

1. I Link

1.1. I link e l'ipertestualità

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

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

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

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

1.2. Link che puntano ad altri documenti

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

Le risorse per webmaster sono su `nome sito`.

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

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

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

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

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

Che dà come risultato: [Mandami un'e-mail](#).

1.3. I percorsi assoluti e relativi

Percorsi assoluti

Fino a quando ci troviamo nella condizione di creare un sito web di dimensioni ridotte (poche pagine) non avremo problemi di complessità, e possiamo anche ipotizzare di lasciare tutti i nostri file in una medesima cartella. È evidente però che – man mano che il nostro sito web cresce – avremo bisogno di un maggior ordine. Si presenterà allora l'esigenza di inserire le immagini del sito in una cartelle diverse (in modo da averle tutte nella medesima locazione), e magari sarà opportuno dividere il sito in varie sezioni, in modo da avere tutti i documenti dello stesso tipo all'interno di un contesto omogeneo.

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

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

Per farlo esistono due tecniche:

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

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

```
Leggi le risorse sui <a href="http://www.nomesito.it/css/index.html">fogli di stile</a>
```

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

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

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

Percorsi relativi

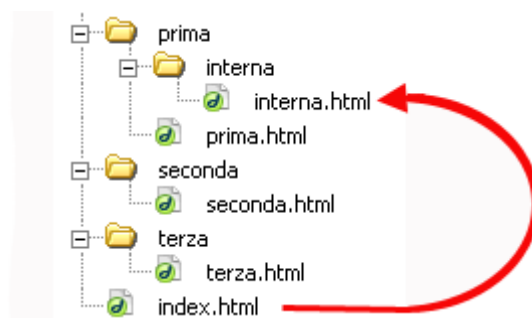
Spesso vi troverete tuttavia a fare riferimento a documenti situati nel vostro stesso sito, e – se state sviluppando il sito sul vostro computer di casa (cioè "in locale") – magari non avete ancora un indirizzo web, e non sapete di conseguenza come impostare i percorsi. È utile allora capire come funzionano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

I percorsi relativi fanno riferimento alla posizione degli altri file rispetto al documento in cui ci si trova in quel momento.

Per linkare due pagine che si trovano all'interno della stessa directory è sufficiente scrivere:

```
<a href="paginaDaLinkare.html">collegamento alla pagina da linkare nella stessa directory della pagina presente</a>
```

Poniamo ora di trovarci in una situazione di questo genere:

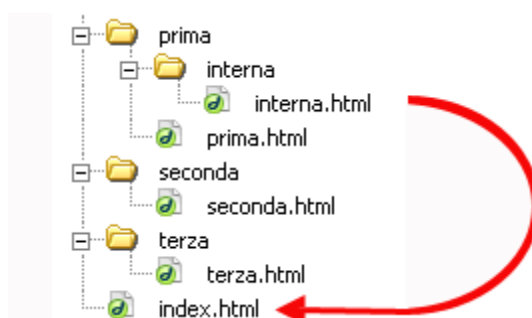


Dalla pagina "index.html" vogliamo cioè far riferimento al file "interna.html", che si trova all'interno della directory "interna", che a sua volta si trova all'interno della directory "prima".

La sintassi è la seguente:

```
<a href="prima/interna/interna.html">Visita la pagina interna</a>
```

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



La sintassi è la seguente:

```
<a href="../../index.html">Visita la pagina interna</a>
```

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

Per far riferimento a un file che si trovi all'interno della stessa directory basta linkare il nome del file	collegamento alla pagina
--	---

<p>Per far riferimento a un file contenuto in una cartella di livello inferiore alla posizione corrente, basta nominare la cartella seguita dallo "slash", e poi il nome del file.</p> <p>Secondo la formula: cartella/nomeFile.html</p> <p>Per tornare su di un livello, è sufficiente utilizzare la notazione: ../nomeFile.html</p>	<pre>Visita la pagina interna</pre> <pre>Visita la pagina interna</pre>
---	--

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

```
../altraCartella/nuovoFile.html
```

Approfondimenti

A volte potrete incontrare la notazione:

```
Leggi le risorse sui <a href="/css/index.html">fogli di stile</a>
```

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

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

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

è sufficiente indicare la directory:

```
http://www.nomesito.it/css/
```

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

Consigli per i nomi dei file

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

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

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

```
<a href="file:///C|percorso\nomeFile.html">testo</A>
```

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

1.4. *I link interni o ancore*

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

```
<a name="primo">Stiamo per esaminare la struttura... Eccetera...</a>
```

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

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

```
<a href="#primo">vai al primo paragrafo</a>
```

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

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

```
<a href="#">link vuoto</a>
```

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

- creazione dell'ancora a cui puntare (****)
- creazione del collegamento all'ancora appena creata e riferimento attraverso il cancelletto (****)

È bene non confondere le due fasi.

1.5. *Gli attributi dei link*

target

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

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

cioè:

```
visita nomesito.IT
```

vedremo questo attributo più in dettaglio quando parleremo dei frames.

title

L'attributo **title** è molto importante, e serve per specificare un testo esplicativo per l'elemento a cui l'attributo è riferito (il title si può infatti utilizzare anche per elementi differenti dalle ancore). Questa spiegazione addizionale favorisce l'accessibilità del sito anche ai disabili, alle persone per esempio che hanno disturbi alla vista.

Se lasciate il cursore del mouse per qualche secondo su un collegamento dotato di title, vedrete comparire una specie di etichetta con il testo specificato nel title:

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

cioè:

[visita nomesito.IT](http://www.nomesito.it/)

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

hreflang

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

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

cioè:

Nel sito del [Word Wide Web Consortium](http://www.w3c.org/) puoi trovare le specifiche dell'HTML in lingua inglese

accesskey

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

Per esempio potreste specificare che:

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

cioè:

[visita nomesito.it](http://www.nomesito.it/)

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

1.6. Colorare i link

Abbiamo già visto come colorare i link in tutta la pagina.

Possiamo però aver bisogno di colorare alcuni link della pagina in modo diverso.

Per farlo è sufficiente annidare il tag all'interno del link:

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

cioè:

[Torna all'home page di nomesito.it](http://www.nomesito.it)

ovviamente il modo giusto per colorare i link non è quello di utilizzare il tag font, ma quello di utilizzare i fogli di stile.

1.7. Il tag **BASE**

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

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

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

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

sicuro che farà riferimento a:

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

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

2. Le immagini e le mappe di immagine

2.1. Inserire le immagini

Finora abbiamo visto come inserire e formattare il testo all'interno delle nostre pagine Web. Naturalmente possiamo inserire anche delle immagini: diagrammi e grafici, fotografie, e in genere immagini create con un programma di elaborazione grafica (come The GIMP, Photoshop o Paint Shop Pro).

I formati ammessi nel Web sono sostanzialmente tre:

- **GIF (Graphic Interchange Format)**. Le GIF sono immagini con non più di 256 colori (dunque con colori piatti e senza sfumature), come grafici o icone
- **JPG**: è l'acronimo del gruppo di ricerca che ha ideato questo formato (il **Joint Photographic Experts Group**), idoneo per le immagini di qualità fotografica
- **PNG (Portable Network Graphic)**. Il PNG è un tipo di immagine introdotto abbastanza di recente, elaborato dal **W3C** per risolvere i problemi di copyright del formato GIF (che è appunto proprietario); tuttavia oggi il PNG è letto oramai da tutti i browser e offre alcune caratteristiche che gli altri formati non hanno (come il supporto al canale alfa, caratteristica questa non ancora perfettamente supportata da ogni browser).

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

Inoltre è importante ricordare che il codice HTML fornisce delle indicazioni al browser su come visualizzare il testo e le immagini - ed eventualmente i video e i suoni - all'interno della pagina: il testo (come abbiamo

visto) è scritto direttamente nel file HTML, le immagini invece sono caricate insieme alla pagina. Attenzione dunque a non inserire immagini troppo pesanti (ricordatevi di ottimizzare sempre i file); bisogna evitare inoltre di sovraccaricare la pagina con troppe immagini. Allo stato attuale dell'arte infatti la maggior parte degli utenti (e non soltanto quelli italiani) naviga ancora con un modem analogico da 56 Kbs: inserire troppe immagini significa dunque creare pagine lente da caricare. Per ottenere un sito web dalla grafica accattivante, spesso è sufficiente giocare con i colori dello sfondo e delle scritte.

La sintassi per inserire un'immagine è:

```

```

dove:

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

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

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

```
  

```

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

```

```

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

L'attributo **alt** è di estrema utilità per rendere il sito accessibile a tutti gli utenti: i disabili che non sono in grado di vedere nitidamente le immagini sullo schermo potrebbero avere delle difficoltà, nel caso in cui l'attributo alt non sia specificato. Gli ipo-vedenti e i non-vedenti sono infatti in grado di comprendere il contenuto delle immagini grazie a dei software appositi (gli **screen reader**) che "leggono" lo schermo tramite un programma di sintesi vocale. Non specificare il testo alternativo significa rendere impossibile la navigazione.

Nel caso in cui la spiegazione dell'immagine sia particolarmente lunga, è possibile espandere la descrizione sintetica - fornita tramite l'attributo "alt" - grazie ad un altro attributo: si tratta di **longdesc (long description)**, che permette di specificare un file con una spiegazione estesa dell'immagine. Ecco la sintassi:

```

```

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

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

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

```

```

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

```
<a href="http://www.nomesito.it"
target=" blank">
  
</a>
```

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

```
<a href="http://www.nomesito.it"
target=" blank">
  
</a>
```

2.2. *Disporre le immagini in un contesto*


Se inserita in un testo, normalmente un'immagine va a capo. Così:

```
<p>Nomesito.it &grave; il primo sito italiano sul web
publishing  con centinaia di esempi e guide esplicative </p>
```

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

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

```
<p> 
Nomesito.it &grave; il primo sito italiano sul web publishing, con centinaia di esempi e guide esplicative </p>
```



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

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

```
<p> 
Nomesito.it &grave; il primo sito italiano sul web publishing, con centinaia di esempi e guide esplicative </p>
```

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



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

Altri valori possibili sono:

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

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

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

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

```

```

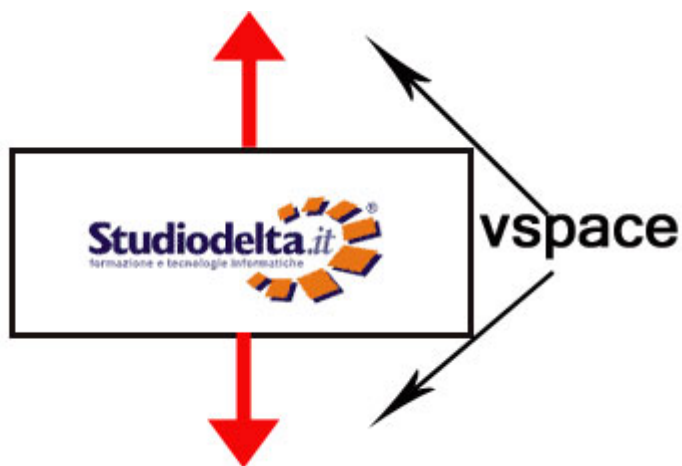


Nel caso di vspace lo spazio è verticale, ma sempre da ambo i lati:

```

```

cioè:



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

```

```

Approfondimenti

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

2.3. Le mappe di immagine

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

In questo caso possiamo utilizzare le mappe.

Ne esistono due tipi:

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

Le mappe lato-client

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

```

```

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

A questo punto non ci resta che creare la mappa:

```
<map name="nomeMappa" >  
...  
</map>
```

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

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



```
  
  
<map name="regioni" id="regioni" >  
  
  <area shape="rect" coords="14,24,35,37" href="http://www.regione.vda.it/"  
target=" blank" alt="Valle d'Aosta" >  
  
  <area shape="circle" coords="45,156,29" href="http://www.regione.sardegna.it/"  
Target=" blank" alt="Sardegna" >  
  
  <area shape="poly" coords="105, 199, 115, 197, 121, 200, 131, 201, 139, 198, 150,  
197, 156, 195, 151, 201, 145, 209, 148, 212, 150, 219, 152, 225, 147, 227, 144, 231, 128,  
221, 119, 219, 113, 212, 108, 212, 102, 210, 98, 205" >  
href="http://www.regione.sicilia.it/" target=" blank" alt="Sicilia" >  
  
</map>
```

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

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

Forma	Descrizione
rettangolare <area shape="rect">	è necessario specificare alcune coordinate del rettangolo per individuare i vertici. In particolare sono da specificare (in quest'ordine): <ul style="list-style-type: none">• la x del lato di sinistra• la y del lato alto• la x del lato destro• la y del lato in basso
circolare <area shape="circle">	è necessario specificare le coordinate del centro (x e y) e la misura del raggio (in pixel)
poligonali <area shape="poly">	è necessario specificare tutte le coordinate del poligono a coppie

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

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

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

In alternativa potete utilizzare dei programmi appositi, come CoffeCup Image Mapper, CuteMap o MapEdit

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

Approfondimenti

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

```
onFocus='this.blur()'
```

da applicare al tag **<AREA>** in questo modo:

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

3. Le tabelle

3.1. Tabella: struttura di base

Le tabelle sono una delle parti più importanti di tutto il codice HTML: nate sin dagli inizi del Web per impaginare dati aggregati, si sono poi trasformate in uno strumento indispensabile per gestire i layout grafici. Il loro ampio utilizzo all'interno dei documenti ha fatto sì che – nel passaggio dall'HTML 3.2 all'HTML 4 - le specifiche delle tabelle venissero estese con una serie di notazioni destinate a "far ordine" all'interno di un codice che rischiava di diventare troppo vasto.

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

`<table>` apre la tabella

`<tr>` "table row": indica l'apertura di una riga

`<td>` "table data": indica una cella all'interno di una riga

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

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

```
<table border="2">
```

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

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

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

Che viene visualizzato così:

prima cella	seconda cella
terza cella	quarta cella

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

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

  <tr>
```

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

Che viene visualizzato così:

prima cella	seconda cella
terza cella	quarta cella

Oppure:

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

Che viene visualizzato così:

prima cella	seconda cella
terza cella	quarta cella

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

Come detto inizialmente le tabelle vanno immaginate come delle griglie, tutto sommato abbastanza rigide: l'eventuale larghezza specificata nelle celle della prima riga avrà effetto dunque anche sulle celle delle righe sottostanti. Viceversa non è possibile variare arbitrariamente le dimensioni delle celle: le misure specificate nelle righe sottostanti non avranno infatti effetto.

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

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

3.2. *Raggruppare celle con rowspan e colspan*

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

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

	<td colspan="2">	

Il cui codice corrispondente è:

```
<table width="430" border="1" bordercolor="#000000">
  <tr>
    <td width="30%"> <br> <br> <br> </td>
    <td width="30%">&nbsp;</td>
    <td width="30%">&nbsp;</td>
  </tr>
  <tr>
    <td><br> <br> <br> </td>
    <td colspan="2">&nbsp;</td>
  </tr>
</table>
```

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

	<td rowspan="2">	

il cui codice corrispondente è:

```
<table width="430" border="1" bordercolor="#000000">
  <tr>
    <td width="30%"> <br> <br> <br> </td>
    <td width="30%" rowspan="2">&nbsp;</td>
    <td width="30%">&nbsp;</td>
  </tr>
  <tr>
    <td><br> <br> <br> </td>
    <td>&nbsp;</td>
  </tr>
</table>
```

3.3. Annidare tabelle

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

```
<table width="430" border="1">
  <tr>
    <td width="50%">&nbsp;</td>
    <td width="50%">&nbsp;</td>
  </tr>
  <tr>
    <td height="24">&nbsp;</td>
    <td><table width="100%" border="1">
      <tr>
        <td>&nbsp;</td>
```

```

        <td>&nbsp;</td>
        <td>&nbsp;</td>
    </tr>
    <tr>
        <td>&nbsp;</td>
        <td>&nbsp;</td>
        <td>&nbsp;</td>
    </tr>
</table></td>
</tr>
</table>

```

che dà come risultato:

	<table border="1"> <tr> <td></td> <td></td> <td></td> </tr> <tr> <td></td> <td></td> <td></td> </tr> </table>						

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

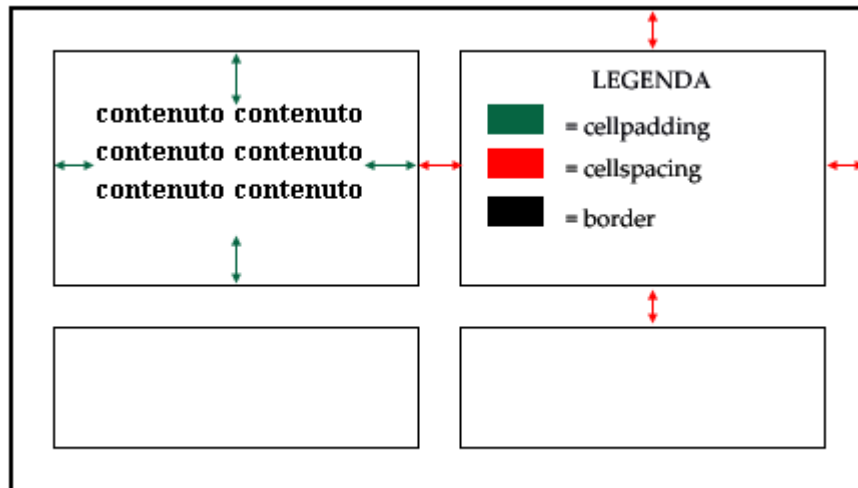
3.4. *Attributi del tag table*

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

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

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

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



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

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

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

Vediamo quali sono i possibili valori e i relativi esempi:

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

azzerati per facilitare la comprensione, dunque ci troveremo nella situazione in cui **frame="void"**)

- **none**: da nessuna parte. È il valore di default
- **groups**: le righe separano i gruppi (siano essi gruppi di righe: `<thead>`, `<tfoot>`, `<tbody>` - o gruppi di colonne: `<colgroup>`)
- **rows**: le righe separano i vari `<tr>`
- **cols**: le righe separano le colonne
- **all**: le righe separano tanto i `<tr>`, quanto le colonne

3.5. *Attributi di <table>, <tr>, <td>*

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

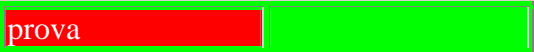
3.6. *Dimensioni*

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

3.7. *Lo sfondo*

Possiamo assegnare un colore di sfondo tramite l'attributo **bgcolor**, oppure un'immagine tramite **background**,

Vediamo un esempio:

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

```

<td width="50%"
      background="tabelle/sfondo.gif">
  <font color="#FFFFFF">prova</font>
</td>
<td width="50%">&nbsp;
</td>
</tr>
</table>

```

visualizzazione:



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

```

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

```

3.8. L'allineamento

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

```

<table align="right">

```

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

```

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

```

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

```

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

```

3.9. Colori dei bordi

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

```

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

```

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

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

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

```
<table width="450" bgcolor="#00CCFF" cellpadding="10" cellspacing="1">
  <tr bgcolor="FFFFFF">
    <td width="50%"><b>contenuto</b></td>
    <td width="50%">&nbsp;</td>
  </tr>
</table>
```

che dà:

contenuto	
------------------	--

3.10. nowrap

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

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

cioè:

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

3.11. Approfondimenti

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

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

cioè:

	contenuto
--	-----------

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

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

3.12. Impaginare un layout con le tabelle

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

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

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

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

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

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

4. I Moduli (forms)

4.1. Struttura del tag form

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

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

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

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

Noi ci occuperemo della sola pagina principale, dal momento che il modo in cui struttura una pagina di programmazione esula dagli obiettivi della presente guida.

4.2. Name e action

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

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

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

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

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

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

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

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

4.3. Method

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

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

```
http://www.nomesito.it/eseempioForm/paginaRisposta.php?nome=Wolfgang&cognome=Cecchin&datiI
nviati=prova+invio
```

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

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

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

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

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

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

4.4. Enctype (tipo di codifica)

Prima di passare i dati alla pagina di risposta, che si trova sul server, questi vengono codificati dal browser in modo da non poter dare adito ad errori (ad esempio gli spazi vengono convertiti in "+"). Normalmente non è necessario specificare come si vuole effettuare la codifica dei dati, perché è sottinteso l'invio di semplice testo. A volte però, come quando è necessario inviare un'immagine, è tuttavia indispensabile dichiarare espressamente quali dati vogliamo inviare. Per farlo è necessario utilizzare l'attributo "**enctype**" ("**encoding type**", cioè "**tipo di codifica**").

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

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

Ma nel caso di invio di immagini dovremo dichiarare:

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

4.5. Target

Grazie all'attributo "**target**" è possibile far aprire i dati del form in una pagina differente rispetto a quella corrente:

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

4.6. Un po' d'ordine: raggruppare i moduli

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

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

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

Possiamo farlo con la seguente sintassi:

```
<fieldset>
  <legend>Dati anagrafici</legend>
  <br><br><br>
</fieldset>
```

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

eccetera

che dà:

Dati anagrafici

Residenza

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

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

Ad esempio:

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

che dà:

Dati anagrafici Anno di nascita:

oppure (cambiando la posizione del testo):

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

che dà:

Dati anagrafici : anno di nascita

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

4.7. Il tag Input

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

Ad esempio:

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

crea un campo di testo.

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

crea un bottone.

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

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

che dà:

4.8. I bottoni (submit, reset, button, image)

Non c'è form che si rispetti senza bottone di invio.

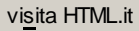
La sintassi tradizionale per creare un bottone di invio è:

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

Ad esempio:

```
<form action=http://www.nomesito.it target=" blank">
  <input type="submit" value="visita Nomesito.it">
</form>
```

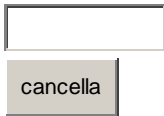
cioè:



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

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

cioè



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

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

che dà:

4.9. Il tag **<button>**

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

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

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

```
<form action=http://www.nomesito.it target=" blank">
  <input type="text"><br>
  <button type="button">
    bottone generico
  </button>
  <button type="reset">
    cancella
  </button>
  <button type="submit">
    invia
  </button>
</form>
```

cioè:



bottone generico

cancella

invia

Ed ecco un esempio complesso:

```
<button name="vai" type="submit">
  invia
  
  <b>adesso</b>
</button>
```

che dà:



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

Es:

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

o anche:

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



4.10. Il campo image

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

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

cioè:

invia!

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

4.11. Inserire testo (campo testo, textarea, password)

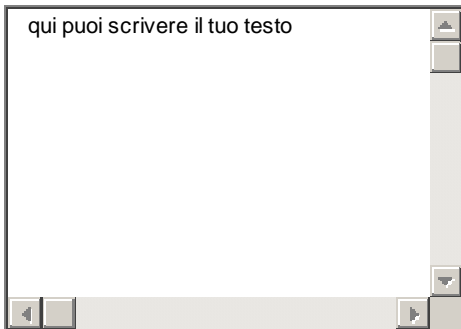
Per consentire all'utente di inserire del testo è possibile utilizzare un **campo testo**.
Se il campo è su una singola linea avremo:

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

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

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

```
<textarea name="testo" cols="40" rows="10">qui puoi scrivere il tuo testo</textarea>
```

A screenshot of a web browser showing a text area input field. The text area is rectangular with a light gray border and contains the text "qui puoi scrivere il tuo testo". On the right side, there is a vertical scrollbar, and on the bottom side, there are horizontal scrollbars.

"**rows**" indica il numero di righe della textarea, "**cols**" il numero di caratteri (cioè di colonne) che ogni riga può contenere.

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

Esiste infine il **campo password** che codifica i caratteri inseriti con degli asterischi:

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

che dà:

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

I campi di testo possono essere anche di sola lettura. Ad esempio:

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

che dà:

O disabilitati:

```
<input name="mioTesto" type="text" value="leggere l'informativa" size="25" maxlength="8" disabled>
```

cioè

4.12. Consentire delle scelte (checkbox, radio, select)

4.13. Checkbox

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

```
<fieldset>
  <legend>Linguaggi conosciuti</legend><br>
  <input type="checkbox" name="html" value="html"> html

  <br>
  <input type="checkbox" name="css" value="css"> css

  <br>
  <input type="checkbox" name="javascript" value="javascript"> JavaScript
</fieldset>
```

che dà:

Linguaggi conosciuti

- html
- css
- JavaScript

Si possono anche selezionare uno o più valori di default:

```
<input name="html" type="checkbox" value="html" checked>
```

cioè

ed è possibile disabilitare una casella:

```
<input name="html" type="checkbox" value="html" disabled>
```

cioè:

4.14. Radio button

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

```
<fieldset>
  <legend>Linguaggi conosciuti</legend>
  HTML<input type="radio" name="linguaggio" value="html">
```

```
CSS <input type="radio" name="linguaggio" value="css">
JavaScript <input type="radio" name="linguaggio" value="javascript">
</fieldset>
```

che viene così visualizzato:

Linguaggi conosciuti HTML CSS JavaScript

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

```
<input type="radio" name="linguaggio" value="html" checked disabled>
```

cioè:



4.15. Menu di opzioni (select)

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

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

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

che da luogo a:

Siti per webmaster

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

```
<select name="siti" >
  <optgroup label="siti per webmaster">
    <option value="http://www.nomesito.it">www.nomesito.it
    <option value="http://freephp.nomesito.it">freephp.nomesito.it
    <option value="http://freasp.nomesito.it">freasp.nomesito.it
  </optgroup>

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

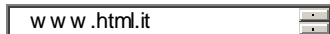
che dà luogo al seguente menu:

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

```
<label>Quale siti visiti?<br>
<select name="siti" multiple>
  <option value="http://www.nomesito.it">www.nomesito.it
  <option value="http://freephp.nomesito.it">freephp.nomesito.it
  <option value="http://freasp.nomesito.it">freasp.nomesito.it
  <option value="http://font.nomesito.it">font.nomesito.it
  <option value="http://cgipoint.nomesito.it" >cgipoint.nomesito.it
</select>
</label>
```

cioè:

Quale siti visiti?

A screenshot of a web browser showing a dropdown menu. The text 'Quale siti visiti?' is above the menu. The menu is open, and 'www.html.it' is selected and displayed in the input field.

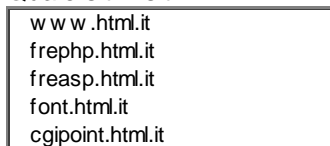
utilizzando il tasto "**ctrl**" l'utente può così effettuare delle scelte multiple.

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

```
<label>Quale siti visiti?<br>
<select name="siti" size="5" multiple>
  <option value="http://www.nomesito.it">www.nomesito.it
  <option value="http://freephp.nomesito.it">freephp.nomesito.it
  <option value="http://freasp.nomesito.it">freasp.nomesito.it
  <option value="http://font.nomesito.it">font.nomesito.it
  <option value="http://cgipoint.nomesito.it" >cgipoint.nomesito.it
</select>
</label>
```

che viene così visualizzato:

Quale siti visiti?

A screenshot of a web browser showing a list box. The text 'Quale siti visiti?' is above the list box. The list box contains five items: 'www.html.it', 'freephp.html.it', 'freasp.html.it', 'font.html.it', and 'cgipoint.html.it'.

4.16. Altri campi (file e hidden)

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

```
<input type="hidden" name="urlDiProvenienza" value="www.nomesito.it">
```

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

```
<input name="fileUtente" type="file" size="20">
```

che dà:

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

Questa riga fornisce alcune informazioni sul documento:

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

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

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

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

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

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

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

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

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

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

5.3. *Gli editor visuali*

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

Ad oggi gli editor visuali più utilizzati sono:

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

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

6. Conclusioni

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

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

6.1. Linguaggi di markup

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

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