Prof. Paolo Molena / scuola media "Quintino di Vona" - Milano

Guida all'utilizzo dell'HTML (03/2013)

area delle Informazioni / 1 media

Questa guida è stata scritta per ragazzi di prima media e va utilizzata insieme alle pagine del corso su divona.it. E' stata studiata per consentire un primo e limitato approccio all'HTML, a Javascript e a PHP

1. Introduzione:

- **1.a – introduzione** — l'uso di Dreamweaver / i tag, proprietà e valori / la pagina web standard / il Doc Type / l'intestazione della pagina (**head**) /meta tag e meta name
- **1.b – il corpo della pagina (body)** — il colore di sfondo (**bgcolor**) / l'immagine di sfondo (**background**)

2. Il testo:

- **2.a – il tag font** — le proprietà **face**, **color** e **size**
- **2.b - lo stile** — grassetto (**b**), corsivo (**i**), sottolineato (**u**)

3. I contenitori:

- **3.a – i titoli** — la dimensione dei titoli con **h1** → **h6**
- **3.b - paragrafi (p), div e span** — caratteristiche, linee divisorie (**hr** con le proprietà **width**, **size** e **align**) andare a capo (**br**)

4. Il tag style per la gestione delle caratteristiche dei contenitori

- **4.a – i bordi** — l'ordine **border** e le proprietà **width**, **color**, **style**
- **4.b – il padding** — l'ordine **padding** e le proprietà **top**, **right**, **bottom** e **left**
- **4.c – i margini** — l'ordine **margin** e le proprietà **top**, **right**, **bottom** e **left**
- **4.d – la posizione con position** — l'ordine **position** e i valori **static**, **absolute** e **relative** le proprietà **top**, **right**, **bottom** e **left**
- **4.e – la posizione con float** — l'ordine **float** e i valori **left** , **right** e **none**

5. Organizzare una pagina:

- **5.a – collocare più div sulla pagina**
- **5.b – i modelli di pagina**
- **5.c – la gestione delle immagini con il tag img** — le proprietà **src**, **border**, **title**, **width**, **height** e **alt**
- **5.d – gli elenchi puntati e numerati** — l'apertura con **ul** (puntati) e **ol** (numerati) gestione delle righe con **li**
- **5.e – l'apertura di tabelle con table** — la gestione delle righe con **tr** e delle colonne con **th**
- **5.e – la gestione dei suoni e dei filmati con embed** — i suoni; la gestione del player per i filmati

6. l'uso del tag a per la gestione dei collegamenti:

- **6.a – per aprire una nuova pagina** — l'ordine **a** e le proprietà **href**, **title**
- **6.b – per aprire una finestra secondaria** — … la proprietà **onclick** per aprire una finestra secondaria
  - l'ordine **window.open**, il percorso, il nome, le dimensioni (**width** e **height**), la posizione (**left** e **top**)
- **6.c – per aprire sia una nuova pagina che una finestra secondaria** — l'uso su una **parola calda** di **href** e di **onclick**
- **6.d – l'uso di un'immagine per i collegamenti**
- **6.e – percorsi assoluti e percorsi relativi**
- **6.f – i collegamenti interni o àncore** — la proprietà **name** per dare un nome al punto di ancoraggio e **href** con **#** rinviare al punto di ancoraggio
- **6.g – le mappe "sensibili"** — il tag **img** e la proprietà **usemap** con **#** per preparare l'immagine il tag **map name** per definire le aree sensibili, il tag **area shape** con **rect**, **circle** e **poly** per scegliere il tipo e **coord** per dare le coordinate

7. la gestione delle proprietà degli elementi di una pagina con i fogli di stile:

- **7.a – il collegamento al foglio di stile** — il tag **link**, l'uso di **id** e di **span**
- **7.b – il foglio di stile** — la possibilità di definire proprietà differenti a quelle di default per **body**, **title**, **p**, **table** (con **th**, **tr** e **td**). La gestione delle proprietà degli identificatori (con **#** e un nome) e delle classi (con **.** e un nome).
- **7.c – modelli per divonasperi.it** — pagina principale - indirizzi - lavoro – galleria
- **7.d – la pubblicazione**

INTRODUZIONE

Come sappiamo l'**HTML** (**HyperText Markup Language**) è il primo linguaggio ideato (e il più utilizzato) per realizzare pagine ipertestuali o **pagine web** cioè pagine di testo da pubblicare su Internet.
Una comune pagina HTML è un documento di testo avente le seguenti estensioni: **.htm** o **.html**. La differenza tra le due estensioni è che la prima veniva usata nel mondo DOS (cioè prima di Windows) perché esso supportava estensioni con al massimo 3 lettere. L'estensione dice al browser che si tratta di un pagina HTML e di interpretarla di conseguenza.
Prima di iniziare il nostro lavoro andiamo su *Pannello di controllo / Opzioni cartella / Visualizzazione / Nascondi le estensioni per i tipi di file conosciuti* e togliamo la spunta in modo da rendere visibili le estensioni dei file.
Proviamo ad aprire il file 1 H.txt dove sono stati memorizzati tutti i nostri dati. Vista l'estensione **.txt** esso viene aperto con Blocco Note, una semplice applicazione presente negli accessori di Windows. Essa viene utilizzata per memorizzare testi senza ordini di formato. Oltre ai codici dei caratteri essa memorizza i codici di alcuni ordini presenti in tastiera come lo spazio, l'invio a capo, la tabulazione che però non sono visibili nel documento aperto.

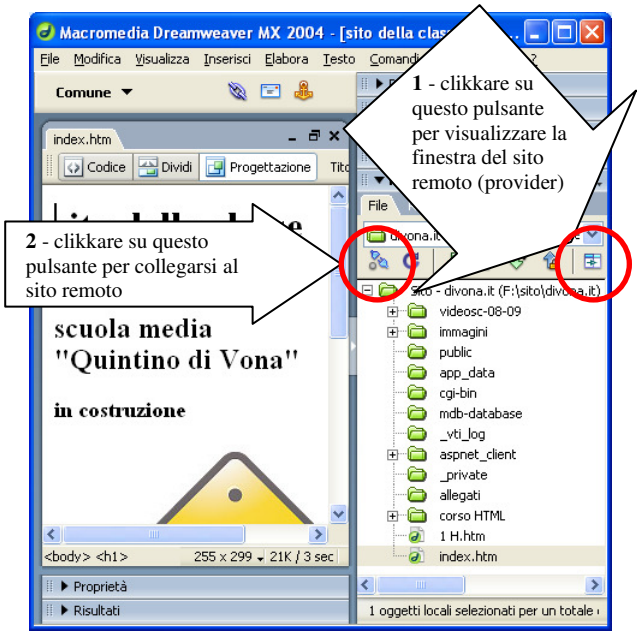
Se vogliamo rendere visibili anche i codici di tastiera dobbiamo modificare l'estensione in **.doc**. Il documento viene aperto con Word. I codici di tastiera diventano visibili sul documento clikkando su ¶ presente nella barra degli strumenti. Come possiamo vedere, la presenza dei codici di tastiera viene visualizzata utilizzando appositi simboli grafici.

Modifichiamo ora l'estensione del file 1 H in **.htm** e proviamo ad aprirlo con un doppio cilk del mouse. Il documento verrà aperto con il browser installato sul calcolatore (il più diffuso è Internet Explorer della Microsoft ma è diffuso anche Mozilla Firefox e si sta affermando sempre di più Google Crome.

La pagina web riporta tutti i nostri dati collocati in sequenza separati da un semplice spazio. Scopriamo così che i browser di Internet interpretano tutti i codici di tastiera come semplici spazi.
Mentre Word rende eventualmente visibili solo i codici di tastiera e nasconde tutti gli ordini che servono per gestire la pagina (dimensione del testo, colore, ecc…), con i browser di Internet è possibile vedere sia la finestra che contiene ciò che si vuole visualizzare (**pagina web**) che quella che contiene, oltre al testo, anche gli ordini utilizzati per la visualizzazione (**finestra del codice**). Per rendere visibile la finestra del codice basta selezionare sulla barra dei menù ***Visualizza*** / ***Origine*** (in Crome: *tasto destro del mouse / visualizza sorgente pagina*).
Comparirà un documento identico a quello che avevamo visto aprendo il file 1 H.txt. Infatti la finestra del codice html è, a tutti gli effetti, un documento di Blocco Note. Mentre la finestra del codice non è modificabile se aperta con un browser, se la riapriamo utilizzando Blocco Note potremo scrivervi sia del testo che degli ordini HTML. Se modificchiamo il testo, salviamo e poi riapriamo la pagina web, vedremo che le modifiche sono state eseguite. Inserendo poi in un punto qualsiasi del testo l'ordine **
** (che in HTML corrisponde all'ordine di invio a capo), salvando e riaprendo vedremo che è stato eseguito l'invio a capo.
Inoltre, inserendo all'inizio dell'elenco: **<title>elenco degli alunni di 1H</title>**, alla successiva riapertura potremo vedere che la pagina web ha una nuova intestazione.

L'uso di Dreamweaver

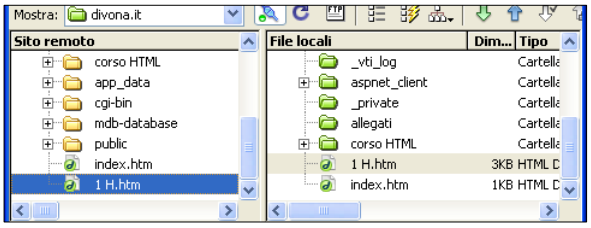
Se sul nostro computer è installato un browser possiamo creare sul nostro calcolatore dei file HTML contenenti del testo e gli ordini per la loro visualizzazione. Se però vogliamo visualizzare le nostre pagine su Internet dobbiamo come prima cosa disporre di un provider che ci metta a disposizione dello spazio su un **dominio**, poi dobbiamo aver installato sul nostro calcolatore un software che disponga dei **protocolli FTP** necessari per collegarsi con il computer del provider e memorizzarvi i file html da noi preparati (**pubblicazione**).
Sul nostro computer la cartella **divona.it** (***sito locale***) è destinata ad ospitare ciò che poi andrà memorizzato sul calcolatore del provider (***sito remoto***) e dunque pubblicato su Internet.
La procedura per organizzare e installare il sito sul nostro computer (compresi username e password per collegarsi al provider) viene descritta a parte.

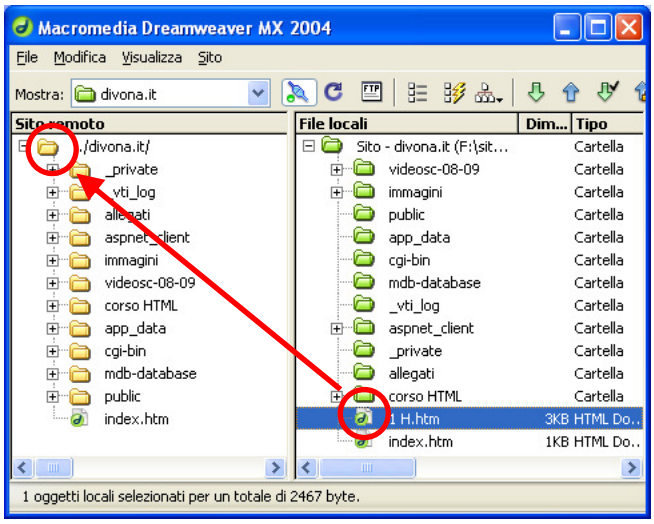


Il software che utilizzeremo, prodotto da Macromedia, è Dreamweaver. Oltre che fornire, in una finestra a sinistra, una mappa completa del sito, esso presenta due finestre di lavoro: la finestra del ***codice***, che abbiamo già visto in precedenza, e quella di ***progettazione*** che corrisponde, in linea di massima, a quanto verrà poi visualizzato sul web. Dreamweaver ci permette di realizzare delle pagine anche senza conoscere il linguaggio HTML. In questo caso traduce in istruzioni HTML ciò che noi facciamo nella finestra di progettazione.
Noi comunque cercheremo di utilizzare al massimo la pagina del codice visto che il nostro scopo è quello di imparare ad utilizzare il linguaggio HTML.

Primo esercizio: pubblicare 1 H.htm

Come prima cosa, senza utilizzare Dreamweaver, collochiamo il file 1 H.htm nella cartella divona.it. Aperto poi Dreamweaver, lo potremo vedere nella mappa del sito locale.
Aperta la visualizzazione e il collegamento con il sito remoto, clikkiamo sul nome del file e lo trasciniamo nella cartella divona.it del sito remoto.
Infatti, quando si pubblica, file e cartelle vanno trascinati sull'icona della cartella che li deve contenere.





Subito dopo vedremo comparire 1 H.htm anche nella mappa del sito remoto. Da questo momento il nostro file è visibile su tutti i computer del pianeta che siano collegati ad Internet.
Visto che il file non è collegato ad altre pagine, per poterlo vedere dovremo raggiungerlo direttamente battendo il suo indirizzo. Su un calcolatore qualsiasi, che sia in rete, apriamo il browser e scriviamo nella barra degli indirizzi: **http://www.divona.it/1 H.htm**
Ci comparirà subito la pagina web che contiene i nostri dati
Da quello che abbiamo visto, realizzare e pubblicare una pagina web può sembrare molto facile. E' così se noi ci proponiamo di far comparire in Internet del semplice testo senza preoccuparci del fatto che esso possa essere trovato anche da chi non conosce il suo **indirizzo diretto**.
Se vogliamo che le nostre pagine siano presentabili dovremo poter *modificare dimensioni, stile e allineamenti del testo*. Dovremo poi *inserire immagini ma anche musiche e video*. Dovremo poi far si che il nostro sito sia *rilevato dai motori di ricerca* e che dalla *pagina principale si possano raggiungere le altre pagine senza conoscerne l'indirizzo diretto*.

Tag, proprietà e valori

Prima di analizzare la struttura della pagina proposta da Dreamweaver è bene soffermarsi su questi tre elementi del linguaggio HTML: il ***tag***, l'***attributo***, e il ***valore***.
Il **tag** è l'unità fondamentale, la potremmo definire l'***istruzione*** che fa capire al browser come interpretare il codice e come visualizzarlo sul monitor. Attraverso i tag possiamo definire molti elementi di un documento: *paragrafi, colore del testo, collegamenti ipertestuali e quant'altro*.
Abbiamo già utilizzato un paio di tag (uno di apertura e uno di chiusura) per dare un'intestazione alla pagina web contenente i nostri dati:

<title>elenco degli alunni di 1H</title>

Come possiamo notare ogni tag è caratterizzato da tre componenti:

< *il segno di minore* **title** *il nome del tag* > *il segno di maggiore*

Un tag di chiusura ha la particolarità che dopo il segno di minore ha uno slash / che comunica al browser che quello è, per l'appunto, un tag di chiusura.
Tra i due tag vi è poi il contenuto che è quello che verrà formattato secondo le regole del tag che lo comprende.
In realtà non tutti i tag necessitano di essere chiusi ve ne sono alcuni che sono a se stanti.
La maggior parte dei tag, quasi tutti, supportano delle **proprietà**, ossia dei parametri che servono a definire ad esempio l'allineamento del testo, il suo colore, ecc... Ogni proprietà è costituita da un **valore** che

determinerà, ad esempio, se questo verrà visualizzato a destra o a sinistra, o se il testo sarà di colore rosso piuttosto che nero.
Facciamo un esempio:
Abbiamo visto che per far comparire del testo in una pagina web basta scriverlo, ci penserà poi il browser a gestirlo dandogli le caratteristiche di default*.

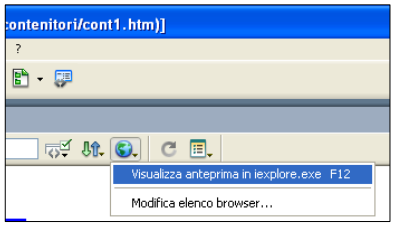
default=modalità con cui lavora un'applicazione alla sua apertura senza che l'utente debba intervenire

Decidiamo invece di intervenire sulle caratteristiche del testo. In questo caso dobbiamo utilizzare il tag (istruzione) **font**, che serve per la gestione del testo. Ad esso possono essere abbinate le *proprietà* **color** (colore), **size** (dimensione), **face** (tipo di carattere), ecc… Infine alle proprietà andrà assegnato un valore. Ad esempio la proprietà color potrà avere i *valori* **"red"**, **"black"**, **"blue"**, **"white"**, ecc…

Ecco il tag ***font*** a cui vengono assegnate le proprietà ***face*** e ***color*** e scelti i relativi valori:

```
<font face="Arial, Helvetica, sans-serif" color="red"> classe prima H </font>
```

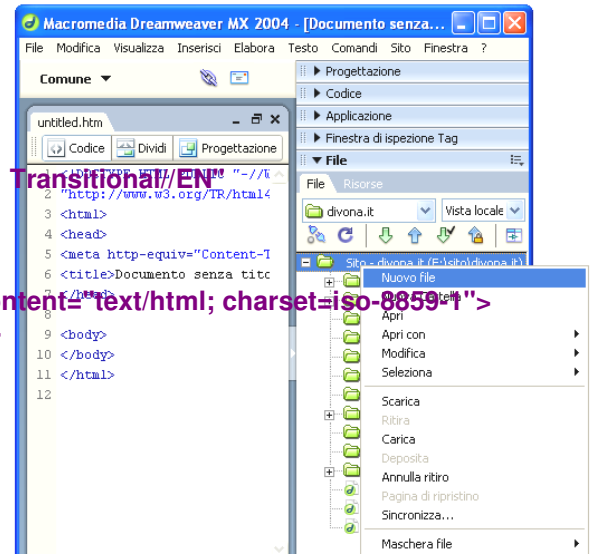
E' bene ricordare che per vedere l'effetto di un tag non dobbiamo accontentarci di quello che vediamo nella schermata di lavoro di Dreamweaver. Spesso è necessario eseguire un'anteprima con Internet Explorer o con Mozilla.



Ora che abbiamo capito com'è strutturato un tag possiamo passare ad analizzare la struttura di una pagina.

La pagina web standard

Dopo aver evidenziato la cartella divona.it, clikkando sul tasto destro appare una finestra. Selezioniamo *Nuovo File*.
Nella mappa comparirà un nuovo file chiamato untitled.htm.
Se ne apriamo la finestra del codice scopriremo la struttura base che deve avere una pagina web:




```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
        <head>
                <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
                <title>Documento senza titolo</title>
        </head>

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

il DocType

- La prima riga di ogni pagina web (quindi ancora prima di qualsiasi tag, nel caso dell'HTML prima del tag <html>), che tenga conto delle specifiche, è il *DocType*. Questa riga fornisce al browser tutte le informazioni per capire le caratteristiche della pagina che interpreterà

La maggior parte delle pagine web, attualmente, ha questo tipo di DocType:
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//IT"
http://www.w3.org/TR/html4/loose.dtd>

Come abbiamo visto, non è obbligatorio specificare il DocType infatti la pagina verrà comunque visualizzata dal browser, e nella maggior parte dei casi correttamente. Tuttavia consigliamo caldamente l'uso di questo tag, così facendo la pagina sarà in armonia con gli standard di validità

Il tag **<html>** indica all'interprete, in questo caso il browser, che il documento è stato formattato in HTML e pertanto il suo contenuto dovrà essere interpretato secondo le specifiche del linguaggio.
I tag **<head>** d'apertura e **</head>** di chiusura, servono a definire l'intestazione del documento, ossia tutte quelle informazioni che servono a definire il contenuto della pagina.

All'interno dei tag **<body>** e **</body>** è presente invece il resto del documento, il *corpo* della pagina, ciò che effettivamente poi verrà visualizzato sul browser.
Il tag di chiusura **</html>** serve invece per comunicare all'interprete che il codice HTML è terminato è pertanto tutto ciò che sarà scritto successivamente a questo tag verrà interpretato come normale testo.
L'HTML è un linguaggio **case-unsensitive** pertanto scrivere i tag tutti in maiuscolo o in minuscolo è indifferente, anche alternandoli non cambia nulla.

Inserire dei commenti

Come ogni buon programmatore sa, è molto importante inserire dei commenti all'interno del codice, così facendo infatti il codice sarà comprensibile anche a distanza di tempo. I commenti sono molto utili e non verranno letti dal browser di conseguenza la pagina non verrà influenzata dalla presenza o meno dei commenti. Nell'HTML inserire commenti è molto semplice, basta usare la seguente sintassi:

```
<!-- corso sull'HTML -->
```

l'intestazione della pagina (head)

meta tag *intestazione*

Oltre al titolo, l'intestazione della pagina comprende i meta tag. Questi sono dei tag speciali che permettono di definire il contenuto della pagina web, in modo, ad esempio, da rendere l'indicizzazione da parte dei motori di ricerca più rapida ed efficace. I meta tag non producono alcun tipo di effetto grafico nella visualizzazione della pagina e sono costituiti da due parti:

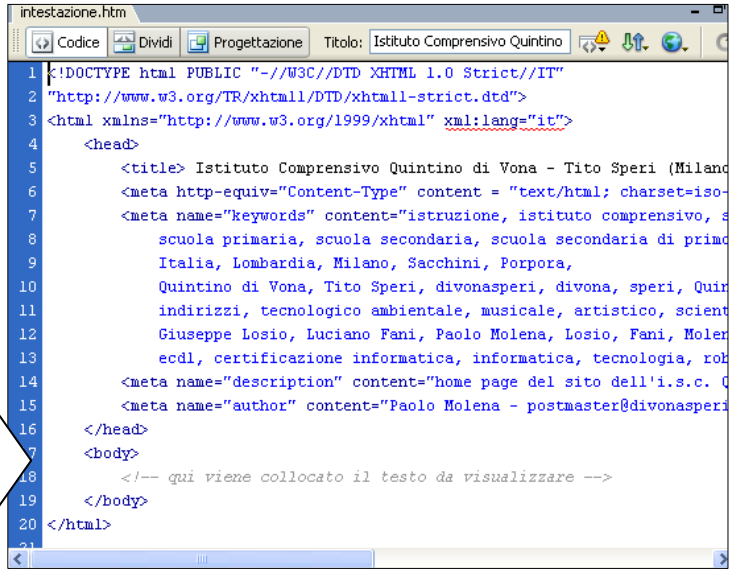
- Nome del meta tag ad esempio: meta name="author"
- Valore del meta tag ad esempio: content="Paolo Molena - postmaster@divonasperi.it"

meta name

i meta tag **name** danno una serie di informazioni che i motori di ricerca utilizzeranno per:

- memorizzare i contenuti della pagina (***keywords***)
- per descriverla (***description***)
- per individuarne l'autore (***author***)

i meta tag collocati sulla home page del sito della nostra scuola. Essi forniscono ai motori di ricerca (Google, Live, Yahoo!, Ask, ecc…) le parole chiave con le quali può essere raggiunto il nostro sito



Passiamo ora ad esaminare i tag utilizzati per gestire ciò che viene visualizzato nella pagina html e che è compresi tra i tag <body> e </body>.
Vedremo più avanti che questa operazione può anche essere fatta al di fuori dalla pagina html utilizzando i fogli di stile.

Il corpo della pagina

Lo sfondo della pagina

Il colore di sfondo pagina

Per poter settare un colore come sfondo di una pagina è sufficiente servirsi dell'attributo **bgcolor** che va inserito all'interno del tag <body>. L'attributo **bgcolor** sta per *background color* che com'è facile intuire corrisponde all'italiano: colore di sfondo. Come valore dell'attributo bgcolor si può impostare qualsiasi colore, sia attraverso il suo valore nominale che attraverso il suo valore esadecimale.
La sintassi per l'uso di **bgcolor**:

```
<body bgcolor="#FFFF00">
```
La pagina avrà uno sfondo di colore giallo

Immagine di sfondo pagina *sfondo*

La collocazione di un'immagine come sfondo alla pagina è un'operazione che va svolta con prudenza. Contrariamente a Word, la pagina HTML non incorpora le immagini. Esse rimarranno separate (preferibilmente memorizzate in file .jpg) e saranno lette e incorporate nella pagina solo quando essa viene aperta dal browser. Le immagini mantengono la dimensione originale e non subiscono nessun adattamento. Dunque, se l'immagine è più grande della pagina, ne comparirà solo una parte. Se è più piccola, essa verrà ripetuta sia orizzontalmente che verticalmente. E' dunque importante ridimensionare in modo opportuno l'immagine con un apposito programma di grafica prima di inserirla in un sito. Inoltre bisognerà stare attenti alla colorazione del testo in modo che questo sia in ogni caso leggibile.
L'attributo per poter inserire un'immagine di sfondo è **background** che segue il tag <body>, il suo valore ovviamente sarà l'url che porterà all'immagine in questione. Nell'esempio (**sfondo**) viene caricata come sfondo l'immagine lago.jpg che è collocata nella cartella immagini:

```
<body background="immagini/lago.jpg">
        ……
</body>
```

IL TESTO

Il tag font e le proprietà face, color e size

face *vedi testi-1*

E' possibile scegliere l'aspetto che assumerà il testo attraverso l'attributo ***face*** i cui valori sono tutti i nomi di font esistenti. Tuttavia è preferibile scegliere sempre le famiglie generiche di font (ogni font appartiene a una famiglia) in maniera tale da garantire la massima compatibilità con tutta l'utenza rinunciando ad un font che probabilmente l'utente non avrà installato sul proprio pc. I possibili valori dell'attributo **face** sono molteplici tra cui: *Arial*, *Verdana*, *Helvetica*, *Times*, più tutte le famiglie generiche di font e così via.

Famiglia di font	Caratteristiche	Esempi di font
serif	Sono proporzionati e hanno le grazie	Times, Georgia
sans-serif	Sono proporzionati e non hanno le grazie	Helvetica, Geneva, Verdana,Arial
monospace	Non sono proporzionati, con o senza grazie	Courier, Courier New
cursive	Hanno le grazie	Comic Sans
fantasy	Non sono classificabili	Woodblock

Color *testi-1a*

Il colore si può impostare attraverso l'attributo **color** i cui valori possono essere tutti i colori disponibili sia in forma nominale che in forma esadecimale. Esempio:

```
<body>
    <font face="Arial, Helvetica, sans-serif" color="red">classe prima H</font>
    <font face="Times New Roman, Times, serif" color="#FF0000">classe prima H</font>
    <font face="Courier New, Courier, mono" color="blue">classe prima H</font>
</body>
```

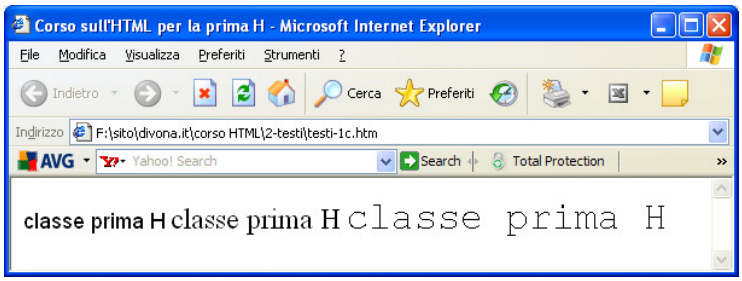
Come abbiamo visto negli esempi si può definire il colore utilizzando una parola chiave. Si tratta di 16 parole che definiscono i colori della palette VGA standard di Windows:

black | navy | blue | maroon | purple | green | red | teal | fuchsia | olive | gray | lime | aqua | silver | yellow | white

In alternativa, si può definire il colore utilizzando il suo codice esadecimale.

Size (dimensioni) *testi-1c*

Le dimensioni sono determinate dal valore dell'attributo **size**, che può essere compreso tra *1* e *7*. Il valore di default è di *3*. Qualora si inserisse un valore minore di uno o maggiore di sette, verrano intrepretati dal browser come dimensione 1 o 7.

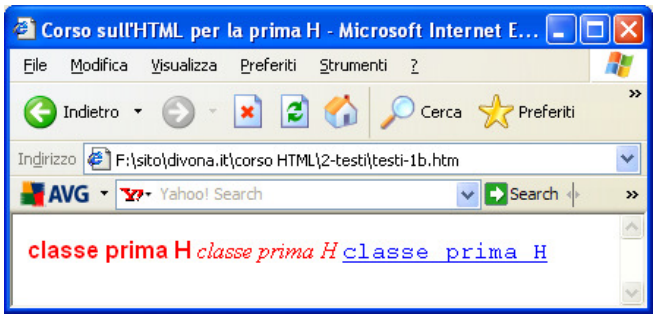



```
<body>
    <font face="Arial, Helvetica, sans-serif" size="+1">classe prima H</font>
    <font face="Times New Roman, Times, serif" size="+2">classe prima H</font>
    <font face="Courier New, Courier, mono" size="+3">classe prima H</font>
</body>
```

Lo stile del testo *testi-1b*

Ci sono diversi valori per impostare lo stile di un testo, di seguito una tabella riassuntiva.

Stile	Ingl.	Significato
<b>	bold	Rende il testo in grassetto
<i>	italics	Rende il testo in corsivo
<u>	underscore	Rende il testo sottolineato




```
<body>
    <font face="Arial, Helvetica, sans-serif" color="red"><b>classe prima H</b></font>
    <font face="Times New Roman, Times, serif" color="#FF0000"><i>classe prima H</i></font>
    <font face="Courier New, Courier, mono" color="blue"><u>classe prima H</u></font>
</body>
```

I CONTENITORI

Anche se all'interno del body è possibile scrivere del testo che verrà visualizzato senza particolari problemi, il testo che si inserisce in una pagina HTML è quasi sempre inserito in contenitori. In questo modo si creano apposite aree che possono essere gestite autonomamente per quanto riguarda formati, font, colori, dimensioni ecc..
Alcuni di questi contenitori, il titolo e il paragrafo, già li conosciamo perché li utilizziamo normalmente per organizzare i nostri testi. Altri, come il div e lo span, sono tipici dell'HTML.
Prima di esaminarli uno per uno, osserviamo le pagine di esempio su ***gestire testi > contenitori***.
In queste pagine, per evidenziare le caratteristiche dei singoli contenitori, è stato loro applicato il tag style con le proprietà border, width, height, dotted, blue che, comunque, tratteremo successivamente.

Titoli (h) *cont1 e cont2*

I titoli sono molto utili nelle pagine per dare maggior visibilità a un elemento che, solitamente, racchiude ciò che sarà possibile leggere nei paragrafi che seguono. Vi sono 6 livelli di titoli, il primo è quello più grande e man mano che si va verso il sesto il titolo risulterà più piccolo. I titoli sono automaticamente visualizzati dall'interprete in grassetto e lasciano prima e dopo di sé, una riga vuota (in modo da essere più leggibile). La sintassi per definire un titolo è la seguente:

```
<h1>Equipaggiamento individuale</h1>
```

Al posto del numero uno si possono inserire i numeri: 2,3,4,5,6 a seconda della grandezza che si desidera visualizzare.

Paragrafi (p), div e span *cont3, cont4 e cont5*

Come abbiamo già visto negli esempi precedenti, vi sono essenzialmente tre tag che svolgono la funzione di suddivisione del testo: <p>, ,e . Le differenze tra questi tag sono minime ma abbastanza determinati, nel contesto della pagina, per ciò che verrà visualizzato dal browser.

Tag	Significato
<p>	sta per *paragraph* ed è infatti l'elemento che rappresenta la nostra concezione di paragrafo. Lascia una riga vuota prima e dopo il testo al suo interno
	sta per *division* e definisce un blocco di testo al suo interno. Non lascia righe vuote, dopo il testo al suo interno va a capo
	fa da contenitore al testo presente al suo interno, viene molto usato per essere altamente personalizzabile attraverso i fogli di stile. Non lascia nè linee vuote, nè va a capo

- <p> sta per *paragraph* ed è infatti l'elemento che rappresenta la nostra concezione di paragrafo.
- sta per *division* e definisce un blocco di testo al suo interno.
- fa da contenitore al testo presente al suo interno, viene molto usato per essere altamente personalizzabile attraverso i fogli di stile (che vedremo in seguito).

Ecco un esempio:

```
<body>
    <h3>questo è un titolo (h3)</h3>
    <p>questo è un paragrafo(p)</p>
    <div>questo è un div </div>
    <span>questo è uno span</span>
</body>
```

linee divisorie

Per impaginare meglio un testo si potrebbe rendere necessaria una linea che divida ad esempio un articolo dall'altro il tag che svolge questa funzione è <hr>. Le sue principali proprietà sono **width** che ne indica la larghezza, richiede un valore in pixel o in precentuale; **size** che ne regola le dimensioni, il cui valore viene espresso in pixel di default 2; e infine **align** che ci permetterà di allineare la barra rispettivamente a destra, al centro o a sinistra.

Nell' esempio visto su **cont1** viene disegnata una linea con larghezza al 100% della pagina e dimensione 3:

```
<body>
    <h1>Equipaggiamento individuale</h1>
    <p>Ogni alunno…..</p>
    <hr width="100%" size="3">
    <div><strong>Equipaggiamento individuale</strong></div>
    <div> …..</div>
</body>
```

Il tag <hr> non prevede un tag di chiusura

andare a capo

Il tag che svolge la funzione per andare a capo è
. Anche questo non prevede un tag di chiusura

IL TAG STYLE PER LA GESTIONE DELLE CARATTERISTICHE DEI CONTENITORI

I bord - border

I bordi possono essere usati per molte cose, per esempio come elemento decorativo o per sottolineare una separazione fra due oggetti. Su **cont3** abbiamo già visto vari esempi di gestione di bordi applicati a vari tipi di contenitori.

Spessore dei bordi - border-width

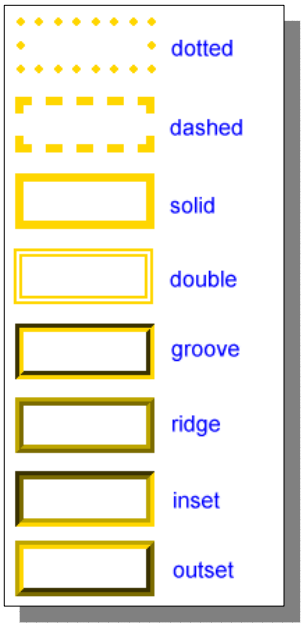
Lo spessore dei bordi viene definito con la proprietà border-width, che può avere valore thin (sottile), medium (medio), e thick (spesso), o un valore numerico indicato in pixel. La figura sotto illustra come:

Colore dei bordi - border-color

La proprietà border-color definisce il colore del bordo. I valori sono i normali valori per i colori, come per esempio "#123456", "rgb(123,123,123)" o "yellow".

Tipi di bordi - border-style

E' possibile scegliere fra diversi tipi di bordi. A fianco sono mostrati 8 diversi tipi di bordi così come vengono interpretati da Internet Explorer 5.5. Tutti gli esempi sono mostrati con colore "gold" e spessore "thick", ma possono essere chiaramente mostrati con altri colori e altri spessori.



Le proprietà dei bordi possono essere date per esteso ma possono essere riassunte in un unico ordine. Negli esempi visti **cont3**, invece di usare:

```
<p style="border-width: 3px; border-style:double; border-color:blue;">
```

È stato usato:

```
<p style="border:3px double blue;">
```

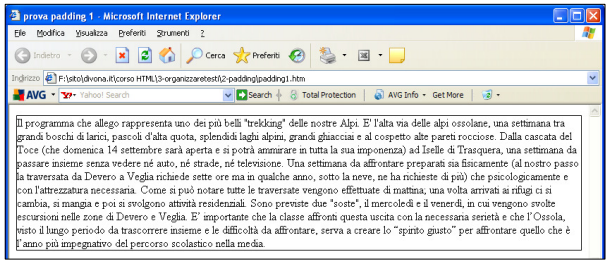
Per definire un bordo di 3 pixel, double, blue.

Il padding *padding1, padding2, padding2a e paddin2b*

Come si è visto negli esempi precedenti, alcuni elementi (i titoli e i paragrafi) creano automaticamente uno spazio intorno ad essi (un **margine**), altri come i div e gli span ne sono privi. Tutti gli elementi visti invece non creano alcuno spazio tra il loro bordo e il testo.
Per svolgere questo compito è stato creato il **tag padding** che può essere utilizzato per tutti i lati dell'elemento o per i singoli lati.

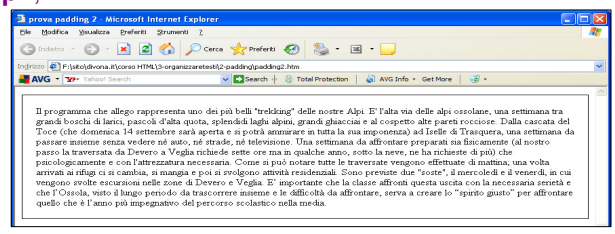
A fianco in **padding1** abbiamo un div con un border e senza padding:

```
<body>
    <div style=" border: 1px solid black; ">
        Il programma ……nella media.
    </div>
</body>
```



Sotto invece lo stesso div con un padding di 20 pixel su tutti i lati.

```
<body>
    <div style=" border: 1px solid black; padding:20px;">
        Il programma …. nella media.
    </div>
</body>
```



In **padding2a e paddin2b** possiamo invece vedere come differenziare il padding sui vari lati con ordini diversi o con un unico ordine.

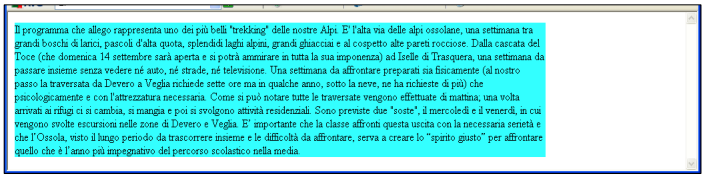
```
<body>
    <div style=" border: 1px solid black; padding-top:10px; padding-right:40px; padding-
        bottom:10px; padding-left:20px;">
        …
    </div>
</body>
```

Il tag **padding** potrà anche essere modificato con un unico ordine dando però le misure nel seguente ordine:
alto, destra, basso, sinistra (in senso orario a partire dall'alto).

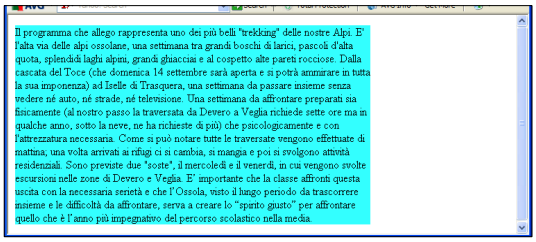
```
<body>
    <div style=" border: 1px solid black; padding:10px 40px 10px 20px;">
    Il programma …… nella media.
    </div>
</body>
```

I margini - margin *margin1*

Il **tag margin** serve per definire lo spazio da collocare oltre i bordi dei contenitori. Può essere utilizzato per quelli che non lo prevedono, ma anche per modificare il margine di default di quelli che lo prevedono.




```
<body>
    <div style=" background-color: #33FFFF; margin-right:200px ">
            Il programma …. nella media.
    </div>
</body>
```



In **margin1** è stato dato ad un div un margine destro di 200 pixel. Inoltre, per rendere evidente lo spazio occupato dal div, gli è stato dato un colore di sfondo.
Come si vede dall'esempio a destra se si ridimensiona la finestra il margine, dato in pixel, rimane fisso.

la posizione degli elementi nella pagina HTML con position

posizione d'un elemento in modo statico, assoluto e relativo

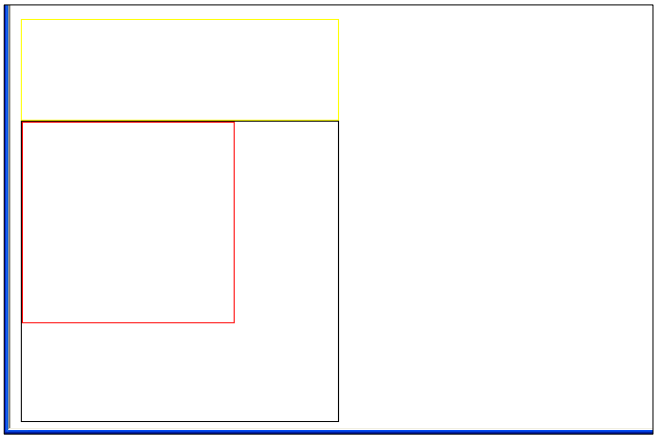
La proprietà **position** con i suoi **valori *static*, *absolute*** e ***relative*** permette di posizionare un elemento in relazione alla pagina che lo contiene o alla sua normale posizione di flusso. Con l'uso di semplici esempi vedremo l'applicazione di questi 3 differenti valori accompagnata dall'uso delle **dichiarazioni *top*** (alto), ***bottom*** (basso), ***left*** (sinistra) e ***right*** (destra).

Per iniziare, vediamo quale sarebbe il ***flusso normale* della pagina** con l'uso di 3 ***div*** con bordi di diverso colore.

- uno giallo ('yellow')
- uno rosso ('red') a quale, di seguito, impostiamo le diverse valore della proprietà **position**
- uno nero ('black') che servirà di ***div*** 'contenitore' a 'red'

Segue il risultato che è anche visibile in ***position1***:

```
<div style="width:300px; height: 100px; border: 1px solid yellow;"></div>
<div style="width:300px; height: 300px; border: 1px solid black;">
        <div style="width:200px; height: 200px; border: 1px solid red;">
        </div>
</div>
```



- position: static;

Il valore **static** (statico) è il valore di default (dunque non è necessario dichiararlo) e corrisponde alla posizione dell'elemento nel flusso normale della pagina.
L'elemento 'statico' non può essere mosso della sua posizione e *ignora le dichiarazioni top, bottom, left e right*.

Il calcolatore collocherà gli elementi static a partire dal vertice in alto a sinistra lasciando però un margine di default di 15 px in alto e di 10px sui bordi.

- position absolute e position relative senza dichiarazioni *position1a*:

Anche se alla proprietà position dell'elemento vengono attribuiti i valori **absolute** (assoluto) o **relative** (relativo) la sua posizione non cambia (visto che non è stata definita una posizione differente da quella di flusso). Se il programma visto precedentemente viene modificato nel seguente modo:

```
<body>
    <div style="width:300px; height: 100px; border: 1px solid yellow;"></div>
    <div style="width:300px; height: 300px; border: 1px solid black;">
        <div style="width:200px; height: 200px; border: 1px solid red; position:absolute;">
        </div>
    </div>
</body>
```

Vedremo che il *div* 'red' verrà collocato nella stessa posizione vista in precedenza.

Può dunque sembrare un lavoro inutile. Vedremo poi che, attribuire ad un elemento una posizione diversa da static senza dichiararne la posizione non ha effetti per lui, ma influisce in modo sostanziale sugli elementi che in esso sono contenuti o che lo seguono.

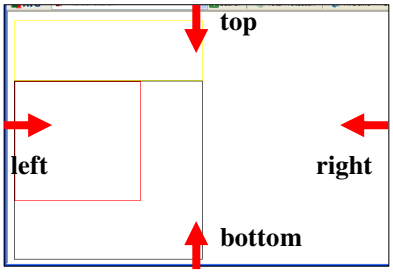
- uso delle dichiarazioni **top**, **bottom**, **left** e **right** con position **absolute**

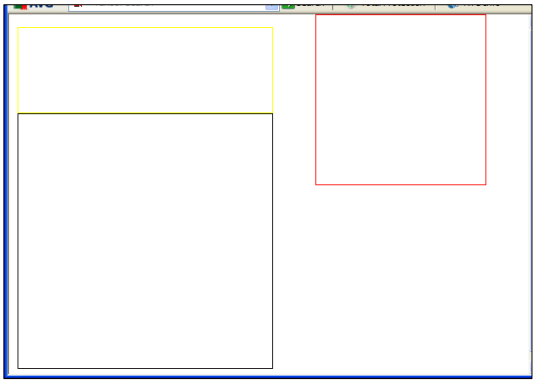
In questo caso è possibile **rimuovere l'elemento dal *flusso normale* della pagina** per poterlo posizionare rispetto alla pagina HTML (normalmente, l'area visiva della finestra del browser) o rispetto all'elemento che lo contiene.
La posizione desiderata va espressa utilizzando le dichiarazioni **top**, **bottom**, **left** e **right** .

primo esempio: position absolute rispetto alla pagina HTLM

L'elemento 'red' è tolto del flusso naturale e si posiziona in alto a 50px a destra in base alla finestra del browser.

```
<body>
    <div style="width:300px; height: 100px; border: 1px solid yellow;"></div>
    <div style="width:300px; height: 300px; border: 1px solid black;">
        <div style="width:200px; height: 200px; border: 1px solid red; position:absolute; top:0; right:50px;">
        </div>
    </div>
</body>
```





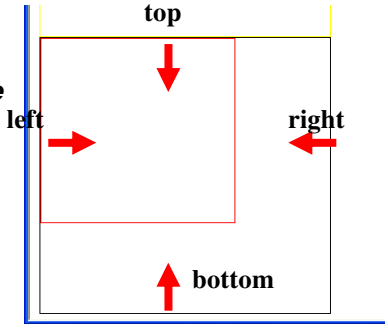
a fianco l'esempio di *position2*

Il risultato sarebbe lo stesso se *div* 'red' non fosse contenuto in *div* 'black' (che ha la posizione di default static).
E' importante notare che in questo caso non viene rispettato il margine di default di 15 px in alto e di 10px sui bordi che il browser lascia quando gli elementi sono static.
Nella gestione della pagina bisogna anche tenere presente che, quando un elemento viene collocato fuori dal flusso normale della pagina, il browser lo gestisce in modo indipendente da quello che viene fatto lavorando sul flusso normale.
Vedi gli esempi in *position2a*, *position2b*, *position2c*

secondo esempio: position absolute rispetto ad un div contenitore

Nel caso l'elemento assoluto sia contenuto in un contenitore con position diversa da static, esso viene posizionato in rapporto al suo ***div*** 'contenitore'.

possibili spostamenti del div 'red' rispetto al suo contenitore 'black'

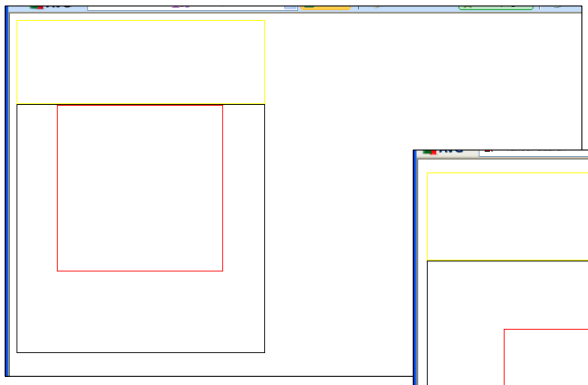


Utilizzando il programma visto nell'esempio precedente e dando al *div* 'black' la proprietà **relative**, l'elemento 'red' si posiziona a 0px dall'alto e 50px da destra rispetto al suo *div* contenitore 'black'.

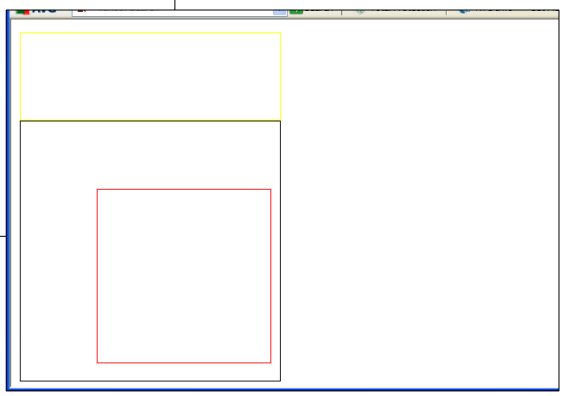
```
<body>
    <div style="width:300px; height: 100px; border: 1px solid yellow;"></div>
    <div style="width:300px; height: 300px; border: 1px solid black;position:relative; ">
        <div style="width:200px; height: 200px; border: 1px solid red; position:absolute; top:0; right:50px; ">
    </div>
</body>
```

esempio in position3

vedi anche l'esempio in ***position3a***



esempio in position3b



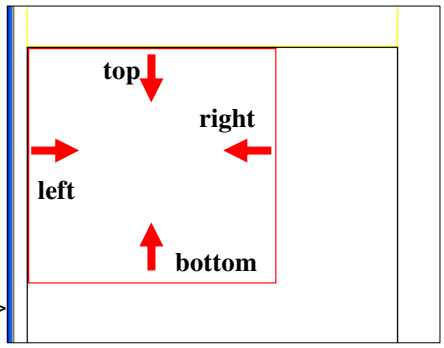
Ripetiamo l'esempio 2 ma invece di **top:0; right:50px,** usiamo ***bottom: 20px; right: 10px;.***

L'elemento 'red' si posiziona a 20 px dal basso e 10px da destra rispetto al *div* 'black'

- uso delle dichiarazioni **top**, **bottom**, **left** e **right** con position **relative**

In questo caso è possibile **spostare** (ma non rimuovere) **l'elemento dal *flusso normale* della pagina** per posizionarlo in base alle sue posizioni iniziali.
Contrariamente alla posizione assoluta, le dichiarazioni **top**, **bottom**, **left** e **right** non rappresentano punti fissi di posizionamento ma sono basate sulla normale posizione di flusso dell'elemento. Esse rappresentano spostamenti rispetto alla posizione normale.

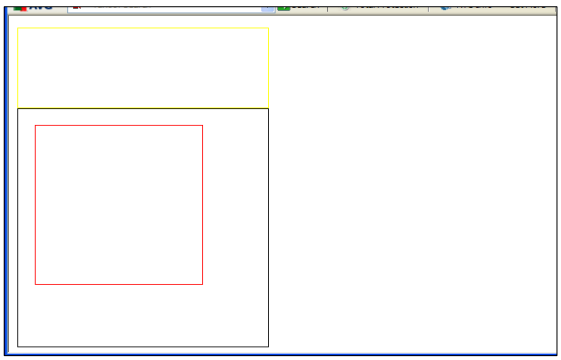
A destra possibili spostamenti del div 'red' rispetto al suo contenitore 'black' >



primo esempio: position relative ***position4***

```
<body>
    <div style="width:300px; height: 100px; border: 1px solid yellow;"></div>
        <div style="width:300px; height: 300px; border: 1px solid black; ">
        <div style="width:200px; height: 200px; border: 1px solid red; position:relative; top:20px; left:20px;">
    </div>
    </div>
</body>
```

Nell'esempio in ***position4*** l'elemento 'red' si sposta dall'alto verso il basso e da sinistra verso destra.
Vedi anche esempio in ***position4a***

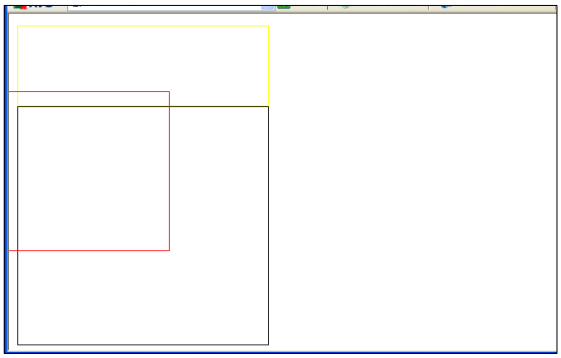


secondo esempio: position relative

Lo stesso che nell'esempio 4 ma invece di:
position: relative; top: 20px; left: 20px;
utilizzeremo:
position: relative; bottom: 20px; right: 40px;

L'elemento 'red' si sposta dal basso verso l'alto aggiungendo 20px alla posizione iniziale *(position1)* della sua parte inferiore e da destra verso sinistra aggiungendo 40px alla posizione iniziale della sua parte destra.
L'elemento rosso "scavalca" così quello nero ed esce con il suo lato sinistro dalla schermata HTML.

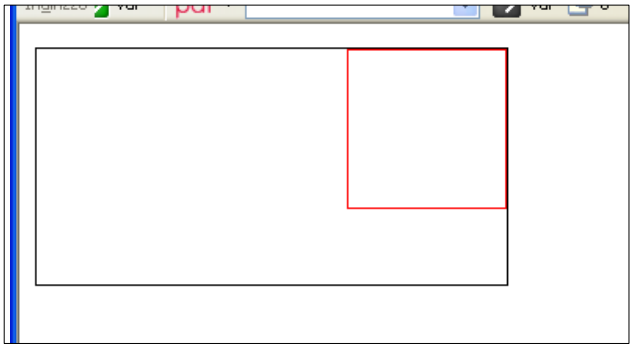
esempio in position4b >



float: un'altra possibilità per stabilire la posizione di un elemento *float1 e float2*

Con questa proprietà è possibile rimuovere un elemento dal normale flusso del documento e spostarlo su uno dei lati (destro o sinistro) del suo elemento contenitore. Il contenuto che circonda l'elemento scorrerà intorno ad esso sul lato opposto rispetto a quello indicato come valore di float.
Valori:

- **left**. L'elemento viene spostato sul lato sinistro del box contenitore, il contenuto scorre a destra.
- **right**. L'elemento viene spostato sul lato destro, il contenuto scorre a sinistra.
- **none**. Valore iniziale e di default in mancanza di una dichiarazione esplicita. L'elemento mantiene la sua posizione normale.



Una nota importantissima. Se usate il float con le immagini non avete problemi perchè esse hanno una dimensione intrinseca che il browser riconosce. Ma se lo applicate ad altri elementi dovete esplicitamente impostare una dimensione orizzontale con la proprietà **width**.

In *float1* abbiamo div 'red' con position absolute contenuto in div 'black' con position relative. E' posizionato in alto a destra.

```
<body>
    <div style="width:300px; height: 150px; border: 1px solid black; position:relative; ">
        <div style="width:100px; height: 100px; border: 1px solid red; position:absolute; top:0px; right:0px;">
        </div>
    </div>
</body>
```

In *float2* abbiamo lo stesso risultato ottenuto utilizzando la proprietà float:

```
<body>
    <div style="width:300px; height: 150px; border: 1px solid black; ">
        <div style="width:100px; height: 100px; border: 1px solid red; float:right;">
        </div>
    </div>
</body>
```

Float è molto utilizzato per posizionare immagini all'interno di altri elementi (paragrafi o div). E' meno frequente la sua applicazione ai div. Per allineare testo e immagini collocate dentro un elemento è anche possibile utilizzare il pannello di controllo di Dreamweaver che crea automaticamente un div dando il valore richiesto alla proprietà '**align**'

ORGANIZZARE UNA PAGINA

Organizzare una pagina HTML con più elementi *div1*

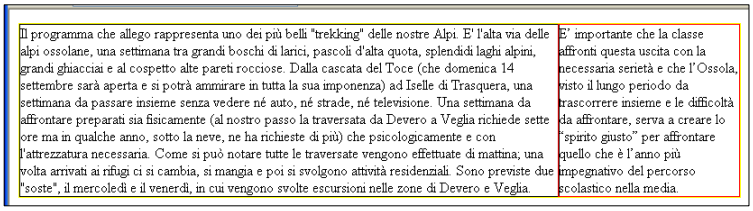
Dopo aver imparato a porre i vari elementi in relazione tra di loro e con la pagina in cui sono collocati, impariamo ora a collocarci del testo e a realizzare una pagina sufficientemente articolata.
Il programma che segue disegna un div con un margine destro di 200px nel quale si colloca un div in posizione assoluta con 10px dal fianco destro della pagina e largo 190px. In questo modo si ricava uno spazio di 10px tra il bordo destro di "black" e quello sinistro di "red".

Perché questa operazione riesca dobbiamo ricordare alcune cose:

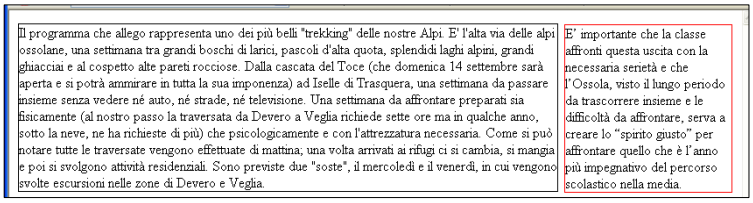
- il div assoluto "red" va scritto, con il suo contenuto, prima del div static "black"
- modificando le dimensioni della finestra HTML la larghezza del div "red" rimane inalterata mentre "black" si allunga o si accorcia a seconda dello spazio a disposizione
- altri div eventualmente aggiunti in coda a "black" seguiranno quest'ultimo ignorando la posizione assoluta di "red"

```
<body>
    <div style=" width:190px; border: 1px solid red; position:absolute; right:10px;">
    E' importante …. nella media.
    </div>
    <div style=" border: 1px solid black; margin-right: 200px;">
    Il programma ……. di Devero e Veglia.
    </div>
</body>
```

A fianco l'esempio che si può trovare in **div1** mentre in **div 2** possiamo vedere gli stessi elementi a cui sono stati aggiunti dei padding sui fianchi destro e sinistro.

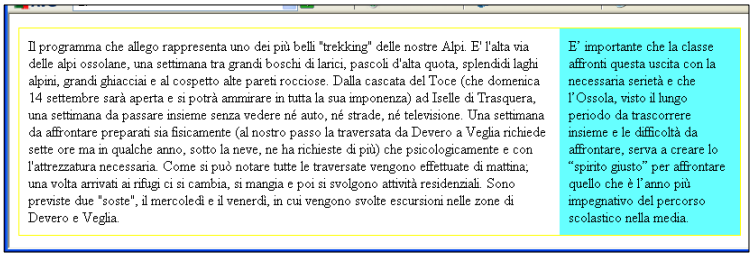


Per creare delle pagine HTML può essere utile inserire i due div contenenti i testi in un altro div che abbia la funzione di contenitore. In **div 3** (a fianco) possiamo vedere i due div "red" e "black", già visti in precedenza, inseriti all'interno di un div "yellow". "Red" occupa tutti i 200 pixel lasciati liberi da "black".



In **div 4** vengono dati gli opportuni padding ai due div che contengono testo.

Infine con **div 5** (a fianco) vengono rimossi i bordi rosso e nero. Per evidenziare le aree dei due div di testo viene colorato di azzurro lo sfondo del div contenitore e di bianco lo sfondo del div di sinistra.
Ecco il programma:



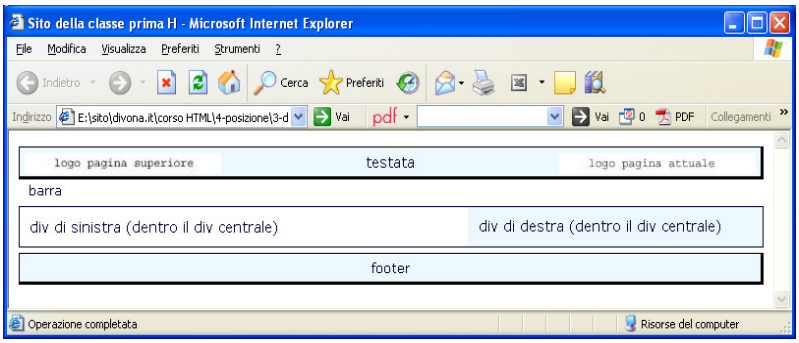

```
<body>
    <div style=" background-color:#66FFFF; border: 1px solid yellow;">
        <div style=" width:180px; padding:10px; position:absolute; right:10px;"> <!-- div di destra -->
            E' importante …….. nella media.
        </div>
        <div style=" background-color:#FFFFFF; padding:10px; margin-right: 200px;"> <!-- div di sinistra -->
            Il programma …….. di Devero e Veglia.
        </div>
    </div>
</body>
```

Ricordare che:

- nella scrittura, il div assoluto collocato a destra ha la precedenza su quello di sinistra
- nel div di destra **width 180** rappresenta lo spazio a disposizione per il testo. Se viene dato un padding di 10px a destra e a sinistra, il div sarà complessivamente largo 200 px e dunque uguale al margine lasciato libero dal div sinistro

I modelli di pagina *div6*

Vediamo ora come organizzare una pagina tipo. In particolare prendiamo in considerazione il modello di pagina con cui vengono organizzate le pagine del sito della nostra scuola.



La pagina inizia con il div 'testata' che ospita l'intestazione della pagina. Sui due lati della testata, utilizzando **float**, ci sono: (da sostituire)
- a sinistra il logo della pagina 'superiore' con un link che ci può riportare ad essa
- a destra il logo della pagina corrente

Subito dopo la barra riporta tutto il percorso sino alla pagina principale del sito. Essa permette un rientro rapido alle pagine superiori.
Segue il div 'centrale' che contiene due div organizzati per ospitare testo. Il div 'centrale' può essere seguito da altri div organizzati in modo diverso. Chiude il div 'footer' che ospita, utilizzando caratteri più piccoli, alcuni dati generali del sito (nome del webmaster, nome del responsabile della pagina, ecc…).
Approfondiremo le tematiche collegate all'organizzazione della pagina nel capitolo dedicato ai fogli di stile.

La gestione delle immagini con il tag img *ima1 e ima2*

Contrariamente a quanto succede nei sw di videoscrittura, le immagini possono essere visualizzate su una pagina HTLM ma non possono far parte di essa. Nel momento in cui la pagina viene attivata, il browser dovrà cercare il file contenente l'immagine e dunque all'interno della pagina web, oltre all'ordine di caricamento, dovrà essere scritto il percorso che porta al file desiderato.
Se non vengono date le dimensioni, il browser caricherà l'immagine con le dimensioni originali e questo potrà causare problemi se l'immagine non è stata ridimensionata con un apposito software (ad es: Photoshop).
Se utilizziamo le proprietà width e height il browser adatterà l'immagine caricata alle dimensioni desiderate. Questa attività però richiede tempo e non garantisce che le proporzioni tra larghezza e altezza vengano rispettate. Dunque è sempre bene che le immagini vengano ridimensionate. Una volta ridimensionate è bene collocare tutte le immagini che andranno collocate su una pagina web in un'apposita cartella che prenderà il nome della pagina in cui verranno caricate.
Nel caso si preveda di far scaricare dall'utente le immagini pubblicate è bene che esse mantengano le dimensioni originali. In questo caso si formano apposite pagine ("***gallerie fotografiche***"), preavvisando l'utente del loro "peso". E' inoltre consigliabile raggruppare le immagini utilizzando dei paragrafi. La sintassi per inserire un'immagine all'interno di una pagina web è la seguente:

<**img src**="url dell'immagine" **title**="testo ">

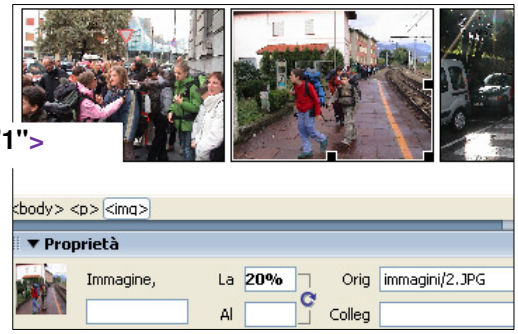
Attributo	**Significato**
border	identifica il bordo che a 0 non è presente, salendo con i numeri aumenta di spessore
title	testo visualizzato se si va con il cursore sopra l'immagine
width	forza la dimensione della foto in larghezza
height	forza la dimensione della foto in altezza
alt	permette di specificare un testo alternativo per l'immagine

È fortemente consigliato inserire testo alternativo per ogni immagine, in modo da migliorare l'accessibiltà per i non vedenti. Il testo alternativo verrà inoltre visualizzato in automatico se il browser non riesce a visualizzare l'immagine.

I valori di **width** e **height** possono essere o espressi in precentuale (rispetto alla pagina) o in pixel.

Nella finestra delle proprietà di Dreamweaver le dimensioni non vengono visualizzate in grassetto; questo vuol dire che le dimensioni sono quelle originali dell'immagine. In questo caso non è necessario scriverle.

Nella pagina **ima1**, viene caricata l'immagine ***2.jpg*** dalla cartella immagini. La larghezza dell'immagine (**20%**) viene prevista in percentuale rispetto alla larghezza della pagina e dunque sarà modificata nel caso di una riduzione della pagina.



<img src="ima/2.JPG" alt="partenza2" width="20%"border="1">

Nella finestra delle proprietà di Dreamweaver la percentale viene visualizzata in grassetto; questo vuol dire che le dimensioni non sono quelle originali dell'immagine. In questo caso è il browser a ridurle in base alla percentuale indicata. Cliccando sulla freccia a fianco verranno riportate a quelle originali.

Nella pagina **ima2**, vengono caricate le stesse immagini viste in **ima1**. Queste però sono state precedentemente ridimensionate a 200*150 e salvate con un nuovo nome. Su ogni immagine è stato organizzato un collegamento verso l'immagine originale.

```
<a href="ima/1.JPG" title="partenza1-scarica l'originale">
    <img src="ima/1picc.jpg" border="1">
</a>
```

In questo caso viene caricata l'immagine **1picc.jpg** (quella ridimensionata) e su di essa è stato organizzato un collegamento con l'immagine originale **1.jpg**.

Il tag <img> non prevede un tag di chiusura

Gli elenchi numerati e puntati *elenchi*

In HTML vengono chiamati elenchi ordinati e non ordinati quelli che in Word vengono chiamati elenchi numerati ed elenchi puntati.
Per definire un elenco o *lista* nell'HTML abbiamo a disposizione diversi tag che possiamo vedere negli esempi seguenti o, all'opera, nella pagina **elenchi** .
Quando vengono organizzati elenchi o tabelle è sempre opportuno utilizzare come contenitori dei div e/o paragrafi. In questo modo sarà più facile regolare la loro posizione.

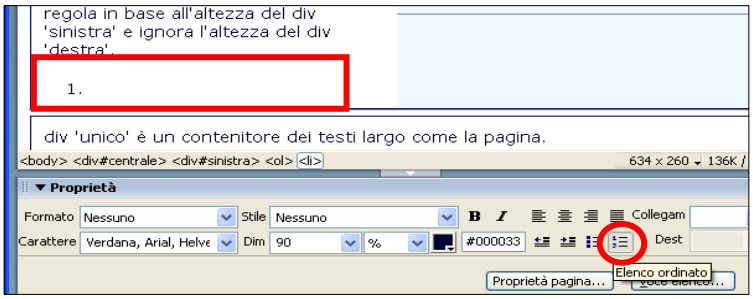
Gli elenchi **ordinati** vengono definiti attraverso il tag <ol> che sta per *Ordered List*. Gli elementi dell'elenco, devono essere inclusi all'interno dei tag ossia *List Item*. Ogni elemento verrà automaticamente preceduto da un numero. Esempio di elenco ordinato:

```
<body>
    <ol>
        <li>1° Elemento in ordine numerico</li>
        <li>2° Elemento in ordine numerico</li>
        <li>3° Elemento in ordine numerico</li>
    </ol>
</body>
```

Gli elenchi **non ordinati**, le cui voci sono precedute da un pallino, vengono definiti attraverso il tag <ul> che sta per *Unordered List*. Gli elementi dell'elenco, devono sempre essere inclusi all'interno dei tag .
Esempio di elenco non ordinato:

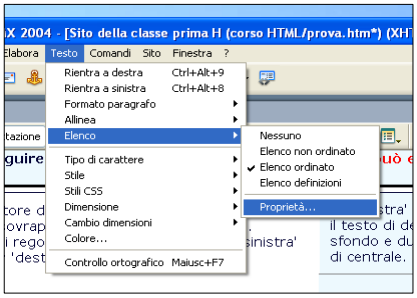
```
<body>
    <ul>
        <li>Elemento uno</li>
        <li>Elemento due</li>
        <li>Elemento tre</li>
    </ul>
</body>
```

Molto più semplicemente possiamo farlo utilizzando il pannello di controllo di Dreamweaver.



Ad esempio lavorando nel div di sinistra:
- ci stacchiamo dal testo precedente con un 'a capo' che apre un nuovo paragrafo
- clikkiamo sul bottone di elenco ordinato. Appare subito il numero uno e il cursore si pone in attesa del testo
- ad ogni 'a capo' il calcolatore inserirà un nuovo numero

Clikkando invece sul bottone a fianco otterremo un elenco non ordinato (puntato).
Se le caratteristiche dei nostri elenchi non ci soddisfa con *Testo > Elenco > Proprietà* possiamo modificarle a nostro piacimento.



L'apertura di tabelle e tabelle *tabelle1 tabelle2*

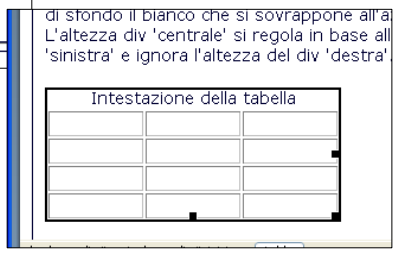
E' anche molto frequente dover utilizzare delle tabelle per organizzare i dati all'interno delle nostre pagine HTML. Anche in questo caso è sempre opportuno utilizzare come contenitori dei div e/o paragrafi per regolare in modo sarà più facile la loro posizione.

Il tag usato per la creazione delle tabelle è **<table>** in coppia con il suo tag di chiusura **</table>**. E' tra questi due tag che si devono inserire colonne e righe, le prime si creano attraverso il tag **<tr>** (*Table Row*) **</tr>**; le seconde, attraverso il tag **<td>** (*Table Data*) **</td>**.
Se si vuole creare una cella d'intestazione con il contenuto in neretto e centrato si può usare il tag **<th>** (*Table Header*), ideato appositamente per questo tipo di funzione.

Possibili proprietà di colonne e righe sono **align** (*alignment*), **valign** (Vertical alignment), **colspan** (*Column span*), e **rowspan**.

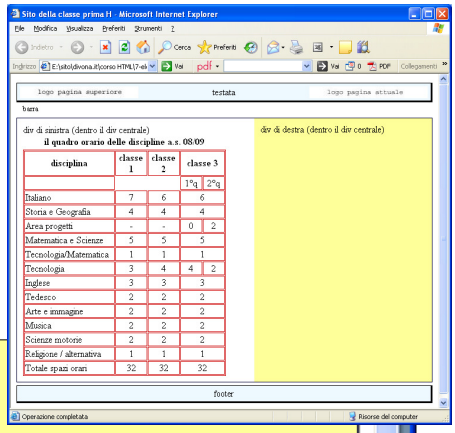
- **align**: allinea il testo della cella a destra (*right*), sinistra (*left*) e centrato (*center*).
- **valign**: allinea il testo della cella sul margine superiore (*top*), sul margine inferiore (*bottom*), e in mezzo (*middle*)
- **colspan** e **rowspan**: il primo indica l'estensione di una colonna, il secondo di una riga

Le celle e le colonne supportano anche le proprietà **height** e **width**.
Vi è un tag interessante che è **<caption>** che permette di aggiungere una descrizione alla tabella (una sorta di didascalia) che supporta l'attributo **align** pertanto si posizionerà o sopra o sotto la tabella secondo il valore specificato in tale attributo.

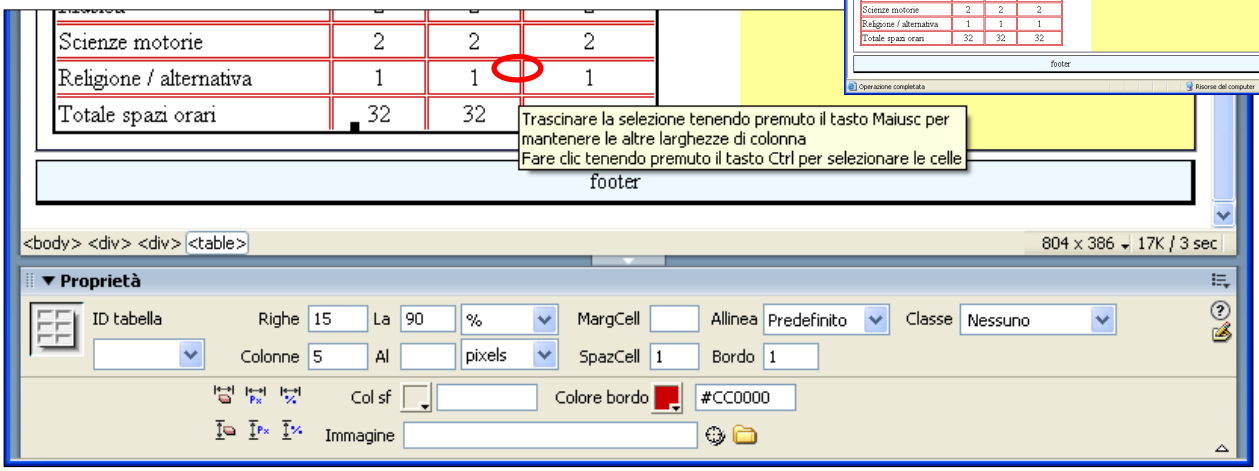


Anche nel caso delle tabelle si può utilizzare Dreamweaver.
Se, come nell'esempio precedente, lavoriamo nel div di sinistra e vogliamo inserire una tabella:
- ci stacchiamo dal testo precedente con un 'a capo' che apre un nuovo paragrafo
- clikkiamo sull'icona della tabella nella barra degli strumenti o utilizziamo *Inserisci > Tabella* nella barra dei menù
- appare una finestra che ci permette di decidere le caratteristiche della tabella
- all'ok comparirà la tabella così come è stata progettata

In *tabelle2* è possibile vedere codice e disegno della tabella utilizzata sul sito della scuola per l'orario scolastico.



Molto comodo è stabilire la larghezza della tabella in percentuale rispetto allo spazio di scrittura del div in cui la tabella è collocata. La larghezza delle colonne, colore e larghezza dei bordi, degli sfondi ecc.. possono essere gestiti tramite il pannello di controllo di Dreamweaver che si ottiene clikkando sul bordo esterno della tabella o con ***Elabora > Tabella > Seleziona tabella.***



La gestione dei suoni e dei filmati con embed

Attraverso l'HTML si possono anche inserire contenuti multimediali all'interno delle proprie pagine. Per permettere di scaricare un file è sufficiente creare un link che come *risorsa* ha il file che vorreste far scaricare.

Suoni *suoni1, suoni2*

Un ordine completo per la gestione dei suoni è **<embed>** che supporta varie proprietà e che rende possibile la gestione del suono con un apposito player.
In *suoni1* vi è un esempio del suo utilizzo. Con:

```
<embed src="suoni1/oboe.mp3" hidden="true" />
```

La musica viene attivata all'apertura della pagina e il player viene tenuto nascosto (**hidden="true"**). E' importante ricordare la chiusura del tag con **/>**.

In *suoni2* due esempi del suo utilizzo. Con:

```
<embed src="immagini-suoni/suoni2/Ulisse.mp3" height=60 width=144 autostart="false" />
```

il suono non parte all'avvio della pagina (autostart="false") ma dovrà essere gestito con il player (60 e 144 sono le dimensioni del player). Con:

```
<embed src="immagini-suoni/suoni2/gr2.mp3" height=60 width=144 autostart="true" ></embed>
```

viene attivato il suono al caricamento della pagina (true su autostart).

Se vogliamo far partire il suono senza visualizzare il player bisognerà inserire la proprietà **hidden*** (nascosto) e non mettere le dimensioni del player. Dunque se scriviamo:

```
<embed src="immagini-suoni/suoni2/gr2.mp3" autostart="true" hidden="true" />
```

All'avvio della pagina potremo ascoltare il suono senza vedere il player.

Da notare anche i due diversi modi di chiudere il tag embed, entrambi validi.
Sia per i suoni che per le immagini, su Internet è importante utilizzare file che abbiano un "peso" ridotto in modo da ridurre i tempi di scaricamento dell'utente. Nell'esempio in *suoni1* il file oboe.mp3 "pesa" 385 KB mentre il file oboe.avi "pesa" 9215 KB.

Le proprietà **hidden** e **autostart** sono facoltative; **hidden** è di default su ***false*** mentre **autostart** è di default su ***true***.
Attenzione! la proprietà autostart non funziona con Google Crome.

Filmati *video1, video2*

Embled può essere utilizzato anche per gestire filmati in formato **.mpg**. Il browser per la proiezione utilizza un apposito player che normalmente viene installato sul computer scaricando Quick Time. Il player ha le dimensioni fisse (larghezza 400 e altezza 300 bordi compresi) che vanno comunque dichiarate. In caso contrario il player non compare o viene deformato.
Ecco l'ordine presente in *video1*.

```
<embed src="video/robot.mpg" width="400" height="300" />
```

Inserendo embed in un div opportunamente dimensionato è possibile proiettare il filmato in una posizione diversa da quella stabilita dal flusso di pagina.
Vediamo in *video 2* il player collocato in alto a destra:

```
<div style="border: 1px solid red; position:absolute; right:10px;">
        <embed src="video/robot.mpg" width="400" height="300" />
</div>
```

Per i filmati diventa indispensabile l'uso di file compressi. Il "peso" di un normale filmato **.avi** diventa un ostacolo insuperabile sul web. Il formato **.mpg** permette già un notevole alleggerimento. Il file robot.mpg "pesa" 13 mb mentre il corrispondente file robot.avi "pesa" 280 mb.
Un altro formato molto utilizzato sul web è **.flv** (flash video). Questo richiede però appositi player che vedremo quando ci occuperemo di Javascript.

L'USO DEL TAG A PER LA GESTIONE DEI COLLEGAMENTI

I links (collegamenti) sono una delle caratteristiche che hanno fatto la fortuna del web, ossia la possibilità di passare rapidamente in un'altra parte della stessa pagina (collegamenti interni) oppure da una pagina ad un'altra dello stesso sito o, anche, di un sito diverso (collegamenti esterni). E' anche possibile aprire una nuova pagina, in genere di dimensioni ridotte, senza chiudere quella presente che rimane attiva.

Collegamenti esterni

Per aprire una nuova pagina

Nell'esempio riportato in *apripagina* la parola calda 'LA NUOVA SOCIETA ...' viene utilizzata per aprire la nuova pagina 'programma.htm'. Per dare il percorso viene utilizzata la proprietà **href**. Come abbiamo già visto per le immagini, spesso viene utilizzata la proprietà title per visualizzare un commento al passaggio del mouse sulla parola calda.
Questo è l'ordine utilizzato:

```
<a href="programma.htm" title="apri il programma dell'incontro"> LA NUOVA SOCIETA ...</a>
```

Per aprire una finestra secondaria

Nell'esempio riportato in *aprifinestra* la parola calda 'LA NUOVA SOCIETA ...' viene utilizzata per aprire una finestra secondaria. Qui non viene utilizzata href (valida per le pagine) ma **onclick**. Al click sulla parola calda viene eseguito l'ordine che segue onclick:

```
onclick="window.open('programma.htm','f','width=600,height=470,left=500, top=10')"
```

Questa è la sintassi precisa dell'ordine:
window.open('percorso', 'nome','dimensioni (width e height)**, posizione in** (left e top)**')**
percorso e nome sono obbligatori mentre le altre caratteristiche sono facoltative.

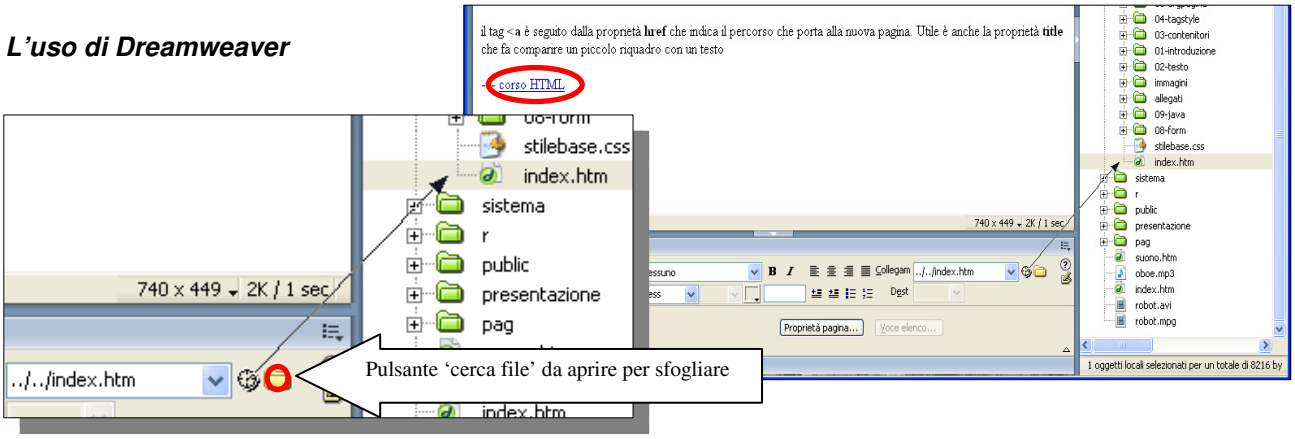
Bisogna tenere presente che, mentre da una pagina HTML possono essere aperte più finestre secondarie che rimangono tutte attive, se da una finestra secondaria viene aperta una nuova finestra secondaria che sostituisce la prima con tutte le sue caratteristiche.
Comunque' in questo caso, al posto di **window.open** è bene utilizzare **location.href**, molto più semplice, che vediamo nell'esempio di *programma*:

```
onclick="location.href = 'achille.htm' "
```

Per aprire sia una nuova pagina che una finestra secondaria

Nell'esempio riportato in *apridue* la parola calda 'LA NUOVA SOCIETA ...' viene utilizzata per aprire sia una nuova pagina (con **href**) che una finestra secondaria (con **onclick).**

L'uso di Dreamweaver



Vediamo sopra come è possibile, utilizzando Dreamweaver, organizzare un link su una parola.
Vi sono due possibilità:

- dopo aver evidenziato la parola su cui si intende realizzare il collegamento, si clikka sul pulsante 'cerca file' che fiancheggia, a destra, lo spazio dei collegamenti. Nella finestra di dialogo che appare selezionare la pagina che si intende collegare. Nome del file e percorso appariranno nello spazio dei collegamenti.
- dopo aver evidenziato la parola su cui si intende realizzare il collegamento, si clikka sull'icona circolare Compare una freccia che dovrà essere trascinata con il mouse sul nome del file da collegare (nella finestra dei file). Nome del file e percorso appariranno nello spazio dei collegamenti.

L'uso di un'immagine per i collegamenti *ima1 e ima2*

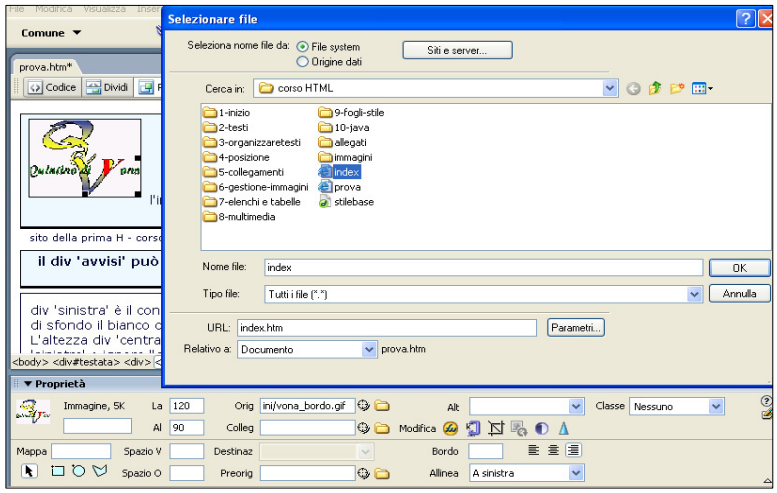
Oltre alle parole calde, anche le immagini possono essere utilizzate per realizzare dei collegamenti con altre pagine HTML. In fondo alla pagina **ima1** vi è un'immagine in formato originale. All'immagine è abbinato un collegamento ad **ima2**:


```
<a href="ima2.htm" title="vai alla pagina della croce del Fop">
<img src="immagini/gruppo.jpg"  width="80" height="60"border="1">
</a>
```

Al collegamento è abbinato anche l'ordine **title** (da non confondere con il tag utilizzato in head per dare un titolo alla pagina). Serve per far comparire un commento al passaggio del mouse.
L'ordine title (che può essere abbinato anche ad un link su parole) può essere inserito di seguito all'ordine di caricamento dell'immagine con gli stessi effetti:

<img src="ima/gruppo.jpg" title="vai alla pagina della croce del Fop" ..... border="1">

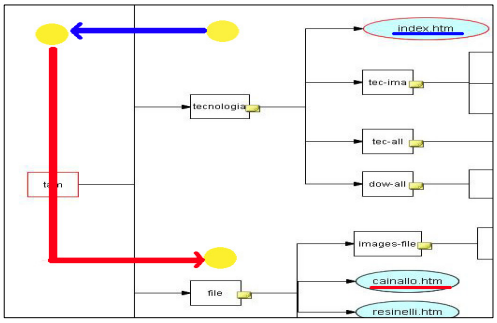
Nell'esempio a fianco vediamo come creare un link su un immagine, utilizzando il pannello di controllo di Dreamweaver:
- si evidenzia l'immagine
- si clikka sul pulsante che fiancheggia, a destra, lo spazio dei collegamenti
- nella finestra di dialogo che appare selezionare la pagina che si intende collegare
- il percorso selezionato apparirà nello spazio dei collegamenti e sarà inserito nella pagina del codice



Ricordarsi di inserire **border="0"** quando non si vuole che l'immagine venga bordata di blu (come di default in caso di immagini utilizzate per collegamenti.
Il tag <img> non prevede un tag di chiusura.

Percorsi assoluti e percorsi relativi

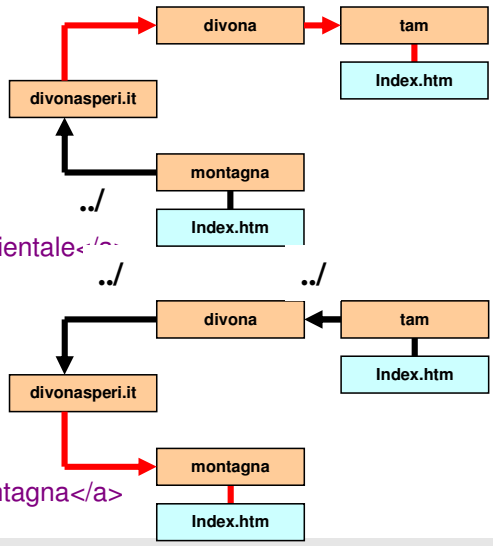
Visto che le immagini sono separate dalla pagina .htm che ospita il testo è molto importante raggrupparle in cartelle che prenderanno nome dalla pagina a cui sono destinate.
Nelle pagine andranno poi inseriti i collegamenti (link) che permetteranno al browser di raggiungere il file contenente l'immagine. Questa verrà letta e visualizzata all'interno della pagina.
I collegamenti ipertestuali dovranno contenere i **percorsi**.
Essi possono essere **assoluti** (con percorsi che partono dalla cartella principale del sito) come: "http://www.divonasperi.it/divona/tam/file/images-file/cainallo/38.jpg" o **relativi** (che permettono di spostarsi a partire da una pagina interna) come "immagini/undici/1.jpg".
In questi casi è spesso necessario "salire" di livello.
Nella mappa a fianco se vogliamo spostarci dalla pagina index.htm di tecnologia alla pagina cainallo.htm di file, potremo inserire il seguente percorso relativo:




```
<a href="../file/cainallo.htm">
```

Dove l'ordine "../" provoca la salita al livello precedente, cioè dalla cartella tecnologia (in cui è ospitato index.htm) si sale alla cartella tam. Da qui poi si scende alla cartella file da dove si apre cainallo htm.

Nell'ordine che segue, collocato sul sito della scuola / pagina del progetto montagna, il click su *indirizzo tecnologico ambientale* porta a risalire dalla cartella del progetto montagna alla cartella principale (***divonasperi.it***) per poi scendere alla cartella ***divona*** e poi alla cartella ***tam*** nella quale viene caricato il file ***index.htm***




```
<a href="../divona/tam/index.htm">indirizzo tecnologico ambientale</a>
```

Se volessimo fare il percorso contrario, cioè spostarci dalla pagina index.htm di tam alla pagina index.htm di montagna dovemmo scrivere:


```
<a href="../../montagna/index.htm"> pagina del progetto montagna</a>
```

Posta elettronica
Attraverso un link è anche possibile aprire il client di posta dell'utente affinchè questo possa mandare una mail con il campo *A:* precompilato. Ecco la sintassi: Parola da visualizzare
Ecco un esempio:

```
<body>
  <a href="mailto:lab4@divonasperi.it">Invia  Mail</a>
</body>
```

I collegamenti interni *link-interni*

I collegamenti interni o *àncore* sono collegamenti che non rimandano a una pagina esterna come abbiamo visto in precedenza, bensì a un contenuto disponibile nella stessa pagina. La loro sintassi è leggermente più complicata dei collegamenti esterni. Per prima cosa, si deve creare un collegamento con l'attributo **name** che farà sì che il collegamento non venga visto come un reale link ma al contrario come un'àncora di cui faremo uso per poter definire un collegamento che punti a quel determinato testo della pagina.
Nell'esempio che compare in **link-interni** abbiamo una sintesi ridotta della pagina dei compiti presente su divona.it. L'ordine:

```
<a href="#1">settimana 8 - 11 settembre</a>
```

Permette lo spostamento all'ancora:

```
<a name="1" ><h4>settimana 8 - 11 settembre - ….. 2010/2011</h4></a>
```

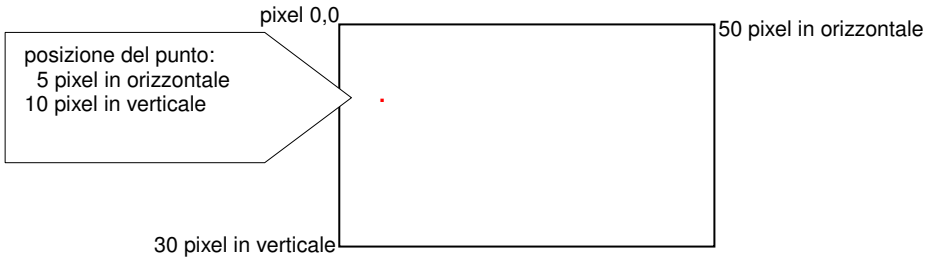
Nella pagina di Dreamweaver compare il segno a fianco che indica l'attivazione dell'ancoraggio a cui si arriva clikkando sulla frase 'settimana 8 - 11 settembre'.

Dunque per poter realizzare un collegamento interno bisogna eseguire le seguenti operazioni:

- Definire l'àncora attraverso l'attributo **name**.
- Richiamarla da un altro link inserendo come valore dell'attributo **href** il valore dell'attributo **name** preceduto dal segno *#*.

Le mappe "sensibili" *mappe*

Può essere utile, in particolari circostanze, delimitare delle aree di un'immagine (che deve avere dimensioni fisse) e renderle sensibili al passaggio del mouse. Con un principio simile a quello studiato in Cartografia, ogni pixel che forma l'immagine avrà una posizione definita dalla riga di pixel e dalla colonna di pixel a cui appartiene. A differenza di Cartografia, l'origine dei punti parte dal punto in alto a sinistra e nella posizione di un punto si legge prima la riga e poi la colonna



Vi sono due fasi:

1. Con il tag **<img src="nome immagine" width="…" height="…." usemap="#nome mappa">** viene caricata un immagine che verrà utilizzata come mappa (usemap). Dall'ordine usemap parte un collegamento interno (#) al tag map che rinchiuderà le definizioni delle aree sensibili contenute nell'immagine. La mappa deve avere un nome che può non corrispondere al nome del file.
2. Si apre il tag **<map name="nome mappa">** che contiene la definizione delle aree sensibili:

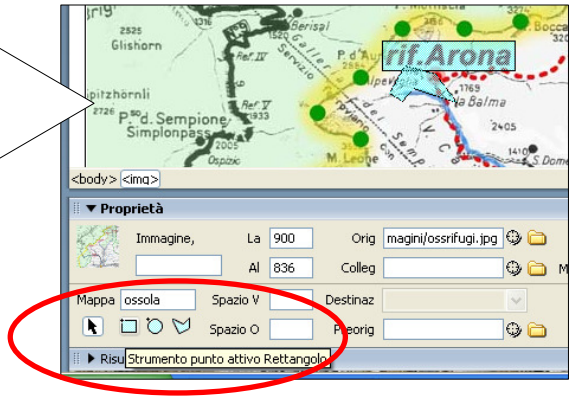
   <area shape="rect" coords=" coordinate " > per un rettangolo (coordinate di due vertici opposti)
   <area shape="circle" coords=" coordinate " > per un cerchio (coordinate del centro e raggio)
   <area shape="poly" coords="coordinate" > per un poligono (coordinate del vertici)

   </map> chiude la definizione delle aree sensibili.

Alla definizione delle aree sensibili possono essere associati vari eventi come in collegamento o la comparsa di un titolo. Ecco, una sintesi, quanto pubblicato su mappe.htm:

```
<img src="mappa-immagini/ossrifugi.jpg" width="900" height="836" >
<map name="ossola">
    <area shape="rect" coords="331,560,454,618" href="file/iselle.htm" title="…..">
    <area shape="circle" coords="517,811,24" title="….." >
    <area shape="poly" coords="517,350,553,319,536,316,524,325,481,326,513,335,507,343" title="….." >
</map>
```

Utilizzando l'area delle proprietà di Dreamweaver è possibile delimitare le aree sensibili individuandone le coordinate



LA GESTIONE DELLE PROPRIETÀ DEGLI ELEMENTI CON I FOGLI DI STILE:

Come abbiamo visto, il linguaggio HTML richiede sempre che il testo venga chiuso dentro differenti tag per formare: **titoli**, **paragrafi**, **div** e **span**. I concetti di titolo e paragrafo sono chiari; il ***div*** potrebbe essere definito un box, una area di testo ben delimitata e separata dalle altre. Lo ***span*** è una parte di testo non separata dalle altre ma che si vuole differenziare in alcune caratteristiche (ad esempio il colore).
Con il tag iniziale dovranno essere descritte tutte le caratteristiche che si vogliono dare al testo che si sta aprendo. Questo fatto determinava un lavoro molto faticoso per ripetere ogni volta tutti gli ordini che dovevano essere dati per assicurare al testo inserito nella pagina un formato accettabile. Inoltre la pagina di codice HTML finiva per essere difficilmente vista la mescolanza tra testo e una gran quantità di ordini.
Fu a questo punto che furono inventati i fogli di stile (dall'inglese Cascading Style Sheet). Con i fogli di stile è possibile separare gli ordini di formato dal testo. Gli ordini di formato sono collocati sul foglio di stile.

Il collegamento al foglio di stile

Nella testata della pagina <head> subito dopo il suo titolo andrà collocato l'ordine con cui associamo quella pagina ad un determinato foglio di stile. Sulla pagina index del nostro corso HTML, l'ordine:

```
<link href="stilebase.css" rel="stylesheet" type="text/css" />
```

Indica al calcolatore che dovrà leggere le caratteristiche degli elementi che formano la pagina sul foglio di stile ***stilebase.css***.

il foglio di stile

Sul foglio di stile le caratteristiche degli elementi che riguardano tutto il **body**, i **titoli** o i **paragrafi** vengono preceduti dal nome dell'elemento con nomi e valori delle proprietà collocati all'interno di parentesi graffe.

Ad esempio a fianco vediamo come viene modificato il titolo h1 dandogli delle caratteristiche diverse da quelle di default.
Osservare con attenzione la sintassi con cui vengono scritti nomi e valori delle proprietà.

```
h1 {
    font-size: 1.4em;
    letter-spacing: 0.1em;
    margin: 5px;
}
```

Un gruppo di nomi e valori che riguarda un **div** viene chiamato ***identificatore*** mentre se nomi e valori riguardano uno **span** vanno a formare una ***classe***.

CSS esterni e interni

E' **esterno** un foglio di stile definito in un file separato dal documento. Si tratta di semplici documenti di testo editabili anche con il Blocco Note ai quali si assegna l'estensione **.css**.
Un foglio di stile si dice invece **interno** quando il suo codice è compreso in quello della pagina HTML. In questo caso tramite il tag **<style>,** posto all'interno della sezione **<head>**, viene creata un'area che comprende gli ordini di stile utilizzati nel documento. E' quello che fa automaticamente Dreamweaver quando cerchiamo di cambiare il colore ad una parte di testo tramite il suo pannello di lavoro.
Noi eviteremo di utilizzare i fogli interni cercando di concentrare tutte le nostre annotazioni di stile su un foglio esterno.
Conosciamo già gli ordini da utilizzare nei fogli di stile visto che sono gli stessi utilizzati sino ad ora per definire le proprietà degli elementi all'interno della pagina HTML. C'è invece qualche cambiamento nelle regole di sintassi.
Inoltre è importante sapere che alcuni ordini, come ad esempio body background se utilizzati in un foglio di stile presentano molte più possibilità rispetto al loro utilizzo diretto nella pagina .html.

Il foglio 'stilebase.css'

Per poter gestire le nostre pagine in modo semplice abbiamo creato il foglio di stile *'stilebase.css'* che cerca di raggruppare quanto può essere per creare diversi tipi di pagine a seconda delle diverse esigenze.

A fianco possiamo vedere la sua parte iniziale che parte con le proprietà da assegnare a body, cioè a tutto ciò che è collocato nel 'corpo' del documento.

stilebase.css

Codice | Dividi | Progettazione | Titolo:

```
body {
    font-family: Verdana, Arial, Helvetica, sans-serif;/*caratt
    font-size: 90%;/*caratteri - riduzione rispetto alle dimens
    color: #000033;/*colore di testo = nero*/
    background-color: #FFFFFF;/*colore di sfondo = bianco*/
}
#testata {
    /* ospita l'intestazione della pagina*/
    padding:5px;
    height: 30px; /* l'altezza va assegnata altrimenti il div i
    Il valore dato è un valore minimo in quanto il div si adegu
    background-color: #EDF9FE;/*colore di sfondo = azzurrino*/
    text-align: center; /*allinea il testo al centro*/
    border-width: 1px 3px 3px 1px; /* alto-top, destra-right, b
    border-style: solid;
    border-color: black;
}
```

I fogli di stile vengono chiamati anche fogli a cascata perché tutto ciò che viene stabilito per un contenitore viene 'ereditato' anche da ciò che è collocato all'interno, salvo ordini differenti.
Nel body dunque stabiliamo il tipo di caratteri da utilizzare e ne riduciamo la dimensione del 10% rispetto alle dimensioni di default (quelle che abbiamo utilizzato sino ad ora). Stabiliamo anche che il colore di testo è il nero e che il colore di sfondo è il bianco anche se queste sono già caratteristiche di default.
Il segno **#** precede, le proprietà degli **identificatori** mentre il punto **.** precede quelle delle **classi**.
Seguono le proprietà della **testata** Il testo viene allineato al centro e, come per tutti i contenitori di testo, viene assegnato un padding. Vi è un altezza (minima) e viene assegnato un colore di sfondo azzurrino. I bordi destro e inferiore vengono disegnati più spessi per dare al div un effetto di tridimensionalità.
Per le due immagini laterali creiamo le due classi **imgsinistra** e **imgdestra** che le spostano sui bordi destro o sinistro con float.

```
#testata {
    /* ospita l'intestazione della pagina*/
    padding:5px;
    height: 30px; /* l'altezza va assegnata altrimenti il div i
    Il valore dato è un valore minimo in quanto il div si adeg
    background-color: #EDF9FE;/*colore di sfondo = azzurrino*/
    text-align: center; /*allinea il testo al centro*/
    border-width: 1px 3px 3px 1px; /* alto-top, destra-right, b
    border-style: solid;
    border-color: black;
}
.imgsinistra {
    float: left;
    /*border: 1px solid black;*/
}
.imgdestra {
    float: right;
    /*border: 1px solid black;*/
}
#barra {
    /* spazio che segue l'intestazione e che ospita il percorso
    font-size: 80%;
```

collegare gli elementi (tabelle titoli, paragrafi, div e span) alle proprietà del foglio di stile

E' frequente che per i titoli e i paragrafi vengano utilizzate le caratteristiche di default previste dal browser. E' invece normale che (div, span e tabelle), presenti nella pagina HTML, vengano collegati alle proprietà stabilite per loro nel foglio di stile.

Questo avviene facendo seguire l'apertura del ***div*** da ***id=****(identificatore)* e dal nome che nel foglio di stile raggruppa le proprietà di quell'identificatore.

```
<div id="testata">
    <h2>Corso sull'HTML per la prima H</h2>
</div>
```

A fianco una ***classe*** creata nel foglio di stile per colorare di rosso una parte di testo e sotto lo ***span*** nella pagina HTML che si collega alla classe con la parola **class** seguita dal nome della classe.
Una classe può essere utilizzata tutte le volte che si presenta la stessa necessità (ad es. colorare di rosso una parte del testo)

```
.testorosso {
    color: #FF0000;
}
```

```
tanti  <span class="testorosso">il testo può essere colorato di rosso con la classe 'testorosso'</span> <
div 'destra', che va impostato per primo con posizione assoluta, e il div 'sinistra' -->
```

Dopo l'id della testata vi è l'identificatore della **barra**, cioè del div collocato dopo il titolo per ospitare il percorso che ha condotto alla pagina corrente e per consentire un rientro rapido alle pagine superiori.
La dimensione dei caratteri viene ridotta di un altro 20% rispetto al 10% stabilito nel body.
E' seguito dall'id degli **avvisi**. E' una barra particolare che viene messa solo per richiamare l'attenzione su avvisi urgenti.

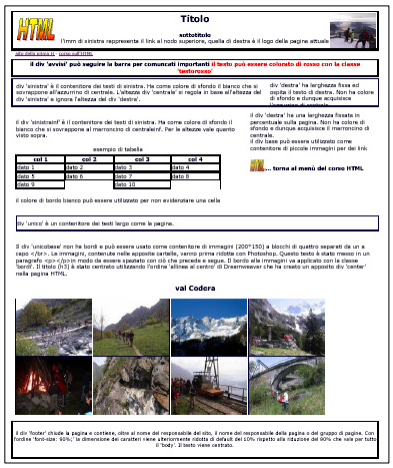
```
#barra {
    /* spazio che segue l'intestazione e che ospita il percors
    font-size: 80%;
    padding: 3px 10px 3px 10px;/* alto-top, destra-right, bass
}
#avvisi {
    /* spazio che segue la barra per ospitare comuncati di int
    font-weight: bold;
```

```
#centrale {
    /*centrale è un contenitore per i box destra e sinistra. No
    la colorazione dello sfondo di centrale invece che di 'dest
    border: 1px solid #000033;
    background-color: #EDF9FE;/*colore di sfondo = azzurrino*/
    position: relative;
    margin: 5px 0px 5px 0px;/*margini verso l'alto e verso il b
}
#destra {
    position: absolute; /*la posizione di questo div è in alto
    top: 0px;
    right: 0px;
    padding: 10px;
    width: 280px; /*la larghezza di questo div è fissa mentre c
    280px rappresenta la larghezza a disposizione del testo cor
    La larghezza reale del div 'destra' è di 300px (280 + 10 di
}
#sinistra {
    background-color: #FFFFFF;
```

Gli identificatori successivi organizzano le proprietà dei div 'centrale', 'sinistra' e 'destra' con le modalità già viste precedentemente in "Organizzare una pagina HTML: il modello" con l'esempio **div6**.
Il div **centrale** con position: relative e di sfondo azzurrino fa da contenitore ai div **destra** (position:absolute) e **sinistra** il cui margine destro è dimensionato per fare spazio al div destra e il cui colore bianco si sovrappone all'azzurrino di centrale.

I div **centraleinf**, **destrainf** e **sinistrainf** funzionano allo stesso modo di quelli già visti ma la larghezza del div di destra non è fissa ma è in percentuale sul totale della pagina. Il colore di sfondo è il marroncino.

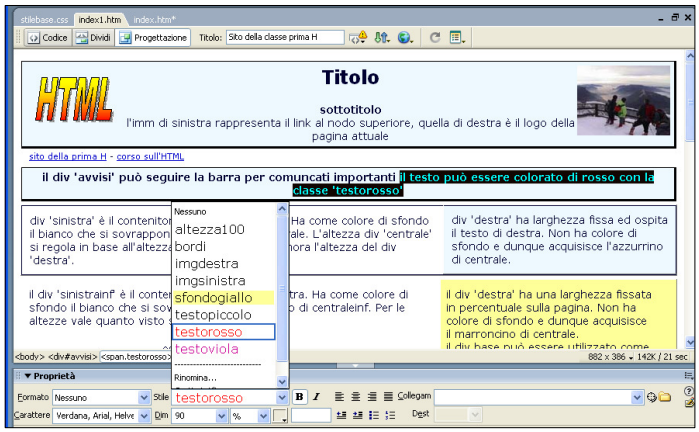
Segue poi l'identificatore di un div, chiamato '**base**', senza alcun attributo, '**unicobase**' da utilizzare quando si vogliono collocare testi e immagini a pagina intera. Non ha bordi.
Vi è poi **unico** che serve per testi e immagini a pagina intera e ha bordi.



La pagina **index1**, corredata da molti commenti, è organizzata utilizzando il foglio *stilebase.css* e rappresenta un modello per le pagine del nostro sito.

E' possibile evitare la fatica di scrivere gli ordini nella pagina di codice. Utilizzando il pannello di controllo di Dreaweaver:
- si evidenzia la parola o la frase a cui si vuole applicare una classe
- si clikka sul segno  collocato di fianco alla finestra della proprietà Stile
- compare una finestra in cui compaiono i nomi di tutte le classi che sono state previste nel foglio di stile collegato alla pagina
- si seleziona con un doppio click il nome dello stile che si desidera applicare
- nella finestra del codice Dreamweaver chiude il testo evidenziato all'interno di un tag span e lo collega alla classe selezionata

Se nel foglio di stile non esiste una classe che soddisfi le nostre esigenze potremo crearla.

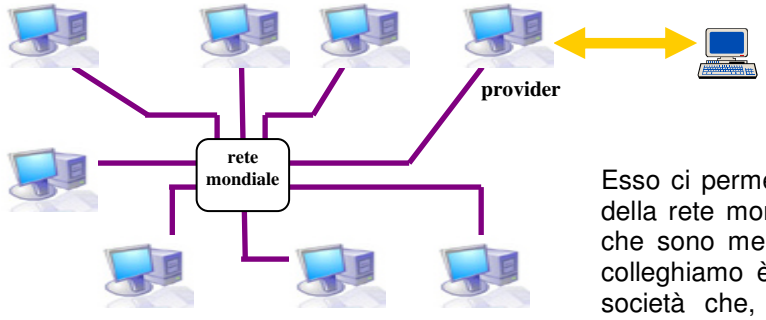


Creare modelli di pagine *i fogli di stile*

Quando si organizza un sito è bene pensare all'organizzazione della pagina che si vuole adottare e che, in linea di massima, sarà estesa a tutto il sito. A questo progetto di pagina corrisponderà un foglio di stile che comprenderà le caratteristiche di base di titoli e paragrafi e il repertorio di **identificatori** e **classi** che si ritiene necessario per la gestione delle pagine.
La creazione di un foglio di stile non è mai un'operazione definitiva. Infatti di fronte a nuove esigenze sarà opportuno aggiungere nuovi elementi. E' però necessario che il sito mantenga la sua "personalità" ed è dunque importante non eccedere nella differenziazione di stili e colori. Per questo motivo è utile creare dei modelli di pagina. Dei contenitori vuoti ma già organizzati secondo lo stile del sito, pronti ad essere duplicati e riempiti.

La pubblicazione

Quando si realizza un sito non lo si fa per tenerlo sul proprio calcolatore. Il sito che noi progettiamo e realizziamo va pubblicato su Internet in modo da poter essere visto in ogni parte del mondo dai calcolatori che hanno accesso alla rete. Per realizzare questo è necessario trasferire quanto da noi realizzato su un calcolatore di un **provider**, cioè di una società pubblica o privata i cui calcolatori siano collegati alla rete mondiale.



Quando ci colleghiamo ad Internet per consultare qualche sito il nostro calcolatore si collega, grazie alla linea telefonica, ad un cavo in fibbra ottica o ad onde radio, ad un calcolatore che ci farà da gateway, cioè da "porta".
Esso ci permetterà di comunicare con gli altri calcolatori della rete mondiale e di poter consultare le informazioni che sono memorizzate in essi. Il calcolatore con cui ci colleghiamo è il calcolatore di un provider, cioè di una società che, ricevendo un compenso, svolge questo servizio.

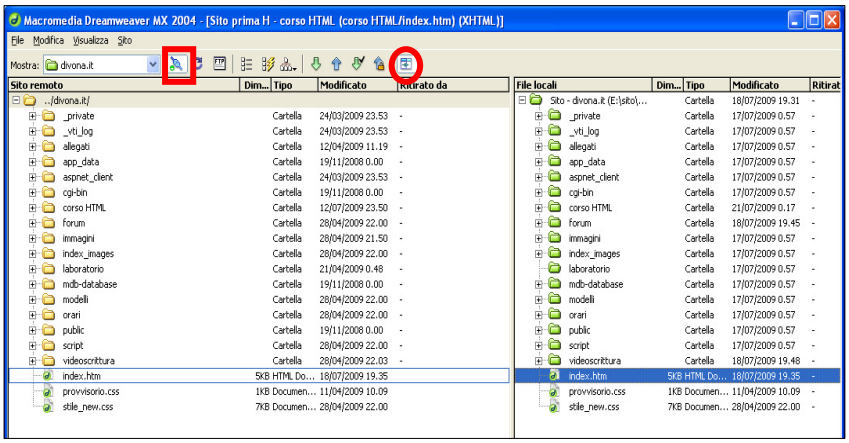
Vi sono alcuni provider che, sempre dietro un compenso, affittano spazi di memoria (chiamati ***domini***) all'interno dei loro computer. Dentro questi spazi di memoria è possibile memorizzare le pagine di un sito che, in tal modo, diventano accessibili a tutti i calcolatori della rete mondiale.
Dunque, per essere visibili in Internet, le pagine del nostro sito vanno copiate dal nostro computer in quello del nostro provider (si chiama Aruba e il suo megacomputer è collocato ad Arezzo - http://webfarm.aruba.it/).

provider

Con *Sito > Gestisci siti* è possibile organizzare, sul nostro calcolatore, un nuovo sito e, nella fase delle impostazioni, fornire l'username e la password che ci permetteranno di scaricare sul computer del provider le pagine da pubblicare. Sempre su Gestisci siti è possibile modificare le impostazioni di un sito già presente sul nostro computer in modo da attivare un collegamento con un provider.

In basso osserviamo il momento della pubblicazione. Sulla destra la situazione del sito sul nostro calcolatore (sito locale), sulla sinistra quella esistente sul computer del provider (sito remoto).

- per collegarsi al sito remoto
- per visualizzare anche il sito remoto



Le date ci permettono di capire quali pagine e cartelle sono state aggiornate e quali no.

Le norme

Le norme di legge equiparano solo in parte la pubblicazione di un sito a quella di un giornale. Anche qui vi è un proprietario (chi acquista il dominio dal provider) e un direttore (il webmaster). La responsabilità su ciò che viene pubblicato non è però del webmaster ma del proprietario del sito.

LAVORARE CON PAGINE DINAMICHE

Nei suoi primi anni di vita il web veniva essenzialmente utilizzato per condividere dati e documenti. L'uso delle pagine era dunque **statico**. Una volta aperte non dovevano fare nulla. Una situazione molto diversa da quella che si crea quando il calcolatore viene utilizzato per risolvere un problema. In questi casi il calcolatore deve interagire con l'operatore per chiedere dati, deve fare calcoli, deve comunicare risultati. Questo utilizzo del calcolatore viene chiamato **dinamico**.
La diffusione sempre più massiccia di Internet ha creato però la necessità che anche le pagine html potessero interagire con l'utente. Il linguaggio html fu arricchito con il tag **<form>** che si occupa di interagire con l'utente, raccogliendo informazioni e reagendo alle sue azioni.
Questo però non era sufficiente. Il tag form poteva essere utilizzato per raccogliere dati dall'utente ma era poi necessario che queste informazioni venissero elaborate e che il calcolatore, in base alle informazioni ricevute, fosse in grado di svolgere diversi tipi di procedimenti. Tutto questo non può essere fatto dall'html (linguaggio nato per gestire testi) ma da un linguaggio di programmazione.
Venne dunque deciso di affiancare all'html l'uso di linguaggi di programmazione che fossero in grado di interagire con esso. La pagina html viene dunque integrata con degli **script**. Cioè con delle istruzioni scritte in uno di questi linguaggi. Uno dei più conosciuti è Javascript, un linguaggio gratuito (come tutto ciò che serve per leggere Internet) che va installato sul proprio computer.
Se sul vostro computer avete installato Javascript, esso "lavorerà" insieme al vostro browser per poter leggere ed eseguire eventuali script contenuti nelle pagine da voi scaricate. Senza di esso molte pagine non sono leggibili.

Lato utente e lato server

Javascript, come abbiamo detto è un linguaggio di programmazione che si installa sul computer dell'utente e che lavora sulle pagine che lui scarica dal web. Si dice dunque che è un linguaggio ***"lato utente"*** *(**client-side**).*
Dopo qualche esempio di utilizzo di **Javascript** per far gestire al calcolatore vari tipi di attività (fare calcoli, proiettare presentazioni e filmati, ecc …) completeremo il nostro lavoro sull'HTML facendo qualche piccola esperienza sul **PHP**, un linguaggio ***"lato server"*** *(**server-side**).*

Che cosa vuol dire?
Mentre uno script Javascript è scaricato ed eseguito dal calcolatore dell'utente, uno script PHP viene eseguito sul calcolatore del provider che invia a quello dell'utente solo quello che è necessario, tradotto in HTML o in Javascript in modo che il suo calcolatore lo possa "capire" .
Per alcune attività, lavorare "lato utente" o "lato server" è indifferente. Per altre no. Se, ad esempio, vogliamo gestire un blog è necessario che tutti possano collegarsi ad un server, ma quello che ognuno scrive deve essere salvato sul server in modo da poter essere letto anche dagli altri.
Dunque serve un file di testo che sia ospitato su una cartella del sito aperta alla scrittura degli utenti (per ragioni di sicurezza le cartelle di un sito sono normalmente aperte solo alla lettura ma il webmaster se lo desidera può modificare la situazione) e un file PHP che raccolga quanto gli utenti scrivono sui loro file HTML e lo scriva nel file. Serve anche un file PHP che, se arriva la richiesta dal file HTML dell'utente, legga ciò che c'è nel file di testo e lo invii al file HTML dell'utente.

Le form *form e form1*

Iniziamo ad occuparci del tag **<form>** che, come abbiamo detto, si occupa di interagire con l'utente, raccogliendo informazioni e reagendo alle sue azioni. La form è un contenitore dentro il quale si possono inserire diversi oggetti utili ad acquisire dati.
Nell'esempio di *form* vediamo una form con vari oggetti ad essa associabili. Questo, semplificato, è il codice:

```
<form>
    Nome
    <input type="text" name="nome">
    Cognome
    <input type="text" name="cognome">
    Sesso
    <input type="radio" name="sesso" value="M"> M
    <input type="radio" name="sesso" value="F"> F
    Hobby
    <input type="checkbox" name="hobby" value="S"> Sport
    <input type="checkbox" name="hobby" value="L"> Lettura
    Paese di nascita
    <select name="paese">
    <option value="I">Italia</option>
    <option value="E">Estero</option>
    </select>
    Commento
    <textarea name="commento" rows="5" cols="30"></textarea>
    <input type="submit" name="invia" value="Invia i dati">
</form>
```

La form racchiude una serie di oggetti utilizzabili per la raccolta di dati dall'utente (oggetti di input):

- le caselle di testo (type="text") utilizzabli per digitare parole o numeri
- i pulsanti (type="radio") che permettono una scelta binaria
- le caselle di spunta (type="checkbox") permettono dimettere un segno di spunta accanto ad un nome. I menù a tendina (select) nei quali si può selezionare una tra più scelte
- le aree di testo che vanno dimensionate con numero di righe e di colonne (rows="5" cols="30" nell'esempio) e che permettono la scrittura di testi più lunghi
- i bottoni (type="button") oppure (type="submit") che, ad un evento (in genere è il click del mouse), permettono l'esecuzione di determinate azioni

E' anche possibile creare bottoni al di fuori dal tag form utilizzando input type="button". Negli esempi di *form1* possiamo vedere due esempi di bottoni abbinati agli eventi **onclick** e **onmouseover** che attivano la finestra dei messaggi **alert**.

JAVASCRIPT: UN LINGUAGGIO DI PROGRAMMAZIONE PER LAVORARE LATO-UTENTE

Gestire problemi

Come abbiamo detto le form servono a poco se non sono abbinate ad uno script che raccolga i dati e li elabori. Javascript è un normale linguaggio di programmazione e possiamo fare con esso quello che facciamo in Basic con qualche differenza di sintassi.

In *form2* è stato inserito, all'interno di un paragrafo, il testo del problema del fruttivendolo. E' seguito da una form che raccoglie i dati parametrici d'entrata e da un bottone da schiacciare alla fine dell'immissione dei dati per dare ordine al calcolatore di eseguire lo script di calcolo:

```
<input type="button" value="Guadagno del contadino (in euro)" onClick="calcola(this)">
```

Nell'ordine onClick ***calcola*** è il nome della **funzione*** che raggruppa le istruzioni Javascript da eseguire quando si verifica l'evento onClick. Il nome della funzione viene stabilito dal programmatore e deve corrispondere a quanto scritto nello script. La parola (**this**) è obbligatoria in situazioni in cui viene stabilito un collegamento tra form e script e quest'ultimo, oltre che ricevere dati dalla form, dovrà poi scrivere in alcuni oggetti i risultati del calcolo. Tutte le istruzioni che fanno parte della funzione devono essere comprese tra parentesi graffe { }.

* con il termine funzione viene indicato un gruppo di istruzioni (procedura) riunite sotto un unico nome. Il fatto che le istruzioni siano sotto un unico nome permette che la funzione venga ***chiamata*** (in altre parti del programma è possibile ordinare al calcolatore di eseguirla). Questa possibilità è presente in tutti i linguaggi di programmazione (anche in Basic) anche se vengono usati nomi e istruzioni diverse.

In *form2* lo script è stato collocato nella parte head della pagina (come è bene fare). Esso può anche essere collocato in body o in una apposita pagina **.js** (come vedremo negli esempi successivi).
Esso inizia con il tag **<script>** e termina con **</script>**.
Vediamo subito due regole importanti:

- la prima è che il codice javascript deve essere contenuto tra i tag <script> e </script>
- la seconda è che ogni istruzione finisce con il punto e virgola (;)

Variabili

Il linguaggio javascript è case sensitive, ossia distingue le lettere maiuscole da quelle minuscole. Occorre dunque fare molta attenzione alle sviste, ad esempio le variabili citta, CITTA e CittA sono considerate come distinte. Inoltre in javascript non deve essere dichiarata la differenza tra numerico e alfanumerico.
Se scriviamo nome="Paolo", la variabile nome sarà alfanumerica. Se num=127, la variabile num sarà numerica. Può accadere che qualche volta il computer collochi in alfanumerico dati che noi vogliamo utilizzare per fare calcoli. In questo caso sarà utile l'istruzione **number** per portare da alfanumerico a numerico (ad es: a=number(b)) e **string** per portare da numerico ad alfanumerico (ad es: a=string(b)). Molte altre regole sono uguali o simili a quelle già viste in Basic.

Input/output

Come abbiamo visto nell'esempio in *form2* lo script può ricevere i dati dalle caselle di testo di una form e può usare le stesse caselle di testo per scrivere i risultati dei calcoli.
Comunque in Javascript l'istruzione *input* è sostituita dall'istruzione ***prompt***, con le regole di sintassi che potete vedere nell'esempio nella pagina di esempio *java*. Questa istruzione attiva una finestra di richiesta del dato cercato che poi viene memorizzato nella variabile indicata all'inizio. La procedura di calcolo è identica a quella che conosciamo mentre la scrittura del risultato non può utilizzare l'ordine *print* (che non esiste). Esso è sostituito, con lo stesso risultato, da ***document.write***. Anche quest'ordine va utilizzato secondo le regole che vedremo di seguito.
Torniamo comunque all'esempio visto in *form2*.

```
<script>
    // f è il nome del canale, è cioè una variabile che indica il rapporto di comunicazione esistente tra la // form e questo script
    function calcola(f){
    // assegnazioni delle variabili - ordino al calcolatore di collocare nella variabile pv il valore presente // nella casella di testo di nome pv della form (f è il nome del canale)
    pv=f.form.pv.value;
    ka=f.form.ka.value;
    ks=f.form.ks.value;
    pa=f.form.pa.value;
        // calcoli
        kv=ka-ks; //chilivenduti = chiliacquistati - chiliscartati
        rv=pv*kv; //ricavovendita = prezzovendita * chilivenduti
        sa=pa*ka; //spesacquisto = prezzoacquisto * chiliacquistati
        guadagno=rv-sa; //guadagno = ricavovendita - spesacquisto
    // il risultato dell'elaborazione viene scritto come valore nell'oggetto chiamato guadagno che fa parte // della form (f è il nome del canale)
    f.form.guadagno.value=guadagno;
    }
```

```
</script>
```

E' bene chiarire che ***document.write*** non si limita a sostituire l'ordine Basic ***print***. Visto che esso colloca sulla pagina html ciò che è collocato tra le parentesi che lo seguono. Possono verificarsi tre situazioni:

document.write("Il guadagno del fruttivendolo è di ");
viene scritto il testo tra virgolette

document.write(guadagno);
viene scritto il valore di una variabile il cui nome è scritto tra parentesi

document.write('<embed src="allegati/power.mp3" autostart="true" hidden="true"/>');
viene scritto un tag html che dunque verrà eseguito dal browser. In questo modo è possibile far gestire da uno script dei tag html (l'esempio è in java > presentazioni > avvio).

Altri esempi con Javascript *java, java1, java2, java3 e java4*

Vi sono altre possibilità. In java possiamo vedere il problema del fruttivendolo con Javascript che gestisce anche la fase di ingresso dei dati grazie alle finestre gestite dall'istruzione **prompt**. Questa istruzione permette di assegnare direttamente le variabili e dunque, a differenza di quanto visto nell'esempio con form2, non è stato necessario assegnare le variabili. Dunque non esiste alcuna form. Nella pagina è stato collocato un bottone che serve ad attivare il calcolo con Javascript (raggruppato nella funzione ***esegui***). In questo esempio l'uscita di dati è gestita dall'ordine **document.write** che scrive all'interno di una pagina .html. Nell'esempio in *java1* l'uscita è gestita grazie ad una finestra di **alert**.

Come abbiamo già visto per i fogli di stile, è buona abitudine non "sporcare" il file html con istruzioni estranee. E' dunque consigliato collocare le istruzioni del programma Javascript in un altro file e collocare nella pagina html solo il collegamento ad esso, come possiamo vedere in *java2* .

```
</p>
<script src="java2.js">
</script>
<p>.</p>
```

Il file ***java2.js*** può essere scritto come normale file di testo utilizzando Blocco Note.
L'importante è che abbia l'estensione ***.js*** che permette al calcolatore di riconoscerlo ed eseguirlo.

java2.js*

Codice | Dividi | Progettazione | Titolo:

```
1  // - - -  inizio file java2.js
2      prezzovendita=prompt("scrivi il prezzo di vendita delle arance in euro ","2");
3      chiliacquistati=prompt("scrivi il numero dei chili acquistati","100");
4      chiliscartati=prompt("scrivi il numero dei chili scartati","10");
5      prezzoacquisto=prompt("scrivi il prezzo di acquisto delle arance in euro ","1");
6      chilivenduti = chiliacquistati - chiliscartati
7      ricavovendita = prezzovendita * chilivenduti
8      spesacquisto = prezzoacquisto * chiliacquistati
9      guadagno = ricavovendita - spesacquisto
10     window.alert("il guadagno del fruttivendolo è di "+guadagno+" euro");
11 // - - -  fine file java2.is
```

L'esecuzione di uno script collocato in una funzione può anche essere ordinato insieme all'apertura del corpo della pagina. In *java3* l'istruzione **<body onLoad="esegui();">** ordina l'esecuzione della funzione ***esegui*** collocata in head.

In *java4* vi è un altro esempio con Javascript. In questo caso vi sono due script, collocati nel body della pagina, che vengono eseguiti all'apertura in sequenza. Vengono chiesti per due volte tre numeri e il calcolatore, ne calcola la media utilizzando la funzione ***media*** collocata in head (notare come vengono passati alla funzione i valori dei dati).

Pulsanti e finestre

aprire finestre

Nell'esempio a fianco il pulsante provoca l'esecuzione della funzione *apri* che contiene l'ordine ***window.open***.

Esso viene utilizzato per aprire una pagina html aggiuntiva dandogli delle dimensioni ridotte.

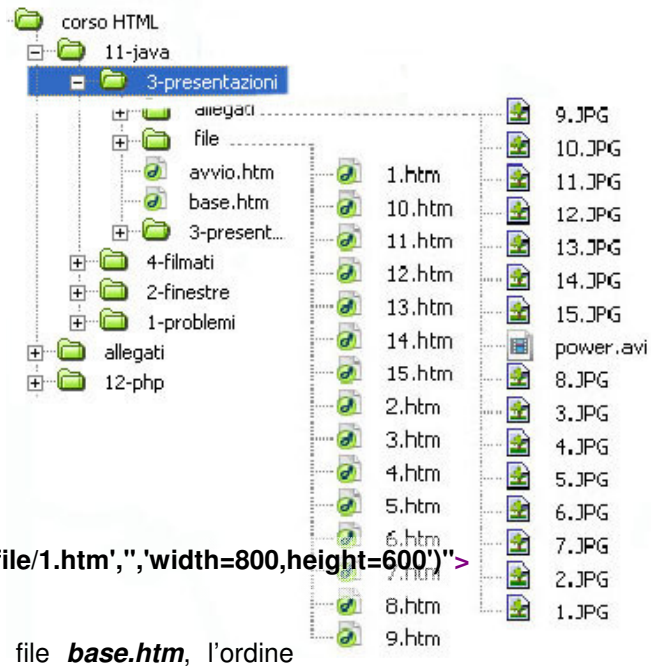
```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<html>
<head>  <!--  -->
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
    <title> Quintino di Vona - corso sull'HTML </title>
    <script>
        function apri(){window.open('javafin.htm','','width=420,height=334')}
    </script>
</head>
<body>
        <h3>clikkare sul pulsante</h3>
        <p>
            <input name="bottone" type='button' onclick=apri() value=" i tecnici  " />
        </p>
    <p><strong>spiegazione:</strong><br>
        Il bottone attiva la funzione <em><strong>apri</strong></em> collocata in head:.
        <em><strong>function apri(){finestra=window.open('javafin.htm','','width=420,height=334')}<
    <p>Questa apre la pagina <em><strong>javalfin.htm,</strong></em> dandogli le dimensioni,
    mantenendo comunque attiva la finestra di partenza. </p>
    <p>--- <a href="../../index.htm">corso HTML</a></p>
</body>
</html>
```

Fare presentazioni in javascript) *fare presentazioni*

Javascript dà anche la possibilità di realizzare delle presentazioni in html. Visto che in questa attività sono coinvolti numerosi file, è bene che essa venga gestita con ordine. Come prima cosa vi sarà una pagina in cui è collocata una parola calda che consente di avviarla e che abbiamo chiamato ***avvio.htm***. Vi sarà anche un'altra pagina (chiamata ***base.htm***) che sarà la **pagina principale**, attiva mentre su una **pagina secondaria** scorreranno le immagini. Questa è necessaria per ospitare la colonna sonora che dovrà rimanere attiva durante tutta la presentazione.

Poi vi sono tante pagine quante sono le pagine (o diapositive) che verranno proiettate. Esse vanno messe in un apposita cartella (file) ed è bene vengano numerate in ordine di proiezione. Se la presentazione richiede degli allegati (immagini, suoni, …), questi vanno messe in un'apposita cartella (allegati) e numerate in modo da abbinarle alle pagine in cui verranno collocate. In questa cartella andrà collocata anche l'eventuale colonna sonora da abbinare alla presentazione.

A fianco possiamo vedere l'organizzazione dei file utilizzati nel nostro esempio.



Nella pagina avviohtm una variante dell'ordine **a href** permette con un solo click di aprire sia la finestra principale (file ***base.htm***) che la finestra secondaria dentro al quale scorreranno le diapositive.

```
<a href="base.htm" onClick="window.open('file/1.htm','','width=800,height=600')">
    <h3>avvia la presentazione</h3>
</a>
```

Mentre viene aperta la finestra principale con il file ***base.htm***, l'ordine **window.open** apre la finestra secondaria caricandovi la pagina ***1.htm*** e dandogli le dimensioni previste.

La finestra principale ha come unico compito quello di ospitare la colonna sonora ma può anche contenere del testo a commento della presentazione. Dovrà in ogni caso contenere il link che permetterà all'utente di tornare alla finestra di partenza.

Vediamo ora come è strutturato il file 1.htm:

```
<head>
   <meta http-equiv=page-enter content=blendTrans(Duration=5.0)>
   <script>
      window.setTimeout ("chiama()", 5000);
      function chiama() { location.href = "2.htm"; }
   </script>
</head>
<body background="../allegati/1.JPG">
</body>
```

Esaminiamolo ora attentamente:

- vi è un **meta tag** che prevede un'apertura della pagina (**page-enter)** con una dissolvenza sfumata (blendTrans) della durata di 5 secondi (Duration=5.0)
- lo script contiene due ordini distinti:
  - **window.setTimeout** serve per gestire le temporizzazioni. Quando incontra questo ordine, il browser attende un certo numero di millesimi di secondo prima di eseguire una determinata funzione (l'uso di funzioni è obbligatorio). Nel nostro esempio il calcolatore attende 5 secondi prima di eseguire la funzione ***chiama***.
  - l'ordine **location.href** permette, di sostituire il contenuto della finestra secondaria aperta. In questo caso il calcolatore sostituirà il contenuto della pagina ***1.htm*** con quello della pagina ***2.htm***. Questo ordine è collocato dentro una funzione per poterlo utilizzare con window.setTimeout.
- l'immagine viene collocata come sfondo di pagina (body background). Essendo la finestra secondaria di 800px per 600 px, tutte le immagini da caricare sono state ridimensionate con Photoshop su queste dimensioni.

Tutti gli altri file sono identici; cambia solo il nome dei file e delle immagini da caricare che saranno quelli successivi. L'unico file diverso sarà quello di chiusura visto che non dovrà caricare un nuovo file ma chiudere se stesso. Nel nostro esempio l'ultimo file è il 15.htm. Ecco il suo script:

```
<script>
   window.setTimeout ("chiama()", 5000);
   function chiama() { f=window.close('15.htm') }
</script>
```

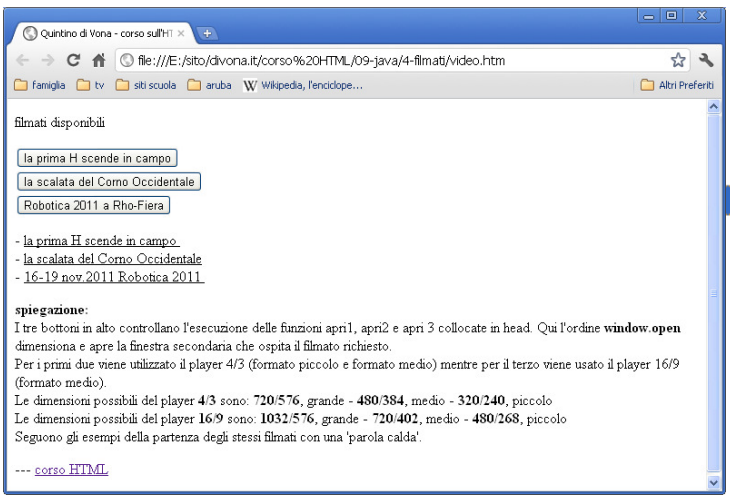
Infatti la finestra f ora dovrà essere chiusa.
Un'ultima considerazione riguarda i tempi di proiezione. Bisogna stimare il tempo necessario perché l'utente possa esaminare una singola pagina senza però annoiarsi. Inoltre la colonna sonora, se esiste, va calcolata in modo che termini alla fine della presentazione. Per manipolare o tagliare la colonna sonora potrà essere utile ***Premiere*****,** mentre per portare i file musicali da *.avi* in *.mp3* potrà servire ***Sound Forge***.

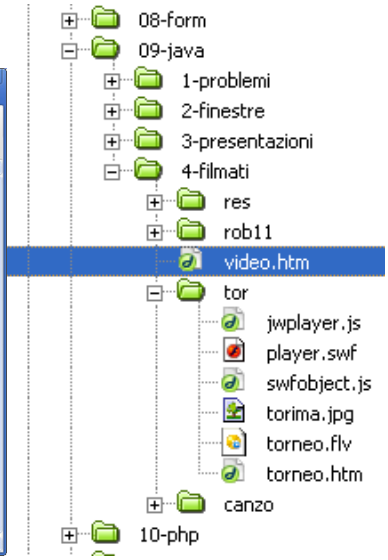
Quando si apre una **finestra secondaria** in genere gli si danno delle dimensioni fisse per fargli occupare solo una parte dello schermo. Se la finestra contiene del testo può essere utile munirla di barre di scorrimento. Il seguente ordine apre la finestra delle circolari nel sito della scuola, munendola di **sbarre di scorrimento**:

```
<script> function apri1() {finestra=window.open('circolari-qdv.htm','','width=600,height=420,scrollbars=yes')}</script>
```

Pulsanti e finestre per gestire filmati

gestire filmati





I sistemi appena visti (un pulsante o una parola calda per aprire una pagina secondaria html di dimensioni ridotte) sono utili anche per gestire dei filmati da collocare nelle pagine del nostro sito. La pagina ***video.htm*** contiene tre filmati che possono essere attivati da *pulsanti* o da *parole calde*. Tutto il necessario per la loro gestione è contenuta in una cartella il cui nome richiama ciò che è contenuto. Noi esaminiamo, ad esempio, il contenuto della cartella **tor** (filmato che ricorda la partecipazione della prima H al torneo del 2009).
Vediamo che cosa contiene:

il filmato ***torneo.flv***

i filmati vengono scaricati dalla telecamera sul computer in vari formati (avi, mpg, ecc ...). Per essere pubblicati su un sito vanno "alleggeriti". Noi li convertiamo in formato **.flv** utilizzando *Macromedia Flash 8 Video Encoder*.
I filmati sono caratterizzati anche dalle dimensioni e dai rapporti tra le dimensioni.

il rapporto tra le dimensioni

è il rapporto tra larghezza e altezza. Il rapporto 4/3 è il rapporto utilizzato nei monitor tradizionali. Oggi è spesso sostituito dal "panoramico" rapporto 16/9.

un'immagine a 4/3

un'immagine a 16/9

le dimensioni - *larghezza/altezza in pixel*

le dimensioni possibili del player 4/3 sono: 720/576 = grande; 480/384 = medio; 320/240 = piccolo
le dimensioni possibili del player 16/9 sono: 1032/576 = grande; 720/402 = medio; 480/268 = piccolo
Il filmato *torneo.flv* è in 4/3 e è di 480/268.

il player

per vedere un filmato è necessario un player, cioè un contenitore che lo gestisca. Il player deve essere in grado di leggere il flusso di immagini provenienti dal file e di proiettarle in un riquadro organizzato sul monitor. Il player, in genere, è anche in grado di controllare il filmato fermando le immagini o spostando la riproduzione in un punto diverso. Il player più diffuso è Windows Media Player che però non può gestire i file *.flv*. Noi dunque utilizziamo ***JW Player*** scaricabile da Internet.
Vediamo ora i suoi componenti:

player.swf - si occupa della gestione dell'oggetto grafico che ospita il filmato mentre
swfobject.js - ospita il software di gestione dell'oggetto grafico
jwplayer.js - si occupa invece di leggere il filmato *.flv* ed adattarlo al player che deve ospitarlo, questo anche in base agli ordini che provengono dalla pagina secondaria *.htm*
torneo.htm - è la finestra secondaria che viene aperta e dimensionata per ospitare il player. E' l'unico file interno alla cartella del fimato su cui dovremo intervenire (gli altri li dobbiamo solo copiare). Vediamo come:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <title>Player in 4/3 piccolo</title>
<!-- finestra secondaria piccola per player in 4/3. Va aperta cor
</head>
<body>
    <div id="f" >...</div>
    <!--  le dimensioni possibi
    <script type="text/javascript" src="jwplayer.js"></script>

    <script type="text/javascript">
        jwplayer("f").setup({
            flashplayer: "player.swf",
            file: "torneo.flv",
            width: 320,
            height: 240,
            image: "torima.jpg"
        });
    </script>
</body>
</html>
```

il titolo della pagina andrà cambiato adattandolo al contenuto

è obbligatorio dimensionare un div destinato ad ospitare il player e dargli un nome (noi usiamo il nome generico di **f**)

con questo script viene caricato *jwplayer.js* il software necessario per la lettura del filmato

inizio del secondo script

viene avviato *jwplayer.js* indicandogli di utilizzare il div **f**

player.swf	nome del contenitore grafico
torneo.flv	nome del file da leggere
320	larghezza (player 4/3 piccolo)
240	altezza (player 4/3 piccolo)

con image (facoltativo) viene indicato il nome di un'immagine da visualizzare nel player in attesa dell'avvio. Se manca, apparirà un rettangolo nero

fine del secondo script

Anche il nome della pagina secondaria va cambiato in base al suo utilizzo.

l'apertura della pagina secondaria

la finestra secondaria che contiene il player verrà aperta utilizzando un *bottone* o una *parola calda* a cui viene abbinato l'ordine window.open. Rispetto alle dimensioni del player, le dimensioni della finestra vanno aumentate di ***16 pixel in larghezza*** e ***16 pixel in altezza*** che rappresentano i bordi della finestra di windows. Per aprire la finestra *torneo.htm* dentro la cartella *tor* è stato utilizzato il seguente ordine:

```
window.open('tor/torneo.htm','','width=336,height=256' )
```

Per aprire il video di robotica 2011 (player 16/9 di 720 per 402) è stato usato:

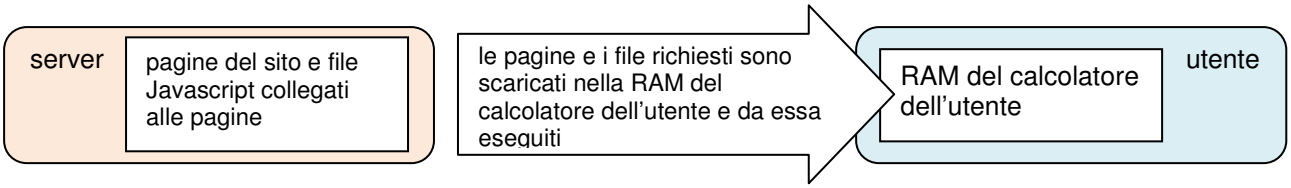
```
window.open('rob11/rob11.htm','','width=736,height=418')
```

La finestra aperta viene sempre collocata in alto a sinistra. Come sappiamo, potremo utilizzare gli ordini **left** e **top** per dargli una posizione diversa.

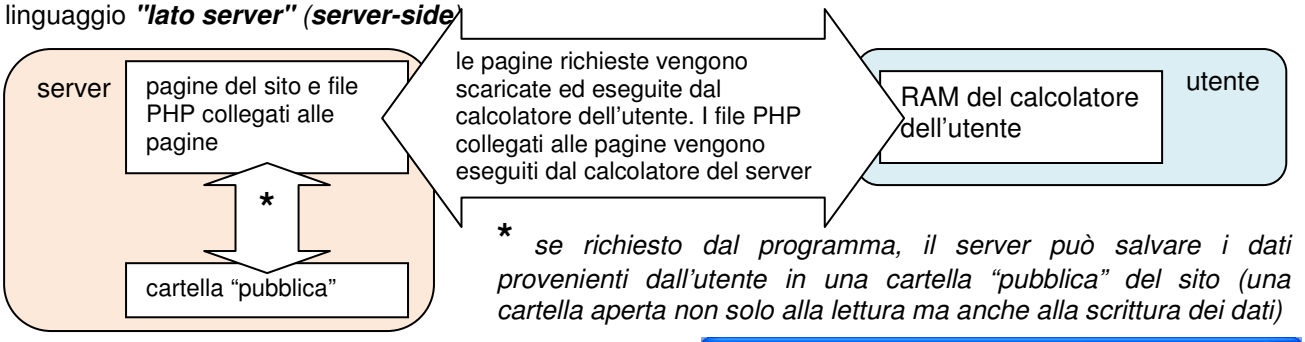
PHP: UN LINGUAGGIO DI PROGRAMMAZIONE PER LAVORARE LATO-SERVER

Completiamo il nostro lavoro sull'HTML facendo qualche piccola esperienza sul **PHP**, un linguaggio ***"lato server"*** *(linguaggio **server-side**).*
Abbiamo già spiegato che, contrariamente a Javascript, uno script PHP viene eseguito sul calcolatore del provider che invia al quello dell'utente solo quello che è necessario (tradotto in HTML o in Javascript in modo che il suo calcolatore lo possa "capire") .

linguaggio ***"lato utente"*** *(**client-side**)*

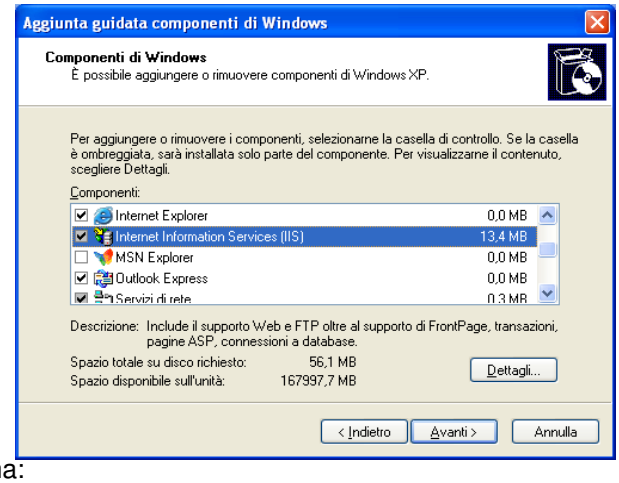


linguaggio ***"lato server"*** *(**server-side**)*



* *se richiesto dal programma, il server può salvare i dati provenienti dall'utente in una cartella "pubblica" del sito (una cartella aperta non solo alla lettura ma anche alla scrittura dei dati)*

Vediamo ora alcune regole del linguaggio PHP. In molti aspetti esso è simile a Javascript però esistono alcune differenze. Esaminiamo ora ciò che può essere utile al nostro lavoro.



L'attivazione

Come prima cosa bisognerà verificare che nel nostro calcolatore sia attivo il software che permette l'attività server-side. In Internet Explorer è l'Internet Information Service (IIS) che va attivato tramite il *Pannello di controllo > Installazione applicazioni > Installazione componenti di Windows*.
Attivato l'IIS sarà necessario scaricare il sw che permette di utilizzare ed interpretare le istruzioni di PHP.
Lo facciamo dal sito ufficiale scaricando la versione italiana:

http://php.softonic.it/download

Ora il nostro calcolatore è pronto per scrivere codice PHP.

Lo script

Uno script PHP inizia con il segno **<?php** e finisce on il segno **?>**. Come per Javascript, anche in questo caso possiamo creare una pagina interamente di codice PHP oppure può essere inserito in una pagina insieme al codice .html. In questo caso però tutta la pagina andrà salvata con un suffisso .php.

La stampa

L’ordine **echo** sostituisce a tutti gli effetti document.write e dunque può anche rinchiudere un tag html. Ad esempio: **echo "<h3>forum di prova</h3>";** provocherà la scrittura nella pagina corrente della frase “forum di prova” come titolo 3.

Variabili, alcune regole

Una variabile inizia sempre con l segno **$** (dollaro). Non deve essere dichiarata la differenza tra numerico e alfanumerico. Il calcolatore cercherà di capire, a seconda del contesto, di quale tipo di dati si tratta.
Le stringhe possono essere racchiuse da apici o da doppi apici. Se una stringa inizia con un tipo di segno deve concludersi con lo stesso segno. Questo significa che se scriviamo:

```
$f = 'Paolo disse: "Alzatevi!" e tutti si alzarono';
echo $f;
```

Il calcolatore scriverà la frase: *Paolo disse: "Alzatevi!" e tutti si alzarono.* Infatti i doppi apici non annullano l’effetto di quelli singoli e dunque vengono gestiti come semplici caratteri.
Per risolvere lo stesso problema è anche possibile utilizzare il cosiddetto 'carattere di escape', cioè la barra rovesciata (backslash: \). Vediamo l’esempio di prima:

```
echo “Paolo disse: \"Alzatevi!\" e tutti si alzarono”;
```

Il calcolatore scriverà *Paolo disse: "Alzatevi!" e tutti si alzarono.* Il segno \ induce il calcolatore a considerare i doppi apici un semplice carattere e non l’ordine di chiusura del testo.
Se il calcolatore trova il segno $ all’interno di una stringa capirà che è stata inserita una variabile. Dunque:

```
$nome=”Paolo”;
echo "$nome è appena arrivato”;
```

Il calcolatore scriverà la frase: *Paolo è appena arrivato.* Per concatenare dati alfanumerici non si usa il segno + (più), come in Basic, ma il segno . (punto). Infatti se:

```
$a=”ciao”;
$b=”Paolo”;
$c=$a.$b."<br>";
echo $c;
```

Il calcolatore scriverà *ciaoPaolo* andando subito dopo a capo.

Input/output

Ma, come abbiamo visto, caratteristica di PHP è quella di poter salvare dei dati sul server in file .txt.
Lo potremo fare utilizzando ordini che abbiamo già visto in Basic, con qualche differenza.
L’ordine **fopen** apre un file. Questo può essere aperto per scrivere (**w**) o per leggere (**r**).
Ad esempio con l’ordine:

```
$fh = fopen(“dati.txt”, 'r');
```

Viene aperto, per leggerne i dati, il file dat.txt. La variabile $fh rappresenta il numero del canale. Come sempre il nome del file (e l’eventuale percorso) può essere sostituito da una variabile che lo contiene.
L’ordine **fread** legge il contenuto del file e lo colloca in una variabile. Attenzione, con questo ordine viene letto tutto il contenuto del file (non esistono segni di fine campo). Inoltre è necessario indicare quanti sono i caratteri da leggere. Questo problema viene risolto con **filesize(nome del file)** che restituisce il numero dei caratteri di un file. Con:

```
$a=fread($fh,filesize(“dati.txt”));
```

Viene letto il contenuto del file ‘dati.txt’ e collocato dentro la variabile $a. Tra parentesi viene riportato il numero del canale ***$fh*** e indicato il numero dei caratteri da leggere ***filesize(“dati.txt”)***. E’ importante ricordare che il file di testo da leggere non deve essere vuoto. In questo caso il calcolatore darà un messaggio di errore.

L'ordine **fwrite** scrive un dato in un file. Mancando i segni di fine campo, con fwrite dovremo memorizzare tutto il contenuto del file. Con:

```
fwrite($fh,$b);
```

viene scritto nel file tutto il contenuto della variabile ***$b*** mentre ***$fh*** è il numero del canale. Con:

```
fclose($fh);
```

viene chiuso il file.

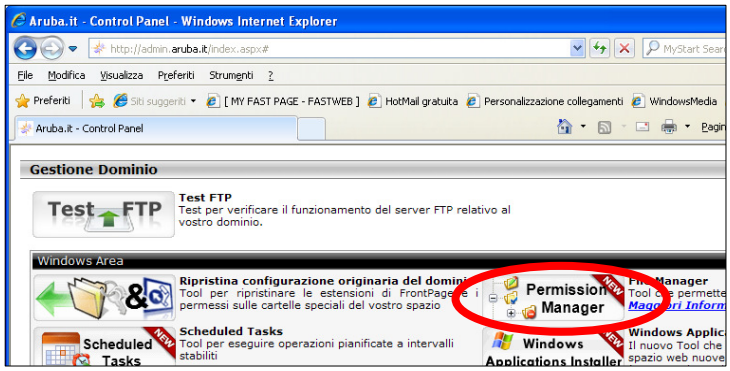
Realizzare un forum con PHP *leggiscrivi.php*

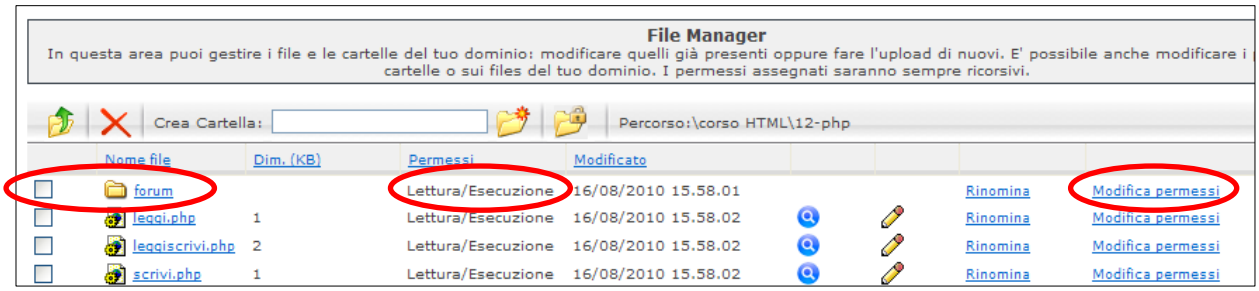
Come esempio di programmazione *server side* abbiamo creato la pagina **leggiscrivi.php** nella cartella **12-php**. La presenza di codice php, insieme a quello html, rende necessario il suffisso .php.
Volendo creare un forum su cui tutti gli utenti possano intervenire, sarà necessario avere, sul server, un file di testo su cui possano essere salvati gli interventi man mano che vengono effettuati. File da cui verranno letti per essere inviati all'utente quando si collega. Creiamo dunque una cartella chiamata ***forum*** dentro la quale mettere il file ***dati.txt*** ed eventuali altri file su cui si deve poter scrivere. Come abbiamo detto all'inizio, il provider non prevede si possa scrivere all'interno di cartelle e file di un sito ma ha messo a disposizione dell'amministratore del sito un pannello di controllo per modificare la situazione

All'indirizzo:

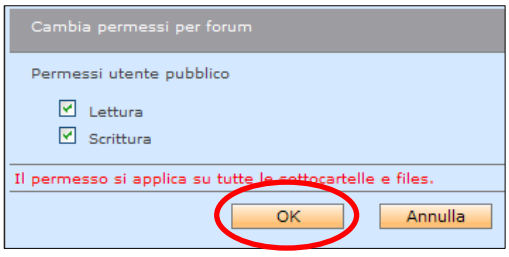
http://admin.aruba.it/login.aspx

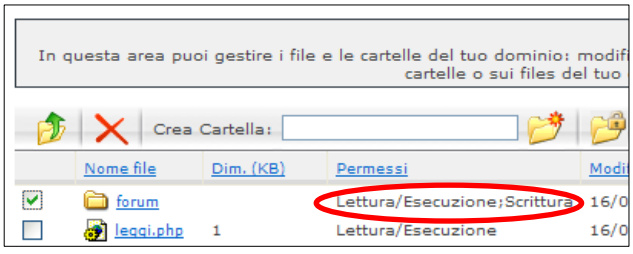
vi è la finestra di accesso al pannello di controllo. Dovremo scrivere il ***nome del dominio***, l'***username (login)*** e la ***password***. Entati nel pannello di controllo andremo a clikkare sul bottone ***Permission Manager***. Qui entreremo nel ***File Manager*** dove cercheremo la cartella **forum**.





Qui possiamo verificare che attualmente la cartella **forum** ha un permesso per la lettura e l'esecuzione. Clikkiamo dunque su ***Modifica permessi*** e ci apparirà la finestra ***Cambia permessi*** dove andremo a spuntare le voci ***Lettura*** e ***Scrittura***.
In un primo momento non succederà nulla. Tornando successivamente sul ***File Manager*** potremo verificare che la cartella **forum** ha l'autorizzazione anche alla scrittura. A questo punto sarà possibile crearvi dei file sui quali memorizzare dati.





La pagina leggiscrivi.php

Dopo aver creato il file dati.txt ricordandoci di non lasciarlo vuoto, passiamo a creare la pagina con cui gestire il forum. Vediamone le parti essenziali:

```
<?php
      $file= "forum/dati.txt"; // il nome del file da aprire viene memorizzato nella variabile $file
      $fh = fopen($file, 'r'); // apertura del file sul server
      $a=fread($fh,filesize($file)); //viene letto il contenuto del file e memorizzato nella variabile $a
      fclose($fh);
      $a=nl2br($a); // traduce in HTML gli invii a capo che trova nel testo presente in $a
      echo "<h3>forum di prova</h3>";
      echo $a; // scrive sul monitor dell'utente il contenuto di $a
?>       <!-- fine dello script di lettura dei dati con php e inizio del form in html -->
<div style="width:30%;  position:absolute; top:0; right:0px;">
      scrivi il tuo intervento
      <form action="scrivi.php" method="get">
            <textarea name='b' cols='30' rows='6' >
            </textarea>
            <br>
            scrivi il tuo nome e cognome
            <textarea name='c' cols='30' rows='1' >
            </textarea>
            <input type='submit' name='bInvia' value='Invia i dati'>
      </form>
      .
      <form action="cancella.php" method="get">
      <input type='submit' name='bInvia' value='Resetta i dati'>
      </form>
</div>
```

All'inizio la parte in codice PHP serve per leggere ciò che è presente nel file dati.txt e scriverlo sul montor dell'utente. Da notare l'ordine **nl2br** che traduce gli 'a capo' fatti dall'utente con Enter in
 validi nella pagina html e il tag h3 *'forum di prova'* scritto in php utilizzando **echo**.
Segue il form con le due aree di testo in cui l'utente può scrivere il suo intervento e il suo cognome e nome. Esso è collocato in un div messo sulla destra in posizione assoluta.
Dopo aver scritto il suo intervento e il suo nome e cognome, l'utente con un click sul bottone attiva il file ***scrivi.php***. Vediamolo:

```
<?php
$file= "forum/dati.txt";
$fh = fopen($file, 'r');
$a=fread($fh,filesize($file));
fclose($fh);
$b=$a."<br>---<br>".$b."<br>".$c; /* il testo letto da 'dati.txt' viene concatenato ai testi proveniente da 'leggiscrivi.php' (contenuto nelle variabili $b e $c) */
$fh = fopen($file, 'w'); //apertura del file
fwrite($fh,$b); // scrittura del testo
fclose($fh);
header("Location: leggiscrivi.php"); /* l'ordine header richiama la pagina leggiscrivi.php che così verrà riscritta e conterrà il testo aggiornato */
?>
```

Nella pagina ***leggiscrivi.php*** abbiamo collocato anche un bottone per resettare quanto contenuto nel file dati attivando il file ***cancella.php***. Questa possibilità viene normalmente prevista solo per l'amministratore del forum e dunque è bene che il bottone sia presente solo sul suo computer