

Sommario

1. Prima di cominciare...	3
2. <HEAD> intestare il documento	5
<HTML>	5
DOCTYPE	5
<HEAD>	6
<TITLE>	6
META	7
3. Scrivere correttamente i meta tag	9
4. <BODY> Impostare lo sfondo del documento	11
<BODY>	11
BGCOLOR	11
BACKGROUND	11
TEXT	12
LINK	12
ALINK	12
VLINK	12
BGPROPERTIES	12
5. Inserire suoni o musiche di sottofondo	13
6. carattere e grandezza del testo	15
Tag di testo logici e fisici	15
Tag FONT e nuovo standard HTML 4	15
Formattare titoli da <H1> ad <H6>	15
	16
, <I>, <U>	16
<STRIKE>	17
<SUP> e <SUB>	17
Testo preformattato <PRE>	17
Breve descrizione degli stili logici	17
<ADDRESS>	17
<BLOCKQUOTE>	18
<CITE>	18
Usato per indicare la fonte della citazione.	18
<CODE>	18
<DFN>	18
	18
<KBD> e <SAMP>	18
	18
<VAR>	18
7. Paragrafi e giustificazione del testo	19
Creare paragrafi con <P>	19
Andare a capo con 	19
Posizionare il testo con <DIV ALIGN> e <CENTER>	19
Linee orizzontali con <HR>	19
8. Creare elenchi puntati e numerati	21
Elenchi ordinati (numerati): e 	21
Elenchi NON ordinati (puntati): e 	22
9. inserire immagini nel documento	23
	23
ALT	23
WIDTH e HEIGHT	23
BORDER	24
HSPACE e VSPACE	24
ALIGN	24
10. <HREF> link verso documenti o immagini	25
<A HREF>	25
TARGET	25
TITLE	25
MAILTO (link a e-mail)	26
Link interni al documento	26

11. <TABLE> creare e gestire Tabelle HTML	27
<TABLE> <TR> e <TD>	27
12. Un modo nuovo per formattare i documenti: i CSS.....	30
13. <FRAME> creare pagine con i Frame.....	31
14. <FORM> creare e gestire moduli HTML.....	37
<FORM></FORM>	37
<INPUT>.....	37
Form anche senza l'ausilio di uno script lato server	41
15. <MAP> creare mappe cliccabili	42
Mappe dal lato server (ISMAP)	42
Mappe dal lato client (USEMAP)	42
16. <APPLET> inserire Applet Java nel documento	44
Gli errori più comuni	44
17. Le nuove specifiche di HTML 4	45

1. Prima di cominciare...

Una premessa prima di cominciare a parlare di HTML

Creare un sito Internet non è difficile, né può essere messo a confronto con veri e propri sistemi di programmazione. Su quest'ultimo punto, in particolare, è bene fin d'ora sottolineare come HTML non sia un linguaggio di programmazione ma un semplice sistema di contrassegno, i cui tag vengono riconosciuti ed interpretati dai browser Web (Netscape, Explorer, Opera, ecc). Questa peculiarità rende HTML un sistema facile da comprendere perché non presuppone alcuna conoscenza tecnica preesistente. Per questo motivo chiunque può avvicinarsi al Web Publishing senza prerequisiti di specializzazione e per questo in Internet esistono milioni di creatori di pagine Web.

Il mercato è popolato di decine di software specializzati nella creazione di pagine HTML: i cosiddetti editor HTML. La complessità di questi programmi ha raggiunto livelli molto alti ma ha seguito strade diverse. In buona sostanza il mercato propone due tipi di editor HTML:

- **Editor testuali**

Si tratta di programmi che propongono modifiche dirette sul codice HTML puro con possibilità di preview del risultato. Per facilitare la scrittura del codice mettono a disposizione comandi preconfezionati attivabili con semplici click. L'utilità maggiore di questi strumenti è la padronanza del codice HTML puro che riescono a comunicare agli utilizzatori. Il difetto maggiore è nell'apprendimento più difficile rispetto agli editor WYSIWYG.

- **Editor WYSIWYG**

L'acronimo sopraindicato sta per: What You See Is What You Get, ed italianizzando significa che ciò che si vede sullo schermo è ciò che si ottiene in un browser Web. In altre parole questi editor non visualizzano il codice HTML ma esclusivamente gli oggetti, le immagini ed il testo. In questo modo lo sviluppatore non si trova a lavorare su codice HTML ma sulla pagina così come verrà visualizzata dal browser: spostando oggetti con il semplice trascinarsi del mouse, sfruttando layout predefiniti ecc.

Il vantaggio di questo tipo di editor è evidente: lo sforzo di comprensione delle specifiche HTML è ridotto al minimo e il tempo di apprendimento è brevissimo.

Secondo chi scrive, però, questo tipo di editor (un esempio concreto è Front Page) non dà la possibilità di comprendere l'HTML e personalizzarne l'uso. Si corre, infatti, il rischio di creare siti fotocopia con un layout identico agli altri e, in presenza di errori che l'editor non riesce a risolvere, di non riuscire ad andare oltre ciò che il programma prevede.

A prescindere da pareri soggettivi su programmi per l'HTML, la presente guida è concepita per far comprendere i rudimenti dell'HTML puro. Continuando questo corso non ti troverai di fronte un tutorial sui comandi degli editor più diffusi, ma davanti codice HTML puro ed universale. Solo in questo modo potrai comprendere le peculiarità dell'HTML e degli altri linguaggi che ne supportano e ampliano l'utilizzo.

Abbandona per tutto il corso di questo tutorial il tuo editor e concentrati sulle nozioni di codice che la guida ti presenta volta per volta. Comprese le basi dell'HTML potrai scegliere se usare editor WYSIWYG o editor testuali, in entrambi i casi avrai consapevolezza dei limiti e delle potenzialità del sistema, potendo usare meglio tutti gli strumenti che il mercato mette a tua disposizione.

Alcuni consigli per comprendere pienamente questa guida:

- I siti Web, prima della pubblicazione, vanno creati in locale, cioè su hard disk del computer di produzione. Solo successivamente vengono pubblicati su Internet, memorizzati sull'hard disk di un Server Web, sfruttando programmi appositi.
- Per i nomi dei file e per le estensioni degli stessi usa caratteri sempre in minuscolo, non lasciare spazi tra le parole, piuttosto tramutali in "_". Esempio: **nome_del_file.htm**
- La modifica del tuo sito Web avviene tramite programmi FTP (File Transfer Protocol) mediante password e UserId. Quindi nessun altro, oltre colui che dispone di appropriate password e UserId, può modificare le pagine.
- Per scrivere codice HTML puoi utilizzare qualsiasi editor HTML testuale (HomeSite, HotDog, DreamWeaver, FrontPage, ecc.), ma anche solo Blocco Note di Windows.

Prima di continuare è importante sottolineare una caratteristica dei tag HTML. Come più volte accennato, HTML è un sistema di contrassegno che produce i suoi effetti su testo e immagini. Perché tali effetti si producano è necessario che elementi specifici (standardizzati in HTML) vengano assegnati ai vari elementi testuali o d'immagine della pagina. Il termine **marcatore** deriva proprio da questa caratteristica. Per una migliore comprensione di questo fondamentale concetto di HTML prendiamo in considerazione il testo seguente:

HTML è un sistema di contrassegno indipendente dalla piattaforma.

Si tratta di un normalissimo testo per il quale non abbiamo specificato alcun tag HTML. In linea con i concetti che reggono il sistema di markup, senza altre indicazioni, il browser formatta il testo come quello che lo precede (infatti si tratta di un font verdana di dimensioni e colore identici a questo stesso testo).

Consideriamo, ora, l'ipotesi di voler dare un colore rosso al testo di esempio. Per fare questo marchiamo il testo con lo specifico tag HTML che determina il colore del testo:

```
<font color="red">HTML è un sistema di contrassegno indipendente dalla piattaforma.</font>
```

Il browser visualizzerà:

HTML è un sistema di contrassegno indipendente dalla piattaforma.

In seguito vedremo il funzionamento del tag . Quello che interessa sottolineare in questa sede è l'uso generale dei marcatori HTML: come si nota agevolmente dall'esempio, il tag **** è posto immediatamente prima e dopo la frase da formattare in rosso. L'unica differenza è che il tag prima della frase è ****, mentre quello che chiude è ****. La barra posta in verticale verso destra indica al browser che il tag **** precedentemente aperto, in questo caso va chiuso. Abbiamo così dato vita ad un piccola marcatura, comunque sintomatica del funzionamento del sistema HTML.

I tag HTML sono **case insensitive**, cioè assolutamente indipendenti da maiuscolo e minuscolo. Per intenderci, il seguente codice:

```
<font color="red"></font>
```

ha gli stessi effetti del codice:

```
<FONT COLOR="RED"></font>
```

In HTML esistono tag che non hanno bisogno di chiusura, perché la loro funzione non è quella di marcare un elemento, ma di fornire informazioni di diverso tipo. Volta per volta indicheremo quali elementi hanno bisogno di chiusura e quali funzionano correttamente anche se ne sono privi.

2. <HEAD> intestare il documento

Come creare da zero un documento HTML; dare un titolo alla pagina e cosa sono e come impostare i meta tag.

Per essere visualizzati su Internet, i documenti vengono salvati in formato testo e contengono i tag necessari ad informare il browser (Netscape, IExplorer o Microsoft Internet Explorer) sulla visualizzazione della stessa. In altri termini un documento contenente testo, salvato in formato .htm senza alcun tag HTML viene comunque visualizzato dal browser, ma privo di qualsiasi formattazione: senza ritorni a capo, paragrafi, testo centrato, grassetto, corsivo ecc.

Lo scopo dell'HTML è quello di fornire, attraverso comandi chiamati TAG , una formattazione al documento, oltre all'inserimento di immagini ed altri elementi multimediali (filmati, applet ecc.). Il lavoro che uno sviluppatore Web produce all'interno di un documento HTML è indirizzato a fornire tutte le informazioni necessarie al browser per interpretare correttamente la pagina.

Un documento HTML si divide in due parti fondamentali: l'intestazione e il corpo del documento. È facile comprendere che il corpo del documento contiene tutti gli elementi della pagina: il testo, le immagini, le applet Java, il codice Javascript e quant'altro viene materialmente visualizzato dal browser.

Al contrario, l'intestazione contiene una serie di informazioni necessarie al browser per una corretta interpretazione del documento, ma non visualizzate all'interno dello stesso. L'intestazione, quindi, ha un ruolo non apparente ma sicuramente fondamentale. Solo per citare alcuni elementi forniti dall'intestazione: il titolo della pagina, i termini chiave per i motori di ricerca, il tipo di HTML supportato ed i link base di riferimento.

In questa sede analizzeremo solo alcuni elementi, omettendo per il momento lo studio degli altri non immediatamente necessari alla comprensione di HTML:

- HTML
- DOCTYPE
- HEAD
- TITLE
- META

In seguito dettaglieremo per ognuno di questi elementi le peculiarità salienti.

<HTML>

Tutti gli elementi ed il contenuto di un documento HTML sono compresi all'interno dei marcatori <HTML></HTML> che, in altre parole, hanno il compito di aprire e chiudere il file. I tag <HTML></HTML> indicano al browser che il documento è marcato in HTML, anche se i browser più recenti (Netscape 3 e successivi, IExplorer 3 e successivi) riescono ugualmente ad interpretare i tag successivi. Detto questo esistono comunque due ragioni per inserire sempre il tag <HTML></HTML> all'interno del documento:

- HTML non è l'unico linguaggio di contrassegno presente sul WWW (si pensi solo a XML) e il browser rischia di male interpretare i tag, confondendoli con altri linguaggi di markup
- Gli utenti che usano vecchi browser rischiano di visualizzare un documento pessimamente formattato.

DOCTYPE

Seguendo le indicazioni del W3C (consorzio internazionale che si occupa di armonizzare l'utilizzo dell'HTML) Doctype deve essere il primo elemento ad aprire il documento. Questo vuol dire che andrebbe posto prima di <HTML>.

Si tratta di un tag che non ha bisogno di chiusura e che ha il compito di fornire informazioni al server Web che ospita la pagina. Le informazioni fornite da DOCTYPE riguardano il tipo di documento visualizzato oltre ad essere necessaria alla comunicazione tra browser e server. DOCTYPE deve essere scritto in una forma standard:

```
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 4.0//EN">
```

Questa riga fornisce alcune informazioni sul documento:

- **HTML PUBLIC**: il documento è pubblico
- **IETF**: il tipo di HTML pubblico è gestito dalla Internet Engineering Task Force
- **DTD HTML 4.0**: la versione di HTML supportata è la 4.0
- **EN**: la lingua del documento è l'inglese

L'uso di DOCTYPE non è obbligatorio e può essere omissis. Certamente un suo utilizzo aiuta il server Web ad interpretare correttamente il documento, ma la sua assenza non è condizionante ai fini della corretta visualizzazione.

Come si nota agevolmente DOCTYPE è un tag che non prevede un elemento di chiusura (non va scritto in questo modo: `<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 4.0//EN"></DOCTYPE>`).

<HEAD>

Gli elementi `<HEAD></HEAD>` sono posti immediatamente dopo l'apertura del tag `<HTML>` e racchiudono l'intestazione vera e propria del documento; cioè tutte le informazioni necessarie al browser, al Web server ed ai motori di ricerca. Si tratta del primo elemento letto dal browser e per questo è il luogo migliore per inserire sintassi script. All'interno di `<HEAD></HEAD>` va inserito il titolo del documento e altre informazioni. Ecco la sintassi HTML di un documento con i comandi finora esaminati:

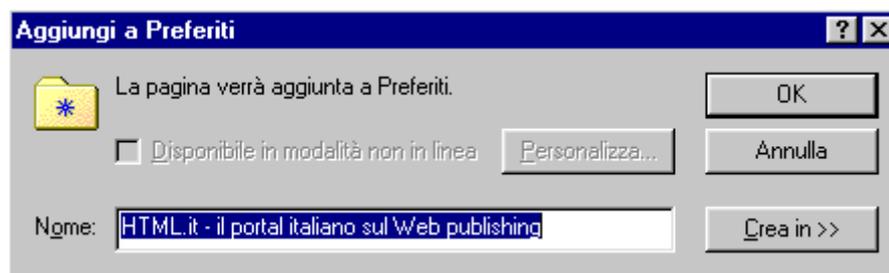
```
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 4.0//EN">
<HTML>
<HEAD></HEAD>
</HTML>
```

<TITLE>

L'elemento `<TITLE></TITLE>` è il più utilizzato all'interno del tag `<HEAD>`, in quanto fornisce il titolo alla pagina. Il titolo viene solitamente visualizzato dai browser nell'intestazione di pagina. Quella che segue è l'immagine del TITLE del sito HTML.it (il testo è: HTML.it - il portale italiano sul Web publishing):



In caso di salvataggio dell'URL con "Aggiungi a preferiti" (per IE Explorer) e "Aggiungi Segnalibro" (per Netscape) il tag TITLE dà il nome al collegamento. In altre parole, quando si salva l'indirizzo, il browser assegna allo stesso quanto presente all'interno di `<TITLE></TITLE>`. L'immagine che segue mostra l'effetto in IE Explorer (del tutto simile per Netscape):



Il contenuto riportato tra i tag `<TITLE></TITLE>` è anche utilizzato da alcuni motori di ricerca per indicizzare la pagina e trovare parole chiave. Altavista è probabilmente l'esempio più eclatante. Per questo è bene fornire nel TITLE una descrizione dettagliata ma sintetica della pagina, con tutte le parole chiave che i motori possono indicizzare.

Un consiglio da tenere a mente è comunque quello di non esagerare con la lunghezza del testo, ma di temperare le esigenze di chi aggiunge il sito al bookmark e dei motori di ricerca. La sintassi per il tag TITLE è la seguente:

```
<TITLE>La mia prima home page</TITLE>
```

META

I motori di ricerca rappresentano una risorsa indispensabile per chi cerca informazioni sulla rete, e siti quali Altavista, Yahoo, Lycos, e Excite sono tra i più visitati su internet. Figurare all'interno di tali motori di ricerca è fondamentale per chi crea pagine web e vuole maggior visibilità. È necessario innanzitutto segnalare le proprie pagine a questi motori di ricerca, e ciò viene fatto attraverso piccoli form presenti sulle pagine web sotto la scritta "Add your site".

Periodicamente questi "spider" monitoreranno le milioni di pagine contenute nei loro database verificando le eventuali modifiche.

Oltre che essere inseriti nei motori di ricerca è importante figurare tra i primi siti che vengono visualizzati nella ricerca (spesso, soprattutto quando i termini da ricercare sono di largo uso e di carattere generale, vengono presentate delle liste con migliaia di siti), e per fare questo è possibile adottare qualche semplice accorgimento.

Fondamentale a questo scopo risultano i META tag, stringhe di codice che figurano in testa al documento, tra i comandi <HEAD></HEAD> e che vengono, per prime, lette dai motori di ricerca. Le "keywords" (parole chiave) sono i termini che, sinteticamente, descrivono il contenuto di una pagina web. Se, ad esempio, il sito si occupa della realizzazione di pagine Web, le keywords saranno:

HTML
realizzazione pagine Web
home page
motori di ricerca
prezzi modici
.....

Il META tag dovrà essere così impostato:

<META name="keywords" Content="HTML, realizzazione pagine Web, home page, motori di ricerca, prezzi modici">

La virgola divide i termini gli uni dagli altri. Le frasi non limitate a singoli vocaboli vanno scritte senza virgola perché il motore di ricerca le trovi in quella stessa forma.

Naturalmente, se il vostro sito è multilingue sarà bene inserire keywords in italiano e nella lingua straniera presente nel sito, facendo sempre molta attenzione a non usare termini troppo generici. Nel caso in cui non si tratti di nomi propri è buona regola inserire la forma singolare e plurale. Il motore di ricerca considera il numero di volte in cui un termine è presente all'interno della pagina e delle keywords, ma ripetere esageratamente una stessa parola nel META tag ha l'effetto opposto a quello voluto, in quanto la gran parte degli spider cancellano dal proprio database questi siti. È bene quindi non esagerare col numero di termini all'interno delle keywords.

Altri META tag riguardano l'autore della pagina web:

<META name="author" content="Nome Cognome">

il titolo che apparirà alla fine della ricerca:

<META name="description" content="La mia prima home page">

e il nome dell'editor con cui il documento HTML è stato generato:

<meta name="GENERATOR" content="Blocco note di Win98">

Se per qualche ragione desideri che una delle tue pagine NON sia indicizzata nei motori di ricerca, devi inserire il seguente META tag:

<META NAME="ROBOTS" CONTENT="NOINDEX">

Un altro META Tag è quello che permette il "refresh" della pagina. Ciò vuol dire che la stessa pagina o altra differente può essere automaticamente lanciata dopo un certo numero di secondi che noi stessi impostiamo. Ecco il codice:

<META HTTP-EQUIV="Refresh" CONTENT="5; url=pippo.htm">

Dove CONTENT="5" è il numero di secondi entro il quale la nuova pagina sarà caricata; mentre url="pippo.htm" è il file che verrà caricato.

Di seguito riportiamo una pagina HTML impostata con il codice visto in questo capitolo. Nota l'indentazione. Questa pagina costituirà un punto di riferimento anche per le lezioni successive:

```
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 4.0//EN">
<HTML>
  <HEAD>
    <META name="keywords" Content="HTML, realizzazione pagine Web, home page,
motori di ricerca, prezzi modici">
    <META name="description" content="La mia prima home page">
    <meta name="GENERATOR" content="Blocco note di Win98">
    <META name="author" content="Nome Cognome">
    <TITLE>La mia prima home page</TITLE>
  </HEAD>
  <BODY>
  </BODY>
</HTML>
```

3. Scrivere correttamente i meta tag

Come scrivere correttamente i meta tag per risultare in buona posizione nei search engines

Lo studio dei dati forniti dai più frequentati Search Engines ha dimostrato come il termine più ricercato sia "sex", e come **l'80% degli utenti non consulta oltre la seconda pagina di ogni ricerca effettuata**. Ciò significa che se una ricerca per parola chiave risponde con 30 pagine di indirizzi Web, soltanto 2 utenti su 10 consulteranno oltre la seconda pagina. In ultima analisi, i siti contenuti nella trentesima pagina hanno una visibilità nulla. Questa considerazione dovrebbe chiarire la vitale importanza di risultare tra i primi posti in queste ricerche. È dimostrato che i Search Engine e le directory sono i mezzi che più di altri (riviste, link su altri siti, televisione ecc.) portano accessi e visitatori in un sito.

Alla luce di queste considerazioni, l'inserimento di un sito all'interno di Search Engine o Directory è un'operazione fondamentale per la riuscita del sito stesso. Accorgimenti e piccoli trucchi possono aiutare in questo, ma non c'è un modo univoco per risultare in testa ad ogni motore di ricerca, per la banale constatazione che ogni motore ha peculiarità proprie. Prima di continuare è utile precisare alcuni concetti:

- non è importante risultare in tutti i motori di ricerca ma al massimo nei 10 più diffusi a livello mondiale e nei 2 a livello nazionale;
- diffidare dei numerosi servizi che per pochi dollari promettono l'inserimento in 500 motori di ricerca, perché spesso non danno risultati e perché, appunto, non ha alcuna rilevanza risultare nei motori minori;
- diffidare dai programmi automatici di inserimento;
- non sperare che le "keywords" dei Meta Tag siano la soluzione unica a questi problemi;
- spesso l'inserimento non è immediato, e in alcuni casi (Lycos e HotBot) è successo addirittura che il database non venisse aggiornato per mesi.

Il numero di persone che negli USA si occupano esclusivamente di inserimento siti nei motori di ricerca è stimato intorno alle 100 unità. Ciò significa che nessuna Directory può materialmente controllare tutti i nuovi siti inseriti, per cui molti vengono indicizzati senza essere previamente verificati. Questo stato di cose genera una certa confusione, oltre ad uno scadimento qualitativo degli inserimenti. Infoseek ha reso noto che giornalmente riceve 30.000 nuove richieste di inserimento, poco meno della metà di Yahoo.

L'inserimento nelle Directory è, per loro stessa natura, più umanizzato e difficilmente oggetto di trucchi più o meno leciti. Anzi, tentare di risultare furbescamente ai primi posti delle Directory può essere controproducente, visto che gli operatori, resisi conto del trucco, inseriscono il sito nel "limbo dei furbi" in posizione quasi invisibile. Diverso, invece, il caso dei Search Engines dove non è un essere umano a ricevere le richieste di inserimento ma uno spider. Nel corso degli anni gli spider, non troppo perspicaci agli esordi, hanno affinato i propri strumenti e in molti casi radicalizzato l'intolleranza verso trucchi e trucchetti. Spesso è sufficiente una keyword ripetuta due volte per causare la cancellazione, tout court, del sito dal database. Prima di tutto, quindi, è buona regola evitare le penalizzazioni. Trucchi che fino ad un anno fa permettevano di risultare primi nei Search Engines oggi sono riconosciuti dagli spider. Per esempio non è una buona idea inserire testo nascosto nella pagina HTML di colore identico al background, come non è una buona idea creare pagine con parole chiave che non hanno nulla a che vedere col contenuto del sito, ma inserite solo perché spesso ricercate dagli utenti (per es. il termine "sex" in un sito che si occupa di informatica). Evitare in modo assoluto di inserire pagine con un meta tag "refresh" che rimanda ad un'altra pagina dopo un certo numero di secondi.

Premesso che il delitto non paga, ecco una serie di suggerimenti per ottenere un buon posizionamento nei Search Engines:

- evitare, per quanto possibile, l'uso di frame HTML;
- non iniziare il documento HTML con un'immagine o una tabella;
- porre particolare attenzione alla home page del sito;
- evitare, per quanto possibile, l'uso di pagine ASP (Active Server Pages) o nominate con un nome di file dove compare un punto interrogativo "?";
- inserire le parole chiave del sito tra i tag <TITLE></TITLE>, perché molti spider (Altavista in primis) danno molta importanza a questo;
- usare i tag <H1><H2> ecc.
- creare "door-page", cioè delle pagine di supporto a quelle principali, contenenti testo e keywords attinenti al sito;
- registrare non più di una pagina al giorno;

- registrare domini di primo livello. Per esempio `www.nome.com` piuttosto che `www.nome.com/direcotry`;
- il meta tag "keywords" non deve contenere termini duplicati e non più di mille caratteri. Il 90% delle ricerche avviene per termini scritti in minuscolo, per cui è preferibile evitare quelli maiuscoli;
- il meta tag "description" è fondamentale perché a questo il Search Engine associa il link al sito e non deve superare i 300 caratteri.

4. <BODY> Impostare lo sfondo del documento

Come inserire un'immagine o un colore di sfondo; il colore di tutti i link attivi e visitati e altre operazioni riguardanti lo sfondo del documento.

Nella precedente lezione abbiamo visto come creare da zero un documento HTML e come impostare il titolo ed i meta tag.

L'operazione successiva all'impostazione del documento è la definizione del colore o dell'immagine di sfondo, oltre ai colori dei link attivi e visitati.

<BODY>

L'elemento <BODY> è posto in posizione immediatamente successiva alla chiusura del tag </HEAD> e comunque all'interno degli elementi <HTML></HTML>; ha un tag di apertura e uno di chiusura, ed all'interno di esso si sviluppa il corpo del documento.

Se l'elemento <HEAD> conteneva metadati (cioè dati non materialmente visualizzati dal browser) la funzione del tag <BODY> è quella di mostrare gli oggetti (testo, immagini, suoni, applet, ecc) della pagina.

La sintassi per l'elemento <BODY> è la seguente:

<BODY>Contenuto del documento</BODY>

Il tag <BODY> è utilizzato, oltre che per fornire al browser indicazioni sulla posizione degli oggetti nel documento, anche per impostare vari attributi di visualizzazione per il documento. Di seguito vediamo quali.

BGCOLOR

L'attributo BGCOLOR imposta un colore unitario di sfondo. La sintassi è la seguente:

<BODY BGCOLOR="red">

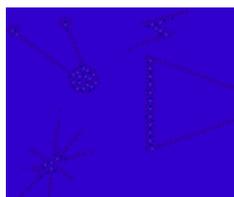
È possibile sostituire al nome in inglese, valori esadecimali. Per esempio, il colore rosso (red) si sostituisce in questo modo:

<BODY BGCOLOR="#ff0000">

L'utilità dei colori esadecimali si ha laddove non si vuole un colore standard ma sfumato o con diversa tonalità. I più diffusi editor HTML prevedono palette per la definizione di colori esadecimali, mentre Paint Shop Pro (software per il foto-ritocco) ad esempio fornisce, oltre al colore, anche il corrispondente valore esadecimale da copiare/incollare.

BACKGROUND

BACKGROUND ha una funzione simile a BGCOLOR, ma mentre il secondo mostra un tinta unica del colore, il primo visualizza sullo sfondo un'immagine in formato grafico .gif o .jpg. Consideriamo, per esempio, di voler costruire uno sfondo con l'immagine seguente:



L'immagine si chiama **sfondo.gif** ed è depositata nella stessa directory del documento. La sintassi per impostare l'immagine come sfondo è:

<BODY BACKGROUND="sfondo.gif">

Il browser visualizza l'immagine sfondo.gif e la ripete su ogni punto dello schermo disponibile. In altre parole non si limita a visualizzare l'immagine una sola volta, magari al centro della pagina, ma riempie ogni spazio disponibile. Per questa ragione è assolutamente necessario creare uno sfondo che, se ripetuto, presenti un aspetto il più possibile uniforme.

È bene scegliere un'immagine di sfondo che non infastidisca la lettura e che sia il più possibile coerente con il colore del testo. Per esempio, inserire un testo arancione su uno sfondo rosso non renderebbe leggibile il testo. Sempre meglio usare il colore nero per il testo e tinte leggere per lo sfondo.

TEXT

Se non stabilito diversamente il colore del testo del documento è nero, in quanto i browser assegnano tale colore di default. Grazie all'attributo TEXT è possibile assegnare al testo un colore diverso dal nero. Questa la giusta sintassi:

```
<BODY BACKGROUND="sfondo.gif" TEXT="red">
```

Anche in questo caso i colori possono esprimersi in nomi o valori esadecimali. All'interno del documento è possibile marcare parte del testo in colori differenti da quello impostato su TEXT.

LINK

Se non stabilito diversamente il colore dei link (non ancora visitati) è blue. Grazie all'attributo LINK è possibile definire colori differenti:

```
<BODY BACKGROUND="sfondo.gif" LINK="red">
```

Tutti i link della pagina non saranno più blue ma rossi (red). Tale colore può essere espresso in valori esadecimali.

ALINK

I link, quando vengono cliccati, assumono un colore diverso da quello impostato su LINK (o dal blue di default). Grazie ad ALINK (la A sta per Active) è possibile modificare questo colore:

```
<BODY BACKGROUND="sfondo.gif" ALINK="yellow">
```

VLINK

Quando un URL associato ad un link viene visitato, quest'ultimo assume un colore diverso da quello di LINK (o dal blue di default). Grazie a VLINK (la V sta per Visited) è possibile agire su questo colore:

```
<BODY BACKGROUND="sfondo.gif" VLINK="green">
```

BGPROPERTIES

Trattando dell'attributo BACKGROUND abbiamo sottolineato come le immagini richiamate siano disposte sullo schermo disponibile. Se, comunque, la pagina è tanto lunga da attivare lo scroller laterale, lo sfondo (e l'immagine associata) scorre insieme alla pagina.

Grazie alla proprietà BGPROPERTIES è possibile rendere lo sfondo immobile rispetto allo scroller di pagina. Questa la sintassi:

```
<BODY BACKGROUND="sfondo.gif" BGPROPERTIES="fixed">
```

Questo espediente **funziona solo con l'Explorer** e non con Netscape che invece continua a scrollare la pagina.

5. Inserire suoni o musiche di sottofondo

Come inserire suoni e musiche di sottofondo compatibili con Netscape e Explorer.

La presenza di file audio sotto forma di MIDI o WAV può essere una piacevole colonna sonora alla navigazione delle pagine web, oppure un fastidioso fardello capace solo di rallentare il caricamento ed innervosire il visitatore. Come spesso viene ripetuto in questa guida è necessario considerare che sì, è importante la piacevolezza grafica e musicale delle pagine web, ma è fondamentale che la navigazione non risulti lenta. Il consiglio è quindi di evitare grossi file MIDI (60Kb di files MIDI ad esempio sono esagerati) ed enormi WAVE (che possono essere sostituiti da altri files più leggeri, come vedremo in seguito).

Consideriamo un file MIDI che si chiami "image.mid"; e consideriamo che si voglia inserirlo come sottofondo ad pagina web non appena questa si apra, automaticamente quindi. Dobbiamo innanzitutto considerare che Navigator e Explorer si servono di comandi diversi per richiamare automaticamente un file MIDI di sottofondo.

Per esempio il comando <BGSOUND> funziona solo con Explorer ma non con Navigator. LOOP indica il numero delle volte che il midi deve essere ripetuto:

```
<BGSOUND SRC="image.mid" LOOP=INFINITE>
```

Oltre ai files MIDI, con il comando <BGSOUND> è possibile inserire files .WAV e .AU. Per rendere possibile l'apertura automatica di un files audio anche con Netscape, si deve usare il comando <EMBED>:

```
<EMBED SRC="image.mid" AUTOSTART=true LOOP=true VOLUME="80" WIDTH="0" HEIGHT="0">
```

Anche con questo comando oltre ai MIDI è possibile inserire files .WAV e .AU. Una delle remore all'inserimento di files midi all'interno delle proprie pagine, è dovuta al fatto che, ogni volta che si cambia pagina il file musicale si interrompe bruscamente. Il problema non persiste nei siti divisi in frame, in quanto il file musicale viene caricato nel frame fisso e quindi non muta al cambiamento delle pagine negli altri frame. E per i siti senza frame? Un piccolo espediente può essere quello mediante il quale, proprio prendendo spunto dai frames, si costruisce una pagina con un frame "invisibile".

Per fare questo si crea una pagina con il seguente codice:

```
<FRAMESET rows="0,*" border=0 frameborder="0" framespacing="0">  
<frame me name="alto" src="top.htm" noresize>  
<frame me name="centrale" src="central.htm" noresize>  
</FRAMESET>
```

Dove nel frame "alto" inseriremo il codice che richiama il file audio:

```
<EMBED SRC="image.mid" AUTOSTART=true LOOP=true VOLUME="80" WIDTH="0" HEIGHT="0">
```

E nel frame "centrale" il contenuto delle pagine.

Altro "stratagemma" per far ascoltare un file audio (**SOLO CON NETSCAPE**) in modo continuativo anche quando si cambia pagina, è quello di caricare tale file in una finestra del browser alternativa. Per fare questo è necessario servirsi di un piccolo ma importantissimo TARGET. I TARGET vengono usati soprattutto nei frame ma possono esserlo anche in altri casi. Se per esempio si desidera che una pagina collegata ad un link venga caricata in una finestra diversa del browser, si dovrà inserire il TARGET="_new".

Facciamo conto che il midi da ascoltare si chiami "image.mid"; ecco il codice:

```
<A HREF="image.mid" TARGET="_new">Ascolta il midi image.mid</A>
```

Cliccando si aprirà una finestra di default la quale si affiancherà a quella principale (da cui si è lanciato il link). Questa procedura è valida solo per Netscape.

In questo modo il visitatore potrà leggere il testo presente sulla finestra da cui ha lanciato il file "image.mid" e nel frattempo l'altra finestra produrrà il midi. Nel momento in cui il visitatore cliccherà su un altro midi da ascoltare, automaticamente tale midi verrà caricato nella finestra precedente aperta per "image.mid" (non verranno aperte, cioè, decine di finestre).

Purtroppo questo espediente funziona solo con Netscape e non con Explorer. Per venire incontro anche agli utenti del browser della software house di Redmond è necessario architettare una procedura un po' più complessa, la quale consiste nell'associare al link non il semplice file midi "image.mid" ma un file HTML che contenga un comando EMBED che faccia automaticamente partire il midi.

6. carattere e grandezza del testo

I tag fisici e logici; come formattare il testo o scegliere un font; come rendere il testo grassetto, corsivo e sottolineato.

Prima di approfondire le tematiche legate alla formattazione del testo è bene precisare alcuni concetti che reggono l'uso di questi strumenti.

Tag di testo logici e fisici

HTML è un sistema di contrassegno con il compito di definire la struttura e l'aspetto di un documento. Questa definizione che senza altri commenti passerebbe inosservata, va invece approfondita per capire meglio i concetti di tag fisici e logici.

Un tag fisico ha il compito di formattare visivamente un documento, cioè di renderlo grassetto, corsivo, sottolineato ecc. La funzione di questi tag non è quella di dare un aspetto al documento, ma di marcare determinati punti per definirne una struttura.

Un tag logico, al contrario ha la funzione di definire una struttura indipendentemente dal suo aspetto. Ovvero, indipendentemente dal modo in cui il browser interpreterà visivamente il contrassegno.

Per comprendere meglio questa differenza prendiamo in esame due elementi volti a formattare testo:

- **<ADDRESS></ADDRESS>**, un tag logico, il quale contrassegna l'informazione di indirizzo nel documento. In altre parole quando si è in presenza di un indirizzo e-mail, di un numero di telefono o di un indirizzo secondo le regole di HTML va inserito questo tag logico.
- **<I></I>**, un tag fisico, il quale rende corsivo (italic) il testo compreso nell'elemento. È quindi, un tag volto a dare un aspetto al documento e non a marcare un elemento di struttura.

Scrivendo:

```
<ADDRESS>pippopluto@provider.it</ADDRESS>
```

oppure

```
<I>pippopluto@provider.it</I>
```

il risultato visivo di entrambi i tag è lo stesso (il testo corsivo):

pippopluto@provider.it

ma da un punto di vista di struttura del documento, solo il primo (ADDRESS) indica che al proprio interno è presente un indirizzo, mentre il secondo non dà alcuna informazione di questo tipo.

Tag FONT e nuovo standard HTML 4

HTML 4 è lo standard che da quasi due anni regge le sorti del linguaggio di markup più famoso al mondo. Senza entrare nel merito delle novità del nuovo standard vogliamo sottolineare come il W3C abbia "deprecato" l'uso del tag FONT nella formattazione del testo in HTML. Il termine "deprecare" (letteralmente tradotto dall'inglese) indica che i browser leggono ancora questo tag, ma il suo uso è decisamente vietato dal nuovo standard. Secondo HTML 4 la formattazione del testo è lasciata unicamente ai "fogli di stile" o CSS (Cascading Style Sheets).

In questo capitolo non terremo conto delle indicazioni del W3C e spiegheremo comunque l'uso dei tag classici di HTML 4. Rimandiamo comunque alla sezione sui "fogli di stile" di questa guida per un approfondimento dell'argomento CSS.

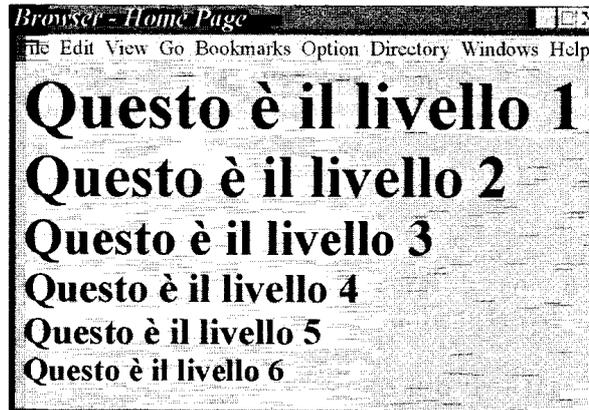
Formattare titoli da <H1> ad <H6>

Il tag <Hn> (dove n è il numero da 1 a 6 che è possibile assegnare all'elemento) ha la funzione di fornire stili ai titoli di pagina, dando maggiore o minore enfasi a seconda del numero inserito. Come accennato i numeri vanno da 1 a 6, con maggior importanza dei numeri più bassi rispetto agli alti. Visivamente tale diversa importanza si traduce in differente grandezza del testo, come nell'esempio che segue:

```

<HTML><HEAD><TITLE>Home Page</TITLE></HEAD><BODY>
<H1>Questo è il livello 1</H1>
<H2>Questo è il livello 2</H2>
<H3>Questo è il livello 3</H3>
<H4>Questo è il livello 4</H4>
<H5>Questo è il livello 5</H5>
<H6>Questo è il livello 6</H6>
</BODY>
</HTML>

```



Oltre a definire lo stile dei titoli, il tag <Hn> inserisce spaziature di paragrafo rispetto agli elementi che seguono.

Il tag fisico FONT è uno dei più usati e frequenti nell'attuale Web publishing. Nella sezione "Impostare lo sfondo del documento (elemento BODY)" abbiamo visto come l'attributo TEXT dia un colore univoco a tutto il testo del documento. Il tag FONT ha in parte una funzione simile, ma molto più ampia e concettualmente differente.

Il tag FONT ha la funzione di formattare tipo e grandezza del testo limitatamente ad alcuni punti del documento. In realtà se il tag font aprisse e chiudesse il documento, la totalità del testo compreso al suo interno sarebbe formattato di conseguenza. Ma concettualmente il tag FONT è stato pensato per definire parti limitate di testo. Mentre, poi, TEXT determina solo il colore del testo, il tag FONT può definire: tipo di font utilizzato, grandezza e colore. Ecco la sintassi del tag:

```
<FONT FACE="arial" SIZE=5 COLOR=red>Carattere da formattare</FONT>
```

L'attributo **FACE="arial"** indica il tipo di carattere da richiamare per la visualizzazione. **SIZE=5** imposta la grandezza del carattere, che può andare da 1 a 7 (con maggiore grandezza dei numeri vicini al 7); **COLOR=red** indica il colore del testo (esprimibile anche in valori esadecimali).

, <I>, <U>

, <I>, <U> sono tre dei più usati tag di formattazione fisica dell'HTML. Tutti vanno aperti e successivamente richiusi:

```
<B> Testo grassetto </B>
```

Il testo compreso tra questi tag è trasformato in grassetto (la B sta per BOLD).

```
<I> Testo corsivo </I>
```

Il testo compreso tra questi tag è formattato in corsivo (la I sta per ITALIC).

<U> Testo sottolineato </U>

Il testo compreso tra questi tag è sottolineato pur non essendo un link (la U sta per UNDERLINE).

<STRIKE>

Esiste anche il tag STRIKE per il testo barrato:

<STRIKE> Testo barrato </STRIKE>

<SUP> e <SUB>

Questi due tag di formattazione fisica creano rispettivamente: apici posti leggermente più in alto del normale testo (SUP) e pedici posti leggermente più in basso (SUB). È possibile annidare diversi tag per ottenere effetti di sovrapposizione successiva. Questi elementi vengono usati soprattutto per indicare note o formule matematiche. Questa la sintassi:

10² = 100

**101₂ = 1x2² + 0x2¹ + 1x2⁰
= 5₁₀**

Il testo visualizzato dal browser sarà rispettivamente:

$10^2 = 100$

e

$101_2 = 1x2^2 + 0x2^1 + 1x2^0 = 5_{10}$

Testo preformattato <PRE>

HTML legge il testo ovviamente da sinistra verso destra e, senza alcun tipo di formattazione, manda a capo alla fine di ogni riga senza soluzione di continuità. La formattazione successiva esaminata in questo capitolo ha il compito di indicare al browser dove andare a capo, quale font utilizzare, quali parole enfatizzare e così via. Spesso, però, un documento creato con un normale editor di testo deve essere importato senza formattazione, così come è stato creato. A questo scopo HTML prevede i tag di testo preformattato (i quali non leggono né interpretano alcun tag di HTML, neanche i caratteri speciali):

<XMP> Testo da formattare </XMP>

<PRE> Testo da formattare </PRE>

Entrambi i tag hanno lo stessa funzione, anche se lo standard di HTML 4 prevede l'uso di XMP e non di PRE.

Breve descrizione degli stili logici

Abbiamo descritto in apertura di lezione la differenza tra tag fisici e logici. Ora elenchiamo i tag logici più interessanti e il corrispondente utilizzo in HTML:

<ADDRESS>

Contrassegna informazioni di indirizzo (mail, telefono ecc.)

<BLOCKQUOTE>

Usato per citazioni più lunghe di due o tre righe

<CITE>

Usato per indicare la fonte della citazione.

<CODE>

Viene utilizzato per formattare righe di codice di programmazione.

<DFN>

Indica che il testo compreso è una definizione.

Enfatizza il testo compreso all'interno del tag.

<KBD> e <SAMP>

Poco utilizzati. Indicano cose che il computer dovrebbe dire all'utente e viceversa, secondo un tipico concetto di Unix.

Enfatizza il testo.

<VAR>

Legato a CODE, indica le variabili di programmazione.

7. Paragrafi e giustificazione del testo

Come posizionare testo e immagini; centrare oggetti, creare linee orizzontali.

Nella capitolo precedente abbiamo visto come formattare del semplice testo in HTML. A questo punto dobbiamo affrontare il modo in cui sistemare, dividere, posizionare il testo e gli altri oggetti di una pagina HTML all'interno del documento.

Creare paragrafi con <P>

Il tag <P> definisce un nuovo paragrafo del testo indicando al browser che lo stesso deve rimanere su una nuova riga ed essere posizionato a destra, a sinistra o al centro. Se non specificato oltre, il tag <P> allinea il testo di default sulla sinistra. Per indicare altri tipi di posizionamento esistono attributi specifici:

<P ALIGN=left>

Definisce un paragrafo e allinea sulla sinistra (left).

<P ALIGN=right>

Definisce un paragrafo e allinea sulla destra (right).

<P ALIGN=center>

Definisce un paragrafo ed allinea al centro (center).

Andare a capo con

 è un tag di interruzione di riga e ritorno a capo. Ha un funzionamento simile al paragrafo visto in precedenza, ma se ne discosta perché non inizia un nuovo paragrafo. In altri termini la sua funzione è simile alla pressione del tasto "invio" della tastiera. Va usato singolarmente senza tag di chiusura, ossia è un *separating tag* (i tag che separano il testo, distinti dai *surrounding tag* che contengono del testo).

Posizionare il testo con <DIV ALIGN> e <CENTER>

L'elemento <DIV> viene utilizzato per allineare il testo in posizione orizzontale a sinistra, destra e centro pagina. L'attributo ALIGN è fondamentale a questo scopo.

<DIV ALIGN=left>Testo e immagini a sinistra</DIV>

Sposta gli elementi contenuti tra i suoi tag sulla sinistra.

<DIV ALIGN=right>Testo e immagini a destra</DIV>

Sposta gli elementi sulla destra.

<DIV ALIGN=center>Testo e immagini centrate</DIV>

Sposta gli elementi in posizione centrale.

Il tag <CENTER> ha un funzionamento del tutto simile a <DIV ALIGN=center> ma il suo uso è stato "deprecato" dallo standard 4 di HTML. L'uso di <CENTER> è stato introdotto da Netscape, proprio quando l'HTML prevedeva l'uscita di DIV. L'enorme diffusione iniziale di Netscape ha reso questo tag altrettanto diffuso e per questa ragione ancora viene usato e riconosciuto dai browser (anche da IExplorer). L'uso di <CENTER> è molto semplice:

<CENTER>Testo da centrare</CENTER>

Linee orizzontali con <HR>

Le linee orizzontali sono uno strumento per dividere parti del documento e rendere il testo più leggibile. La sintassi necessaria al loro inserimento in un documento HTML è la seguente:

`<HR align="CENTER" size="2" width="400" color="Red" noshade>`

Il tag HR (acronimo che sta per Horizontal Rule) non ha bisogno di chiusure successive, è un *separating tag*. Si compone di diversi attributi:

- **aling="CENTER"**: definisce la posizione della riga (center, right, left).
- **size="2"**: definisce l'altezza, in pixel, della riga.
- **width="400"**: definisce lunghezza orizzontale, in pixel, della linea. Può anche esprimersi in percentuale di spazio disponibile: **width=80%**.
- **color="RED"**: definisce il colore della linea.
- **noshade**: se presente, questo attributo elimina l'effetto 3D della linea. Se omesso, produce tale effetto.

8. Creare elenchi puntati e numerati

Indicizzare risorse all'interno di elenchi puntati e numerati.

Gli elenchi rappresentano uno dei modi più diffusi per organizzare informazioni all'interno di siti Web. Una delle loro caratteristiche principali è sicuramente quella di fornire un quadro chiaro e sintetico dell'argomento trattato.

HTML mette a disposizione diversi tipi di elenchi che di seguito andiamo a trattare singolarmente.

***Elenchi ordinati (numerati): e ***

Gli elenchi ordinati numerati sono costituiti da un singolo tag di apertura e chiusura `` e tanti tag di apertura per quante sono le voci di menu ``. Questa è la sintassi per creare elenchi ordinati numerati:

```
<OL>
<LI> Prima voce di menu
<LI> Seconda voce di menu
<LI> Terza voce di menu
</OL>
```

Dall'esempio si nota come è possibile omettere il tag `
` per il ritorno a capo, visto che è automaticamente inserito da ``. Se non stabilito diversamente (come vedremo in seguito), il tipo di elenco ordinato che il browser visualizza è numerato (cioè l'elenco è costituito da una serie di numeri che da 1 crescono progressivamente). Le ultime versioni di HTML prevedono la possibilità di creare elenchi ordinati che contengono un sistema di indicizzazione diverso dai numeri, come appena visto.

Indicizzazione alfabetica maiuscola:

```
<OL TYPE=A>
<LI> Prima voce di menu
<LI> Seconda voce di menu
<LI> Terza voce di menu
</OL>
```

Indicizzazione alfabetica minuscola:

```
<OL TYPE=a>
<LI> Prima voce di menu
<LI> Seconda voce di menu
<LI> Terza voce di menu
</OL>
```

Indicizzazione con numeri romani maiuscoli:

```
<OL TYPE=I>
<LI> Prima voce di menu
<LI> Seconda voce di menu
<LI> Terza voce di menu
</OL>
```

Indicizzazione con numeri romani minuscoli:

```
<OL TYPE=i>
<LI> Prima voce di menu
<LI> Seconda voce di menu
<LI> Terza voce di menu
</OL>
```

**Elenchi NON ordinati (puntati): e **

Gli elenchi non ordinati funzionano in modo simile a quelli ordinati. La differenza di fondo è che le risorse indicizzate non sono legate da stretti rapporti di successione gerarchica, per cui non sono previsti elenchi progressivi quali numeri o lettere.

Gli elenchi non ordinati (o puntati) si compongono di un tag unico di apertura e chiusura e tanti tag di elenco per quante sono le voci da indicizzare . La sintassi per definire un elenco puntato è la seguente:

```
<UL>  
<LI> Prima voce di menu  
<LI> Seconda voce di menu  
<LI> Terza voce di menu  
</UL>
```

Se non definito diversamente l'elenco è costituito da una serie di pallini. Come per gli elenchi numerati anche in questo caso è possibile indicare diversi tipi di elenco.

I pallini pieni visti in precedenza sono ottenibili con disc:

```
<UL TYPE=disc>  
<LI> Prima voce di menu  
<LI> Seconda voce di menu  
<LI> Terza voce di menu  
</UL>
```

L'attributo circle imposta pallini vuoti all'interno:

```
<UL TYPE=circle>  
<LI> Prima voce di menu  
<LI> Seconda voce di menu  
<LI> Terza voce di menu  
</UL>
```

L'attributo square imposta elenchi definiti da quadratini pieni:

```
<UL TYPE=square>  
<LI> Prima voce di menu  
<LI> Seconda voce di menu  
<LI> Terza voce di menu  
</UL>
```

9. inserire immagini nel documento

Come inserire immagini all'interno di un documento e usare gli attributi del tag IMG.

Internet, solo negli ultimi 10 anni affermata come media di massa, è stato per lunghi anni un sistema di comunicazione ad esclusivo utilizzo di apparati militari prima, e universitari dopo. È agevole immaginare quanto poco inclini alla grafica ed all'estetica fossero i vecchi navigatori della Rete. Alla fine degli anni ottanta Internet ha subito una radicale trasformazione che ha portato sulle "autostrade telematiche" un'utenza di massa, con esigenze diverse da quelle militari o accademiche. Con la trasformazione della Rete si è imposta l'esigenza di mettere a disposizione strumenti di navigazione più "user friendly" per esigenze non più meramente di studio e ricerca. Così, nel 1989 nasce il WWW (World Wide Web) che grazie al primo browser della storia, Mosaic del 1992, trasforma il testo bianco su sfondo nero nell'attuale Web, fatto di colori e interattività. La rivoluzione dell'immagine (criticata da alcuni puristi accademici che videro nell'Internet di massa la fine del sistema di comunicazione) fu anche merito di un tag oggi molto comune , cioè l'elemento necessario a richiamare immagini all'interno della pagina.

Prima di trattare nello specifico il tag principale per l'inserimento di immagini negli ipertesti è bene precisare alcuni concetti di HTML. A differenza di molti editor di testo (Ms Word, per esempio) gli ipertesti non vengono "fusi" con le immagini a corredo grafico, ma si limitano a richiamarle da un percorso specifico in locale o su Web. In altre parole esiste una distinzione molto chiara tra file .htm e immagini (ma anche file audio, applet Java, oggetti Flash, ecc). I documenti HTML si limitano a prevedere uno spazio al proprio interno entro il quale vanno inserite le immagini richiamate. Il tag non ha bisogno di chiusura e la sua sintassi è la seguente:

```
<IMG SRC="immagine.gif">
```

Dove **SRC** sta per "search" ed è il percorso dal quale il browser ricava l'immagine (in questo caso "immagine.gif"). Come detto si tratta di un tag singolo, nel senso che NON va chiuso con un corrispondente !

I browser (Netscape, MsIe, Opera, ecc.) riconoscono molti formati grafici, anche se sono due quelli più utilizzati: **GIF** (Graphics Interchange Format) e **JPEG** (Joint Photographics Experts Group). Negli ultimi tempi sta diffondendosi su scala mondiale un altro formato: **PNG** (Portable Network Graphics).

L'elemento è corredato da diversi attributi molto utili ai fini del suo utilizzo. Vediamo insieme quali.

ALT

L'utilizzo di commenti testuali alle immagini è una fondamentale regola da osservare nella creazione di siti Web per alcune buone ragioni:

- taluni browser potrebbero essere impostati per non richiamare le immagini;
- i browser testuali per non vedenti non riuscirebbero ad interpretare le immagini senza un commento.
- è possibile evitare didascalie inserendo commenti all'interno dell'immagine stessa. In tutti i casi considerati l'uso di commenti risolve il problema e consente di ottimizzare l'utilizzo di una pagina Web. La sintassi per i commenti è la seguente:

```
<IMG SRC="immagine.gif" ALT="Opera di P. Picasso">
```

Per verificare gli effetti basta passare con il mouse sull'immagine in questione.

WIDTH e HEIGHT

Negli esempi finora indicati non abbiamo specificato le misure dell'immagine che il browser ha provveduto a cercare automaticamente. È possibile definire altezza e larghezza dell'immagine con gli attributi width ed height:

```
<IMG SRC="immagine.gif" WIDTH=178 HEIGHT=180 ALT="Opera di P. Picasso">
```

Dove **WIDTH=178** è la dimensione orizzontale (larghezza) dell'immagine espressa in pixel, e **HEIGHT=180** la dimensione verticale (altezza).

Con questi attributi è possibile definire dimensioni differenti da quelle effettive dell'immagine. È comunque, consigliabile inserire immagini di dimensioni naturali soprattutto se in formato GIF, visto che ridimensionando questo formato la qualità decade enormemente.

BORDER

Con l'attributo **BORDER** è possibile assegnare un bordo all'immagine. I valori sono numerici ed espressi in pixel. Il valore **BORDER** impostato su 0 ha l'effetto di non visualizzare alcun bordo. Se questo attributo viene omesso il browser non assegna alcun bordo, ma se l'immagine è anche un link viene automaticamente assegnato un **BORDER=1**. Questa è la sintassi:

```
<IMG SRC="immagine.gif" WIDTH=178 HEIGHT=180 BORDER=2 ALT="Opera di P. Picasso">
```

HSPACE e VSPACE

Con questi due attributi è possibile stabilire la distanza in pixel dell'immagine dagli oggetti che si trovano ai quattro lati della stessa:

- **HSPACE** definisce la distanza dai lati destro e sinistro dell'immagine degli oggetti più vicini (testo, immagini, applet ecc.).
- **VSPACE** definisce la distanza dai lati superiore e inferiore dell'immagine degli oggetti più vicini (testo, immagini, applet ecc.). Questa la corretta sintassi:

```
<IMG SRC="immagine.gif" WIDTH=178 HEIGHT=180 BORDER=2 VSPACE=3 HSPACE=3 ALT="Opera di P. Picasso">
```

Questi attributi risultano utili quando si vuole distanziare l'immagine dagli oggetti più vicini.

ALIGN

L'attributo **ALIGN** definisce la posizione dell'immagine rispetto al testo posto immediatamente prima o dopo la stessa. Esistono varie opzioni per l'attributo **ALIGN**:

- **ALIGN=top**: allinea la prima riga di testo sulla sinistra al top dell'immagine.
- **ALIGN=middle**: allinea la prima riga di testo sulla sinistra al centro dell'immagine.
- **ALIGN=bottom**: allinea la prima riga di testo sulla sinistra nella parte più bassa dell'immagine.
- **ALIGN=left**: allinea il testo sulla destra dell'immagine partendo dal top.
- **ALIGN=right**: allinea il testo sulla sinistra dell'immagine partendo dal top.

10. <HREF> link verso documenti o immagini

Come creare collegamenti tra documenti HTML con l'elemento Ancora <A>e i suoi attributi.

Gli ipertesti sono il modo più comune di visualizzazione delle informazioni sul Web. Un ipertesto è un modo di formattare documenti in forma non sequenziale. Per comprendere questa caratteristica si pensi ad un libro cartaceo che si struttura in: sommario, capitoli e indice analitico. Ha, dunque, una struttura sequenziale nel senso che si legge il sommario per passare ai capitoli (dal primo all'ultimo), terminando con l'indice analitico dei termini. Il lettore che legge un libro segue generalmente un percorso di lettura che lo porta necessariamente a leggere il primo capitolo per giungere all'ultimo. Con gli ipertesti questa struttura sequenziale cessa di esistere grazie ai collegamenti ipertestuali o link, che permettono di leggere il documento senza seguire necessariamente un ordine sequenziale. La peculiarità di saltare da un punto all'altro del documento è tipica del Web, dove spesso seguendo un documento si passa da un sito all'altro a seconda dei propri interessi.

I documenti HTML sono degli ipertesti il cui funzionamento è in massima parte dovuto agli HyperLink o ancoraggi, il cui tag specifico è <A>.

<A HREF>

L'elemento <A> (la A sta per Anchor) ha bisogno di un tag di apertura e chiusura e al suo interno è possibile inserire testo, immagini o altri elementi multimediali.

Perché funzioni, l'elemento <A> deve essere associato ad altri attributi. Il più importante di questi è HREF (abbreviazione di Hypertext Reference) contenente l'URL o la pagina da raggiungere. Questa la sintassi:

```
<A HREF="http://www.nomesito.it">Visita nomesito.it</A>
```

Nel codice sopra citato, cliccando sul testo "Visita nomesito.it" (compreso tra i tag A in apertura e chiusura) si raggiunge l'URL http://www. nomesito.it indicato dall'attributo HREF.

È possibile sostituire al testo un'immagine con effetto identico:

```
<A HREF="http://www.nomesito.it"><IMG SRC="nomesito.gif"></A>
```

HREF può contenere sia link a risorse esterne (come nel caso precedente), oppure link ad altri documenti dello stesso sito. Per esempio, se ci troviamo nella pagina 1.htm e vogliamo inserire un link alla pagina 2.htm presente nella stessa cartella, il codice corretto è:

```
<A HREF="2.htm">Clicca qui per raggiungere la nuova pagina</A>
```

TARGET

TARGET è un attributo implementato per esigenze legate alla gestione dei frame. In una pagina divisa in frame, infatti, questo attributo indica in quale di questi debba essere visualizzato il documento. Non entriamo nel merito dei frame.

In questa sede possiamo, però, trattare di un uso ulteriore di questo attributo. Grazie a TARGET è possibile caricare un pagina associata a HREF in un'altra finestra del browser, attraverso la seguente sintassi:

```
<A HREF="http://www.nomesito.it" TARGET="_new">Visita nomesito.it</A>
```

L'attributo TARGET="_new" indica al browser di caricare il link a http://www.nomesito.it in una nuova (_new) finestra.

Il lancio di nuove finestre da link può essere molto utile nel caso in cui si rimandi a risorse esterne e non si voglia perdere il visitatore, che in questo modo terrà aperte due finestre.

TITLE

Come per l'elemento IMG, anche per i collegamenti ipertestuali è possibile definire un testo di commento, attivato quando il mouse passa sull'area di link. Questa la sintassi:

`Visita nomesito.it`

MAILTO (link a e-mail)

È possibile inserire collegamenti anche verso indirizzi e-mail, attraverso una sintassi leggermente diversa da quella indicata per le URL. Questo il codice:

`Scrivi al webmaster di nomesito.it`

Cliccando su questo link viene automaticamente aperto il tuo programma di posta predefinito con il campo "To:" già impostato su webmaster@nomesito.it.

Link interni al documento

Finora abbiamo analizzato link a risorse esterne o ad altre pagine di uno stesso sito. È possibile creare collegamenti in punti specifici all'interno di un certo documento, grazie al codice ``.

Ecco la procedura per creare collegamenti a punti interni del documento:

- Inserisci il tag `` nel punto del documento da raggiungere, dove la parola "ancora" è il nome identificativo del punto in cui il browser dovrà visualizzare la pagina. Ovviamente puoi sostituirlo con altri termini.
- Nel collegamento dal quale vuoi raggiungere il punto del documento specificato dalla parola "ancora" inserisci la seguente sintassi: `Vai al link`

Il cancelletto (#) indica che si tratta di un link interno. L'esempio considerato rimanda ad un punto della stessa pagina, perché se a destra del cancelletto c'è il nome del link interno, a sinistra non è scritto nulla.

Se invece volessimo raggiungere un punto preciso di un documento esterno a quello nel quale ci troviamo, la sintassi è:

`Vai al link`.

Dove "nome_file.htm" è il nome del documento da raggiungere, e "ancora" il punto preciso all'interno dello stesso file.

11. <TABLE> creare e gestire Tabelle HTML

Creare tabelle HTML per gestire dati e formattare il layout di pagina.

La realizzazione di tabelle all'interno di una pagina HTML viene realizzata per mezzo del gruppo di tag <TABLE> all'interno del quale stanno <TR> e <TD> per creare la struttura della tabella stessa.

<TABLE> <TR> e <TD>

La larghezza e l'altezza complessiva di una tabella viene indicata all'interno dell'etichetta <TABLE>, la quale è quella che deve aprire e chiudere una tabella. Le dimensioni vengono definite con WIDTH (larghezza) e HEIGHT (altezza) e possono essere espresse in pixel o in percentuale della pagina corrente:

```
<TABLE WIDTH=500>  
</TABLE>
```

In questo esempio di codice la larghezza della tabella viene espressa in pixel (in questo caso 500). Se si sceglie questa opzione, a qualsiasi risoluzione venga vista la pagina, la misura della tabella rimarrà invariata, cioè di 500 pixel di larghezza.

Cosa diversa accade quando si fornisce una grandezza in percentuale della pagina:

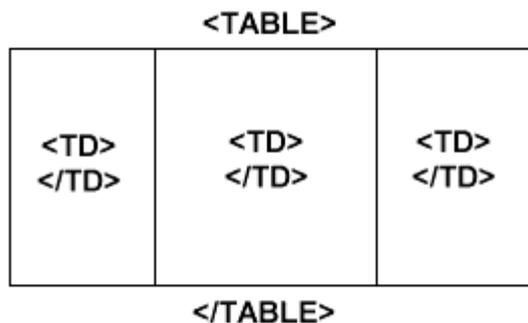
```
<TABLE WIDTH=50%>  
</TABLE>
```

In questo caso la larghezza della tabella sarà diversa a seconda della risoluzione video di colui che ne visualizzerà il contenuto.

Per esempio chi ha una risoluzione video di 640x480 vedrà una tabella di circa 320 pixel (perché abbiamo impostato la tabella di una grandezza pari al 50% della pagina), mentre chi ha una risoluzione di 800x600 la vedrà grande 400 pixel.

Mantenere il controllo assoluto sulla grandezza delle tabelle adottando una misurazione in pixel piuttosto che configurarla in percentuale dipende dalle esigenze di visualizzazione della pagina.

Ora diamo uno sguardo al modo in cui deve strutturarsi una tabella:

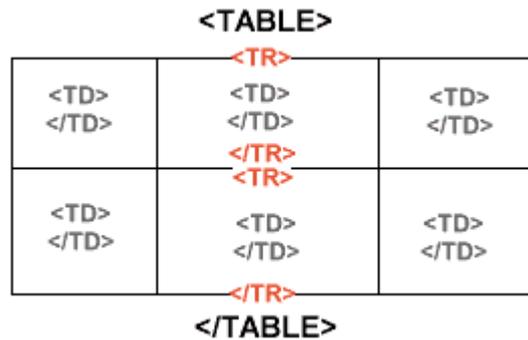


Come si vede da questa immagine, l'attributo <TABLE> genera la tabella, con <TR> si definiscono i vari livelli di riga, mentre con <TD> si definiscono i campi o colonne presenti all'interno della tabella. Ecco come il disegno può essere trasformato in codice e quindi in una vera e propria tabella:

```
<table width="100%" border="1">  
  <tr>  
    <td>prova</td>  
    <td>prova</td>  
    <td>prova</td>  
  </tr>  
</table>
```

prova	prova	prova
-------	-------	-------

Consideriamo un esempio più generico di tabella:



Come si nota il comando <TR> è una sorta di "a capo" all'interno della tabella. Ecco come il disegno può essere trasformato in codice e quindi in una vera e propria tabella:

```

<table width="51%" border="0" cellspacing="0" cellpadding="0">
  <tr>
    <td width="100">prova</td>
    <td width="100">prova</td>
    <td width="100">prova</td>
  </tr>
  <tr>
    <td width="100">prova</td>
    <td width="100">prova</td>
    <td width="100">prova</td>
  </tr>
</table>

```

prova	prova	prova
prova	prova	prova

Lo spazio tra i vari campi di una tabella è definito all'interno del comando </TABLE> con CELLSPACING=X e CELLPADDING=X (dove X è la distanza in pixel):

```

<TABLE CELLPADDING=10 CELLSPACING=8>
</TABLE>

```

La posizione del testo o delle immagini all'interno dei vari campi <TD> della tabella può essere modificata in diversi modi:

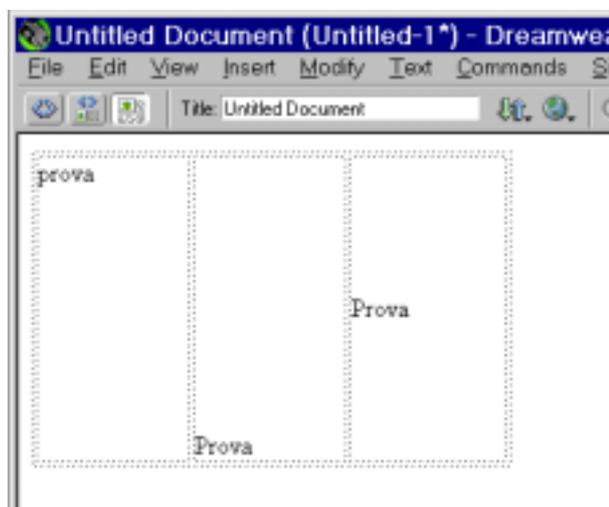
```

<TABLE WIDTH=300
HEIGHT=200>
  <tr>
    <TD width=100
VALIGN=TOP>
prova </TD>

    <TD WIDTH=100
VALIGN=BOTTOM>
Prova </TD>

    <TD WIDTH=100
VALIGN=MIDDLE>
Prova </TD>
  </tr>
</TABLE>

```



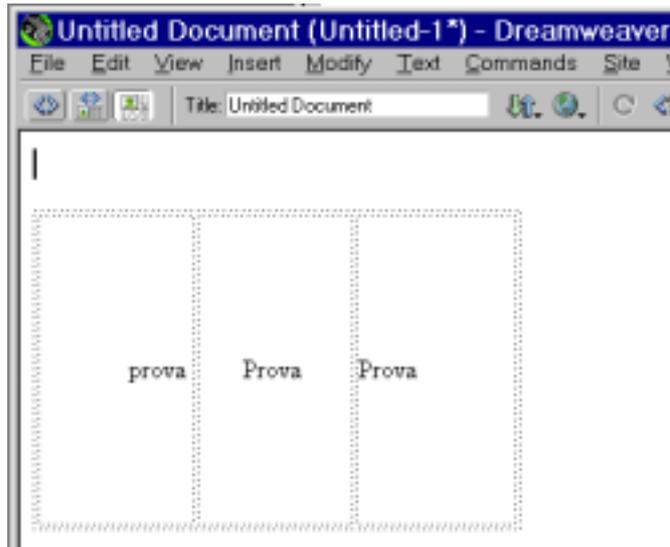
```

<TABLE
WIDTH=300
HEIGHT=200>
<tr>
<TD width=100
ALIGN=RIGHT>
prova </TD>

<TD WIDTH=100
ALIGN=CENTER>
Prova </TD>

<TD WIDTH=100
ALIGN=LEFT>
Prova </TD>
</tr>
</TABLE>

```



Ogni campo può avere un colore di background diverso dagli altri ed addirittura uno sfondo come delle normali pagine web (gli sfondi possono essere sostituiti da GIF animate):

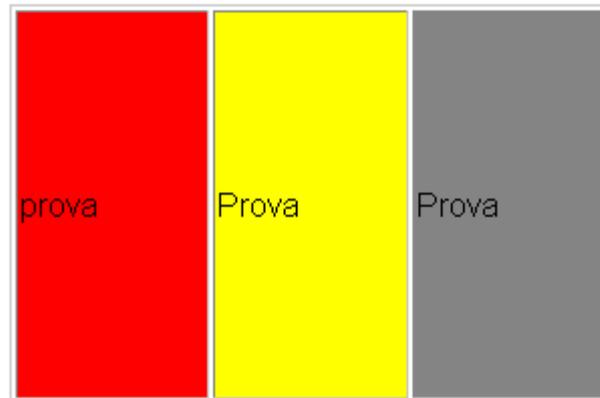
```

<TABLE WIDTH=300
HEIGHT=200>
<tr>
<TD width=100
BGCOLOR="red">
prova </TD>

<TD WIDTH=100
BGCOLOR="yellow">
Prova </TD>

<TD WIDTH=100
BGCOLOR="gray">
Prova </TD>
</tr>
</TABLE>

```



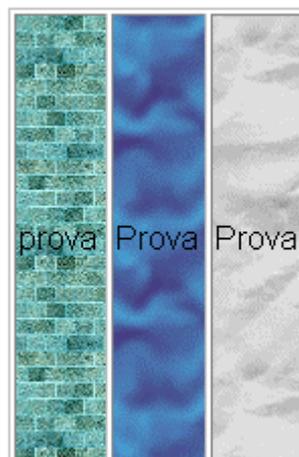
```

<TABLE WIDTH=300 HEIGHT=230>
<tr>
<TD width=100
BACKGROUND="sfondo15.jpg">
prova </TD>

<TD WIDTH=100
BACKGROUND="sfondo26.jpg">
Prova </TD>

<TD WIDTH=100
BACKGROUND="sfondo61.jpg">
Prova </TD>
</tr>
</TABLE>

```



12. Un modo nuovo per formattare i documenti: i CSS CSS o fogli di stile.

HTML soffre di limiti propri di un sistema di contrassegno ideato per scopi molto lontani da quelli attualmente richiesti dal Web design. Se questi limiti appaiono marginali agli occhi dei neofiti, risultano fastidiosi, e molto spesso immobilizzanti per i professionisti formati nella grafica tradizionale. Posizionare un'immagine, creare una banda laterale, giustificare del testo in HTML diventa un problema risolvibile esclusivamente con strumenti nati per tutt'altro scopo (le tabelle <TABLE>, per esempio, nel 90% dei casi vengono utilizzate per posizionare elementi nella pagina, invece che per ordinare dati).

Il problema, in termini più tecnici, riguarda la classica separazione SGML (Standard Generalized Markup Language) tra stile, contenuti e struttura. L'**SGML** è un **meta-linguaggio**, ovvero un insieme di regole generalizzate usate per creare molteplici linguaggi speciali che prendono il nome di markup language. L'HTML è una delle applicazioni più note di **SGML**, rivolto al Web, ma non soddisfa la condizione di separare stile, contenuti e struttura (intesa come il modo in cui si definiscono sia gli elementi con i rispettivi attributi, che le relazioni tra tali elementi stessi), che anzi si trovano tutti inclusi nello stesso file HTML.

Ma cosa significa separare lo stile dai contenuti? Consideriamo un sito Web mediamente complesso, con un numero di pagine HTML pari a 100. Poniamo che il font adottato per l'esposizione degli argomenti sia un "arial" corsivo. Con gli strumenti classici di HTML il codice per ottenere questo risultato è:

```
<FONT FACE="ARIAL"><I>Testo della pagina</I></FONT>
```

ripetuto in tutte le 100 pagine del sito, a chiusura e apertura del testo da formattare. Semplice, ovvio e per molti versi banale. Ma cosa succede nel momento in cui scegliamo di modificare il tipo di carattere a tutte le pagine del sito? Non c'è altra soluzione che aprire le 100 pagine e procedere ad altrettante modifiche, che sostituiscano il nome "arial" con il nuovo font scelto. Un webmaster di medie capacità può impiegare solo qualche decina di minuti, che diventano ore per siti di grandi dimensioni. Questa perdita di tempo è diretta conseguenza della promiscuità tra stile e contenuto, laddove il primo (il tag FONT) non è separato dal secondo (il testo della pagina).

Viene da sé che la soluzione a questo problema è nella separazione tra i due elementi sopra citati, che nella pratica si risolve adottando i **Cascading Style Sheets**. D'ora in poi ci serviremo dell'acronimo CSS per richiamare questi "fogli di stile a cascata" che da tempo sono stati introdotti nel Web publishing, ma che solo ultimamente hanno conosciuto una grande diffusione.

Il termine "a cascata" (cascading) richiama una delle caratteristiche principali di questa tecnologia, per cui è possibile incorporare nel documento differenti fogli di stile, ognuno dei quali, in base a regole gerarchiche, prevale sull'altro.

I CSS sono stati introdotti da Microsoft dalla versione 3 di Internet Explorer, e parzialmente supportati da Netscape soltanto dalla versione 4. Chi accede con un browser obsoleto ad un documento formattato con fogli CSS, si troverà di fronte una pagina gestita dalle opzioni di default del browser (lo sfondo, per esempio, sarà grigio, il font di testo "times new roman", ecc).

I CSS sono stati ufficialmente riconosciuti e standardizzati dal W3C (consorzio internazionale per lo sviluppo del Web) nelle raccomandazioni "CSS1" prima, e "CSS2" poi. Le raccomandazioni sono i documenti ufficiali del W3C, liberamente consultabili per approfondimenti.

L'argomento dei fogli di stile è abbastanza complesso da richiedere una trattazione molto più ampia di quella che si potrebbe fornire in questa sede.

13. <FRAME> creare pagine con i Frame

Come creare e gestire correttamente pagine HTML con frame

I frame (cornici, riquadri) sono strumenti entrati nella consuetudine del Web e perfettamente gestiti da tutti i browser. I denigratori dei frame affermano l'inutilità di suddividere ulteriormente le pagine Web in più pagine che coesistono al loro interno; pagine che in ultima analisi possono risultare poco leggibili. Altri, invece, pensano che i frame possano rendersi utili evitando di caricare parti di pagina che non cambiano mai durante la navigazione e mantenendo ordinata la struttura e i contenuti del sito. Naturalmente abusare dei frame può portare a traguardi di pessima impostazione grafica, giungendo ad un risultato opposto rispetto a quello preventivato.

Una buona soluzione è quella di creare una versione frame ed una versione senza frame. Come si creano i frame?

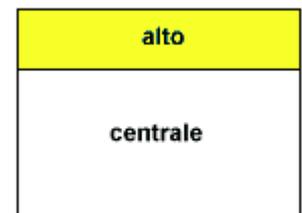
Diamo uno sguardo ai comandi HTML per la gestione dei frame.

Documento Frame	<FRAMESET></FRAMESET>	(al posto di <BODY>)
altezza in righe	<FRAMESET ROWS=,,,></FRAMESET>	(pixel o %)
altezza in righe	<FRAMESET ROWS=*></FRAMESET>	(* = misura relativa)
larghezza in colonne	<FRAMESET COLS=,,,></FRAMESET>	(pixel o %)
larghezza in colonne	<FRAMESET COLS=*></FRAMESET>	(* = misura relativa)
larghezza della cornice	<FRAMESET BORDER=?>	
cornice	<FRAMESET FRAMEBORDER="yes no">	
colore della cornice	<FRAMESET BORDERCOLOR="#\$\$\$\$\$">	
Definizione del Frame	<FRAME>	(contenuto di ogni singolo riquadro)
documento da visualizzare	<FRAME SRC="URL">	
denominazione del frame	<FRAME NAME="*" _blank _self _parent _top>	
larghezza dei margini	<FRAME MARGINWIDTH=?>	(margine destro e sinistro)
altezza dei margini	<FRAME MARGINHEIGHT=?>	(margine alto e basso)
barra di scorrimento?	<FRAME SCROLLING="YES NO AUTO">	

dimensione non modificabile	<FRAME NORESIZE>	
cornice	<FRAME FRAMEBORDER="yes no">	
colore della cornice	<FRAME BORDERCOLOR="#\$\$\$\$\$">	
contenuto in assenza di frame	<NOFRAMES></NOFRAMES>	(per i vecchi browsers)

Per creare una pagina divisa in frame è necessario creare più file HTML richiamati da un unico file principale che è quello che ne permette la gestione. Prima di tutto quindi bisogna impostare questo file "sorgente", e poi successivamente gli altri files che compongono il frame.

Consideriamo di dover creare una finestra divisa in frame come da figura, con un frame in alto fisso (nel quale caricheremo il file "top.htm" da creare a parte) e un frame centrale (nel quale caricheremo il file "central.htm" da creare a parte) il quale cambi a seconda della pagina che verrà visualizzata. Come scritto sopra, questi due frame devono essere gestiti da un terzo file il quale dovrà visualizzarli, assegnando loro una parte della pagina. Ecco il codice di questa pagina che chiamiamo "index.htm":



```
<FRAMESET rows="80,*">
  <frame name="alto" src="top.htm">
  <frame name="centrale" src="central.htm">
</FRAMESET>
```

Come si vede il codice del frame è racchiuso tra i comandi <FRAMESET></FRAMESET> che si comportano come gli usuali comandi <HTML></HTML>.

La grandezza dei frame (o meglio lo spazio che ognuno di essi deve occupare della pagina) è stabilita dal comando **rows="80,***, che sta a significare che il frame alto (che in questo caso è una riga, "rows") deve essere di 80 pixels, mentre quel "*" sta a significare che tutto il resto deve essere assegnato al frame centrale. Avremmo potuto esprimere anche in percentuali le grandezze dei frames, in questo modo:

```
<FRAMESET rows="20%,*">
```

Impostati i due parametri <FRAMESET></FRAMESET> all'interno di essi si definiscono i nomi e i files da richiamare nei due frames creati. È necessario dare un nome ai frames (name="alto") e indicare il file HTML che dentro al frame dovrà essere caricato (SRC="top.htm"). Si devono quindi previamente creare due files di nome "top.htm" e "central.htm".

Nota come sia importante la posizione all'interno del codice per la corretta interpretazione da parte del browser. Se infatti si invertono gli ordini in questo modo:

```
<FRAMESET rows="80,*">
  <frame name="centrale" src="central.htm">
  <frame name="alto" src="top.htm">
</FRAMESET>
```

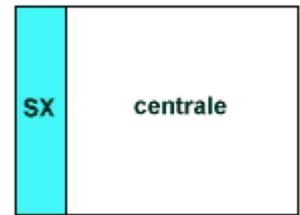
Il browser invertirà l'ordine di assegnazione e caricherà il file "central.htm" nel frame superiore e il file "top.htm" nel frame centrale.

Per creare due frame verticali è sufficiente sostituire a "rows" il termine "cols":

```

<FRAMESET cols="100,*">
  <frame name="sx" src="sx.htm">
  <frame name="centrale" src="central.htm">
</FRAMESET>

```



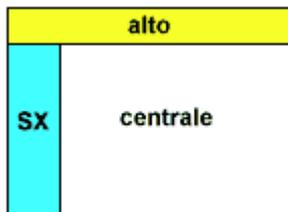
I vecchi browser non supportavano i frame per cui, nella prospettiva che le pagine vengano visualizzate da uno di questi "vecchi" software, e' utile inserire del codice che avverta della presenza di frame e della impossibilit  da parte del browser di visualizzarle. Ecco il codice da inserire:

```

<noframe>
  <HTML>
    <body>
      Attenzione. Il tuo browser non supporta l'opzione dei frame. Per visualizzare queste pagine e' necessario scaricare un browser che supporti tali opzioni. Ti consiglio Netscape 3.0 o superiore.
    </body>
  </html>
</noframe>

```

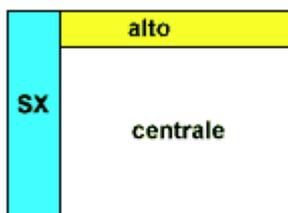
È possibile adottare contemporaneamente una divisione sia in colonne (cols) che in righe (rows), in modo tale da creare una finestra divisa in pi  frame. Vediamo come si deve operare sul codice HTML del documento a seconda del numero e della posizione dei frame che si intendono creare.



```

<frameset rows="100,*">
  <frame name="alto" src="top.htm">
  <frameset cols="150,*">
    <frame name="sx" src="sx.htm.htm">
    <frame name="centrale" src="central.htm">
  </frameset>
</frameset>

```



```

<frameset cols="120,*">
  <frame name="sx" src="sx.htm">
  <frameset rows="100,*">
    <frame name="alto" src="top.htm">
    <frame name="centrale" src="central.htm">
  </frameset>
</frameset>

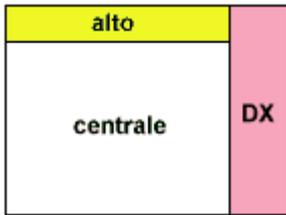
```



```

<frameset cols="120,*">
  <frame name="sx" src="sx.htm">
  <frameset rows="20%,60%,20%,*">
    <frame name="alto" src="top.htm">
    <frame name="centrale" src="central.htm">
    <frame name="basso" src="basso.htm">
  </frameset>
</frameset>

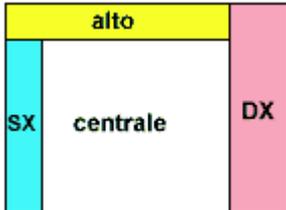
```



```

<frameset cols="75%,25%">
<frameset rows="20%,80%*">
<frame name="alto" src="top.htm">
<frame name="centrale" src="central.htm">
</frameset>
<frame name="dx" src="dx.htm">
</frameset>

```



```

<frameset cols="75%,25%">
<frameset rows="20%,80%*">
<frame name="alto" src="top.htm">
<frameset cols="20%,80%*">
<frame name="sx" src="sx.htm">
<frame name="centrale" src="central.htm">
</frameset> </frameset>
<frame name="dx" src="dx.htm">
</frameset>

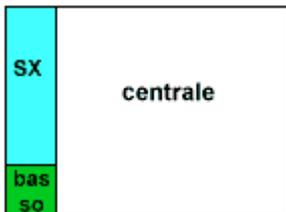
```



```

<frameset cols="75%,25%">
<frameset rows="20%,80%*">
<frame name="alto" src="top.htm">
<frame name="centrale" src="central.htm">
</frameset> <frameset rows="24%,76%">
<frame name="top" src="top2.htm"> <frame name="dx"
src="dx.htm">
</frameset> </frameset>

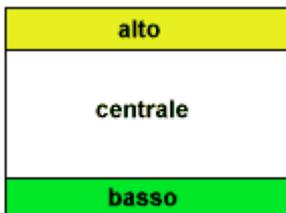
```



```

<frameset cols="25%,75%">
<frameset rows="80%,20%">
<frame name="alto" src="top.htm">
<frame name="basso" src="basso.htm">
</frameset>
<frame name="centrale" src="central.htm">
</frameset>

```



```

<frameset rows="20%,60%,20%">
<frame name="alto" src="top.htm">
<frame name="centrale" src="central.htm">
<frame name="basso" src="basso.htm">
</frameset>

```



```

<frameset cols="20%,60%,20%">
<frame name="sx" src="sx.htm">
<frame name="centrale" src="central.htm">
<frame name="dx" src="dx.htm">
</frameset>

```

Per eliminare il bordo grigio dei frame si deve inserire il seguente codice:

```
<frameset cols="20%,60%,20%" border=0>
```

Per impedire che i frame vengano ridimensionati dal visitatore:

```
<frame name="alto" src="top.htm" noresize>
```

Per eliminare sempre le barre di scorrimento (scrollbars):

```
<frame name="alto" src="top.htm" scrolling="no">
```

Per renderle sempre visibili:

```
<frame name="alto" src="top.htm" scrolling="yes">
```

Per renderle visibili solo quando servono:

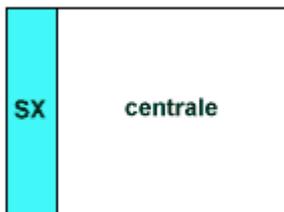
```
<frame name="alto" src="top.htm" scrolling="auto">
```

Per regolare la distanza dal contenuto del frame dal margine alto (marginheight) e sinistro e destro (marginwidth):

```
<frame name="alto" src="top.htm" marginheight=2 marginwidth=5>
```

Per quanto riguarda i link all'interno dei frame (cioè come caricare una pagina in una frame diverso da quello in cui si trova il link) si deve far riferimento al nome che in fase di realizzazione abbiamo assegnato ai vari frame. Nome che non si riferisce al file ma a quanto scritto dopo "name=". Per esempio, nel caso seguente: **<frame name="alto" src="top.htm">** il nome assegnato è "alto".

Consideriamo la seguente pagina divisa in frame:



```
<frameset cols="20%,60%">  
<frame name="sx" src="sx.htm">  
<frame name="centrale" src="central.htm">  
</frameset>
```

Consideriamo che da un link presente su "sx", dobbiamo caricare un'altra pagina sul frame "Centrale". Se il link, presente sul frame "sx", fosse semplicemente:

```
<A HREF="nuova.htm">Clicca</A>
```

la pagina verrebbe caricata all'interno dello stesso frame (cioè "sx"), perché in assenza di comandi adatti il browser capisce di dover caricare la nuova pagina nello stesso frame in cui è presente il link. L'esatto codice da scrivere si ottiene usando l'attributo TARGET:

```
<A HREF="nuova.htm" TARGET="centrale">Clicca</A>
```

Altro uso fondamentale del comando TARGET è quello di richiamare un link ad un'altra pagina la quale verrà visualizzata a pieno schermo, eliminando tutti i frame preesistenti. Ecco il codice:

```
<A HREF="nuova.htm" TARGET="_parent">Clicca</A>
```

oppure

```
<A HREF="nuova.htm" TARGET="_top">Clicca</A>
```

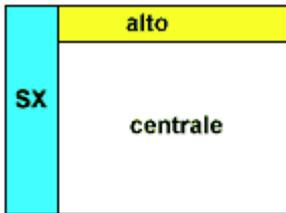
Se inserisci il codice:

```
<base target="_top">
```

in testa al documento HTML tutti i link presenti nella pagine elimineranno i frame esistenti, senza necessità di inserirli in ogni link.

Un'altra necessità può essere quella di caricare, con un solo click, due o più frame (naturalmente la finestra deve essere divisa in almeno tre frame).

Consideriamo una pagina suddivisa come nella figura:



```
<frameset cols="120,*">  
<frame name="sx" src="sx.htm">  
<frameset rows="100,*">  
<frame name="alto2" src="top.htm">  
<frame name="centrale3" src="central.htm">  
</frameset>  
</frameset>
```

La nostra necessità è di inserire un solo link nel frame di "sx" che carichi contemporaneamente due pagine diverse nei due frame di destra: "alto" e "centrale". Per fare questo è necessario inserire qualche riga di codice Javascript. La prima parte del codice va inserita tra <HEAD></HEAD>:

```
<HEAD>  
  <script language="JavaScript">  
    <!--  
      function loadtwo(page2, page3)  
      {  
        parent.alto2.location.href=page2;  
        parent.centrale3.location.href=page3;  
      }  
    // -->  
  </script>  
</HEAD>
```

Mentre la seconda parte va inserita tra <BODY></BODY> dove si vuole inserire il link:

```
<BODY>  
  <FORM NAME="buttons">  
    <INPUT TYPE="button" VALUE="Clicca"  
onClick="loadtwo('nuovo1.htm','nuovo2.htm')">  
  </FORM>  
</BODY>
```

14. <FORM> creare e gestire moduli HTML

Inserire moduli HTML nelle pagine Web.

Prima di passare alla trattazione dei form (moduli), è bene precisare che le difficoltà di comprensione dei moduli HTML sono giustificabili con la singolarità che essi costituiscono nella normale programmazione ipertestuale. Il grosso degli elementi messi a disposizione da HTML, infatti, permette al visitatore di visualizzare i contenuti del nostro sito, ma non di far interagire l'utente con esso. In altre parole il rapporto utente-pagina rimane unidirezionale e statico.

Con i moduli, al contrario, l'utente può interagire con il sito, spedendo un proprio commento, avanzando richieste senza necessità di scrivere via e-mail, firmando guestbook, rispondendo a sondaggi, modificando dinamicamente le pagine secondo i propri interessi, e così via. Tutto questo rende il rapporto bidirezionale ed è possibile solo grazie all'intervento di programmi residenti su server (come ad esempio i CGI Common Gateway Interface, scritti in C o in Perl) o di script eseguibili lato server, (come le ASP Active Server Pages, le PHP Personal Home Page tool, le JSP Java Server Page) che rendono possibile lo scambio di informazioni tra server e client. Questo vuol dire che in realtà la gestione dei moduli dipende in misura preponderante dal server, piuttosto che dalla programmazione HTML in sé e per sé.

A dire il vero, un form può essere creato anche evitando il passaggio per software lato server, ma con svantaggi e limiti che rendono un tale utilizzo poco affidabile e stilisticamente poco piacevole. Nelle pagine seguenti parleremo nel dettaglio anche di questa possibilità.

La creazione di un modulo consta di due fasi:

- impostazione dei tag per la creazione del modulo, dei campi e del tasto di spedizione;
- creazione di uno script CGI su server o di uno script ASP, PHP, JSP incluso nella pagina HTML ma anch'esso eseguibile solo dal server.

I moduli sono stati introdotti dalla versione HTML 2.0. Vediamo, in particolare, tutti i tag che HTML 4.0 prevede per la creazione di form.

<FORM></FORM>

Questo tag apre e chiude il modulo e raccoglie il contenuto dello stesso, che può variare a seconda dei tag che vedremo oltre. Non è possibile inserire un modulo all'interno di un altro. In altre parole i form non permettono nidificazioni.

La sintassi usuale dei tag analizzati è la seguente:

```
<FORM method="get|post" action="http://www.tuosito.com/cgi-bin/nome_script.cgi">
```

Se method è impostato come get i dati vengono spediti al server e separati in due variabili. Per questo metodo il numero massimo di caratteri contenuti nel form è di 255. Utilizzando "method=post" i dati vengono ricevuti direttamente dallo script CGI senza un preventivo processo di decodifica. Questa caratteristica fa sì che lo script possa leggere una quantità illimitata di caratteri.

Impostato il primo tag <FORM> del modulo è possibile inserire gli altri elementi utili al fine di realizzare un modulo per l'acquisizione e la gestione dei dati. Poi, quando l'utente avrà dato conferma all'invio del modulo, la cosa più ovvia da fare sarà di visualizzare una pagina di conferma "il tuo form è stato correttamente inoltrato", ma si può anche rimandare direttamente alla pagina principale del sito.

<INPUT>

Il tag di base per la definizione degli elementi di un form è <INPUT>, che viene utilizzato per aggiungere pulsanti, menu di scelta, password ecc. Gli possono venire assegnati 8 valori differenti.

type="TEXT"

`<INPUT type="TEXT" name="nome" maxlength="40" size="33" value="Il tuo nome">`

Questo valore crea i tipici campi di testo, dove usualmente vengono richiesti dati quali il nome o l'indirizzo e-mail. È un valore usato soprattutto per informazioni non predefinite che variano di volta in volta. TEXT ha tre attributi aggiuntivi presenti anche nell'esempio: **maxlength** (il numero massimo di caratteri inseribili nel campo, oltre il quale non è possibile aggiungere), **size** (la larghezza della stringa all'interno della pagina) e **value** (visualizza un testo di default all'interno della stringa).

ESEMPIO:

`type="PASSWORD"`

`<INPUT type="PASSWORD" name="nome" maxlength="40" size="33">`

Questo campo funziona similmente a TEXT visto in precedenza, ma con una piccola differenza che ne giustifica il nome: quando si digita all'interno della stringa bianca, non appaiono le lettere ma i classici asterischi delle password. In realtà i dati non vengono in alcun modo codificati, per cui rimane un'insicurezza di fondo che questo comando non elimina.

ESEMPIO:

`type="CHECKBOX"`

`<INPUT type="CHECKBOX" name="gusti" value="yes" checked>pasta`
`
<INPUT type="CHECKBOX" name="gusti" value="yes">pizza`
`
<INPUT type="CHECKBOX" name="gusti" value="yes">torte`

Questo attributo viene solitamente utilizzato per informazioni del tipo "si/no" e "vero/falso". Crea delle piccole caselle quadrate da spuntare o da lasciare in bianco. Se la casella è spuntata, input restituisce un valore al CGI, al contrario non restituisce alcun valore. **Value** impostato su **"yes"** significa che di default la casella è spuntata. **Checked** controlla lo stato iniziale della casella, all'atto del caricamento della pagina.

ESEMPIO:

pasta
 pizza
 torte

`type="RADIO"`

`<INPUT type="RADIO" name="giudizio" value="sufficiente"> SUFFICIENTE`
`
<INPUT type="RADIO" name="giudizio" value="buono"> BUONO`
`
<INPUT type="RADIO" name="giudizio" value="ottimo"> OTTIMO`

Questo attributo ha funzioni simili al **CHECKBOX** visto in precedenza, ma presenta più scelte possibili. Selezionando una voce tra quelle presenti, qualora abbiano tutte valore **"name"** identico, si deselezionano automaticamente le altre.

ESEMPIO:

SUFFICIENTE ^{jn}
BUONO ^{jn}
OTTIMO ^{jn}

type="SUBMIT"

`<INPUT type="SUBMIT" value="spedisci">`

È il bottone che invia il form con tutti i suoi contenuti. La grandezza del bottone dipende dalla lunghezza del testo.

ESEMPIO:



type="BUTTON"

`<INPUT type="BUTTON" value="Bottone">`

È il bottone a cui si può assegnare un compito all'interno di un file HTML, generalmente viene associato all'esecuzione di uno script (Visual Basic Script, or JavaScript). La grandezza del bottone dipende dalla lunghezza del testo.

ESEMPIO:

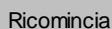


type="RESET"

`<INPUT type="RESET" value="Ricomincia">`

Bottone che reimposta l'intero form eliminando i dati inseriti.

ESEMPIO:



type="IMAGE"

`<INPUT type="IMAGE" src="freccia.gif">`

Funzione simile a quella del tasto "SUBMIT" ma con la differenza che al posto del bottone di default, viene visualizzata un'immagine, scelta da noi.

ESEMPIO:



`type="HIDDEN"`

`<INPUT type="HIDDEN" name="nome" value="valore nascosto">`

Definisce un valore nascosto all'utente che naviga sulla pagina Web, è usato per far viaggiare informazioni di servizio tra il client e il server.

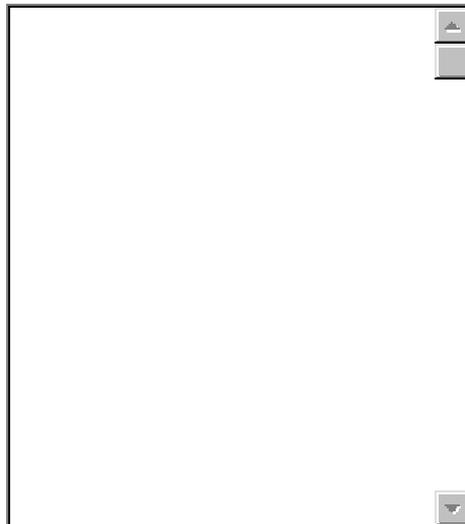
TEXTAREA

`<TEXTAREA cols=40 rows=5 WRAP="physical" name="commento"></textarea>`

Textarea è utilizzato per commenti o campi che prevedono l'inserimento di molto testo. La larghezza è impostata da `"cols"` e l'altezza da `"rows"`.

`WRAP="physical"` stabilisce che qualora il testo inserito superi la larghezza della finestra, venga automaticamente riportato a capo.

ESEMPIO:

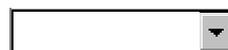


SELECT

```
<SELECT size=1 cols=4 NAME="giudizio">
  <OPTION selected Value=nessuna>
  <OPTION value=buono> Buono
  <OPTION value=sufficiente> Sufficiente
  <OPTION Value=ottimo> Ottimo
</select>
```

Select è un elemento che permette la creazione di elenchi a discesa con varie possibilità di scelta. Nel nostro esempio abbiamo simulato la richiesta di un giudizio su un sito Web, cliccando sulla freccia, troveremo in fila Buono, Sufficiente e Ottimo, quello su cui ci si fermerà, rilasciando il bottone del mouse, sarà il giudizio scelto.

ESEMPIO:



Form anche senza l'ausilio di uno script lato server

Come accennato precedentemente in questo capitolo, l'uso di form HTML dà i migliori risultati quando è affiancato da un programma CGI residente su server o da uno script eseguito anch'esso server side. Questo non preclude, comunque, la possibilità di usare form anche senza l'ausilio di tutto ciò. Per spedire un modulo usando esclusivamente i comandi messi a disposizione da HTML, si deve inserire un'intestazione simile alla seguente:

```
<form action="mailto:nome@provider.it" method="post">
```

Dove a "nome@provider.it" va sostituito il nostro indirizzo email, verso il quale devono essere indirizzati i moduli compilati.

L'inconveniente maggiore di questa scelta è nella mancata formattazione del contenuto del form, nella mancata sistematica archiviazione dei dati ricevuti in un database, nella impossibilità di una risposta immediata alle richieste dell'utente, ecc. In questo, invece, gli script server side agiscono in modo molto efficace. Quando un form viene inviato al server, il valore di ciascuno dei campi <input> corrispondenti è incluso in una singola stringa. Tale stringa è composta da coppie di valori nome-valore ed è delineata da "&" commerciali. Ad essa vengono assegnati tutti i valori degli elementi denominati, in una sorta di concatenamento.

Ecco, di seguito, il confuso risultato di un form nel quale, senza il passaggio per uno script, sono stati inseriti alcuni dati personali:

```
Nome=Pippo+Pluto&E-mail=pippopluto@provider.it&citta%27=Roma&eta%27
```

Se per un modulo molto semplice la scrematura del contenuto vero e proprio è un'operazione rapida, per form lunghi e complessi (e soprattutto numerosi) è una strada non percorribile e molto poco professionale.

15. <MAP> creare mappe cliccabili

Cosa sono, come si creano, quali programmi usare: tutto sulle image map

Le mappe cliccabili sono elementi usati dagli sviluppatori per la facilità d'utilizzo e l'evidente utilità. Il concetto di mappa cliccabile è semplice e parte da una altrettanto semplice constatazione: spesso collegare un unico link ad un'intera immagine limita il web-designer che invece potrebbe linkare diverse pagine sfruttando un singolo gadget grafico. Le mappe cliccabili rimuovono questo limite e consentono, all'interno di una singola immagine, di creare differenti aree di ritaglio, ognuna delle quali con un link diverso. Il concetto fondamentale sul quale si reggono le mappe cliccabili è quello delle coordinate, che uniche possono indicare al browser punti precisi dell'immagine. Esistono due tipi di mappe, ma prima di passare alla loro esposizione è bene precisare che l'uso di estensioni GIF o JPG è ininfluente e non crea alcuna incompatibilità.

Mappe dal lato server (ISMAP)

Non sono molte diffuse. Affinché queste mappe funzionino è necessario che sul tuo server siano presenti le estensioni CGI necessarie. Si tratta di programmi gratuitamente scaricabili in Rete. Le ISMAP, questo il termine tecnico che indica questo genere di mappe, individuano le coordinate dell'immagine attraverso un file esterno. Gli URL creati dalle ISMAP è seguito dal valore delle coordinate x e y, separate da una virgola e precedute da un punto interrogativo. Ecco un esempio:

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

Questo URL indica che nel punto x=25 e y=55 il link è quello stabilito all'interno della ISMAP. Il comando ISMAP sarà dato ad esempio da:

```
<A HREF=" http://www.nome_del_sito.it/cgi-bin/menu.map ">  
  <IMG SRC="menu.gif" ISMAP>  
</A>
```

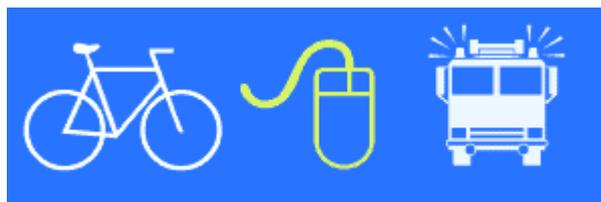
Mappe dal lato client (USEMAP)

Le mappe da lato server non sono molte diffuse soprattutto per la possibilità di ottenere un risultato del tutto simile con una procedura da lato client mediante le USEMAP, cioè le mappe che vengono interpretate senza alcun intervento da parte del server. La sintassi corretta che HTML indica per inserire USEMAP in un documento è la seguente:

```
<IMG SRC="img/home/menu.gif" WIDTH=609 HEIGHT=87 BORDER=0  
usemap="#menu">
```

Dove la prima parte del codice richiama l'immagine e ne stabilisce la grandezza; mentre #menu dà il riferimento al MAP che contiene le informazioni relative alle coordinate. In altre parole le USEMAP non fanno altro che richiamare una mappa dove per ogni link dell'immagine è indicato il relativo URL a cui si andrà a finire.

Per comprendere meglio il funzionamento di una USEMAP prendiamo in considerazione l'immagine che segue:



simuliamo che la nostra esigenza sia quella di creare tre differenti link d'immagine per ognuno dei tre elementi grafici: la bicicletta, il mouse e i pompieri. Se usassimo un normale ancoraggio d'immagine <A HREF> avremmo un unico link per tutti gli elementi grafici. Usando un USEMAP, al contrario, è possibile creare tre differenti aree di link per ognuna delle immagini. Il codice MAP così generato viene inserito all'interno dello stesso documento:

```

<map name="06a">
<area shape="poly" alt="Questa è una bicicletta"
coords="69,26,47,25,52,15,31,15,27,19,35,28,37,34,34,39,20,37,12,
44,7,51,5,57,5,65,8,71,9,75,16,79,24,81,31,81,40,79,44,73,50,67,
55,65,65,56,65,66,68,72,73,79,91,84,98,80,110,64,106,53,100,43,87,
37,82,38,75,23,85,24,82,17,68,17,71,25,71,25,71,25,71,25" href="bicicletta.htm"
title="Questa è una bicicletta">
<area shape="poly" alt="Questo è un mouse"
coords="181,27,172,14,163,10,150,14,136,23,136,34,128,40,123,32,112,
31,115,42,121,52,132,54,139,49,146,38,150,33,152,48,148,64,153,80,
161,82,172,82,180,76,183,53,181,34,181,24,181,25" href="mouse.htm" title="Questo è
un mouse">
<area shape="rect" alt="Camion dei pompieri" coords="207,7,279,82"
href="pompieri.htm" title="Camion dei pompieri">
</map>

```

Il codice è contenuto all'interno dei tag MAP, che chiudono e aprono la sintassi. Le tre aree di ritaglio sono generate da altrettanti campi AREA. Esistono tre differenti modi per ritagliare un'immagine:

- **poly**: ritaglia immagini frastagliate attraverso linee rette
- **rect**: ritaglia aree quadrate o rettangolari di 4 lati
- **circle**: ritaglia aree circolari

Nel nostro esempio sono presenti aree di ritaglio poly e rect. I commenti alle singole aree di ritaglio sono generati dai campi TITLE e ALT e fanno in modo che il browser visualizzi testo descrittivo per ogni singola area di link.

Creare il codice MAP è necessario inserire all'interno dell'immagine un riferimento allo stesso. Per fare questo, nel nostro esempio abbiamo impostato il seguente codice:

```



```

La dicitura usemap="#06a" richiama il codice impostato in precedenza ed assegna alle immagini le aree di ritaglio.

Creare delle aree di ritaglio non è affatto semplice se non si dispone di un programma adatto allo scopo che eviti un calcolo manuale. La gran parte degli editor HTML presenti sul mercato prevede generatori automatici di mappe, ed ai file di help di ognuno di essi rimandiamo per un approfondimento ulteriore.

16. <APPLET> inserire Applet Java nel documento

Come inserire le applet Java in un file HTML

Le applet Java sono file .class pre-compilati da un compilatore Java e inseribili nelle pagine Web. Tale inserimento richiede di fare attenzione ai parametri modificabili da richiamare all'interno del file HTML che ospita l'applet.

Il codice di richiamo dell'applet va inserito compreso tra:

```
<APPLET>
```

```
</APPLET>
```

Vanno copiati i file .class e le eventuali immagini utilizzate dall'applet nello spazio Web del sito. Se si vogliono richiamare i file .class da una directory diversa da quella in cui è depositato il file HTML, bisogna impostare il parametro Codebase, come segue:

```
<APPLET CODE="XXXXX.class" CODEBASE="http://www.nomesito.com/java">
```

Ovviamente a "XXXXX.class" va sostituito il nome del file class, e a "http://www.nomesito.com/java" il percorso per raggiungere lo stesso.

Il Java è supportato dalle Versioni 2 e successive di Netscape, dalle versioni 3 e successive di MS Explorer, HotJava e altri browser, in sistemi operativi a 32 bit (Windows 95/NT, Mac OS, Sparc, Linux, etc.). Le applet Java non appariranno su Windows 3.1, o in navigatori a 16 bit (Netscape 4 per Win 3.1 supporta il java ma non bene come Win 95). Inoltre, il supporto del Java deve essere abilitato nei browser, altrimenti vedrai solo un messaggio che dice che il browser non supporta il java.

Gli errori più comuni

A) NOMI LUNGHICI DI FILE SPEZZATI E CAMBIAMENTO DI MAIUSCOLE/MINUSCOLE: Molti utenti MS-DOS usano ancora il vecchio PKUNZIP per decomprimere gli archivi zip, oppure programmi a 16 bit per copiare i files sul sito, senza prendere in considerazione la differenza tra FILE.ExE, FILE.EXE e file.eXe. Questo è male per varie ragioni:

La prima è l'uso dei nomi lunghi per i file: l'MS-DOS (e i vecchi programmi a 16 bit per win 3.1) sono in grado di supportare solamente i nomi di files con 8+3 caratteri di estensione. I sistemi moderni invece supportano i nomi lunghi di file, per esempio MyNiceApplet.class, che ha un suffisso (class) di 5 caratteri. Se si usa PKUNZIP per scompattare l'archivio delle applet, o si copiano i files .class sul server con un vecchio programma FTP per Windows 3.1 o simili, il nome del file sarà troncato e il risultato sarà MYNICEAP.CLA e provando a far partire l'applet, sarà visualizzato un errore "java.lang.ClassNotFound". Assicuratevi quindi di scompattare usando "winzip 32" o simili sotto win95, e di copiare i files con i nuovi programmi che non troncano il nome dei files.

Occorre inoltre considerare anche la differenza tra maiuscole e minuscole, perché molti server su Internet (Unix) sono case sensitive, ovvero distinguono anche tra maiuscole e minuscole. Per esempio, se il nome dell'applet è "XxXxxxxxx.class" e si scrive, invece, "xxxxxxxxx.class", l'applet non funzionerà. Gli errori più comuni sono nei nomi delle immagini, perché in locale Windows 95 non fa distinzione tra maiuscole e minuscole e quindi carica le immagini anche se su Internet sarebbero considerate con nome diverso.

Ad esempio, se si salva un'immagine come "image1.jpg", se si usa il sistema operativo Windows, si potrà caricarla dall'hard disk locale anche indicandola come IMAGE1.JPG, image1.JPG o IMAGE1.jpg. Ma una volta trasferita sul server del provider (o comunque sul vostro sito) funzionerà soltanto se indicata come image1.jpg. Assicuratevi sempre quindi che i nomi delle immagini caricate da un'applet siano identici, comprese la maiuscole e minuscole.

B) Dimensione sbagliata di immagini o applet. Le immagini normalmente devono essere di dimensioni stabilite (ad esempio 64*64, 128*128 ecc.). Inoltre, è bene non allargare le applet troppo, e immagini più larghe di 600 pixels non saranno visibili pienamente a chi usa un modo video in 640x480.

17. Le nuove specifiche di HTML 4

Le maggiori novità introdotte dalla quarta versione dello standard HTML

L'affermazione su scala planetaria della Rete quale strumento comunicativo orizzontale ha creato, tra gli altri, l'inedita figura dell'utente programmatore. A chi scrive non sfugge la differenza tra l'HTML e i veri e propri linguaggi di programmazione (VBasic, Java e altri), ma il punto che preme sottolineare è l'affermazione di un utente attivo e "propositivo", che non si limita a raccogliere informazioni, ma a proporre piccoli o grandi contenuti verso la comunità.

Questo sviluppo rivoluzionario della distribuzione di informazioni è in massima parte dovuto alle peculiarità del nuovo media Internet, ma anche alla semplicità con cui è possibile creare documenti per il Web. Ovvio che si stia parlando di HTML che dal 1989 regge ininterrottamente le sorti del Web.

L'HTML ha avuto, ed ha, uno sviluppo impetuoso e per molti versi simile a quello di internet. Uno sviluppo che si è cercato di regolare per la prima volta nel novembre del 1995, con la definizione da parte dell'*Internet Engineering Task Force* (IETF) dell'HTML 2.0. Questa prima versione ufficiale nacque già sorpassata dallo sviluppo di questo linguaggio, tanto da non tenere in conto le tabelle, che i browser presenti sul mercato avevano già adottato da tempo. Nel marzo del 1995 il W3C (*World Wide Web Consortium*) iniziò a lavorare su una versione 3.0 destinata ad apportare numerose e fondamentali modifiche alla versione precedente. Ma il W3C si trovò ad operare mentre Microsoft e Netscape rilasciavano versioni dei rispettivi browser contenenti nuovi tag i quali, in ultima analisi, rendevano impossibile la definizione di uno standard. Così il W3C abbandonò la stesura della versione 3.0 per passare direttamente alla 3.2. Il risultato, a dire il vero, non fu certo dei più esaltanti. Le maggiori novità della versione 3.2 sono state: le tabelle, i font colorati, gli applet java, i tag superscript e subscript e le Client Side Image. Come si vede, tutti elementi da tempo supportati dai browser e adottati dai webmaster per la creazione di siti internet.

Ripercorrendo la storia dell'HTML, quindi, è di tutta evidenza il gap tra ciò che le "istituzioni ufficiali" stabiliscono come standard, e ciò che in realtà avviene nel web. Questa situazione è imputabile, da un lato alle strategie commerciali dei due grandi concorrenti (MS e Netscape), dall'altro alla genetica "frenesia" della rete. Questa sorta di "doppia velocità" non ha finora permesso la definizione di uno standard per l'HTML. Non è infrequente che i due browser principali adottino, per generare uno stesso effetto, comandi diversi ed incompatibili; a danno, è ovvio affermarlo, dei webmaster che si trovano a fare una scelta di campo tra uno dei due prodotti.

Il W3C si è impegnato a definire ufficialmente il nuovo standard (in fase di studio chiamato "Cougar"), finalmente espresso nel "*W3C Proposed Recommendation*". È nato quindi l'HTML 4.0 con propositi e prospettive diverse dai precedenti tentativi. A differenza del passato sia Netscape che Microsoft si sono impegnate ad implementare i lavori del W3C, e soprattutto ad accettarne i risultati rendendo i propri browser compatibili. È bene ricordare che il W3C è un consorzio formato da: Adobe, Hewlett Packard, IBM, Microsoft, Netscape, Novell, SoftQuad e Sun Microsystems.

L'HTML 4.0 segna il definitivo avvicinamento al Dynamic HTML, ma non è certo questa l'unica importante innovazione:

Caratteri

È stato adottato il set di caratteri [ISO/IEC: 10646](#), che include qualsiasi tipo di carattere in qualsiasi lingua. Ciò vuol dire che il W3C ha tenuto conto dello sviluppo planetario del web, e che sarà possibile inserire in una stessa pagina idiomi o caratteri di lingua differenti.

Accessibilità

HTML 4.0 tiene conto delle limitazioni incontrate dai portatori di handicap, e prevede diversi strumenti per facilitarne l'accesso a documenti ipertestuali:

- distinzione tra struttura base e contenuto di un documento attraverso i fogli di stile;
- tabelle che prevedono la possibilità di accedere ai non vedenti;
- form che permettono di essere selezionati attraverso tasti di scelta rapida;
- mappe cliccabili che possono visualizzare testo in alternativa alle immagini;
- supporto dei comandi [title](#) e [lang](#) per tutti gli elementi;
- descrizioni lunghe per tabelle, immagini, frames ecc.;
- supporto degli elementi [ABBR](#) e [ACRONYM](#);
- possibilità di usare, all'interno di fogli di stile, linguaggi di comunicazione quali tty, braille ed altri.

Object

Nell'HTML 4.0 potranno essere inseriti elementi eseguibili e multimediali, per fare questo il marcatore **OBJECT** sostituirà gli attuali **IMG** e **APPLET**. Questo marcatore omnicomprendente permette di includere immagini, video, suoni e programmi eseguibili nel documento, attraverso una tecnica gerarchica per specificare varie alternative in base all'ambiente in cui si trova ad operare il browser dell'utente.

Facendo un esempio riguardo il marcatore **OBJECT** e la sua caratteristica di generare alternative, verrà di seguito riportato un codice che tenta di visualizzare in alternativa (1) un applet che genera l'immagine animata di una bandiera sventolante; la stessa animazione in formato Mpeg (2), ed infine un'immagine fissa JPG della bandiera:

```
<OBJECT title="Una bandiera" classid="java:Flag.class" width="300" height="200">  
<OBJECT data="flag.mpeg" type application/mpeg">  
<OBJECT src="flag.jpg">
```

```
<FONT SIZE=4>Una bandiera</FONT>
```

```
</OBJECT>  
</OBJECT>  
</OBJECT>
```

Fogli di stile CSS

L'HTML è stato progettato per gestire singole pagine. Se ciò ha generato la definizione di un linguaggio semplice e diretto, non è più aderente alle necessità di siti spesso composti da migliaia di pagine. A questo si affianca il problema incontrato dai grafici professionisti riguardo la difficoltà di posizionamento all'interno del documento di immagini e testo.

A queste problematiche HTML 4.0 cerca di far fronte adottando i fogli di stile, i quali possono essere inseriti all'interno di un documento ipertestuale in tre modi:

- attraverso un file esterno (metodo preferibile);
- attraverso l'intestazione generale di ogni singolo documento;
- o attraverso la specifica degli attributi nel singolo tag.

Sfruttando i fogli di stile è possibile modificare la struttura di diverse pagine agendo su un unico file esterno che ne determina l'aspetto.

I fogli di stile sono destinati a sostituire l'abitudine, sempre più frequente, di layout di pagina creati attraverso strumenti impropri quali le tabelle. La marcatura HTML dovrà, quindi, tornare alla sua originaria funzione di definizione della struttura logica del documento, lasciando ai fogli di stile la gestione del layout di pagina.

I fogli di stile non necessariamente debbono essere HTML. Alcuni linguaggi specializzati come il CSS (Cascading Style Sheets) permettono di riunire direttive di stile provenienti da fonti diverse, facendo prevalere le une alle altre in base alle scelte del webmaster che compone il file HTML.

Tabelle

Il modello definitivo di tabelle dell'HTML 4.0 riprende la prima stesura dell'HTML+ (tale modello è stato una estensione temporanea per rispondere a requisiti di controllo nella presentazione dati, richiesti dalle nuove necessità della rete).

HTML 4.0 permette un notevole controllo sulle tabelle (basate sulla RFC1942), che sono esclusivamente orientate alla presentazione dati, e non, come avviene attualmente, alla definizione del layout di pagina.

È stato introdotto, tra le altre cose, il nuovo elemento COLGROUP che permette di creare gruppi di colonne, specificandone allineamento e larghezza.

Nello specifico, riprendendo quanto ufficialmente riportato nel *W3C Proposed Recommendation*, le più importanti innovazioni alle tabelle sono:

- possibilità di allineamento per caratteri indicati come "." e ":" (per esempio, sarà possibile allineare una colonna di numeri sul punto decimale);
- introduzione di maggior flessibilità;
- possibilità di ottenere tabelle scorrevoli con testate fisse, più un supporto adeguato per le tabelle interrotte durante le pagine di stampa;
- è stato introdotto un nuovo elemento che permette di raggruppare una serie di colonne con larghezza e proprietà di allineamento differenti da uno o più elementi, e `rules="basic"` è stato sostituito da `rules="groups"`.

Form

Sono stati introdotti nuovi attributi destinati a risolvere le lacune dell'HTML 3.2:

- l'attributo `accesskey` provvede a specificare un accesso diretto tramite tastiera ai campi del form;
- con `readonly`, un attributo supplementare, i webmaster possono vietare i cambiamenti ai campi del form;
- l'elemento `LABEL` aggiunge un'etichetta con un particolare form di controllo.
L'elemento `FIELDSET` raggruppa campi in relazione fra di loro e, in associazione con l'elemento `LEGEND`, può essere usato per nominare il gruppo. Entrambi questi nuovi elementi, permettono una migliore interpretazione e interazione;
- una nuova serie di attributi `onchange-INPUT`, in associazione con il supporto per i linguaggi script, permette ai form dei provider di verificare i dati inseriti dall'utente;
- l'elemento `INPUT` ha un nuovo attributo `accept` che permette agli autori di specificare una lista di media o tipi di struttura per l'input;
- il nuovo elemento `BUTTON` può essere usato per arricchire i form in modo diverso oltre ai classici "submit" e "reset".

Scripting

HTML 4.0 prevede la possibilità di associare script ai form, in modo da generare una maggior interattività con l'utente.

Stampa

HTML 4.0 permette di stampare pagine Web senza per questo ottenere una pura e semplice riproduzione della pagina, tout court. Il programmatore HTML 4.0 ha la possibilità di definire ciò che verrà stampato, e come verrà stampato, quando l'utente farà click sul pulsante "Stampa" del proprio browser.

Modifiche degli elementi rispetto a HTML 3.2

I nuovi elementi dell'HTML 4.0 sono:

[Q](#), [INS](#), [DEL](#), [ACRONYM](#), [LEGEND](#), [COLGROUP](#), [BUTTON](#), [FIELDSET](#).

Gli elementi disapprovati:

[ISINDEX](#), [APPLET](#), [CENTER](#), [FONT](#), [BASEFONT](#), [STRIKE](#), [S](#), [U](#), [DIR](#), [MENU](#).

Gli elementi ritenuti obsoleti (deprecati):

[XMP](#), [PLAINTEXT](#), [LISTING](#)

al posto dei quali va usato esclusivamente il comando [PRE](#).