

Basi di programmazione HTML

HTML (Hyper Text Markup Language) è un semplice linguaggio basato su un sistema di etichettatura (tagging scheme), derivato dal più complesso linguaggio SGML (Structured Generalized Markup Language).

HTML è un linguaggio ad interprete: i programmi HTML (o codici sorgenti) sono dei testi con estensione .html, che vengono presi come ingresso da un programma particolare (il *browser*), il quale interpreta le istruzioni HTML e fornisce la visualizzazione della pagina e gli strumenti di interazione e multimedia richiesti dal codice stesso (cioè, ad esempio, i collegamenti ad altre pagine o le immagini).

Non essendoci una fase di compilazione (come invece avviene per linguaggi come il Basic o il C), non abbiamo nessuna segnalazione di errori! Gli errori nel codice HTML si rilevano solo osservando il risultato presentato dal browser e notando le differenze tra questo ed il risultato sperato.

Un programma HTML è quindi un file di testo in cui alcuni elementi (le istruzioni, o *tag*) svolgono le funzioni di marcatori (mark-up), ovvero forniscono al programma interprete le istruzioni per la visualizzazione e l'uso dell'ipertesto.

Strumenti di sviluppo

Per programmare in HTML utilizziamo un editor di testi (in ambiente Windows usiamo Wordpad), salviamo in formato testo (NON in altri formati, come DOC o RTF!!) dando al file un nome con estensione .html, dopodiché apriamo tale file con un browser (Netscape Navigator ed Internet Explorer sono i due più diffusi al momento).

Uno degli standard per l'editing visuale, invece, è Macromedia® Dreamweaver MX.

Gestione dei file

I nomi dei file HTML devono contenere solo lettere non accentate (dalla "a" alla "z"), maiuscole o minuscole, possono contenere anche numeri, il segno di sottolineatura (_, detto anche *underscore*) ed il segno meno (-). Non sono ammessi nei nomi di file le lettere accentate, i segni di punteggiatura, le parentesi, i caratteri speciali (come ad esempio +, *, &, %, ", ...) e lo spazio bianco.

Istruzioni

Le istruzioni HTML (dette anche *tag*) hanno una struttura precisa, che comprende il nome dell'istruzione, seguito eventualmente da alcuni parametri.

Tutte le istruzioni HTML sono delimitate da parentesi angolari (< e >).

Le istruzioni prive di parametri hanno la forma:

```
<istruzione>
```

Le istruzioni con parametri hanno la forma:

```
<istruzione parametro1="valore" parametro2="valore">
```

(il numero di parametri varia da istruzione a istruzione ed anche in base al contesto)

L'HTML non è *case sensitive*, cioè non distingue tra maiuscole e minuscole. Le istruzioni <HTML>, <html>, <Html> o ancora <HtMl> sono assolutamente equivalenti. La scelta di usare maiuscole o minuscole dipende dal gusto del programmatore.

Le uniche istruzioni *case sensitive* sono quelle che definiscono i caratteri speciali; ad esempio, per visualizzare una "à" dobbiamo usare l'istruzione à mentre &AGRAVE; non verrà riconosciuto come istruzione valida.

Tassonomia delle istruzioni HTML

Le istruzioni HTML si possono raggruppare in tre categorie:

istruzioni di formattazione; permettono di formattare la pagina (ovvero di darle forma), ad esempio stabilendo il colore, la dimensione e il tipo dei caratteri da visualizzare, la loro posizione, ...

istruzioni di inserimento multimediale; permettono di inserire elementi multimediali (come immagini o audio) nella pagina;

istruzioni di ipertestualità; permettono di creare i collegamenti con le altre parti dell'ipertesto.

Inoltre le istruzioni HTML possono agire puntualmente (ovvero nel *punto* in cui sono inserite), o su una zona di influenza delimitata dall'apertura e dalla chiusura dell'istruzione (che avviene ripetendo l'istruzione preceduta da una barra: `` apre l'istruzione per scrivere in grassetto, `` la chiude).

Un esempio di istruzione ad azione puntuale è `
`, che permette di andare a capo:

codice HTML

```
... da visualizzare nella
pagina; testo da
visualizzare nella pagina.
<br/> Testo da visualizzare
nella pagina, ...
```

risultato nel browser

... da visualizzare nella pagina; testo da visualizzare nella pagina.
Testo da visualizzare nella pagina, ...

Un esempio di istruzione a zona di influenza è ``, che permette di scrivere in grassetto (detto anche neretto o bold):

codice HTML

```
... da visualizzare nella
pagina; testo da
<b>visualizzare nella
pagina</b>. <br/> Testo da
visualizzare nella pagina,
...
```

risultato nel browser

... da visualizzare nella pagina; testo da **visualizzare nella pagina**.
Testo da visualizzare nella pagina, ...

L'esistenza di istruzioni a zona d'influenza crea il fenomeno della nidificazione, quando una istruzione viene inserita all'interno dell'area di influenza di un'altra.

Possiamo immaginare che le aree di influenza delle istruzioni siano delle scatole, che possiamo inserire le une dentro le altre ma **non** possiamo intersecare!

Vediamo un esempio di nidificazione corretta:

codice HTML

```
... da visualizzare nella
pagina; testo da
<b>visualizzare <i>nella
pagina</i></b>. <br/> Testo
da visualizzare nella
pagina, ...
```

risultato nel browser

... da visualizzare nella pagina; testo da **visualizzare nella pagina**.
Testo da visualizzare nella pagina, ...

Ed uno di nidificazione scorretta:

```
... da visualizzare nella pagina; testo da <b>visualizzare <i>nella</b>
pagina</i>. <br/> Testo da visualizzare nella pagina, ...
```

In questo secondo esempio l'area di influenza dell'istruzione per il grassetto interseca quella dell'istruzione per l'italico, generando un errore. In alcuni semplici casi questi errori non vengono notati dal browser, ma possono creare danni rilevanti in codici HTML più complessi.

Struttura del programma HTML

I codici HTML sono definiti all'interno dell'istruzione `<HTML>`, che indica al browser il tipo di linguaggio usato, e sono costituiti da sue sezioni: l'intestazione (HEAD) e il corpo (BODY):

```
<html>
<head>
...
</head>
<body>
...
</body>
```

La sezione HEAD contiene informazioni che riguardano il documento e consentono di risalire all'autore, di collegarlo a un motore di ricerca o di potenziarlo con funzioni aggiuntive.

In particolare, in questa sezione troviamo la definizione del titolo del documento, che viene visualizzato nella barra del titolo del browser:

```
<head>
<title>Titolo del documento</title>
</head>
```

La sezione BODY contiene il corpo del programma HTML, ovvero il testo da visualizzare e tutte le istruzioni su come visualizzarlo, come inserire multimediali e collegarlo al resto dell'ipertesto o di Internet.

Un primo esempio di programma HTML è **primo.html**.

Il colore in HTML

L'HTML ci permette di modificare il colore di alcuni elementi, come vedremo più avanti, utilizzando una rappresentazione del colore basata sulla combinazione di tre colori primari: rosso (red), verde (green) e blu (blue), grazie alla diversa intensità di emissione luminosa di ognuno dei fosfori che costituiscono la parte visibile di un monitor.

Questi tre colori sono alla base della gestione del colore nella grafica computerizzata, nella quale parliamo di spazio di colore RGB (dalle iniziali dei nomi in inglese dei tre colori primari), e ci permettono di definire quindi ogni colore desiderato con una terna di numeri:

Colore_x=(R_x, G_x, B_x)

La variazione del valore di intensità luminosa per ognuno dei tre colori di base varia da 0 a 255, permettendoci così di avere 256 possibili tonalità su ognuno dei colori di base (chiamati anche *canali colore*). In HTML esprimiamo i numeri in base esadecimale, per la cui corretta gestione rimandiamo alla parte teorica relativa ai cambiamenti di base numerica, ricordando qui che vale l'equivalenza:

| | | | | | | | | | | | | | | | | | | | | |
|-------------------------|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|----|----|----|----|-----|
| base decimale | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | ... |
| base esadecimale | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | a | b | c | d | e | f | 10 | 11 | 12 | 13 | ... |

La combinazione dei valori dei tre canali di colore ci permette di creare numerosi colori dello spettro, ricordando che la massima intensità vale 255 in base 10 e ff in base 16, o, in termini matematici: $[255]_{10}=[ff]_{16}$. Vediamo quindi alcune rappresentazioni cromatiche:

| colore | R | G | B |
|--------|----|----|----|
| bianco | ff | ff | ff |
| nero | 00 | 00 | 00 |
| rosso | ff | 00 | 00 |
| verde | 00 | ff | 00 |
| blu | 00 | 00 | ff |

| | | | |
|--------|----|----|----|
| grigio | xx | xx | xx |
|--------|----|----|----|

Le diverse tonalità di grigio di esprimono con lo stesso valore numerico su tutti i canali di colore. La scelta dei colori per lo sfondo della pagina ed il testo deve essere fatta in modo da non compromettere la leggibilità dei contenuti. Ricordiamo quindi che un testo scuro su sfondo chiaro è più leggibile di uno chiaro su fondo scuro, e che alcuni accostamenti di colore possono essere tali da compromettere la leggibilità del documento, vuoi per fastidio (rosa su verde, rosso su verde, ...) vuoi per eventuali problematiche della percezione del colore (come ad esempio il daltonismo) che rendono difficile distinguere tra tonalità diverse di uno stesso colore.

Affrontiamo ora una breve panoramica sulle istruzioni fondamentali del linguaggio.

Istruzioni di formattazione

Per andare a capo: `
`

Per aprire e chiudere un nuovo paragrafo: `<p> . . . </p>`

Aprire un nuovo paragrafo significa riprendere la visualizzazione del testo con uno spazio di interlinea 2 (ovvero, lasciando una riga vuota). Se si desidera lasciare più di una riga vuota di spazio tra due elementi della pagina, occorre usare una ripetizione del `
`, o altri metodi più complessi.

Scrivere il testo in neretto: ` . . . `

Scrivere il testo in italico: `<i> . . . </i>`

Modificare l'aspetto dei caratteri

Per modificare l'aspetto dei caratteri usiamo il tag ``.

La dimensione del carattere viene variata con il parametro `size`:

```
<font size=+1>c</font>
```

I valori del parametro variano da `-4` a `+6`, con variazioni in base al tipo di carattere utilizzato.

Un esempio di uso di questa opzione è il programma `f_size.html`

Il colore del carattere viene modificato con il parametro `color`:

```
<font color="#rrggbb">testo in color rrggbb</font>
```

Un esempio di uso di questa opzione è il programma `f_color.html`

Da notare con attenzione che il valore numerico del colore è composto di sei cifre (due per il rosso, due per il verde e due per il blu), anche nel caso tali valori siano nulli; l'assenza di una o più di tali cifre genera errori. Inoltre, il colore va inserito tra doppi apici e preceduto dal carattere `#`.

Modificare il colore dello sfondo della pagina

Il colore dello sfondo della pagina viene modificato dal parametro `bgcolor` dell'istruzione `body`:

```
<body bgcolor="#rrggbb">
```

Modificare il colore del testo nella pagina

Il colore del testo in tutta la pagina viene modificato dal parametro `text` dell'istruzione `body`:

```
<body text="#rrggbb">
```

Inserire una linea orizzontale

Una linea orizzontale viene inserita nella pagina con l'istruzione `hr`, la quale è dotata di alcuni parametri:

```
<hr size=numero> permette di modificare la profondità della riga;
```

```
<hr width=numero|percentuale> permette di modificare la larghezza della riga; il valore può essere espresso in pixel o in percentuale;
```

`<hr align=left|right|center>` permette di modificare l'allineamento della riga nella pagina;

`<hr noshade>` permette di non visualizzare la profondità della riga.

Un esempio di uso di questo tag è nel file **hr.html**

Centrare il contenuto della pagina

Il testo e gli altri elementi contenuti nella pagina si possono allineare al centro della stessa con il tag `center`:

```
<center>elementi da centrare</center>
```

Inserire commenti

Nel codice HTML possiamo inserire dei commenti, ovvero delle annotazioni che non verranno visualizzate dal browser. Un commento si apre con i caratteri `<!--` e termina con `-->`:

```
<!--Questo e' un commento -->
```

Caratteri speciali

Nel codice HTML possiamo inserire dei caratteri speciali, ovvero caratteri non presenti nella tabella ASCII standard, come ad esempio le lettere accentate.

Vediamo come rappresentare alcuni caratteri particolarmente frequenti:

| carattere | codice HTML |
|---------------|---------------------------|
| à | <code>&agrave;</code> |
| è | <code>&egrave;</code> |
| é | <code>&eacute;</code> |
| ò | <code>&ograve;</code> |
| ù | <code>&ugrave;</code> |
| ì | <code>&igrave;</code> |
| < | <code>&lt;</code> |
| > | <code>&gt;</code> |
| & | <code>&amp;</code> |
| È | <code>&Egrave;</code> |
| spazio bianco | <code>&nbsp;</code> |

Alcuni esempi di uso di caratteri speciali è nel file **special.html**

Livelli di testo

Il testo può essere visualizzato in diversi *livelli di importanza*, detti anche *heading*.

Prima e dopo uno heading viene inserita una riga vuota. Lo heading di livello più alto (1) viene usato in genere per i titoli, il successivo per il sottotitoli. I livelli più bassi (fino al 6, il minimo) vengono usati per inserire note.

```
<h1>Testo di livello 1</h1>
```

...

```
<h6>Testo di livello 6</h6>
```

Un esempio di uso di livelli è nel codice **livelli.html**

Liste

Il testo può essere organizzato in liste di diversi tipi.

La lista più semplice è la **lista non numerica** (o non ordinata, *unordered*), che si definisce con il tag `ul` e nella quale gli elementi sono definiti con il tag `li` (*list item*):

```
<ul>
<li> primo elemento della lista non ordinata
<li> secondo elemento della lista non ordinata
<li> terzo elemento della lista non ordinata
</ul>
```

La lista non ordinata può avere diversi modi di indicazione degli elementi, definiti con il parametro `type`:

```
<ul type=disc|circle|square>
```

Vediamo alcuni esempi, che ritroviamo nel file `liste.html`:

| | |
|--|---|
| <pre><ul type=disc> primo elemento della lista non ordinata secondo elemento della lista non ordinata terzo elemento della lista non ordinata </pre> | <ul style="list-style-type: none"> • primo elemento della lista non ordinata • secondo elemento della lista non ordinata • terzo elemento della lista non ordinata |
|--|---|

| | |
|--|---|
| <pre><ul type=square> primo elemento della lista non ordinata secondo elemento della lista non ordinata terzo elemento della lista non ordinata </pre> | <ul style="list-style-type: none"> ▪ primo elemento della lista non ordinata ▪ secondo elemento della lista non ordinata ▪ terzo elemento della lista non ordinata |
|--|---|

La **lista ordinata** (*ordered list*) si definisce con il tag `ol` e presenta gli elementi accompagnandoli ad un numero progressivo del quale possiamo definire il tipo ed il valore di partenza. Anche in questo caso, gli elementi sono definiti con il tag `li` (*list item*):

```
<ol>
<li> primo elemento della lista ordinata
<li> secondo elemento della lista ordinata
</ol>
```

Il parametro `type` permette di modificare il tipo di numerazione:

```
<ol type=1|A|a|I|i>
```

dove “1” è il valore di default, che può quindi non essere specificato; “A” ed “a” indicano la numerazione in lettere dell’alfabeto maiuscole o minuscole; “I” ed “i” indicano la numerazione in numeri romani maiuscoli o minuscoli.

Vediamo alcuni esempi, che ritroviamo nel file `liste.html`:

| | |
|--|---|
| <pre> primo elemento della lista ordinata secondo elemento </pre> | <ol style="list-style-type: none"> 1. primo elemento della lista ordinata 2. secondo elemento |
|--|---|

| | |
|---|---|
| <pre><ol type=A> primo elemento della lista ordinata secondo elemento </pre> | <ol style="list-style-type: none"> A. primo elemento della lista ordinata B. secondo elemento |
|---|---|

| | |
|---|--|
| <pre><ol type=I start=6> primo elemento della lista ordinata secondo elemento </pre> | <ol style="list-style-type: none"> VI. primo elemento della lista ordinata VII. secondo elemento |
|---|--|

Un terzo tipo di lista è la **lista a definizioni** (*definition list*), nella quale gli elementi sono definiti come coppie di termine da definire e righe di descrizione.

Questa lista si definisce con il tag `dl`, mentre ogni coppia è definita con il tag `dt` per l’elemento da definire e `dd` per la sua descrizione.

Vediamo un esempio, che ritroviamo nel file `liste.html`:

| | |
|---|--|
| <pre><dl> <dt>Primo elemento della lista</pre> | <p>Primo elemento della lista sua descrizione a parole, che viene</p> |
|---|--|

| | |
|--|--|
| <pre><dd>sua descrizione a parole, che viene visualizzata indentata rispetto all'allineamento generale del testo <dt>Secondo elemento della lista <dd>sua descrizione a parole, che viene visualizzata indentata rispetto all'allineamento generale del testo </dl></pre> | <p>visualizzata indentata rispetto all'allineamento generale del testo</p> <p>Secondo elemento della lista</p> <p>sua descrizione a parole, che viene visualizzata indentata rispetto all'allineamento generale del testo</p> |
|--|--|

Collegamenti

Per la creazione di collegamenti portiamo avanti la metafora della navigazione, considerando un collegamento (o *link*) come un'ancoraggio (*anchor*) ad un altro documento, che ci permette di spostarci dalla visualizzazione del documento di partenza alla visualizzazione di un altro documento, realizzando quindi l'ipertestualità.

I link possono essere esterni (collegando un documento ad un altro) o interni (collegando diverse parti di un'unico documento).

Il codice per realizzare un collegamento esterno ha questa forma:

```
<a href="URL">testo</a>
```

dove *a* significa *anchor* ed *href* significa *hypertextual reference* (riferimento ipertestuale); il "testo" è ciò che utilizzeremo per spostarci sul documento di destinazione, del quale dobbiamo specificare la posizione nella rete (l'URL, Uniform Resource Locator).

Vediamo un esempio (che ritroviamo nei file `link_ex1.html` e `link_ex2.html`):

| | |
|---|--|
| <pre>... si arriva cos'igrave; al secondo esempio per gli ...</pre> | <p>...si arriva così al <u>secondo esempio</u> per gli ...</p> |
|---|--|

| | |
|--|---|
| <pre>... si torna cos'igrave; al primo esempio per gli ...</pre> | <p>...si torna così al <u>primo esempio</u> per gli ...</p> |
|--|---|

Il codice per realizzare un collegamento interno si basa sullo stesso principio del precedente, ma richiede due fasi:

1. l'attribuzione di un nome alla parte del documento sulla quale vogliamo arrivare:

```
<a name="etichetta">...</a>
```

2. l'uso di tale nome nel collegamento:

```
<a href="#etichetta">testo</a>
```

L'uso del carattere "#" serve per indicare al browser che il collegamento è destinato ad una parte dello stesso file e non ad un altro file. La sua omissione spinge il browser a cercare nel file system un file chiamato "etichetta" ed a produrre quindi un errore.

Vediamo un esempio che ritroviamo nel file `link_ex3.html`:

| | |
|---|---|
| <pre>...alla fine del documento.contenuti.</pre> | <p>...alla <u>fine</u> del documento.</p> <p>...</p> <p>...contenuti.</p> |
|---|---|

I due metodi di collegamento (esterno ed interno) si possono ovviamente combinare, specificando il nome di una etichetta di destinazione in un altro file, al momento della definizione del collegamento esterno.

Vediamo un esempio che ritroviamo nel file `link_ex1.html`:

| | |
|---|--|
| <pre>... fine di tale documento.</pre> | <p>... alla <u>fine</u> di tale documento.</p> |
|---|--|

La convenzione universalmente accettata fa sì che i collegamenti siano evidenziati, rispetto al testo circostante, tramite sottolineatura. Una linea guida della programmazione e della progettazione di interfacce per ipertesti web è quindi di non sottolineare testi che non siano link, anche se i collegamenti sono, nella pagina, evidenziati in altro modo.

Colore dei collegamenti

A meno di specifiche particolari nel documento, i collegamenti vengono visualizzati dai browser con un codice di colore che specifica lo *stato* del link:

non visitato; collegamento ad una pagina ancora non visitata: blu

visitato; collegamento ad una pagina già visitata: viola

attivo; nel tempo che passa tra il “click” e l’apertura del collegamento: rosso.

Esistono numerosi metodi per modificare questa visualizzazione. Il più semplice ed utilizzato agisce a livello del corpo del programma:

```
<body link="#rrggbb" vlink="#rrggbb" alink="#rrggbb">
```

Dove `link="#rrggbb"` indica il colore del collegamento non visitato, `vlink="#rrggbb"` quello del collegamento visitato e `alink="#rrggbb"` quello del collegamento attivo.

Descrizione del collegamento

Abbiamo detto che nel codice del collegamento dobbiamo specificare il nome del file di destinazione, utilizzando la sigla URL (Uniform Resource Locator).

Un URL identifica in modo *uniforme* (ovvero con lo stesso metodo su tutti i computer collegati in rete) la *locazione* (posizione) di una *risorsa*. Nel nostro caso, le risorse sono file HTML o gli elementi multimediali ad essi collegati.

L’URL è composto di tre parti:

protocollo - indirizzo_del_computer - posizione_nel_file_system

ad esempio:

```
http://www.nomedelcomputer.net/users/start.html
```

Lavorando con ipertesti, il protocollo è sempre HTTP (*HyperText Transport Protocol*) e viene spesso ommesso per comodità.

Il nome del computer è indispensabile se stiamo costruendo un collegamento a documenti archiviati su un’altra macchina, ed è invece inutile se ci stiamo collegando a documenti residenti sulla stessa macchina.

Possiamo limitarci al *percorso* nel file system, omettendo il nome del file, nel caso in cui questi sia `index.html`, dato che i server web sono programmati in modo tale da fornire tale file in caso di mancata specifica.

Ogni file ha una *posizione* all’interno del file system generale del computer su cui è archiviato. Il percorso che va dalla *radice* dell’albero del file system fino al file stesso è detto percorso (o *path*) assoluto.

Nella prima delle due immagini che seguono, vediamo un dettaglio di file system, nel quale osserviamo che (ad esempio) il file `brick.gif` è locato in una posizione particolare, il cui path assoluto (che appare nella barra della finestra della gestione risorse) è:

```
C:\Programmi\CosmoSoftware\Shared\textures\color1\brick.gif
```

Quando si progetta e si implementa un ipertesto, è buona norma mantenere tutti i file all’interno di una directory e delle sue sottodirectory, in modo tale da formare un sottoinsieme facilmente trasportabile.

Nella seconda immagine, consideriamo il file `ipertesti.htm`, con path assoluto `C:\sito\ipertesti.htm`, ed il file `ggg.gif`, che ha path assoluto `C:\sito\img\ggg.gif`.

Se nel codice del file `index.htm` inseriamo il path assoluto di `ipertesti.htm` come riferimento ipertestuale, e ci riferiamo all'immagine tramite il suo path assoluto, la semplice azione di cambiare nome alla directory (da `sito` a `ipertesto`, ad esempio) genera un errore di impossibilità di reperire i file, rendendo inutilizzabile tutto il sistema.

Usando il path relativo (cioè la strada da percorrere per raggiungere il file partendo da un *punto specifico*), invece, restiamo isolati nella directory `sito` e possiamo tranquillamente rinominarla o spostarla.

Il path relativo di `ipertesti.htm` in questo caso è `ipertesti.htm` (essendo i due file nella stessa cartella), quello dell'immagine è `img/ggg.gif`, mentre il path relativo da `index.htm` all'immagine dell'esempio precedente è:

```
..\Programmi\CosmoSoftware\Shared\textures\color1\brick.gif
```

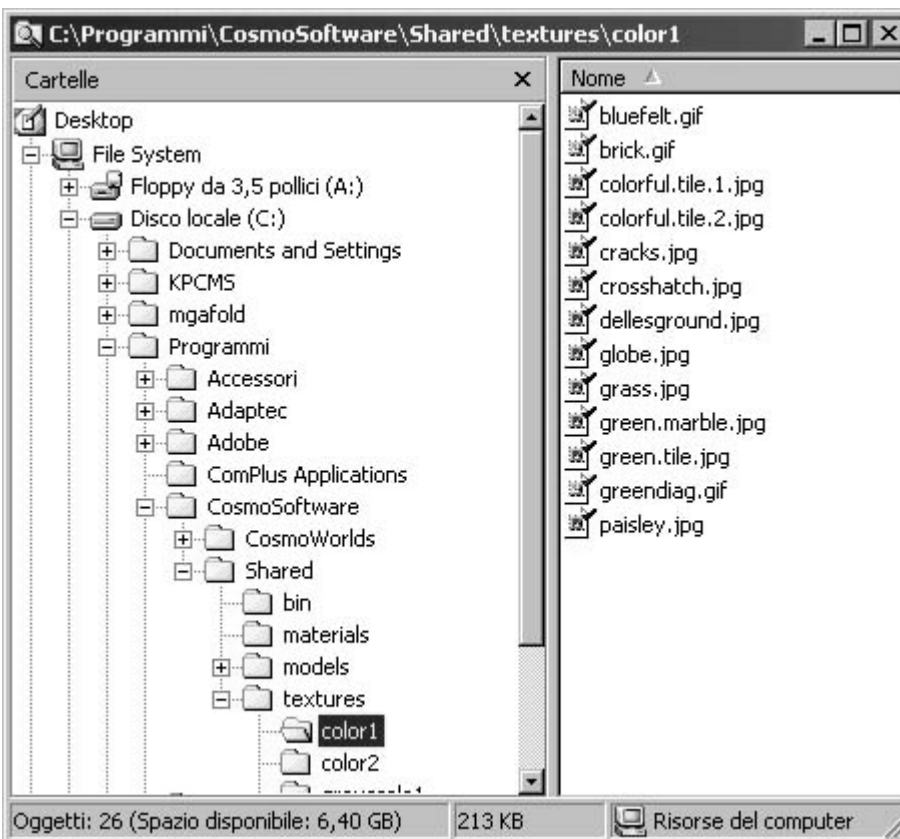


Immagine 1 – La posizione di un file nel file system è determinata dal suo nome assoluto

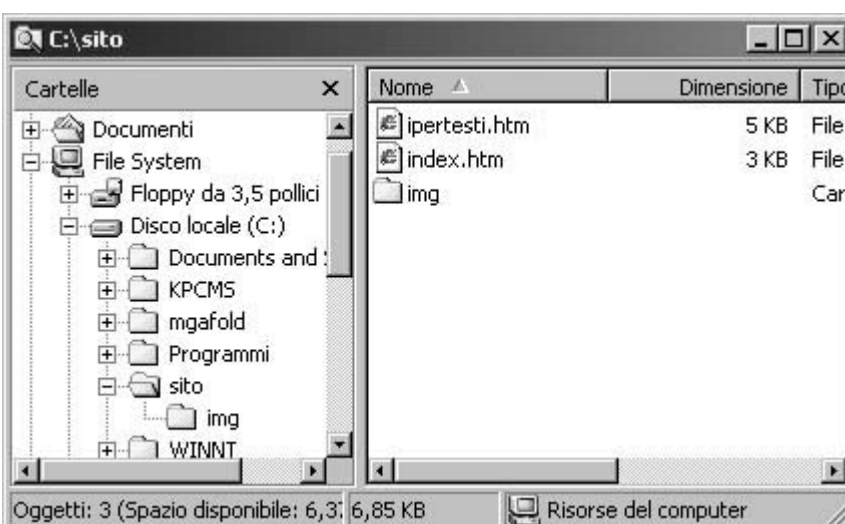


Immagine 2 – La posizione di un file relativamente ad un altro permette di costruire riferimenti "sicuri" tra le parti di un ipertesto

Tabelle

L'HTML ci permette di inserire *tabelle* nelle pagine, ovvero strutture più o meno complesse in grado di contenere testi, immagini od altri elementi.

Le tabelle hanno una grande importanza nella progettazione di pagine ipertestuali, essendo lo strumento adatto alla costruzione di griglie di layout e permettendoci di avere un elevato grado di controllo sull'aspetto delle pagine stesse e sulla posizione degli elementi in esse contenuti.

Una tabella (*table*) è formata da un insieme di righe (*table rows*), a loro volta composte di celle (*table definition*). Il codice necessario è più complesso dei casi già studiati e richiede particolare attenzione.

Una tabella è sempre racchiusa nello spazio delimitato dalla coppia di tag `<table>...</table>`.

Le righe della tabella sono specificate con il codice `<tr>...</tr>`, mentre le singole celle sono delimitate da `<td>...</td>`. Il numero di righe è determinato dal numero di coppie `<tr>...</tr>`, mentre il numero di celle per ogni riga è determinato dal numero di coppie `<td>...</td>`. Per una corretta visualizzazione dei contenuti, il numero di celle deve essere lo stesso per tutte le righe.

I contenuti della tabella devono essere inseriti all'interno della coppia di tag che specifica la cella nella quale vogliamo vengano visualizzati.

Vediamo alcuni esempi, che ritroviamo nel file `tabelle.html`.

Il codice

```
<table>
<tr>
  <td>testo della prima cella della prima riga</td>
  <td>testo della seconda cella della prima riga</td>
</tr>
<tr>
  <td>testo della prima cella della seconda riga</td>
  <td>testo della seconda cella della seconda riga</td>
</tr>
</table>
```

produce questa tabella:

| | |
|--|--|
| testo della prima cella della prima riga | testo della seconda cella della prima riga |
| testo della prima cella della seconda riga | testo della seconda cella della seconda riga |

Possiamo inserire tabelle dentro altre tabelle, nidificandole:

```
<table>
<tr>
  <td>testo della prima cella della prima riga</td>
  <td>
    <table>
    <tr>
      <td>testo della prima cella della prima riga</td>
      <td>testo della seconda cella della prima riga</td>
    </tr>
    <tr>
      <td>testo della prima cella della seconda riga</td>
      <td>testo della seconda cella della seconda riga</td>
    </tr>
    </table>
  </td>
</tr>
<tr>
  <td>testo della prima cella della seconda riga</td>
  <td>testo della seconda cella della seconda riga</td>
</tr>
</table>
```

Le istruzioni per la costruzione di tabelle hanno numerosi attributi che ci permettono di migliorare e rendere più completo e complesso il codice. Vediamo i principali.

`border=valore`

attributo di `table`; rende visibile il bordo della tabella, con uno spessore di “valore” pixel; il valore 0 è molto utilizzato nei casi in cui la tabella serve per impaginare gli elementi ma non debba essere visualizzata

`align=left|center|right`

attributo di `tr` e `td`; allinea orizzontalmente il contenuto della riga o della cella

`valign=top|middle|bottom|baseline`

attributo di `tr` e `td`; allinea verticalmente il contenuto della riga o della cella

`colspan=valore`

attributo di `td`; permette di creare celle che occupino più di una colonna

`rowspan=valore`

attributo di `td`; permette di creare celle che occupino più di una riga

`cellspacing=valore`

attributo di `table`; permette di modificare la larghezza delle linee interne alla tabella, ovvero della distanza tra le celle

`cellpadding=valore`

attributo di `table`; indica la quantità di spazio tra il bordo della tabella e il contenuto delle celle

`width=valore, height=valore`

indicano la dimensione della tabella, della riga o della singola cella; il valore può essere espresso in percentuale (rispetto alla larghezza della pagina o dell'elemento in cui la tabella attuale è contenuta) o in pixel

`bgcolor="#rrggbb"`

attributo di `table, tr` e `td`; permette di attribuire un colore di sfondo

`background="URL"`

attributo di `table, tr` e `td`; permette di attribuire un'immagine di sfondo

Il file **tabelle.html** contiene alcuni esempi di uso degli attributi.

Inserimento di immagini

Le pagine HTML possono essere migliorate ed arricchite tramite l'inserimento di immagini, siano esse destinate a veicolare contenuti, a decorare o a partecipare al funzionamento dell'interfaccia.

I formati di immagini supportati dai browser sono attualmente tra: GIF, JPEG, PNG. La scelta del formato per ogni singola immagine dipende da numerosi fattori, che esulano dallo scopo di questa dispensa. Dando quindi già per esistenti le immagini, analizziamo come utilizzarle nei nostri documenti ipertestuali.

L'istruzione per l'inserimento di un'immagine è:

```

```

dove, come già visto, URL significa il nome del file che vogliamo visualizzare. Questa istruzione è di tipo puntuale e non deve essere chiusa.

Una immagine può essere *supporto* per un collegamento:

```
<a href="URL1"></a>
```

dove ovviamente URL1 è l'indirizzo del file da visualizzare attivando il collegamento e URL2 è il nome dell'immagine.

`alt="testo"`

testo viene visualizzato al posto dell'immagine durante il suo caricamento, ed in una sorta di *nuvoletta* posizionata accanto al puntatore del mouse, quando questi si posiziona sopra l'immagine caricata

`align=left|right|top|texttop|middle|absmiddle|baseline|bottom|absbottom`

consente di modificare l'allineamento di testo e immagine;

`left` allinea l'immagine a sinistra del testo, fino alla fine in altezza dell'immagine o fino all'uso di `<br clear=left>`; analogamente si comporta `right`;

`top` allinea l'immagine con la parte alta del più alto elemento della linea di testo;

`texttop` allinea con la parte alta del testo della linea (a volte ciò è differente da `top`);

`middle` allinea la base della linea di testo corrente con il centro dell'immagine, alla fine della linea il testo andrà a capo sotto l'immagine;

`absmiddle` allinea il centro della linea di testo corrente con il centro dell'immagine;

`baseline` allinea il fondo dell'immagine con la base della linea corrente;

`bottom` ha lo stesso effetto di `baseline`;

`absbottom` allinea il fondo dell'immagine con la base della linea di testo corrente

`width=valore, height=valore`

specificano la dimensione dell'immagine; è importante ricordare che il browser non è un programma di elaborazione delle immagini, ed è quindi controproducente usare questi attributi per ridimensionare un'immagine!

Il loro uso corretto si ha quando servono ad indicare la dimensione occupata dall'immagine, permettendo al browser di impaginare correttamente il testo nell'attesa che l'immagine venga scaricata, oppure quando si usa un'immagine trasparente di dimensione minima (ad esempio 1x1 pixel) per spaziare alcuni elementi della pagina di una quantità di pixel specificata, appunto, da `width` ed `height`

`border=valore`

specifica la dimensione in pixel del bordo dell'immagine; è particolarmente utile con valore 0 (zero) per non visualizzare il bordino che il browser aggiunge di default alle immagini che supportano collegamenti ipertestuali

`vspace=valore, hspace=valore`

permettono di specificare altezza o larghezza in pixel dello spazio da lasciare libero intorno all'immagine

Infine, possiamo specificare un'immagine che faccia da sfondo per l'intero documento HTML, utilizzando l'opzione `background="URL"` all'interno dell'istruzione `body`.

L'immagine specificata verrà usata per riempire l'intera finestra del browser, utilizzando un meccanismo detto *tiling* (o di tassellazione): l'immagine viene ripetuta in file e colonne, come se fosse una mattonella con cui *pavimentare* tutta la finestra.

Ricordiamo, nello scegliere l'immagine di sfondo, che la pagina nasce allo scopo di veicolare informazioni e che lo sfondo non dovrà, ovviamente, impedirne la fruizione. Dovrà quindi avere contrasti di colore molto bassi, per evitare che i contenuti perdano leggibilità, e dovrà essere di piccole dimensioni, per non appesantire la fase di trasmissione dell'intero documento.

Frame

L'HTML, infine, ci permette di suddividere la finestra del browser in sottofinestre indipendenti, nelle quali visualizzare documenti diversi contemporaneamente.

La suddivisione in cornici (o *frame*) si genera costruendo una pagina di definizione delle cornici stesse. Il codice HTML di questa pagina non ha più la sezione `body`, che viene sostituita da una sezione `frameset` (insieme di cornici), che definisce il numero e la forma dei frame da generare nella finestra del browser:

```
<frameset cols="30%, *">
<frame src="URL1">
<frame src="URL2">
</frameset>
```

dove `URL1` ed `URL2` sono gli indirizzi dei file da visualizzare nelle due cornici appena create. Anche questo metodo prevede numerose opzioni:

```
cols="valore, valore, ...", rows="valore, valore, ..."
```

attributo di `frameset`, specifica se dividere la finestra in colonne o in righe; le dimensioni possono essere date in percentuali o in pixel, il carattere `*` consente di lasciare al browser il calcolo dello spazio non specificato dal codice: `"*, *"` equivale a `"50%, 50%"`, `"50%, *, *"` equivale a `"50%, 25%, 25%"`

```
name="stringa"
```

attribuisce un nome al `frame` di cui è attributo; il nome viene utilizzato nella fase di costruzione degli ancoraggi ipertestuali: l'opzione `target` del comando `anchor` ci permette di specificare il frame in cui visualizzare il documento a cui si riferisce il collegamento

```
noresize
```

attributo di `frame`, impedisce all'utente di modificare la dimensione del frame stesso

```
scrolling=yes|no|auto
```

attributo di `frame`, permette di stabilire se visualizzare o meno la barra di scorrimento

Una volta creato il file di definizione dei frame occorre creare, con le tecniche già viste, i file HTML da visualizzare all'interno della struttura.

Un esempio di struttura a frame è composto dai file `frame_1.html`, `frame_2.html`, `frame_3.html`, `frame_4.html`.