fare una prova provate a impostare diversamente il tema delle finestre. Dal pannello di controllo: **Schermo > Aspetto > Combinazione** e poi scegliere:

"nero a contrasto elevato", oppure "prugna".

Infine visualizzate questa pagina - che è senza sfondo – e vedrete che la pagina HTML prenderà la colorazione che avete impostato nel tema delle finestre.

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), come è possibile vedere nell'esempio.

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).

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, visualizzabile anche in questa pagina:

```
<!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>HTML.it</title>
</head>

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

Testo di prova
</body>
</html>
```

Approfondimenti: lo sfondo con i CSS

Tutti gli attributi del tag `<body>` che abbiamo visto finora (da eccezione dell'attributo `lang`) servono a modificare caratteristiche che riguardano il layout della nostra pagina HTML. Come si può vedere, con un markup 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.

Le regole su come impostare lo sfondo con i CSS vengono spiegate dettagliatamente nella relativa lezione della guida ai CSS.

I fogli di stile sono estremamente potenti, e dando un'occhiata al link segnalato poco sopra si può leggere che è anche possibile fissare lo sfondo in modo che non si ripeta:

```
<body style="background-image: url(sfondo.gif);
             background-repeat: no-repeat;" >
```

si tratta di una sintassi che funziona bene persino con Netscape 4.x, come si può vedere nella pagina di esempio.

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;" >
```

come si può vedere anche nell'esempio.

Impostare il colore del testo e dei link per tutta la pagina

Il testo

Se non viene impostato 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. Immaginiamo ad esempio di voler utilizzare proprio il nero come sfondo: con la pagina nera e il 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">
```

come nell'esempio consultabile in questa pagina.

I link

Spiegare che cosa siano i link sembra superfluo, in effetti è l'esperienza della navigazione nel Web ad averci già insegnato che il link è un collegamento, un ponte tra una pagina è l'altra.

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

Status	Codifica in HTML 4.01	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`	Il collegamento è attivo nel momento in cui il link è stato cliccato e sta avvenendo il passaggio da una pagina all'altra. 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. Con Internet Explorer è possibile vedere anche una linea tratteggiata attorno al collegamento attivo.

		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".
Collegamento al passaggio del mouse	`assente`	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, utilizzando la parola chiave `hover`" nei CSS)

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">
```

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.

Si tratta di un **elemento di blocco** (cfr. lezioni precedenti). Eccone un esempio.

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>
```

Ecco come viene visualizzato l'esempio.

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>
```

Si tratta dell'**elemento di tipo block per eccellenza**. Ecco come viene visualizzato l'esempio.

Lo

Lo `<span>` è un contenitore generico che può essere annidato (ad esempio) all'interno dei `<div>`. 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 `<span>` 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, come si vede nell'esempio.

Montare gli elementi insieme

Le caratteristiche più evidenti di `<p>`, `<div>` e `<span>` sono quindi:

- **<p>** 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

Un esempio dovrebbe chiarire il tutto.

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.

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	`<p align="left">testo</p>`	Nel mezzo del cammin di nostra vita mi ritrovai per una selva oscura ché la diritta via era smarrita
Testo allineato a destra	`<p align="right">testo</p>`	Nel mezzo del cammin di nostra vita mi ritrovai per una selva oscura ché la diritta via era smarrita
Testo giustificato	`<p align="justify">testo</p>`	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 `<br />` ("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" />
```

Risultato:

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
<b>testo in	Questo **testo** è in	Formatta il testo in

grassetto **Esempio:** Questo testo è in grassetto	grassetto	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 **Esempio:** 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.
<strike>testo barrato</strike> **Esempio:**	Questo ~~testo~~ è barrato	Con il testo barrato, vengono indicate (ad esempio) le correzioni.

Questo <strike>testo</strike> è barrato		
^{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 nostra vita mi ritrovai per una selva oscura ché la diritta via era smarrita </blockquote>	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.
<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: "Il coraggio, uno non se lo può dare"	Indica una citazione breve all'interno del testo. Nessun rendering del

Come diceva Don Abbondio: <q>***&quot***;Il coraggio, uno non se lo può dare***&quot***;</q>.		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

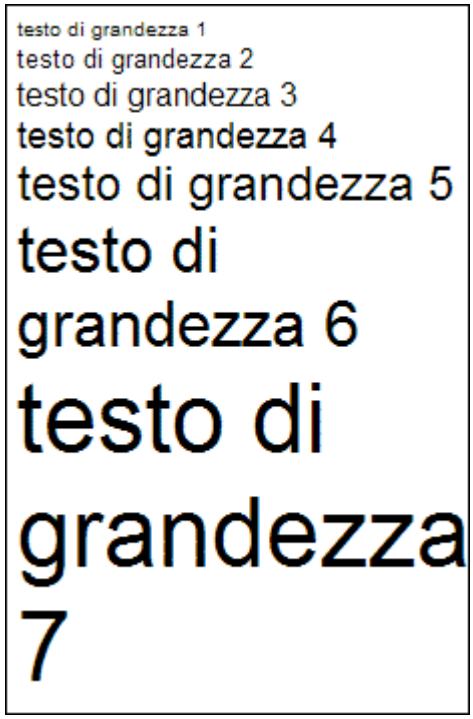
La seconda codifica è preferibile alla precedente, perché la scelta del tipo di carattere viene effettuata una sola volta, evitando così di scrivere del codice inutile. È importante notare che per evitare la ripetizione i due tag sono annidati l'uno dentro l'altro.

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

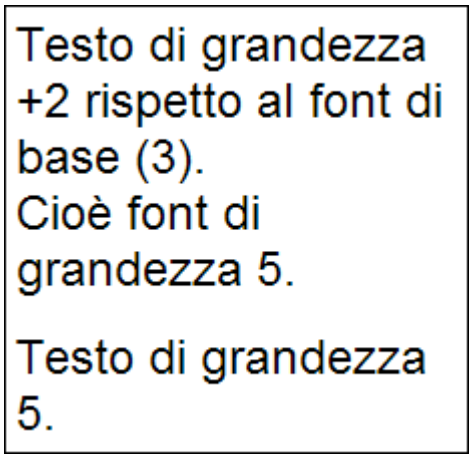
Nel caso dei valori relativi alla dimensione di base è possibile "spostarsi" nella scala di grandezza del `&lt;font&gt;` utilizzando i segni "`+`" e "`-`".

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	Effetto

```
<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>
```



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>
```

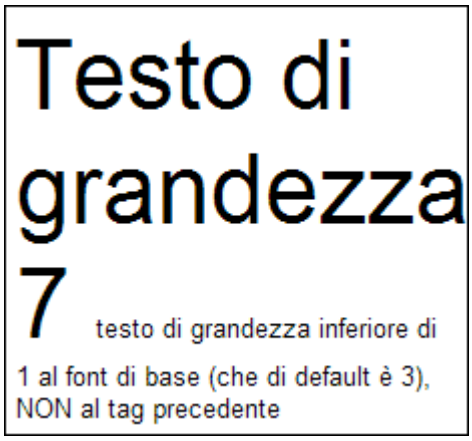
Come si può vedere nella pagina esemplificativa.

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	Effetto

Testo di grandezza 7

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



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, come si può vedere nell'esempio.

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.

NOTA BENE

Quando state utilizzando il tag <font> - 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, esprimendo le dimensioni in pixel.

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

```
<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...
```

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. Chi fosse interessato ad approfondire può consultare la relativa lezione della guida ai fogli di stile.

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'.

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.

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

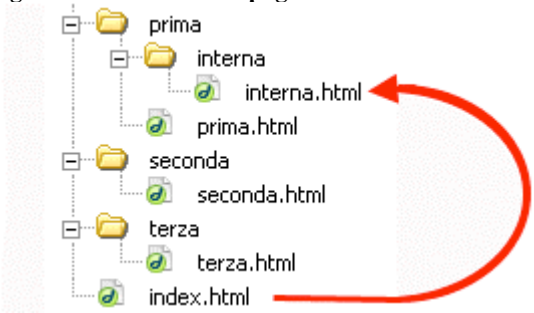
Spesso vi troverete tuttavia a fare riferimento a documenti situati nel vostro stesso sito, e – se state sviluppando il sito sul vostro computer di casa (cioè "in locale") – magari non avete ancora un indirizzo web, e non sapete di conseguenza come impostare i percorsi. È utile allora capire come funzionano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



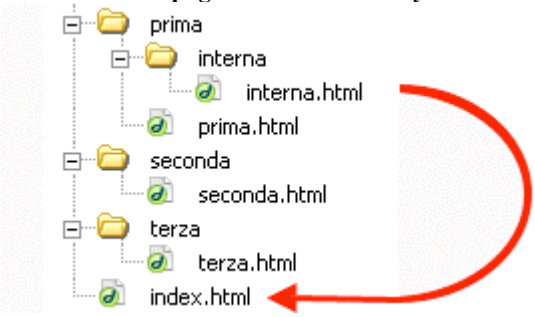
Dalla pagina "index.html" vogliamo cioè far riferimento al file "interna.html", che si trova all'interno della directory "interna", che a sua volta si trova all'interno della directory "prima".

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.

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.

I link interni o ancore

È possibile anche creare un indice interno al documento, utilizzando le àncore. Ciascuna àncora può avere infatti un nome:

Stiamo per esaminare la struttura…. Eccetera…

Da notare che in mancanza dell'attributo che indica il collegamento (href) le àncore non vengono viste come link, ma la loro formattazione è indistinguibile dal "normale" testo.

In un ipotetico indice è allora possibile far riferimento all'àncora presente all'interno del documento attraverso un link che punti ad essa:

vai al primo paragrafo

il cancelletto indica che il collegamento deve cercare un àncora chiamata "primo" all'interno della pagina stessa.

Se non si specifica il nome dell'àncora a cui si vuol puntare, viene comunque creato un link che punta ad inizio pagina (viene cercata un'àncora il cui nome non è specificato). Questo infatti è un ottimo escamotage per creare link "vuoti" (in alcuni casi vi occorreranno). Ad esempio:

link vuoto

Per creare un indice interno alla pagina si procede dunque in due fasi distinte:

- creazione dell'ancora a cui puntare (**<a name="mioNome">**)
- creazione del collegamento all'ancora appena creata e riferimento
- attraverso il cancelletto (**<a href="#mioNome">**)

È bene non confondere le due fasi.

Un esempio di quanto appena esposto lo potete trovare nella pagina dell'esempio.

Gli attributi dei link

target

È anche possibile specificare in quale finestra la pagina linkata deve essere aperta: di default infatti la pagina viene aperta all'interno del documento stesso, ma è possibile specificare che la pagina sia aperta in una nuova finestra:

```
<a target="_blank" href="http://www.html.it">visita HTML.IT</a>
```

cioè:

visita HTML.IT

vedremo questo attributo più in dettaglio quando parleremo dei frames.

title

L'attributo **title** è molto importante, e serve per specificare un testo esplicativo per l'elemento a cui l'attributo è riferito (il title si può infatti utilizzare anche per elementi differenti dalle ancore). Questa spiegazione addizionale favorisce l'accessibilità del sito anche ai disabili, alle persone per esempio che hanno disturbi alla vista. Se lasciate il cursore del mouse per qualche secondo su un collegamento dotato di title, vedrete comparire una specie di etichetta con il testo specificato nel `title`:

```
<a title="in HTML.it puoi trovare risorse per webmaster"
    href="http://www.html.it/" target="_blank" >
```

```
    Visita HTML.it
</a>
```

cioè:

Visita HTML.it

L'attributo `title` è anche utilissimo per migliorare la propria presenza nei motori di ricerca, che ne vanno a leggere il contenuto.

hreflang

Con "hreflang" si indica la lingua del documento: si tratta di un attributo che migliora l'accessibilità del sito, oltre ad essere potenzialmente utile per i motori di ricerca (l'attributo può essere utilizzato ad esempio per specificare la presenza di una sezione del proprio sito in lingua inglese):

```
Nel sito del <a href="http://www.w3c.org/" hreflang="eng" target="_blank">World
Wide Web Consortium</a> puoi trovare le specifiche dell'HTML in lingua inglese
```

cioè:

Nel sito del World Wide Web Consortium puoi trovare le specifiche dell'HTML in lingua inglese

accesskey

Le **accesskey** sono delle scorciatoie "da tastiera" che potete utilizzare nel vostro sito. Si tratta di scegliere delle lettere della tastiera che - quando vengano digitate dall'utente - permettono di andare direttamente a determinate pagine. Per esempio potreste specificare che:

```
<a href="http://www.html.it/" accesskey="h" target="_blank" >Torna all'home page
di HTML.it</a>
```

cioè:

Torna all'home page di HTML.it

In questa pagina digitando "**ALT + h + invio**" con Internet Explorer, oppure direttamente "**h + invio**" con Mozilla si accede direttamente all'home page di HTML.it. Si tratta di un'altra tecnica per migliorare l'accessibilità, ma un uso improprio e indiscriminato di questa tecnica può risultare davvero deleterio per la navigazione. Diciamo che le accesskey dovrebbero essere riservate per la navigazione dei menu che portano alle parti principali del sito.

Colorare i link

Abbiamo già visto 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

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.

Il tag <base>

I percorsi relativi fanno di norma riferimento alla directory in cui si trova il file HTML che stiamo scrivendo. Se tuttavia vogliamo far riferimento a un differente percorso per tutti i percorsi relativi, possiamo farlo specificandolo grazie al tag `<base>`, che va incluso nella head del documento. Ad esempio con:

```
<base href="http://www.mioSitoWeb.com/miaCartella">
```

specifico che d'ora in poi tutti i percorsi relativi faranno riferimento al percorso indicato. E poi nel documento potrò scrivere:

```
<a href="mioFile.html">collegamento al mio file</a>
```

sicuro che farà riferimento a:

```
http://www.mioSitoWeb.com/miaCartella/mioFile.html
```

Si tratta di una caratteristica particolarmente utile quando bisogna mandare ad esempio delle mailing list in formato HTML: possiamo infatti utilizzare i percorsi relativi per sviluppare la pagina della mailing list in locale, e mantenerli inalterati grazie all'utilizzo di questo tag. Grazie ad esso siamo infatti sicuri che anche l'utente che riceverà la mail potrà visualizzare le immagini e i link con un percorso corretto.

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 The GIMP, Photoshop o Paint Shop Pro).

I formati ammessi nel Web sono sostanzialmente tre:

- **GIF** (**Graphic Interchange Format**): le GIF sono immagini con non più di 256 colori (dunque con colori piatti e senza sfumature), come grafici o icone
- **JPG**: 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 per risolvere i problemi di copyright del formato GIF (che è appunto 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).

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

Inoltre 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. Allo stato attuale dell'arte infatti molti utenti (e non soltanto quelli italiani) navigano ancora con un modem analogico da 56 Kbs: inserire troppe immagini significa dunque creare pagine lente da caricare.

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

La sintassi per inserire una immagine è:

<img src="http://html.it/guide/img/guida_html/miaImmagine.gif">

dove:

- **img** significa **image**, cioè **immagine**
- **src** significa **source**, cioè origine

Il tag <img> è un tag "vuoto", che non ha la necessità di essere chiuso.

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):

<img src="logo.gif">	

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

<img src="../img/logo.gif">
<img src=" http://www.html.it/img/logo.gif ">

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:

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

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:

<img src="logo.gif" **alt="HTML.it"** width="224" height="69">	

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:

<img src="logo.gif" alt="HTML.it" longdesc="descrizione.html" width="224" height="69">

Nell'esempio allegato è possibile visualizzare il codice di una pagina con la descrizione estesa dell'immagine. 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**:

<img src="logo.gif" **title="HTML.it"** alt="HTML.it" width="224" height="69">	

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

<img src="logo.gif" **border="3"** width="224" height="69">	

Si noti che i link lasciano **sempre** di default 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):

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

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

 <img src="logo.gif" **border="0"** width="224" height="69"> 	

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.

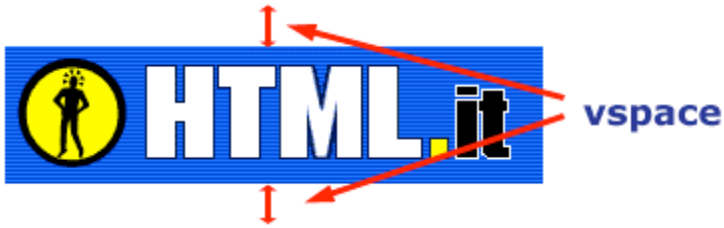
Nel caso di hspace impostiamo uno spazio orizzontale da ambo i lati, come in questo caso:

<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.

Le mappe di immagine

A volte è necessario far sì che solo una determinata parte di un'immagine sia collegata a un link. È il tipico caso delle Regioni d’Italia: abbiamo una cartina e abbiamo la necessità che alla sagoma di ciascuna regione corrisponda un differente link.

In questo caso possiamo utilizzare le mappe. Ne esistono di due tipi:

- **le mappe lato client**
- **le mappe lato server (non più utilizzate)**

Le mappe lato-client

Questo tipo di mappa è contraddistinto dall'attributo **usemap** del tag img:

<img src="miaImmagine.gif" usemap="nomeMappa">

come valore dell'attributo usemap bisogna specificare il nome della mappa a cui l'immagine fa riferimento.

A questo punto non ci resta che creare la mappa:

<map name="nomeMappa">
…
</map>

All'interno del tag **<map>** dobbiamo poi specificare le aree sensibili a cui corrisponderanno i nostri link, con relativi forme, coordinate e collegamenti. Per farlo si utilizza il tag **<area>**, per ogni zona sensibile che vogliamoc reare.

Vediamo un esempio: abbiamo preso la cartina dell'Italia e – a scopo puramente didattico – abbiamo deciso di mappare la Regione **Valle D'Aosta** con una forma rettangolare, la **Sardegna** con un cerchio, e la **Sicilia** con un poligono (per rendervene conto passate il mouse su una di queste regioni).

Figura 1. Esempio di immagine con mappe

<img alt="Esempio di immagine con mappe" src="italia.gif" width="220" height="235" border="1" usemap="#regioni">

<map name="regioni" id="regioni">

<area shape="rect" coords="14, 24, 35, 37" href="http://www.regione.vda.it/" target="_blank" alt="Valle d'Aosta">

<area shape="circle" coords="45, 156, 29" href="http://www.regione.sardegna.it/" target="_blank" alt="Sardegna">

<area shape="poly" coords="105, 199, 115, 197, 121, 200, 131, 201, 139, 198, 150, 197, 156, 195, 151, 201, 145, 209, 148, 212, 150, 219, 152, 225, 147, 227, 144, 231, 128, 221, 119, 219, 113, 212, 108, 212, 102, 210, 98, 205" href="http://www.regione.sicilia.it/" target="_blank" alt="Sicilia">

</map>

le coordinate fanno riferimento all'immagine stessa, e il vertice **in alto a sinistra** è l'ipotetico punto con coordinate **0,0**. Le coordinate dei punti che descrivono le varie forme si riferiscono alla distanza in pixel da quel punto (si tratterà di volta in volta della x o della y).

Come si può vedere per definire un'area è necessario specificare una forma, che può essere di tre tipi:

Forma	**Descrizione**
rettangolare <area shape="rect">	è necessario specificare alcune coordinate del rettangolo per individuare i veritici. In particolare sono da specificare (in quest'ordine): • la x del lato di sinistra • la y del lato alto • la x del lato destro • la y del lato in basso
circolare <area shape="circle">	è necessario specificare le coordinate del centro (x e y) e la misura del raggio (in pixel)
poligonali <area shape="poly">	è necessario specificare tutte le coordinate del poligono a coppie

In ciascun tag **<area>** è inoltre possibile specificare l'attributo **alt** per il testo alternativo (ed eventualmente il **longdesc**).

Per il resto, il tag **<area>** si comporta esattamente come il tag **<a>**, con la possibilità di specificare ad esempio il **target** in cui aprire i link.

In realtà non è difficile disegnare le mappe, perché ci sono già software che lo fanno al posto nostro. Se utilizzate un editor visuale ad (esempio Dreamweaver) potete trovare degli strumenti integrati nell'ambiente di sviluppo che vi consentono di disegnare le mappe in tutta tranquillità.

In alternativa si possono usare programmi appositi, come CoffeCup Image Mapper, CuteMap o MapEdit

Per quel che riguarda il luogo in cui posizionare la mappa così creata, dipende dalle vostre preferenze: è una buona norma però situare la mappa in prossimità dell'immagine, in modo da poterla reperire facilmente.

Approfondimenti

Con Internet Explorer le mappe a volte lasciano un fastidioso tratteggio sull'area che è stata appena cliccata. Per eliminarlo è sufficiente utilizzare la seguente sintassi:

onFocus='this.blur()'

da applicare al tag <AREA> in questo modo:

<area shape="circle" coords="45, 156, 29" href="http://www.regione.sardegna.it/" target="_blank" alt="Sardegna" onFocus=?this.blur()?>

Le mappe di immagine lato server

In questo caso la mappatura dell'immagine non è visibile all'interno della pagina HTML, ma è presente all'interno di un programma sul server, ad esempio un cgi.

La presenza di una mappa sull'immagine è specificata dall'attributo **ismap** del tag **<img>** e l'indicazione della posizione in cui si trova la mappa sul server viene data attraverso un link. Ad esempio:

<img src="immagine.gif" ismap>

(notare che l'attributo **ismap** non ha alcun **valore**).

Il link fa riferimento alla locazione della mappa sul server, e quando l'utente cliccherà sul collegamento, sulla barra degli indirizzi comparirà qualcosa di questo genere:

http://www.html.it/cgi-bin/menu.map?25,55

dal momento che il programma sul server avrà tradotto la mappa in coordinate da passare al mouse.

C'è da dire che le mappe lato server non vengono più usate, perché estremamente scomode (è poco pratico mantenere in differenti file le indicazioni sull'immagine e quelle sulla mappa), dunque non le vedremo nel dettaglio. Venivano utilizzate soprattutto agli albori del web, quando non tutti i browser erano in grado di interpretare le mappe lato-client: con le mappe lato-server si aveva la certezza, allora, che qualsiasi utente potesse leggere correttamente l'immagine (dal momento che il lavoro di mappatura veniva fatto dal server, appunto, e non dal computer dell'utente).

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, come si può vedere nell'esempio allegato, che non è corretto.

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, come si può vedere nella pagina di esempio.

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.

Creare gruppi di righe: <caption>, <thead>, <tfoot>, <tbody>

Come dicevamo, la struttura delle tabelle ha letteralmente invaso le pagine HTML, che si sono riempite di **<tr>** e di **<td>**.

Per portare un po' di ordine in questo caos nelle specifiche sono stati introdotti dei tag (opzionali) che consentono di capire facilmente quali siano le diverse parti della tabella.

Per individuare facilmente i gruppi di righe sono stati introdotti i seguenti tag:

<caption>	è l'intestazione, il titolo con un commento esplicativo sulla tabella
<thead>	è l'intestazione, la parte iniziale della tabella, quella che contiene ad esempio indicazioni sul contenuto delle celle
<tfoot>	è il piede, la conclusione della tabella, quella che consente ad esempio di tirare le somme
<tbody>	è il corpo, la parte centrale con il contenuto vero e proprio della tabella

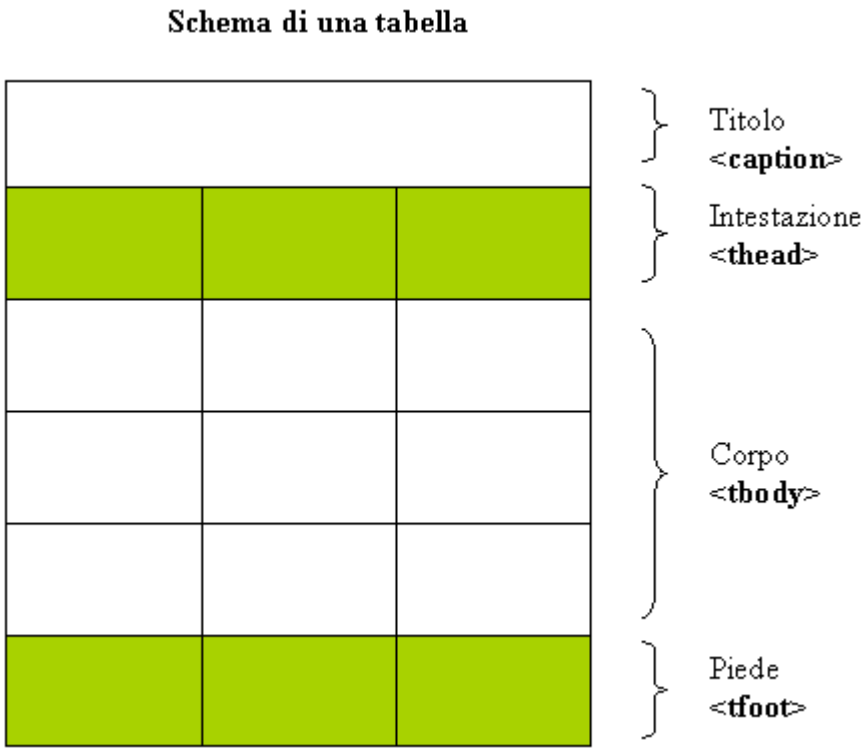
<thead>, **<tfoot>**, **<tbody>** sono tag che consentono di individuare gruppi di righe (**"row group"**).

Da notare che – contrariamente a quello che si potrebbe pensare – il tag **<tfoot>** che chiude la tabella, è **anteposto** rispetto al **<tbody>**. L'idea di base è che il browser nell'eseguire il rendering del codice tenga conto della parte iniziale e della parte finale della tabella, e il corpo vero e proprio sia sviluppato nella sua interezza tra le due estremità.

Un'altra particolarità è che le celle all'interno del tag **<thead>** possono essere indicate con **<th>** (**"table header"**), al posto del consueto **<td>** (**"table data"**), in questo caso il contenuto delle celle è automaticamente formattato in grassetto e centrato.

Ecco comunque uno schema che riassume la struttura delle tabelle secondo l'HTML 4:

Figura 1. Elementi della tabella



A questa pagina è possibile visualizzare un esempio.

Raggruppare gli stili delle colonne

Come è possibile suddividere le righe di una tabella in gruppi ordinati, allo stesso modo è possibile raggruppare gli stili delle colonne. Il tag da usare è il **<colgroup>** e serve per fornire indicazioni su come le colonne debbano essere visualizzate.

Purtroppo questo tag nel momento in cui scrivo non funziona ancora correttamente con tutti i browser: il tag è correttamente visualizzato da Internet Explorer, è supportato solo in parte da Opera, ma non è ben interpretato da Mozilla (e dunque neanche da Netscape 7), che lo ignora tranquillamente. Tuttavia trattandosi di una specifica del W3C in futuro il supporto a questa sintassi dovrebbe essere maggiore.

Prima di vedere nel dettaglio questo tag anticipiamo due concetti che vedremo in seguito quando ci occuperemo più approfonditamente della formattazione delle tabelle e che ora ci servono per meglio comprendere l'uso di **<colgroup>**.

- L'attributo **align** può essere riferito sia a **<tr>**, sia a **<td>** e serve per definire l'allineamento dei contenuti a sinistra, a destra o al centro (**left**, **right**, **center**)
- L'attributo **bgcolor** può essere riferito tanto a **<table>**, **<tr>**, o **<td>** e – come abbiamo visto a proposito del **<body>** - consente di impostare un colore di sfondo

Il tag **<colgroup>** - la cui chiusura è facoltativa - va posto subito dopo il tag **<caption>** e prima di **<thead>**, e consente di impostare un layout unico per le colonne senza avere la necessità di specificare allineamento del testo, o il colore di sfondo per ogni singola cella.

Con l'attributo **span** (da non confondere con il tag ****) possiamo impostare il numero di colonne che fanno parte del gruppo.

Per avere un layout di questo genere:

Figura 1. Esempio di uso dell'attributo «span»

Fatturato dell'azienda XYZ

Mesi di attività	Fatturato da attività 1 (in euro)	Fatturato da attività 2 (in euro)
gen	10.000	1.000
feb	20.000	2.000
Totale	30.000	3.000

basterà allora scrivere:

```
<table width="75%" border="1">

 <colgroup bgcolor="#00FFCC" width="20%" align="right"></colgroup>
 <colgroup span="2" bgcolor="#33CCFF" width="30%" align="center">

 <caption>
  <div align="center"><b>Fatturato dell'azienda XYZ</b></div>
 </caption>

 <thead>
  <tr>
   <th>Mesi di attivit&agrave;</th>
   <th>Fatturato da attivit&agrave; 1 (in euro)</th>
   <th>Fatturato da attivit&agrave; 2 (in euro)</th>
  </tr>
 </thead>

 eccetera....

</table>
```

come illustrato dettagliatamente in questa pagina.

<colgroup> ha l'indubbio vantaggio di poter attribuire in una sola volta la formattazione a un numero elevato di colonne. Ad esempio:

```
<colgroup span=”40” align=”right”>
```

Se si preferisce attribuire più esplicitamente lo stile ad una colonna si può usare il tag **<col>** (che non necessita chiusura) all'interno di **<colgroup>**. In questo caso l'attributo span va riferito a **<col>** e non a **<colgroup>**.

Ad esempio:

```
<colgroup span="5" width="20" bgcolor="red"></colgroup>
<colgroup width="60">
   <col span="5" bgcolor="blue" align="left">
   <col span="5" bgcolor="green" align="right">
</colgroup>
```

come mostrato in dettaglio nella pagina di esempio.

Approfondimenti

I tag **<colgroup>** e **<col>** in teoria supportano anche la possibilità di creare delle celle larghe proporzionalmente. Ad esempio:

```
<colgroup width=”1*”>
<colgroup width=”2*”>
```

questa sintassi dovrebbe dividere lo spazio a disposizione in tre parti e assegnare una parte alla prima cella e due parti alla seconda cella. In realtà questa sintassi non è ancora supportata da nessun browser.

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>&amp;nbsp;</td>
  <td colspan="2" align="center" valign="middle">
   <b>&amp;lt;td colspan=&amp;quot;2&amp;quot;&amp;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%">&amp;nbsp;</td>
  <td width="30%" rowspan="2" align="center" valign="middle">
   <b>&amp;lt;tdrowspan=&amp;quot;2&amp;quot;&amp;gt;</b>
  </td>
  <td width="30%">&amp;nbsp;</td>
 </tr>
 <tr>
  <td>&amp;nbsp;</td>
  <td>&amp;nbsp;</td>
 </tr>
</table>
```

Per un esempio approfondito esaminare il codice di questa pagina.

Annidare tabelle

È anche possibile annidare le tabelle le une dentro le altre. Come in questo esempio:

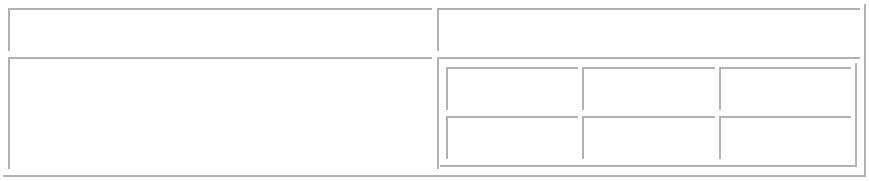
```
<table width="430" border="1">  <tr>
  <td width="50%">&amp;nbsp;</td>
  <td width="50%">&amp;nbsp;</td>
 </tr>
 <tr>
  <td height="24">&amp;nbsp;</td>
  <td>

    <table width="100%" border="1">
     <tr>
      <td>&amp;nbsp;</td>
      <td>&amp;nbsp;</td>
```

```
    <td> </td>
   </tr>
   <tr>
    <td> </td>
    <td> </td>
    <td> </td>
   </tr>
  </table>

 </td>
</tr>
</table>
```

che dà come risultato:



Per evitare che compaiano nel layout degli spazi indesiderati è consigliabile aprire e chiudere la tabella a ridosso del tag della cella che la contiene.

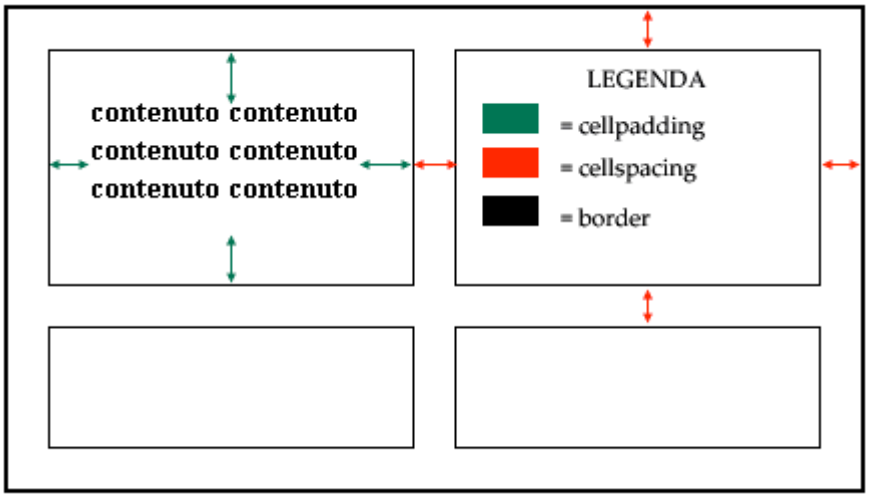
Attributi del tag table

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">

come si può vedere nell'esempio.

per quel che riguarda l'attributo **border,** a partire da Internet Explorer 4 e da Netscape Navigator 6 è possibile modificare l'aspetto dei bordi esterni della tabella (tramite l'attributo **frames**) e delle righe fra le celle (tramite l'attributo **rules**).

Vediamo quali sono i possibili valori e i relativi esempi:

esempio	spiegazione
<table border="1" frame="above"> (nelle pagine di esempio qui a fianco le righe interne tra le celle sono state azzerate per facilitare la comprensione, dunque ci troveremo nella situazione in cui **rules="none"**)	Il bordo della tabella è presente: • **void**: in nessun lato. È il valore di default • **above**: solo nel lato superiore • **below**: solo nel lato inferiore • **hsides**: solo nei lati superiore e inferiore • **vsides**: solo a sinistra e a destra • **lhs**: solo nel lato sinistro (left-hand side) • **rhs**:solo nel lato destro (right hand side). • **box**: in tutti e quattro i lati

	• **border**: in tutti e quattro i lati
<table border="1" rules="rows"> (nelle pagine di esempio qui a fianco i bordi esterni della tabella sono stati azzerati per facilitare la comprensione, dunque ci troveremo nella situazione in cui **frame="void"**)	Le righe interne alle celle sono presenti: • **none**: da nessuna parte. È il valore di default • **groups**: le righe separano i gruppi (siano essi gruppi di righe: <thead>, <tfoot>, <tbody> - o gruppi di colonne: <colgroup>) • **rows**:le righe separano i vari <tr> • **cols**:le righe separano le colonne • **all**:le righe separano tanto i <tr>, quanto le colonne

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

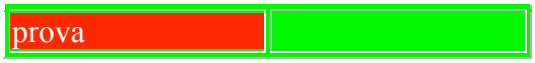
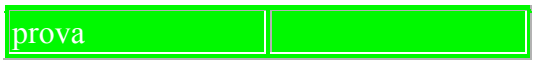
Abbiamo già esaminato gli attributi **width** e **height** che determinano la larghezza e l'altezza (in pixel o in percentuale) di tabelle, righe o celle.

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: 
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: 

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%">contenuto</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.
   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.

Impaginare un layout con le tabelle

Le tabelle, grazie alle loro molteplici e multiformi caratteristiche, si sono rivelate uno strumento indispensabile non solo per impaginare i dati ma soprattutto per visualizzare i layout grafici: grazie alle tabelle è infatti possibile costruire delle griglie in cui inserire i vari contenuti di un sito e per mezzo degli sfondi, dei margini è possibile riprodurre un’impostazione accattivante.

Visualizzando la pagina dell'esempio è possibile vedere il layout di HTML.it impaginato grazie alle tabelle: questo primo esempio utilizza una tabella unica suddivisa nelle classiche sezioni **<thead>**, **<tbody>** e **<tfoot>**.

In questo secondo esempio il medesimo risultato è ottenuto sovrapponenedo una serie di tabelle, senza così vincolare tutto a un'unica struttura.

Le possibilità di ottenere il layout che abbiamo appena esaminato comunque sono molteplici. Grazie alle tabelle è possibile anche progettare **layout liquidi**, che si adattino cioè alla risoluzione del monitor dell'utente.

È vero che l'impaginazione a tabelle ha fatto il suo tempo:

- perché mischia la visualizzazione dei dati ai dati stessi, e dunque è difficile da gestire
- perché riempie le pagine con molto codice rallentando lo scaricamento

Oggi siamo in un periodo di transizione. Gli sviluppatori dai "vecchi" modi di costruire i siti web (a tabelle), dovrebbero migrare verso qualcosa di nuovo: verso una impaginazione che utilizzi i fogli di stile e l'(x)html.

L'impaginazione a tabelle rimane, tuttavia, senz'altro una pietra miliare del web.

Comporre una pagina in frame

Comporre una pagina in frame

I frame ("riquadri") comparvero per la prima volta con Netscape Navigator 2: si tratta della possibilità di suddividere una medesima finestra del browser in vari riquadri indipendenti.

Questa soluzione all'epoca si rivelò un successo, dal momento che permetteva notevoli vantaggi:

- Fino a qualche tempo fa la velocità di navigazione era ben poca cosa, e si navigava con modem analogici molto lenti (anche da 14.4 kbs): i frame hanno l'indubbio vantaggio di non costringere a ricaricare tutta quanta la pagina, accelerando così la navigazione dell'utente all'interno di un sito web. D'altro canto il fatto che solo una parte del contenuto sia ricaricata fa risparmiare banda anche dal punto di vista del server che deve erogare le pagine
- Per quel che riguarda i webmaster, i frame hanno la caratteristica di utilizzare una struttura che consente di non ripetere le parti comuni nelle varie pagine di un sito, dal momento che il contenuto della pagina (per sua natura) è organizzato "a zone"
- Il fatto di poter mantenere fisso su un lato del monitor il menu di navigazione e far scorrere sull'altro lato il contenuto piace a molti utenti, soprattutto quando la risoluzione del monitor è bassa (800 x 600 o 640x480, magari su un monitor da 15'')

Tutte queste caratteristiche hanno causato un vero e proprio boom dei frames, tanto che subito dopo l'invenzione della Netscape, anche Microsoft si trovò a "copiare" la possibilità di strutturare le pagine in questo modo; in seguito (con l'HTML 4) i frames divennero una specifica ufficiale del W3C.

Struttura di un frameset

Per utilizzare i frame, è necessario creare una pagina che contenga la dichiarazione della struttura che vogliamo utilizzare. Vediamo subito il codice:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//IT"
"http://www.w3.org/TR/html4/frameset.dtd">

<html>
<head>
 <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
 <title>HTML.it</title>
</head>

<frameset rows="50%,50%" cols="50%, 50%">
 <frame src="prima.html">
 <frame src="seconda.html">
 <frame src="terza.html">
 <frame src="quarta.html">

 <noframes>
  <p>Qui può essere indicato il link a
   <a href="senzaFrame.html"> una versione del sito</a>
   che non utilizzi un layout a frame
  </p>
 </noframes>

</frameset>
</html>
```

L'esempio completo si trova qui.

Come vi sarete accorti, rispetto alle pagine che abbiamo studiato finora cambiano alcune cose. In primo luogo cambia il **doctype**, cioè il **tipo di documento** di riferimento.

All'inizio del documento al posto di questa riga:

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

compare ora infatti questa dicitura:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//IT"
"http://www.w3.org/TR/html4/frameset.dtd">
```

stiamo indicando semplicemente al browser che facciamo riferimento alle specifiche che servono per regolare il comportamento dei frame.

Avrete notato inoltre che scompare il tag **<body>** e al suo posto troviamo il tag **<frameset>** ("set di riquadri"), che ci permette di indicare come devono essere indicati i frames all'interno della pagina

Il tag <frameset> ha sostanzialmente due importanti attributi: **rows** e **cols**:

- **rows** permette di specificare il numero e la grandezza delle righe, nel caso in cui venga omesso, significa che ci troviamo di fronte a una struttura a colonne . Ad esempio:

  <frameset cols="33%, 33%,*">

- **cols** permette di specificare il numero e la grandezza delle colonne e, nel caso in cui venga omesso significa che ci troviamo di fronte una struttura a righe

  <frameset rows="33%, 33%,*">

Nell'indicare la grandezza di ciascuna riga (o colonna) possiamo poi lasciare che una o più righe si auto-dimensionino, occupando tutto lo spazio che rimane, in questo caso utilizzeremo l'asterisco ("wild card"); normalmente invece potremo esprimere la grandezza dei riquadri secondo uno dei seguenti sistemi di misura (da scegliere a nostra discrezione):

dimensione fissa	Questa sintassi crea un frameset di 2 righe con 3 colonne ciascuna, per un totale di 6 riquadri: <frameset rows="150,*" cols="100,200,*"> L'altezza della 1a riga è di 150 pixel, mentre la seconda si adatta al resto della pagina. Le tre colonne sono larghe rispettivamente: 100 pixel, 200 pixel, e la terza colonna si adatta al resto della pagina
percentuale	Questa sintassi crea un frameset che si adatta alla risoluzione. La grandezza dei riquadri è espressa in percentuale: <frameset rows="20%,80%" cols="25%,25%,50%"> come si può vedere la prima riga occupa il 20% dell'altezza, la seconda il rimanente 80%. Le 3 colonne si dividono la larghezza: la prima colonna occupa il 25%, la seconda nuovamente il 25%, la terza il 50% dello spazio
proporzionale	In questo caso la ripartizione è proporzionale: <frameset rows="1*,3*" cols="3*,2*,1*"> • per ciò che riguarda le righe: l'altezza viene suddivisa in 4 parti (1+3); la prima riga ne occupa 1 parte e la seconda riga ne occupa 3 • per ciò che riguarda le colonne: l'altezza viene suddivisa in 6 parti (3+2+1); la prima colonna occupa 3 parti, la seconda riga ne occupa 2 e la terza 1

Una volta creata la nostra griglia con il tag **<frameset>**, incrociando le righe e le colonne, dobbiamo specificare dove si trova il file di origine di ciascun frame. Possiamo farlo con la sintassi:

<frame src="prima.html">

come si può vedere l'origine di ciascun frame è un documento HTML standard (come quelli che abbiamo analizzato finora): avrà dunque la sua dichiarazione di documento, la sua <head> e il suo <body>.

Se le dimensioni del riquadro non sono sufficienti a mostrare il documento nella sua interezza, il frame avrà delle barre di scorrimento, a meno che non sia stato esplicitamente specificato il contrario negli attributi (che vedremo tra breve).

Per visualizzare il codice HTML di ciascun frame è sufficiente andare nel riquadro desiderato e poi digitare il tasto destro del mouse. Quindi:

- Con Internet Explorer:
  selezionare HTML
- Con Mozilla:
  selezionare this frame > view frame source

È possibile anche individuare un frame in modo più preciso, assegnandogli un nome:

<frame id="primoRiquadro" name="primoRiquadro" src="prima.html">

la sintassi corretta per dare un nome a un frame dovrebbe essere:

id="primoRiquadro"

tuttavia per questioni di retro-compatibilità (con Netscape 4) è oramai entrato nell'uso utilizzare anche **name="primoRiquadro"**.

Frameset annidati

È possibile annidare diversi frames l'uno dentro l'altro. In questo caso, al posto di uno dei tag <frame> è sufficiente includere le indicazioni del nuovo frameset. Così:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//IT"
"http://www.w3.org/TR/html4/frameset.dtd">

<html>
<head>
 <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
 <title>HTML.it</title>
</head>
 <frameset rows="15%,70%,15%">
  <frame src="11.html">

   <frameset cols="25%,50%,25%">
    <frame src="21.html">
    <frame src="22.html">
```

```
  <frame src="23.html">
 </frameset>

 <frame src="12.html">

<noframes>
<p>Qui può essere indicato il link a <a href="senzaFrame.html"> una versione del sito</a> che
non
utilizzi un layout a frame</p>  </noframes>
</frameset>
</html>
```

L'esempio completo si trova qui.

Attributi dei frames per la visualizzazione

Attributi del frameset

Il tag frameset non ha (secondo le specifiche ufficiali) attributi per la visualizzazione. Alcuni attributi tuttavia sono entrati nell'uso e sono correttamente supportati dai browser attuali:

<frameset frameborder="no" cols="25%,75%">	L'attributo **frameborder** (di default impostato a "yes") permette di specificare se nel frameset devono essere presenti i bordi. L'esempio è qui
<frameset framespacing="20" border="20" cols="25%,75%">	**framespacing** funziona solo con Internet Explorer e permette di impostare lo spazio tra un frame e l'altro. Di fatto equivale all'attributo **border**, che permette di specificare lo spessore dei bordi in pixel. Per mantenere la compatibilità con Internet Explorer 4 (che non legge l'attributo border), di solito si specificano sia il **framespacing**, sia il **border**. L'esempio è qui
<frameset border="10" framespacing="10" bordercolor="#FF0000" cols="25%,75%">	**bordercolor** permette di specificare il colore dei bordi del frameset. L'esempio è qui

Attributi dei frame

A differenza degli attributi del tag frameset, che sono dovuti alla convenzione, i seguenti attributi del tag frame sono invece descritti nelle specifiche del W3C e permettono di modificare l'aspetto dei riquadri e il modo in cui l'utente può interagire con essi:

<frame src="prima.html" **scrolling="no"**> <frame src="prima.html" **scrolling="auto"**>	L'attributo **scrolling** (di default impostato a **"yes"**) indica se si vuol consentire all'utente di poter scorrere il frame oppure no. Nel caso sia impostato a **"no"**, il frame non ha la barra di scorrimento anche nel caso in cui il contenuto della pagina HTML vado oltre la cornice, come si può vedere nell'esempio. L'esempioè qui **scrolling** può anche essere impostato ad **"auto"**. In questo caso la barra di scorrimento compare in automatico, ma solo se necessario
<frame src="prima.html" **scrolling="no**"> <frame src="prima.html" **scrolling="no" noresize**>	**noresize** impedisce al singolo frame di essere ridimensionato. L'esempio è qui Se usato in unione con **scrolling="no"**,di fatto "blocca" il contenuto del frame. L'esempio è qui Un uso maldestro di questa tecnica potrebbe però impedire all'utente la corretta visualizzazione dei contenuti
<frame src="prima.html" **frameborder="0"**>	**frameborder** conesente di far apparire o meno i bordi del frame (i valori ammessi sono "0" e "1", ovvero "no" e "yes"). Se **frameborder** è impostato a "0" i bordi non sono visibili L'esempio è qui Attenzione però a come impostate i bordi nei vari frame, dal momento che i bordi di frame adiacenti non sempre vanno d'accordo(provate) Questo attributo permette di specificare

	un valore differente da quello impostato nel **frameborder** del **<frameset>**
<frame **marginwidth="50" marginheight="50"** src="prima.html">	**marginheight** e **marginwidth** permettono di impostare la distanza erticale (**marginheigth**) e orizzontale (**marginwidth**) ra i bordi del frame e il suo contenuto. L'esempio è qui

Approfondimenti

Ovviamente sarebbe meglio impostare i bordi e gli spazi tra i frame attraverso i CSS. Nella lezione della guida ai fogli di stile dedicata ai bordi con i CSS è spiegato come fare.

I link in un frameset e il tag noframes

Il target dei link in un frameset

In una struttura a frames può apparire difficile caricare il contenuto di un link nella sezione appropriata. Grazie all'attributo **target** possiamo specificare qual'è la destinazione del link; con questa sintassi siamo dunque in grado di caricare il contenuto di un collegamento nel riquadro che riteniamo più opportuno:

```
<a href="paginaDaLinkare.html" target="nomeDelFrame">
```

come si può vedere nell'esempio.

Ci sono poi delle "parole chiave" che ci consentono di ricaricare i link in destinazioni predefinite:

Codice	Descrizione
target="_blank"	Apre il link in una nuova finestra, senza nome L'esempio è qui
target="_self"	Apre il link nel frame stesso (è così di default) L'esempio è qui
target="_parent"	Il documento viene caricato nel frameset precedente a quello corrente (più esattamente nel frame genitore) L'esempio è qui Il comportamento di `_parent` è particolarmente evidente in una struttura annidata in cui alcune pagine HTML contengono a loro volta dei frameset, come nell'esempio: in questo caso viene caricato il contenuto del link nel frameset immediatamente precedente alla pagina del link.
target="_top"	Il documento viene caricato nella finestra originale, cancellando ogni struttura a frame.

L'esempio è qui

(si noti la differenza con **"_parent"**).

Più esattamente il documento viene caricatao nella parte piùalta ("top") della struttura, ed è questo il motivoper cui il frameset stesso viene annullato e sostituito dal contenutodel link.

Può essere noioso specificare per tutti i link il target appropriato. Ci viene in aiuto allora il tag **<base>** che ci consente di specificare la destinazione dei tutti link in una sola volta. Il tag va inserito nella pagina contenente i link. Ad esempio:

<base target="_blank">

a questo indirizzo è possibile visualizzare l'esempio completo.

Abbiamo già incontrato il tag **<base>** per impostare un percorso predefinito di default. Ovviamente è possibile combinare i due attributi di **<base>**:

<base href="http://www.html.it" target="_blank">

Questa sintassi indica che l'indirizzo di base per i percorsi della pagina è http://www.html.it e che tutti i link vanno aperti in una nuova pagina.

Il tag noframes

All'interno della dichiarazione del **<frameset>** abbiamo sempre visto comparire l'elemento **<noframes>**: questo tag serviva per browser particolarmente obsoleti (come Mosaic 2) che non erano in grado di leggere e visualizzare una struttura frameset. In previsione di situazioni di questo genere è possibile inserire all'interno del tag **<noframes>** un contenuto appositamente studiato per questa eventualità, ad esempio l'avvertimento che il sito è strutturato a frame, o un contenuto alternativo, oppure una descrizione per i motori di ricerca (come vedremo tra breve).

La struttura è la seguente:

```
<frameset rows="50%,50%">
  <frame src="prima.html">
  <frame src="seconda.html">

  <noframes>
    <p>Qui può essere indicato il link a
    <a href="senzaFrame.html">una versione del sito</a>
    che non utilizzi un layout a frame, oppure un
    contenuto alternativo, o anche una descrizione
    per i motori di ricerca.</p>
  </noframes>
</frameset>
```

È importante capire una volta per tutte come funziona questo genere di tag, dal momento che un tipo di struttura di questo genere ricorre in diverse occasioni nell'HTML: il tag <noscript> e i contenuti alternativi nel caso dell'**<iframe>** utilizzano infatti gli stessi principi di funzionamento.

I browser obsoleti non sono in grado di interpretare né la struttura di un frameset, né tanto meno il **<noframes>**: non essendo un tag previsto nelle specifiche rilasciate all'epoca in cui questi browser sono stati costruiti, si tratta infatti di una sintassi del tutto oscura per loro.

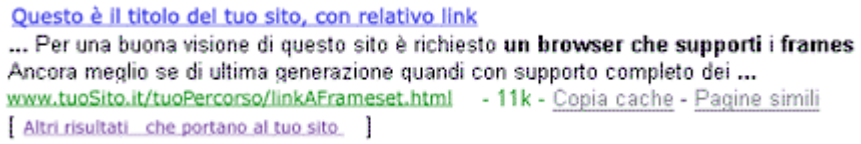
Di tutto il codice in questione l'unica parte comprensibile a questo genere di browser è l'HTML standard compreso all'interno del **<noframes>**: e questo si limitano a leggere.

Sono invece i browser moderni che sanno di dover ignorare il <noframes>, dal momento che il suo contenuto non li riguarda.

C'è da dire che, se tutti i browser odierni sono in grado di interpretare correttamente la struttura di un **frameset**, viceversa sotto molti punti di vista i motori di ricerca sono paragonabili a browser obsoleti, e non è raro che un sito organizzato a frame sia inserito nell'indice dei motori di ricerca in maniera non corretta: per questo motivo alcuni di essi "catturano" il contenuto del <noframes> come descrizione di un sito.

Quindi per evitare messaggi di questo genere a seguito di una ricerca in rete:

Figura 1. Messaggio del motore di ricerca per pagine senza "noframes"



è bene utilizzare il tag noframes per indicare la descrizione del sito: in questo modo miglioriamo anche il posizionamento nei motori di ricerca. Ad esempio:

```
<frameset rows="50%,50%">
  <frame src="prima.html">
  <frame src="seconda.html">

  <noframes>
    <p>Su PRO.HTML.it - Approfondimenti sul Web Publishing e articoli per
webmaster</p>
  </noframes>
</frameset>
```

L'iframe

“Iframe” significa **“inline frame”**: in qualsiasi momento in un documento che non utilizzi una struttura a frame è possibile creare un frame al volo grazie a questo tag.

Possiamo specificare la larghezza e l’altezza del tag, mentre gli attributi di visualizzazione sono gli stessi del tag **<frame>**: si tratta di una vera e propria finestra verso l’esterno all’interno di un documento ordinario.

Questo tag è correttamente supportato da tutti i browser moderni (Netscape 4 non lo supporta, ma questo browser oramai sta scomparendo). La sintassi è:

```
<iframe src="http://pro.html.it" width="300" height="300">
  Contenuto alternativo per i browser che non leggono gli iframe.
</iframe>
```

Come si può vedere tra l'apertura e la chiusura del tag è possibile specificare un contenuto alternativo per i browser che non siano in grado di leggere l'**<iframe>**: in realtà questi browser sono ciechi a questo tag (come abbiamo visto per il **<noframes>**) e dunque leggono direttamente il contenuto al suo interno. Sono invece i browser che supportano questa sintassi a ignorare volutamente quanto viene compreso tra apertura e chiusura del tag.

Anche in questo caso sarò opportuno utilizzare la possibilità di inserire un contenuto alternativo per migliorare il posizionamento nei motori di ricerca. Ad esempio:

```
<iframe src="http://pro.html.it" width="300" height="300">
  <p>Su <a href="http://pro.html.it">PRO.HTML.it</a> -
  Approfondimenti sul Web Publishing e articoli per webmaster</p>
</iframe>
```

Un esempio completo è disponibile a questa pagina.

Vantaggi e svantaggi dei frames

Abbiamo già visto all'inizio delle lezioni sui frames quali sono stati alcuni dei motivi di successo dei frames. E cioè:

- Dal punto di vista dell'utente: evitare di ricaricare le parti comuni
- Dal punto di vista del webmaster: includere il layout comune in pochi files
- Dal punto di vista dell'utente: mantenere in vista alcuni punti del layout

Tuttavia gli svantaggi che comporta un uso scorretto di un layout a frame sono superiori ai vantaggi che possono derivare dal loro utilizzo.

I motori di ricerca infatti navigano di link in link attraverso il web per catturare contenuti da indicizzare.

È frequente allora che una struttura a frame sia inserita all'interno di un motore di ricerca in modo errato: a volte viene catturato solo un menù (come in questo caso), altre volte compare soltanto la parte interna con il contenuto del frame e dunque viene perso ogni menu di navigazione (come in questo caso).

Per evitare problematiche di questo genere, è meglio evitare di utilizzare una struttura a frame; o nel caso in cui la si desideri utilizzare è bene prevedere sin da subito dei metodi che ricostruiscano il frameset, nel caso in cui sia catturata soltanto una pagina interna. È possibile farlo utilizzando JavaScript (un linguaggio di programmazione lato-client).

C'è anche da dire che oggi molti dei motivi che rendevano vantaggioso l'utilizzo dei frames sono venuti meno: la banda a disposizione si è ampliata, i CSS alleggeriscono la struttura dei siti e rendono possibile alcune soluzioni che prima erano difficili (come quella di mantenere un menu di navigazione sempre a portata di mano), e **la gestione dei contenuti può essere semplificata utilizzando le inclusioni lato server**.

Viceversa una struttura a frame risulta molto vantaggiosa nel caso in cui si utilizzino delle vere e proprie applicazioni che utilizzano internet (come le piattaforme di e-learning, la webmail, eventuali aree riservate del sito con accesso tramite login e password). In questo caso la suddivisione dei contenuti evita di sovraccaricare il server (dal momento che così vengono ri-caricati solo le parti strettamente necessari), semplifica la gestione, e quindi si rivela estremamente utile.

Impaginare a livelli con i CSS

Abbiamo già visto due stili di impaginazione: le **tabelle** e i **frame**. Come accennato più volte all'interno del corso esiste una terza via: l'impaginazione tramite **i fogli di stile** (ovvero "CSS", cioè **"Cascading Style Sheets"**, che significa **"fogli di stile a cascata"**).

La sintassi dei CSS esula dall'argomento del corso presente, è importante tuttavia afferrare il concetto che un elemento può essere disposto all'interno della pagina, semplicemente specificando le sue coordinate, o indicando il modo in cui deve essere spostato rispetto agli elementi che lo circondano.

Nel caso dell'impaginazione tramite i fogli di stile abbiamo per lo più a che fare con i tag **** e ** **, che possono essere accuratamente posizionati e visualizzati specificando ad esempio:

- larghezza
- altezza
- distanza dall'alto
- distanza da sinistra
- colore o immagine di sfondo
- colore, tipo e grandezza dei bordi
- distanza tra il contenuto e il bordo (padding) e tra il bordo e l'esterno (margin)

Per "livello" (o "layer") in HTML si intende appunto un posizionato tramite i CSS all'interno della pagina.

La sintassi dei CSS è molto diversa da quella dell'HTML solito.

Ad esempio per posizionare la testata di HTML.it a 50 pixel dall'alto della pagina e 200 pixel da sinistra, con lo sfondo a righe, bordo nero, e una dimensione di 600 x 80 pixel di è necessario utilizzare questa sintassi.

Nella **<head>**:

```
<style type="text/css">
#logo {
  position:absolute;
  left:200;
  top:50;
  width:600px;
  height:80px;
  background-image: url(sfondo.gif);
  border: 1px solid #000000;
}
</style>
```

nel **<body>**:

```
<div id="logo">
 <img src="logo.gif" alt="HTML.it" title="HTML.it" width="224" height="69">
 <img src="logo1.gif" alt="HTML.it" title="HTML.it" width="59" height="70">
</div>
```

L'esempio completo si trova a questo indirizzo.

Il vantaggio di questa impostazione è quella di avere una pagina molto pulita: infatti il file HTML è composto soltanto dai (o dagli span) con i contenuti, mentre lo stile di visualizzazione dei contenuti è affidato ai fogli di stile (per lo più espressi in un file separato).

Per gli approfondimenti vi rimandiamo alla guida sui fogli di stile di HTML.it che approfondisce questo argomento.

Struttura del tag form

Uno dei fattori che ha decretato il successo del Web è senz'altro la possibilità di interagire: la possibilità cioè di iscriversi a servizi di vario tipo (ad esempio mailing list), ma soprattutto di partecipare a vere e proprie comunità virtuali, come il forum di HTML.it .

Per organizzare questo genere di servizi è necessario raccogliere in qualchemodo i dati dell'utente: per farlo si utilizzano, in maniera molto semplice, i moduli (cioè i form).

L'invio dei dati è solitamente organizzato in due parti:

- una **pagina principale** che contiene i vari campi dei form, che consentono all'utente di effettuare delle scelte, scrivere del testo, inserire un'immagine
- una **pagina secondaria** che viene richiamata dalla principale e che effettua "il lavoro" vero e proprio di processare e raccogliere i dati. Di norma sitratta di una pagina di programmazione che si trova sul server. Può essere un cgi, oppure una pagina asp, php, jsp o altro

Noi ci occuperemo della sola pagina principale, dal momento che il modo incui struttura una pagina di programmazione esula dagli obiettivi della presenteguida.

Name e action

Per creare una pagina con dei moduli, bisogna utilizzare l'apposito tag **<form>**: si tratta di un elemento di blocco, come il **<p>**, quindi il tag **<form>** lasciauno spazio prima dell'apertura e dopo la chiusura.

```
<form name="datiUtenti"action="paginaRisposta.php">
...
</form>
```

Nel caso in non si desideri avere dello spazio superfluo è possibilemodificare i bordi del tag utilizzando i fogli di stile. Con questa semplice sintassi:

```
<form name="datiUtenti" style="border:0px"action="paginaRisposta.php">
```

Come si può vedere, **"name"** serve per indicare il nome del form, **"action"** indica l'URL del programma o della pagina dirisposta che processerà i dati.

Grazie all'**"action"** è anche possibile far sì che i dativengano inviati in e-mail al webmaster (si tratta infatti a tutti gli effetti di un riferimento a un URL). Il codice è questo:

```
<form action="mailto:tuamail@nomeDominio.it?subject=Oggetto predefinito"
enctype="text/plain" method="POST">
```

vedremo in una delle prossime lezioni come utilizzare concretamente questa sintassi.

Method

Quando creiamo un form possiamo scegliere due metodi di invio: **GET**e **POST**.

Con il metodo **GET** la pagina di risposta viene contattata e idati vengono inviati in un unico step. Nell'URL della pagina di risposta potremo allora vedere tutti i parametri nella barra degli indirizzi (piùprecisamente nella "**query string**", cioè nella "**stringa di interrogazione**") secondo questa forma:

```
paginaRisposta.php?nome=Wolfgang&cognome=Cecchin&datiInviati=prova+invio
```

i dati (nella forma **nome del campo = valore del campo**) sono appesi alla pagina dopo il punto interrogativo.

Alcuni server hanno tuttaviahanno delle limitazioni per quel che riguarda il metodo **GET** e non consentono di inviare form con valori superiori a **255 caratteri**complessivi. Il metodo **GET** è dunque particolarmente indicato per form con pochi campi e pochi dati da inviare. La sintassi perl'invio in get è:

<form name="datiUtenti" action="paginaRisposta.php"method="GET">

Nel metodo **POST** invece l'invio dei dati avviene in due stepdistinti: prima viene contattata la pagina sul server che deve processare i dati, e poi vengono inviati i dati stessi. Per questo motivo i parametri noncompaiono nella query string (dunque se non si desidera che i parametri siano mostrati all'utente questo metodo è preferibile).

In questo caso non ci sonolimiti sulla lunghezza dei caratteri. La sintassi è:

<form name="datiUtenti" action="paginaRisposta.php"method="POST">

Enctype (tipo di codifica)

Prima di passare i dati alla pagina di risposta, che si trova sul server, questi vengono codificati dal browser in modo da non poter dare adito ad errori(ad esempio gli spazi vengono convertiti in "+"). Normalmente non è necessario specificare come si vuole effettuare la codifica dei dati, perché è sottintesol'invio di semplice testo.

A volte però, come quando è necessario inviare un'immagine, è tuttavia indispensabile dichiarare espressamente quali dativogliamo inviare. Per farlo è necessario utilizzare l'attributo **"enctype"** (**"encoding type**", cioè **"tipodi codifica"**).

Come dicevamo normalmente non è necessario farlo, perché viene sottintesoquesto tipo di sintassi:

<form name="datiUtenti" action="paginaRisposta.php"enctype="text/plain">

Ma nel caso di invio di immagini dovremo dichiarare:

<form name="datiUtenti" action="paginaRisposta.php"method="post" enctype="multipart/form-data">

target

Grazie all'attributo **"target"** è possibile far aprire i dati del form in una pagina differente rispetto a quella corrente (o in una determinata parte di un frameset):

<form name="datiUtenti" action="paginaRisposta.php" method="get" target="_blank">

Un po' d'ordine: raggruppare i moduli

Per la loro natura di "raccoglitori di informazioni", i moduli tendono a ingigantirsi e diventare lunghissimi. Per questo, con l'HTML 4 sono statiintrodotti dei tag per fare un po' d'ordine all'interno dei form.

Grazie al tag **<fieldset>** possiamo creare dellemacro-aree all'interno dei form, e grazie al tag **<legend>**, possiamo indicare il nome di ciascunamacro-area.

Poniamo ad esempio di dover raccogliere i dati di un utente, raccogliendodati anagrafici, residenza, domicilio e reperibilità sul lavoro.

Possiamo farlo con la seguente sintassi:

```
<form action=ìì>
<fieldset>
  <legend>Datianagrafici</legend>
  <br><br><br>
</fieldset>

<fieldset>
  <legend>Residenza</legend>
  <br><br><br>
</fieldset>

eccetera

</form>
```

che dà:

Datianagrafici

Residenza

come si può vedere vengono creati dei riquadri con un indicazione del tipo dicontenuto.

Un altro tag particolarmente utile - si può utilizzare con ogni tipo di campoche vedremo d'ora in poi - è il tag **<label>**, che permette di indicare un'etichetta per il nome del campo.

Ad esempio:

```
<form action=ìì>
 <fieldset>
  <legend>Datianagrafici</legend>
  <label>Anno di nascita: <input type="text"></label>
 </fieldset>
</form>
```

che dà:

Datianagrafici Anno di nascita: 

oppure (cambiando la posizione del testo):

```
<fieldset>
  <legend>Dati anagrafici</legend>
  <label><input type="text">: anno di nascita</label>
</fieldset>
```

che dà:

Dati anagrafici : anno di nascita

Come si può vedere il campo su cui si vogliono dare delle indicazioni deve essere compreso all'interno del tag **label** stesso.

Il tag Input

Per quel che riguarda i campi dei form il tag più utilizzato è l'**<input>**, che è senza chiusura. Per specificare undeterminato tipo di campo è sufficiente indicare il tipo di input.

Ad esempio:

```
<input type="text">
```

crea un campo di testo.

```
<input type="button">
```

crea un bottone.

I vari **<input>** sono dotati di attributi che consentonodi indicare il tipo di campo, il nome (ad esempio per interagire con JavaScript), e il valore (per lo più il testo visualizzato).

```
<input type="text" name="tuoTesto" value="qui il tuo testo">
```

che dà:

qui il tuo testo

I bottoni (sumbit, reset, button, image)

Non c'è form che si rispetti senza bottone di invio. La sintassi tradizionale per creare un bottone di invio è:

```
<input type="submit" value="invia I dati">
```

Ad esempio:

```
<form action=http://www.html.it target="_blank">
  <input type="submit"value="visita HTML.it">
</form>
```

cioè:

visita HTML.it

Un altro bottone utile è il **"reset"** che - una volta premuto - consente di riportare il form allo stato originario, cancellando ogni cosa scritta finora dall'utente. Ecco un esempio:

```
<form action=ìì>
    <inputtype="text"><br>
    <input type="reset" value="cancella">
</form>
```

cioè

cancella

Esiste infine un tipo di bottone generico, che non esegue nessuna azioneparticolare, ma che può essere ad esempio utilizzato per associare degli eventi tramite JavaScript.

```
<form action=ìì>
    <input type="button"value="bottone generico">
</form>
```

che dà:

Il tag <button>

Con l'HTML 4 è stato introdotto il tag **<button>** cheoffre la possibilità di creare dei bottoni con un aspetto particolarmente ricco.

Il tag **<button>**, a differenza del tag **<input>**, dà la possibilità di inserire il testo del bottone tra l'apertura e la chiusura del tag medesimo. Questo ci consente di specificare anche del codice HTML all'interno del tag.

I bottoni che abbiamo appena visto dovrebbero dunque avere questa forma:

```
<form action=http://www.html.ittarget="_blank">

 <input type="text"><br>

 <buttontype="button">
   bottone generico
```

```
 </button>

 <buttontype="reset">
   cancella
 </button>

 <button type="submit">
   invia
 </button>

</form>
```

cioè:

bottone generico cancella invia

Ed ecco un esempio complesso:

```
<form action=http://www.html.it target=_blank>
 <button name="vai" type="submit">
   invia

   <img src="puntoEsclamativo.gif" width="23" height="67" title="invia adesso" border="1"
vspace="5"
alt="invia adesso" align="middle">
   <b>invia adesso</b>
 </button>
</form>
```

che dà:

invia  **invia adesso**

Grazie all'attributo **"disable"** è infine possibile disabilitare i bottoni.

Es:

```
<input type="submit" value="invia" disabled>
```

o anche:

```
<form action=ìì>
 <input disabled="disabled" type="submit" value="invia">

 <button disabled="disabled" type=submit>
   invia
```

</button>
</form>

cioè:

invia

Il campo image

Il campo **"image"** ci consente di utilizzare come bottoni del form delle vere e proprie immagini e assegnare loro un valore grazie a JavaScript; in questo caso non si tratta propriamente di un bottone ma la funzionalità è la medesima. Ecco il codice:

<form action=http://www.html.it target=_blank>

<input name="invia il modulo" type="image" src="invia.gif" alt="invia il modulo" title="invia il modulo" width="78"
height="38">
</form>

cioè:

come si può vedere, se non si specifica nulla, l'immagine ha valore di submit.
Gli attributi del campo immagine sono molto simili a quelli del tag **<img>**.

Inserire testo (campo testo, textarea, password)

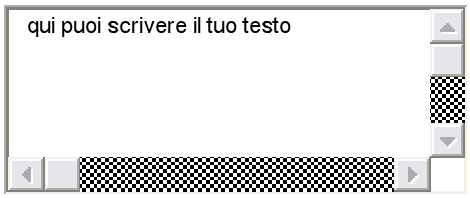
Per consentire all'utente di inserire del testo è possibile **utilizzare un "campo testo"**. Se il campo è su una singola linea avremo:

```
<input name="mioTesto" type="text" value="qui il tuo testo" size="40"
maxlength="200" />
```

L'attributo `maxlength` indica il numero massimo di caratteri che l'utente può inserire, con `size` si esprimono invece le dimensioni del campo di testo (la larghezza è data dal numero di caratteri).

Se si ha la necessità di indicare un campo che consenta di inserire una grande quantità di testo conviene invece **utilizzare una "textarea"** (area di testo). Ecco la sintassi:

```
<textarea name="testo" rows="5" cols="40">
  qui puoi scrivere il tuo testo
</textarea>
```



L'attributo `rows` indica il numero di righe della textarea, `cols` il numero di caratteri (cioè di colonne) che ogni riga può contenere.

Come si può vedere, se si vuol indicare del testo predefinito in questo caso bisogna inserirlo fra l'apertura e la chiusura del tag.

Esiste infine il **campo password** che mostra degli asterischi (o palline) al posto dei caratteri inseriti :

```
<input type="password" maxlength="8" size="18" value="mia_password"
name="mioTesto" />
```

risultato:

Nota: la codifica fornisce una protezione soltanto per chi eventualmente stia sbirciando sul monitor dell'utente. L'invio dei dati attraverso il Web, se non vengono adottate altre misure di sicurezza, avviene 'in chiaro'.

Possiamo prevedere **campi di testo accessibili solo in lettura**. Ad esempio:

```
<input readonly="readonly" maxlength="8" size="25" value="leggere l'informativa"
name="mioTesto" />
```

che risulta:

Oppure possiamo impostare le aree di testo come **campi disabilitati**:

```
<input disabled="disabled" maxLength="8" size="25" value="leggere l'informativa"
name="mioTesto" />
```

cioè

Consentire delle scelte (checkbox, radio, select)

Checkbox

Con le checkbox possiamo consentire all'utente di operare delle scelte multiple. Ad esempio:

```
<form action=ìì>
<fieldset>
 <legend>Linguaggi conosciuti</legend><br>
 <input type="checkbox" name="html" value="html"/> html
 <br>
 <input type="checkbox" name="css" value="css"/> css
 <br>
 <input type="checkbox" name="javascript" value="javascript"/> JavaScript
</fieldset>
</form>
```

che dà:

Linguaggi conosciuti

- ☐ html
- ☐ css
- ☐ JavaScript

Si possono anche selezionare uno o più valori di default:

```
<form action=ìì>
 <input name="html" type="checkbox" value="html" checked="checked"/>
</form>
```

cioè

☑

ed è possibile disabilitare una casella:

```
<form action=ìì>
 <input name="html" type="checkbox" value="html" disabled="disabled"/>
</form>
```

cioè:

☐

Radio button

I **"radio button"** (**"bottoni circolari"**) invece consentono di effettuare una scelta esclusiva. In questo caso quindi una scelta esclude l'altra. Per ottenere questo effetto i campi devono avere lo stesso nome e differente valore:

```
<form>
 <fieldset>
  <legend>Linguaggi conosciuti</legend>
  HTML<input type="radio" name="linguaggio" value="html"/>
  CSS <input type="radio" name="linguaggio" value="css"/>
  JavaScript <input type="radio" name="linguaggio" value="javascript"/>
</fieldset>
</form>
```

che viene così visualizzato:

Linguaggi conosciuti HTML CSS JavaScript

Anche in questo caso è possibile assegnare un valore di default o disabilitare un pulsante.

```
<form action=ìì>
 <input type="radio" name="linguaggio" value="html" checked="checked" disabled="disabled"/>
</form>
```

cioè:

Menu di opzioni (select)

Grazie al tag **<select>** è possibile costruire dei menu di opzioni. In questo caso ciascuna voce deve essere compresa all'interno del tag **<option>** (la chiusura del tag è opzionale) e il valore deve essere specificato attravero l'attributo **"value"**. Con l'attributo **"selected"** si può indicare una scelta predefinita:

```
<form>
 <fieldset>
  <legend>Siti per webmaster</legend>

  <select name="siti" >
   <option value="http://www.html.it" selected="selected">www.html.it  </option>
   <option value="http://freephp.html.it">frephp.html.it  </option>
   <option value="http://freasp.html.it">freasp.html.it  </option>
  </select>
 </fieldset>
</form>
```

che da luogo a:

Siti per webmaseter 

Siccome i menu di scelta tendono a diventare particolarmente lunghi, nell'HTML 4 è stato introdotto il tag **<optgroup>** che consente di suddividere le varie possibilità di scelta in gruppi tramite l'utilizzo di apposite etichette. Ecco l'esempio:

```
<form action=ìì>
 <select name="siti" >
  <optgroup label="siti per webmaster">
   <option value="http://www.html.it">www.html.it  </option>
   <option value="http://freephp.html.it">frephp.html.it  </option>
   <option value="http://freasp.html.it">freasp.html.it  </option>
  </optgroup>

  <optgroup label="risorse per webmaster">
   <option value="http://font.html.it">font.html.it  </option>
   <option value="http://cgipoint.html.it">cgipoint.html.it  </option>
  </optgroup>
 </select>
</form>
```

che dà luogo al seguente menu:

Infine con il tag **select** è possibile impostare anche delle scelte multiple. Come si può vedere, utilizzando l'attributo **"multiple"** l'aspetto del tag select cambia notevolmente:

```
<form action=ìì>
 <label>Quale siti visiti?<br>

  <select name="siti" multiple="multiple">
   <option value="http://www.html.it">www.html.it  </option>
   <option value="http://freephp.html.it">frephp.html.it  </option>
   <option value="http://freasp.html.it">freasp.html.it  </option>
   <option value="http://font.html.it">font.html.it  </option>
   <option value="http://cgipoint.html.it" >cgipoint.html.it  </option>
  </select>
 </label>
</form>
```

cioè:

Quale siti visiti?

Utilizzando il tasto **"ctrl"** l'utente può così effettuare delle **scelte multiple**.

Tramite l'attributo **"size"** si può specificare il numero delle voci che devono comparire nel menu, e conseguentemente regolare l'altezza del menu, aggiungendo o togliendo la barra di scorrimento verticale.

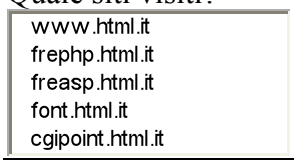
```
<form action=ìì>
 <label>Quale siti visiti?<br>

  <select name="siti" size="5" multiple="multiple">
```

```
   <option value="http://www.html.it">www.html.it  </option>
   <option value="http://freephp.html.it">frephp.html.it  </option>
   <option value="http://freasp.html.it">freasp.html.it  </option>
   <option value="http://font.html.it">font.html.it  </option>
   <option value="http://cgipoint.html.it" >cgipoint.html.it  </option>
  </select>
 </label>
</form>
```

che viene così visualizzato:

Quale siti visiti?



Altri campi (file e hidden)

Potremmo avere la necessità di passare dei parametri "di servizio", senza far percepire la loro presenza all'utente. In questo caso possiamo utilizzare dei campi nascosti, presenti all'interno del form ma invisibili all'utente (ricordiamoci sempre di specificare la coppia **"nome-valore"**):

<input **type="hidden**" name="urlDiProvenienza" value="www.html.it">

Il campo **"file"**, consente invece di inviare un file sul server, nel caso in cui la pagina di risposta sia stata programmata correttamente. La sintassi è:

```
<form action=ìì>
 <input name="fileUtente" type="file" size="20">/
</form>
```

che dà:

"size" indica la larghezza del campo. Come si può vedere, a fianco del modulo compare il pulsante **"sfoglia**" o **"browse"** (a seconda della lingua del browser dell'utente).

Un esempio concreto

Riprendendo un esempio accennato in precedenza, possiamo vedere come sia possibile consentire all'utente di inviarci il contenuto di un questionario tramite e-mail.
Dal punto di vista dell'utente si aprirà un messaggio che domanda se si vuole inviare una mail, ma ciò è inevitabile se si utilizza questo metodo: per evitare questa eventualità bisognerebbe infatti usare dei programmi che inviino e-mail lato-server.

```
<form name="datiUtente" enctype="text/plain" method="post"
action="mailto:tuamail@nomeDominio.it?subject=Questionario proveniente dal web">/
```

```
 <fieldset>
  <legend>Dati Utente</legend>
  <label>Nome: <input name="nome" type="text" size="20" maxlength="30"></label>/
  <label>Cognome: <input name="cognome" type="text" size="20" maxlength="30"></label>/
  <label>Professione: <input name="cognome" type="text" size="20" maxlength="30"/> </label>
 </fieldset>
<br/>
 <fieldset>
  <legend>Questionario</legend>
  <label>Siti visitati:<br>
   <select name="siti" size="5" multiple="multiple">
    <option value="http://www.html.it">www.html.it</option>
    <option value="http://freephp.html.it">frephp.html.it</option>
    <option value="http://freasp.html.it">freasp.html.it</option>
    <option value="http://font.html.it">font.html.it</option>
    <option value="http://cgipoint.html.it" >cgipoint.html.it </option>
   </select>
  </label>
 </fieldset>
<br/>
 <fieldset>
  <legend>Campo libero</legend>
  <label>Lasciami un parere:<br>
   <textarea name="parere" cols="20" rows="10">
   </textarea>
  </label>
 </fieldset>
<br/>
 <button type="reset">annulla</button>
    
 <button type="submit">invia</button>
```

che dà luogo a questo form:

Dati Utente Nome: Cognome: Professione:
Questionario Siti visitati:

www.html.it
frephp.html.it
freasp.html.it
font.html.it
cgipoint.html.it

Campo libero Lasciami un parere:

annulla

Approfondimenti sui form

L'attributo tabindex

Utilizzando il tasto **"tab"** della tastiera l'utente può passare da un campo del form all'altro. Per varie ragioni di impaginazione l'ordine così ottenuto potrebbe però non essere quello desiderato. Grazie all'attributo **"tabindex"** che si applica ai campi dei moduli è possibili specificare in quale ordine deve avvenire il passaggio da un campo all'altro. Il valore di questo attributo può variare tra 0 e 32767. Vediamo un esempio:

```
<form action="datiUtente">
 <fieldset>
  <legend>Dati utente</legend>

  <table width="300" border="1" cellspacing="0" cellpadding="5">
   <tr>
    <td>
     <label>Nome:
      <input tabindex="1" name="nome" type="text" size="30" maxlength="30"/>
     </label>
    </td>
    <td>
     <label>Professione:
      <input tabindex="3" name="professione" type="text" size="30" maxlength="100"/>
     </label>
    </td>
   </tr>

   <tr>
    <td>
     <label>Cognome:
      <input tabindex="2" name="cognome" type="text" size="30" maxlength="30"/>
     </label>
    </td>
    <td> </td>
   </tr>
  </table>
 </fieldset>
</form>
```

che viene così visualizzato:

Dati utente

Nome: **MostraTesto non può essere più lungo di una ri**	Professione: **MostraTesto non può essere più lungo di una ri**
Cognome: **MostraTesto non può essere più lungo di una ri**	

come si può vedere, digitando il tasto **"tab"**, l'ordine di passaggio da un campo all'altro non è quello indicato nell'HTML, ma è modificato secondo il valore di **"tabindex"**.

Il layout dei form

Se siete alle vostre prima pagine HTML, può apparire difficile avere il controllo perfetto dei form. Si trovano validi suggerimenti tra gli articoli correlati alla guida:

- I Form: segreti e trucchi di personalizzazione
- I Form: risposte a domande frequenti

Premessa: il tag object

Se volete inserire file multimediali (audio e video), oppure effetti grafici particolari scritti in qualche linguaggio di programmazione, ricordatevi sempre di fare attenzione al peso dei file che state inserendo. Siamo infatti sul web e dunque **tutti** i file, in un modo o nell'altro, dovranno essere scaricati dal visitatore del vostro sito per essere correttamente visualizzati.

Bisogna anche considerare che - sebbene la banda a disposizione del pubblico si stia allargando grazie all'adsl e alle fibre ottiche - non tutte le zone sono "coperte" da questa tecnologia, e la maggioranza dei visitatori naviga ancora con modem analogici da 56 Kb.
In ogni caso - banda larga o no - inserire ad esempio un file mp3 da 3 Mb come musica di sottofondo sarebbe esagerato anche per una connessione adsl.

Quindi: **attenzione al peso dei file che inserite!**. Per approfondimenti vi consigliamo la lettura della lezione dedicata alla leggerezza dei siti Web della nostra guida all'Usabilità.

Nelle lezioni successive ci occuperemo di come includere in pagine Web elementi multimediali o file di scripting. Per chi fosse interessato alle inclusioni di file HTML in file HTML rimandiamo all'articolo «Come includere codice esterno nelle pagine web». Per chi volesse invece includere del codice XML in pagine HTML vi consigliamo la lettura dell'articolo: «Data binding client-side con XML Data Islands».

La maggior parte dei file multimediali che vedremo nel corso delle lezioni si inserisce all'interno delle pagine con il tag **<object>**, che è il tag corretto - secondo le indicazioni del W3C - per inserire elementi multimediali, tanto che nelle specifiche dell'XHTML 2 (l'evoluzione dell'HTML) persino le immagini devono essere inserite tramite questo tag.

Un altro tag che spesso viene utilizzato per la multimedialità è **<embed>**: si tratta di un elemento che non è nelle specifiche del W3C, ma che è stato a lungo utilizzato, perché supportato sia da Internet Explorer, sia da Netscape Navigator, a differenza di <object>, che ha dei problemi di compatibilità.

Vediamo i principali attributi di <object>:

data	questo attributo può essere utilizzato per specificare il percorso dell'oggetto da inserire nella pagina
classid	dà indicazioni sul percorso dell'oggetto, ed è utile per identificare il tipo di plugin con cui eseguire l'oggetto
codebase	serve per indicare l'URL di base, a cui il codice indicato in **"data"** o in **"classid"** fa riferimento
type	è il tipo di oggetto da inserire (più esattamente è il **MIME type** dell'oggetto)
archive	si può indicare una lista di URL, separati da virgola, contenti risorse relative all'oggetto inserito
width, height	se necessario, si possono indicare una larghezza e una altezza

All'interno del tag <object> è possibile specificare una sintassi alternativa per i browser che non leggono questo tag. Inoltre all'interno del tag è possibile specificare eventuali parametri necessari all'esecuzione dell'oggetto.

In molti casi il tag object si occupa di attivare un "plug-in", cioè un componente aggiuntivo che si integra nel browser, per lo più fornito dal produttore del software multimediale (es. Flash), in grado di leggere il file multimediale (qualsiasi esso sia).

Vedremo nelle lezioni seguenti quali sintassi specifiche utilizzare per includere questi oggetti nelle vostre pagine Web.

Includere un file Audio

Per impostare un suono di sottofondo si può utilizzare il tag `<bgsound>`

Sintassi di <bgsound>

<bgsound src="*url del file audio*" loop="*numero di ripetizioni*" />

Basta quindi inserire il riferimento del file audio (es. wav) e lasciare che il suono venga riprodotto:

- una volta: `loop="1"` oppure omettendo l'attributo `loop`
- due volte o anche di più: `loop="2"` basta indicare il numero di ripetizioni desiderato
- infinite volte: `loop="infinite"`

In realtà questo non è il modo migliore, né il più efficace, per inserire un file audio, perché obsoleto e **compatibile soltanto con Internet Explorer**.

Un'altro modo obsoleto, ma crossbrowser prevede l'uso di <embed>.

Sitassi di <embed>

<embed src="*url del file audio*" autostart="true" />

Un esempio in questa pagina.

Se invece vogliamo seguire le specifiche dell'HTML 4.01, utilizziamo il tag **<object>**.

Sintassi di <object>

```
<object data="url del file audio" type="audio/wav" autostart="true">
  <embed src="url del file audio" autostart="true">
</object>
```

tuttavia, affinché tutto funzioni perfettamente, spesso conviene indicare il tipo di plugin da utilizzare grazie all'attributo `classid`.

Vediamo il codice necessario ad aprire la barra del lettore multimediale **RealOne** (se RealOne non è presente, l'utente verrà avvisato). Notare che all'interno del tag `<object>` vengono espressi i parametri che indicano come devono essere visualizzati i controlli di RealOne, e la sintassi alternativa per browser obsoleti (indicata tramite `<embed>`):

```
<object id="sound1" classid="clsid:22D6F312-B0F6-11D0-94AB-0080C74C7E95">
  <param name="src" value="multimedia/audio/jet_pubb.wav">
  <param name="controls" value="All">
  <param name="console" value="sound1">
  <param name="autostart" value="false">

  <embed src="http://html.it/guide/esempi/guida_html/multimedia/audio/jet_pubb.wav"
     type="audio/wav" console="sound1" controls="All" autostart="false" name="sound1">

</object>
```

Quando inserite dei file audio, fate sempre caso ad essere i proprietari dei diritti d'autore, o ad avere l'autorizzazione a utilizzare la musica (o il suono) in questione.

Ricordiamo anche che l'uso di suoni o musica di sottofondo se utilizzata in modo eccessivo può compromettere l'usabilità del sito.

Includere un file Video

Come per l'audio anche per i video si dovrebbe utilizzare il tag **<object>**. La sintassi è identica a quella dei file audio:

```
<object data="filmato.mov" type="video/quicktime" width="164" height="140">
 <embed src="filmato.mov" type="video/quicktime" width="164" height="140">
</object>
```

come abbiamo visto per i filmati audio, è possibile utilizzare l'attributo **"classid"** con gli appropriati parametri per aprire barre di visualizzazione e controlli vari ed eventuali.

Data la grande dimensione che questi file possono raggiungere, per i filmati (ma anche per l'audio) è possibile utilizzare lo **streaming attraverso il web**. Si tratta di un procedimento che consente di scaricare il filmato gradualmente dal server, a mano a mano che lo si sta vedendo e in modo del tutto trasparente all'utente.

Per maggiori informazioni sullo streaming video, vi rimandiamo alla sezione «streaming» di PRO.HTML.it.

Finora abbiamo visto come inserire singoli filmati audio e video all'interno di una pagina web, se invece volete sincronizzare diversi filmati audio e video, dovete utilizzare un linguaggio apposito che prende il nome di «SMIL».

Includere un file Flash

Flash è un software molto potente sviluppato da **Macromedia** a partire dal 1996. Si tratta di un programma che utilizza **grafica vettoriale**: significa che le immagini non vengono descritte attraverso mappe di bit (bitmap), ma attraverso formule matematiche. Per questo motivo i filmati in Flash pesano molto meno della grafica tradizionale, e per lo stesso motivo i colori sono "piatti".

Inoltre, proprio perché le immagini sono espresse da formule matematiche, non c'è il problema dell'effetto "sgranato" dovuto al ridimensionamento proprio della grafica tradizionale, perché i filmati in flash si adattano automaticamente alle dimensioni indicate nel codice HTML (o se le dimensioni sono espresse in percentuale, si adattano alla pagina).

Ovviamente il fatto che le immagini siano espresse grazie a delle formule è una caratteristica di questo software che non tocca minimamente il webmaster, il quale - quando sviluppa filmati in flash - si ritrova ad usare un "normale" software visuale.

I file sorgenti dei filmati (quelli su cui gli sviluppatori lavorano) hanno estensione **.fla**, i file compilati (quello che si possono vedere in giro per il web) hanno invece estensione **.swf** (cioè **"Shockwave Flash"**): è quest'ultimo tipo di file che dovremo inserire dunque nelle nostre pagine HTML.

Per inserire un filmato in flash in una pagina HTML è sufficiente utilizzare la seguente sintassi:

```
<object type="application/x-shockwave-flash"
data="http://html.it/guide/esempi/guida_html/multimedia/flash/bottone_in-out.swf" width="450"
height="164">
<param name="movie"
value="http://html.it/guide/esempi/guida_html/multimedia/flash/bottone_in-out.swf"/>
</object>
```

Maggiori informazioni su Flash si possono trovare sul nostro sito dedicato a Flash.

Includere un file Java

Java è un linguaggio di programmazione rilasciato dalla **Sun** nel 1995, ma in corso di sviluppo sin dal 1991. Il suo successo nel web è dovuto alle famose **"applet"** (fusione delle parole "application" e "gadget") che permettono di aggiungere interattività alle pagine web.

Le applet Java hanno tracciato la strada verso una migliore **esperienza utente** per i siti Web. Sono le antesignane dei diversi plugin come il Flash Player, Silverlight e del canvas di HTML 5.

Tutto quello che si può fare con le applet Java oggi lo si può fare anche con Flash o JavaScript e in molti casi si preferisce utilizzare questi linguaggi.

Java rimane una valida alternativa per applicazioni web complesse (ad esempio i dati e i grafici delle quotazioni finanziarie, che devono essere aggiornati in tempo reale), o anche per applicazioni come la chat (i client in Java sono tuttora insuperati).

I file con il codice sorgente hanno estensione **.java**, i file compilati (da inserire nelle nostre pagine web) hanno invece estensione **.class**.

Come per tutti i plugin oggi possiamo inserire un applet Java grazie al tag `<object>` ed una sintassi simile a questa:

```
<object id="appletLake"          codetype="application/java"
codebase="applet_dir/"          width="500" height="400" >    <param
name="image" value="myimage.jpg">      <!--html alternativo -->   <img
src="http://html.it/guide/esempi/guida_html/multimedia/java/logo.gif"
width="263" height="65" alt="html.it">   <!-- fine html alternativo -->
</object>
```

che dà questo output: ???

Diamo uno sguardo alle proprietà che abbiamo inserito:

Proprietà	Descrizione
`codetype`	Stabilisce il tipo di oggetto (nel nostro caso `application/java`) che stiamo inserendo
`codebase`	Indica l'indirizzo della cartella che contiene il file `.class` della nostra applet
`width` e `height`	Sono le dimensioni (larghezza e altezza) del box in cui eseguire l'applicazione

Oltre alle proprietà necessarie a definire l'applicazione da caricare, possiamo utilizzare anche un tag `<param>` per passare dei valori all'applet. Nel nostro caso le passiamo l'url di un'immagine da visualizzare.

Infine, per coloro che non hanno installato il plugin per la visualizzazione delle applet, abbiamo inserito del markup HTML alternativo, per mostrare comunque l'immagine.

Nota: In passato si utilizzava il tag <applet>, che però, è stato da tempo deprecato dal W3C e può causare anche problemi di visualizzazione su alcuni browser.

Una raccolta di applet Java è presente sul sito Java.HTML.it.

Includere file di scripting o CSS

JavaScript e VbScript

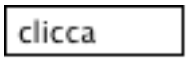
Cominciamo subito con il dire che JavaScript non è Java. JavaScript è un linguaggio di scripting, eseguito dal browser, che permette di creare interattività all'interno della pagina.
La sintassi JavaScript deve essere inserita all'interno del tag **<script>**. Così:

```
<script type="text/javascript"> function ciao() {   alert ("ciao"); } </script>
```

e poi nella pagina:

```
<input type="button" value="clicca" onClick="ciao()">
```

che dà il seguente output:

HTMLDirect 

Per imparare a programmare in JavaScript è possibile iniziare con la nostra guida. Inoltre su HTML.it è disponibile una numerosa raccolta di JavaScript pronti da usare così come sono, o utili per prendere spunto e costruire nuove funzioni.

VbScript ("Visual Basic Script") è anch'esso un linguaggio di scripting eseguito dal browser, ma è possibile utilizzarlo soltanto con Internet Explorer.

I CSS (i fogli di stile)

Infine i fogli consentono di impostare il layout di un documento. La sintassi per includerli all'interno della pagina è:

<style type="text/css">
...
</style>

Anche in questo caso vi rimando alla nostra guida ai CSS.

I meta tag

Adesso che abbiamo terminato il nostro sito possiamo occuparci di farlo trovare dai motori di ricerca.

È utile allora impostare correttamente i meta tag all interno della **<head>** del documento: si tratta di una serie di parole chiave e descrizioni, che aiutano i motori di ricerca a classificare il sito.

Abbiamo già visto il **<title>**, che è il titolo della pagina; ma il testo ivi contenuto può comparire anche in seguito alla ricerca in un motore, come titolo del link. Sarà dunque importante impostarlo in modo pertinente:

<title>HTML.it - il sito italiano sul webpublishing</title>

C'è poi il **meta-tag "description"** che permette di impostare una descrizione sintetica del sito stesso. Anche in questo caso, la descrizione compare talvolta nei risultati della ricerca:

<meta name="description" content="HTML.it - il sito italiano sul Web publishing">

Infine il meta-tag "**keywords"** permette di indicare alcuni contenuti relativi al sito stesso.
Le keywords (a seconda del webmaster) compaiono separate da virgola, da punto e virgola, oppure senza alcun segno di interpunzione:

<meta name="keywords" content="html wml xml smil javascript js dhtml dynamic xhtml vbscript coldfusion photoshop paint shop pro risorse webmaster webdesigner flash grafica css applet java asp cgi perl guida free corso php mysql tutorial lezioni sql database realizzazione siti web leggi mailing list newsletter gif jpg publishing editor iis webserver apache linux raccolte script news chat forum fogli di stile hdml wap linux mac apple palmari computer c++ delphi visual basic vb vbasic">

È fortemente sconsigliabile l'inserimento di keyword "astute" non relative al contenuto effettivo del sito per migliorare il posizionamento (tipo le ricercatissime "sesso", "mp3", ecc.). Quando i motori di ricerca se ne accorgono, per lo più cancellano il sito dalle loro liste.

Su HTML.it sono presenti molte risorse sull'argomento. L'articolo «I Meta-tag: come scriverli correttamente», ad esempio, è un approfondimento su come impostare i meta-tag.

Un buon posizionamento all'interno dei motori di ricerca è una meta difficile da raggiungere e l'argomento non si esaurisce certo in poche righe. Per cui, se siete interessati all'argomento, è utile consultare i nostri articoli sui motori di ricerca e il topic "Motori di Ricerca e Web Marketing" sul nostro forum.

Il DocType (DTD)| Guida HTML | HTML.it

Finora abbiamo tralasciato l'analisi della prima riga di una pagina HTML (quella che consente di specificare di che tipo di documento si tratta). Il **<!DOCTYPE>** assume un aspetto di questo genere:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
http://www.w3.org/TR/html4/loose.dtd>
```

Questa riga fornisce alcune informazioni sul documento:

Elemento	Descrizione
HTML	il tipo di linguaggio utilizzato è l'HTML
PUBLIC	il documento è pubblico
W3C	il documento fa riferimento alle specifiche rilasciate dal W3C
-	(è il segno "meno") le specifiche non sono registrate all'ISO (organizzazione di standardizzazione nternazionale). Se lo fossero state, ci sarebbe stato un "+"
DTD HTML 4.01 Transitional	il documento fa riferimento a una DTD ("Document Type Definition" cioè "Definizione del tipo di documento"); la versione di HTML supportata è la 4.01 "transitional"
EN	la lingua con cui è scritta la DTD è l'inglese

Inoltre, se necessario, è possibile specificare l'indirizzo di riferimento a cui è possibile trovare la DTD: per l'HTML non lo si fa quasi mai, perché gli URL a cui trovare la documentazione sono universalmente noti.

Per quel che riguarda l'HTML le indicazioni possibili sono tre:

- **Strict**: è una DTD particolarmente rigorosa: esclude ogni elemento che riguarda il layout (la cui formattazione è affidata all'utilzzo dei CSS) e non è consentito l'uso degli elementi deprecati:

  <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Strict//EN"
  http://www.w3.org/TR/html4/strict.dtd>

- **Transitional**: è una versione temporanea, per consentire il passaggio da una specifica all'altra. Nella DTD transitionali tag deprecati sono ammeesi. Questa DTD andrà bene nella maggior parte dei casi:

  <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  http://www.w3.org/TR/html4/loose.dtd>

- **Frameset**. È la DTD che riguarda i frames:

  <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"
  http://www.w3.org/TR/html4/frameset.dtd>

Nelle ultime versioni il tipo di `<!DOCTYPE>` utilizzato influisce sulla visualizzazione della pagina da parte del browser. Tale tecnica, chiamata **<!DOCTYPE> switch**, è una delle principali cause di visualizzazione delle pagine sul Web. A questo argomento HTML.it ha dedicato un lungo e dettagliato approfondimento nell'articolo «Il <!DOCTYPE> ed il <!DOCTYPE> switch nei moderni browser»

Configurare un programma FTP

Abbiamo detto sin dall'inizio che un sito internet, per essere visibile da tutti, deve essere presente su un computer che faccia da server, che sia in grado cioè di distribuire i contenuti di un sito a chi ne faccia richiesta nel web.

È giunto il momento di spostare il nostro sito internet sul server e per farlo dobbiamo utilizzare un protocollo che si chiama **FTP ("file transfer protocol")**.

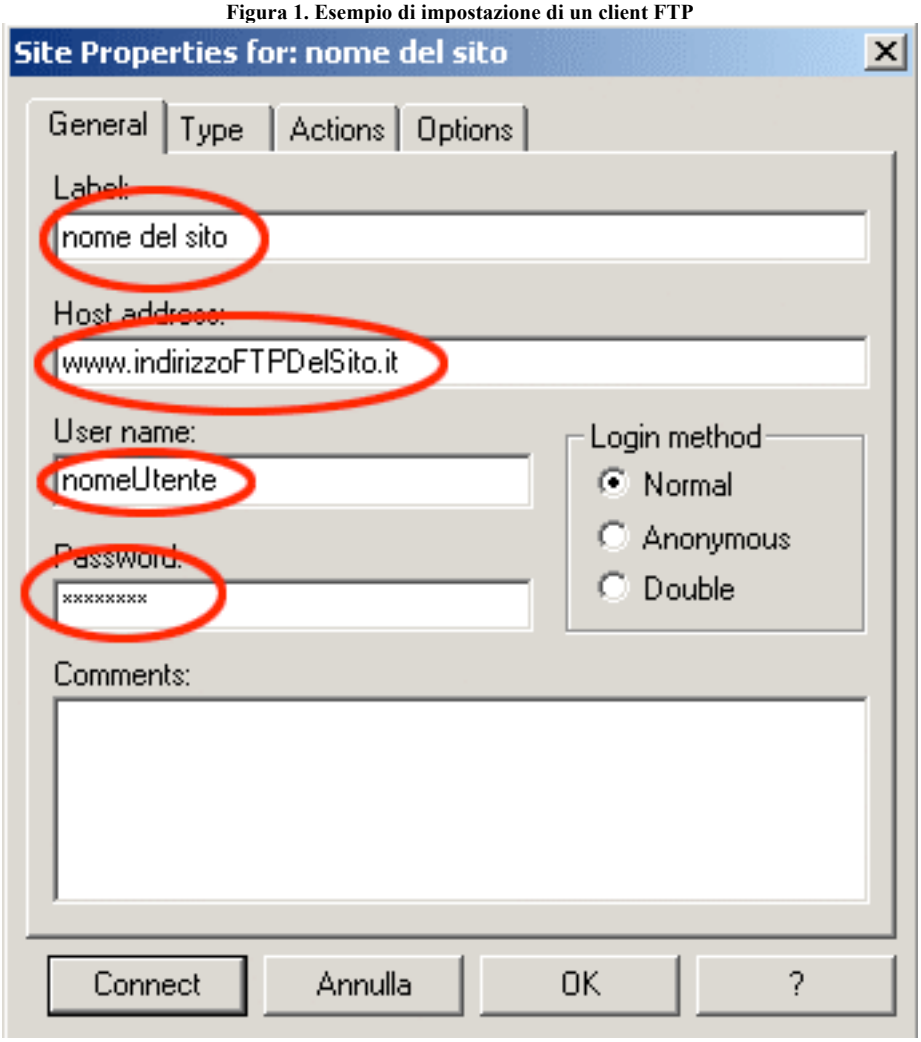
Per prima cosa, cercate uno spazio web in cui trasferire il vostro sito (ce ne sono di gratuiti, ad esempio su Digiland). Dobbiamo quindi procurarci un programma che ci consenta di trasferire i file in maniera visuale dal nostro computer al computer in remoto. Nella sezione «Download» ce ne sono alcuni: i più famosi sono CuteFtp e WS-Ftp.

Durante la configurazione del programma dovrete inserire alcuni dati che vi fornirà il gestore dello spazio:

- indirizzo ftp del sito
- nome utente
- password

Come evidenziato in questa schermata:

Figura 1. Esempio di impostazione di un client FTP



Una volta che effettuata correttamente la procedura di login, non avete che da spostare i file dal vostro computer (di solito alla vostra sinistra) al computer in remoto (a destra).

Se avete impostato i collegamenti in modo corretto - in modo che non facciano riferimento al vostro computer di casa, ma a riferimenti relativi - avete messo il vostro primo sito nel Web!

Gli editor visuali

Finora abbiamo scritto tutto il codice a mano, ma vi accorgerete presto che esistono dei programmi che permetto di inserire immagini, tabelle, frame, form e quant'altro in maniera più intuitiva: si tratta degli editor visuali, quelli che gli anglosassoni chiamano editor **WYSIWYG** (**"What you see is what you get"**, che significa "ciò che vedi è quello che ottieni").

Ad oggi gli editor visuali più utilizzati sono:

- **Dreamweaver** della Macromedia: un editor molto potente e pieno di funzionalità, ma forse proprio per questo inizialmente difficile da usare. Sicuramente il migliore.
- **FrontPage** della Microsoft: è l'editor che tutti solitamente utilizzano, perché incluso nel pacchetto Office. In realtà "sporca" molto il codice, visto che la sua attenzione è concentrata su Internet Explorer.

- **Golive** di Adobe: negli ultimi anni ha perso notevoli quote di mercato, rimane tuttavia un editor serio e una valida alternativa a FrontPage

Una precisazione sul WWW e sui linguaggi

Il WWW (Word Wide Web) come lo conosciamo oggi fu inventato da **Tim Berners Lee** al Cern di Ginevra nel 1991. Egli inventò sostanzialmente tre procedimenti standard grazie ai quali far colloquiare gli elaboratori fra loro:

- **HTTP** ("Hyper Text Tranfer Protocol"): è il protocollo grazie a cui due computer differenti si scambiano le informazioni
- **URI** ("Uniform Resource Identifiers") e **URL** ("Unified Resource Locator"): sono due sistemi per individuare in modo univoco la collocazione di una determinata macchina, di un determinato documento o di una determinata risorsa all'interno del Web. Un esempio di URI è l'indirizzo web http://www.html.it.
- **HTML**: un linguaggio standard. Oramai dovrebbe essere chiaro di cosa si tratta

Linguaggi di markup

A scanso di equivoci, ecco qui una piccola panoramica dei linguaggi che sono strettamente parenti dell'HTML:

- **SGML** ("Standard Generalized Markup Language"). in pratica l'HTML è stato scritto seguendo le specifiche di questo linguaggio. L'SGML serve infatti per creare linguaggi di contrassegno. La maggior parte di noi non se ne occuperà mai, ma l'SGML ha fornito la struttura per creare l'HTML
- **XML** (Extensible Markup Language). L'XML (modellato anch'esso sull'SGML) è nato per superare i limiti dell'HTML: è infatti possibile creare dei tag personalizzati, che si adattino ad ogni esigenza. Questa caratteristica facilita l'interscambio dei dati tra piattaforme differenti grazie all'uso dell'XML.
  In pratica si tratta di un meta-linguaggio, in grado di creare altri linguaggi, adattabili per le esigenze più disparate.
  Es: il WML (Wirless markup Language) per i telefonini, MathML per descrivere espressioni matematiche, l'SVG (Scalable Vector Graphics) per la grafica vettoriale, lo stesso XHTML
- **XHTML** ("Extensible HyperText Markup Language"): l'XHTML non è nient'altro che la riformulazione dell'HTML come linguaggio XML. Infatti - dopo l'HTML 4.01 - non ci saranno più nuove versione dell'HTML, perché l'HTML si è evoluto in XHTML

Conclusioni

A questo punto dovreste essere in grado di costruire i vostri siti con le vostre stesse forze.

È una buona abitudine quella di validare il vostro codice HTML, per vedere se fate degli errori. Per farlo potete utilizzare lo stesso validatore del W3C (che è il più noto), ma ne esistono anche altri.

Se costruire un sito per voi è un'occasione sporadica, questo corso dovrebbe essere abbastanza esauriente per tutto quello che dovete fare. Ma se "da grandi" volete fare i Webmaster, il prossimo passo da affrontare è la «guida ai fogli di stile».

Se poi volete aggiungere effetti grafici particolari ai vostri siti web, potete anche dedicarvi alla «guida a JavaScript».

In ogni caso nel sito HTML.it potete trovare dei validi suggerimenti per i vostri prossimi passi.

Se avete dei dubbi sul codice, potete sempre cercare aiuto nel «forum di discussione di HTML.it».