

Una breve guida introduttiva  
all'HTML

# Guida HTML

Corso Pratico

Andrea Aralla

---

# Guida HTML

## Sommario

1	Risorse.....	6
2	INTRODUZIONE.....	8
2.1	Prerequisiti.....	8
2.2	Le pagine HTML e i TAG.....	8
3	Hello HTML! .....	9
3.1	Creare la prima pagina .....	9
3.2	Estensione .html .....	10
3.3	Apriamo il file HTML.....	11
3.4	Struttura generale di una pagina HTML.....	12
3.5	I tag <head> e <body>: la struttura minimale della pagina HTML .....	13
3.6	Sbirciare l'HTML nelle pagine Web.....	14
4	Elementi e tag in HTML .....	15
4.1	Tag della head .....	15
4.2	Tag del body.....	16
4.3	Come è fatto un tag .....	16
4.4	Elementi contenitori e autonomi, il DOM.....	16
4.5	Gli attributi .....	18
5	Titoli, paragrafi e testi.....	18
5.1	I paragrafi <p>.....	18
5.2	Andare a capo,   .....	19
5.3	Headings, i titoli h1, h2, h3, etc.....	20
5.3.1	Gerarchia degli headings.....	20
5.3.2	Gerarchia degli headings nelle home page .....	20
5.4	Grassetto e corsivi .....	21
5.4.1	Esempi di utilizzo dei tag u, em, b e i.....	22
5.5	Altri tag per enfatizzare il testo .....	23

5.5.1	Il tag <s> e <strike> - barrato .....	23
5.5.2	i tag <big> e <small> .....	23
5.5.3	I tag <sup> e <sub>.....	23
5.6	Aggiungere un commento in HTML.....	24
5.7	La dimensione e lo stile dei caratteri il tag <font>.....	24
5.8	I colori .....	25
5.8.1	Colore di sfondo a tinta unita in HTML .....	25
5.8.2	Colore del testo in HTML.....	25
5.8.3	Esempio sull'uso dei colori.....	26
5.8.4	Tabella colori HTML.....	27
6	I link in HTML, l'ipertesto .....	28
6.1	Link che puntano ad altri documenti.....	28
6.1.1	Web link HTML: Percorsi assoluti .....	30
6.1.2	Link HTML con Percorsi relativi.....	31
6.2	Href: index.html .....	33
6.3	Consigli per i nomi dei file.....	33
6.4	I link interni o ancore in HTML.....	34
6.4.1	HTML anchor senza href .....	34
6.4.2	Anchor link a inizio pagina.....	34
6.5	Esempio di uso delle ancore all'interno dello stesso documento.....	35
6.6	Esempio di uso dei percorsi relativi .....	36
6.7	Esempio di percorso assoluto.....	40
6.8	Gli attributi dei link .....	40
6.8.1	title .....	40
6.8.2	hreflang, rel .....	41
6.8.3	target.....	41
6.8.4	accesskey.....	42

6.9	Il tag <base> .....	42
7	Elenchi puntati e numerati .....	44
7.1.1	UL, gli elenchi non ordinati (o elenchi puntati).....	45
7.1.2	Annidare le liste non numerate .....	46
7.2	OL, gli elenchi ordinati (o elenchi numerati).....	47
7.2.1	Attributo type, per descrivere il tipo di elenco.....	47
8	Le tabelle.....	49
8.1	I tag HTML per la creazione di tabelle .....	50
8.2	Il tag <table> .....	51
8.3	Il tag <td>: gli attributi colspan e rowspan .....	52
8.4	Altri attributi del tag <td> .....	54
8.5	Il tag <tr>.....	54
8.6	Un esempio completo di tabella HTML.....	55
8.7	Gestire l'allineamento con HTML: il tag center e gli attributi align e valign .....	56
8.7.1	Il tag <center> .....	56
8.7.2	Gli attributi align e valign .....	57
8.8	Approfondimenti sulle tabelle .....	58
8.9	Il template di una tabella .....	60
8.9.1	caption, thead, tbody, tfoot.....	60
8.9.2	colgroup e col, raggruppare le colonne.....	63
8.10	Esempi di tabelle che utilizzano i fogli di stile.....	64
9	Le immagini .....	66
9.1	I formati delle immagini per il Web.....	66
9.2	Il tag <img>.....	67
9.3	L'attributo src.....	67
9.4	L'attributo alt .....	67
9.5	Altri attributi del tag <img> .....	68

9.6	Considerazioni sugli attributi width e height del tag <img> .....	69
9.7	Immagini con link .....	70
9.8	Link ad un'immagine .....	71
10	I Form.....	72
10.1	Il tag form .....	72
10.1.1	I tag utilizzati per creare moduli HTML.....	72
10.2	Esempio di Form.....	74
10.3	Alcune osservazioni in merito ai tag per la creazione di moduli.....	76
10.4	Inviare file attraverso un modulo.....	77
10.5	I form e l'interazione con l'utente .....	78
10.6	Un esempio .....	79
10.7	I tag utilizzabili in un form .....	80
10.8	La validazione dei dati nei form .....	82
11	I fogli di stile CSS.....	84
12	I contenitori HTML e i selettori CSS.....	84
13	L'accessibilità e l'usabilità.....	84

# 1 Risorse

## VIDEO CORSO

- Video Lezioni: <https://www.youtube.com/playlist?list=PL7A4A3449C649048F>

## RISORSE WEB

- Una delle risorse migliori in italiano: <https://www.html.it/guide/guida-html/>
- Ottimo sito (in inglese): <https://www.w3schools.com/html/>
- Elenco dei TAG: <https://www.w3schools.com/TAGS/>
- Un altro sito molto bello in italiano: <https://www.mrwebmaster.it/html/guida-html/>

## LIBRO DI TESTO

- PRO.SIA per la Classe 3 – Lorenzi, Giupponi – ATLAS

## LINK UTILI

- Editor Note Pad ++: <https://notepad-plus-plus.org/>
- Browser Firefox: <https://www.mozilla.org/it/firefox/new/>



## 2 INTRODUZIONE

### 2.1 Prerequisiti

- Avere un PC
- Notepad ++
- Browser

### 2.2 Le pagine HTML e i TAG

**HTML** è l'acronimo di *HyperText Markup Language* (“Linguaggio di contrassegno per gli Iper testi”) e non è un linguaggio di programmazione. Si tratta invece di un **linguaggio di markup** (di ‘contrassegno’ o ‘di marcatura’), che permette di indicare come disporre gli elementi all’interno di una pagina.

Queste indicazioni vengono date attraverso degli appositi marcatori, detti **tag** (*‘etichette’*), che hanno la caratteristica di essere inclusi tra parentesi angolari (ad es, `<img>` è il segnaposto di un’immagine).

Con HTML quindi indichiamo, attraverso i *tag*, quali elementi dovranno apparire su uno schermo e come essi debbano essere disposti. Tutte queste indicazioni sono contenute in un documento HTML, spesso detto “**Pagina HTML** “. Una pagina HTML è rappresentata da un *file di testo*, ovvero un file che possiamo modificare con programmi come notepad e in genere hanno un nome che finisce con l’estensione `.html`.

I TAG del linguaggio HTML sono delle stringhe comprese fra i caratteri “<” e “>”

Esempio `<HTML>`, `<BODY>`, `<P>`

Il tag di chiusura sono contraddistinti dal simbolo “/”

Non essendo un vero e proprio linguaggio di programmazione eventuali errori non ci verranno segnalati come capita, ad esempio, nell’editor DEV C++.

## 3 Hello HTML!

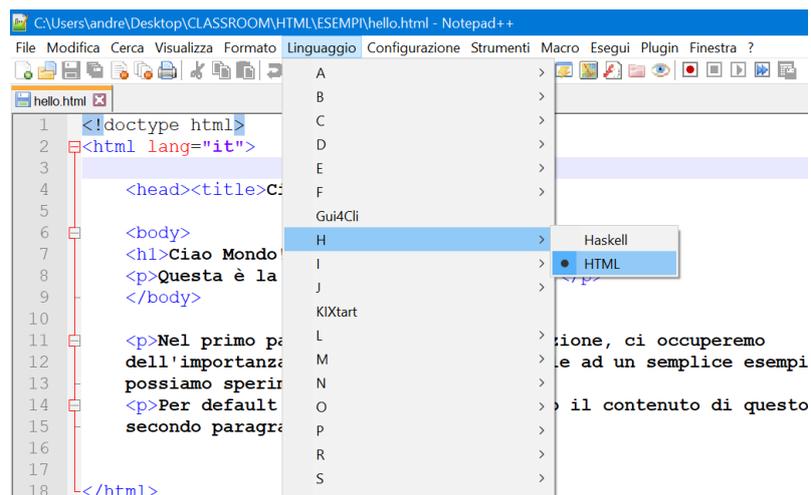
### 3.1 Creare la prima pagina

Seppure molto potente, HTML non è un linguaggio complesso e può offrire soddisfazioni immediate. Per iniziare basta pochissimo, quasi niente in realtà:

- Un **browser**, ne abbiamo almeno uno già installato nel PC o nei device che utilizziamo.
- Un **editor di testi**, anche di questi ce ne sono a bizzeffe e tipicamente ce n'è uno in ogni ambiente operativo.

L'importante è ricordare che non dobbiamo utilizzare programmi come Word, OpenOffice o WordPad, che sono “word processor” evoluti e non servono allo scopo.

Per iniziare a fare qualcosa in HTML utilizzeremo Notepad++ selezionando il linguaggio HTML.

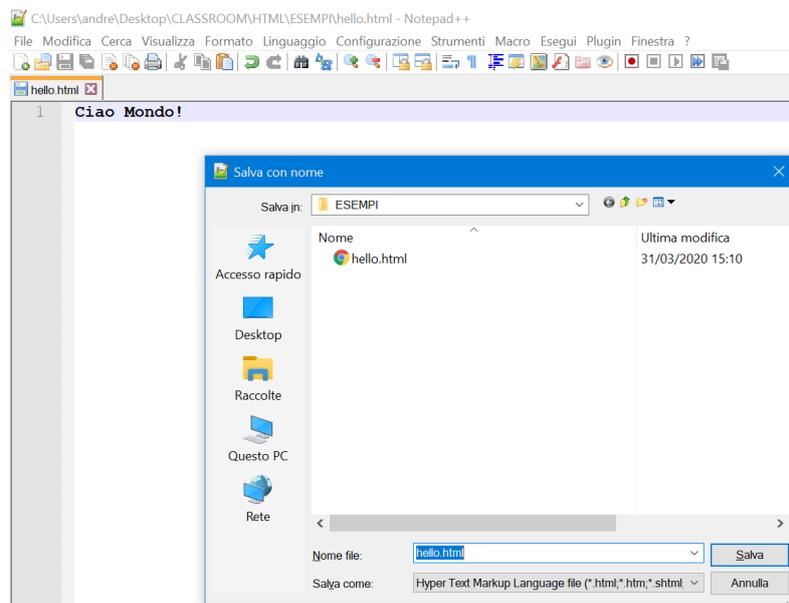


## 3.2 Estensione .html

Lanciamo Notepad++, impostiamo come lingua html, e scriviamo un semplice testo come il seguente:

```
Ciao Mondo!
```

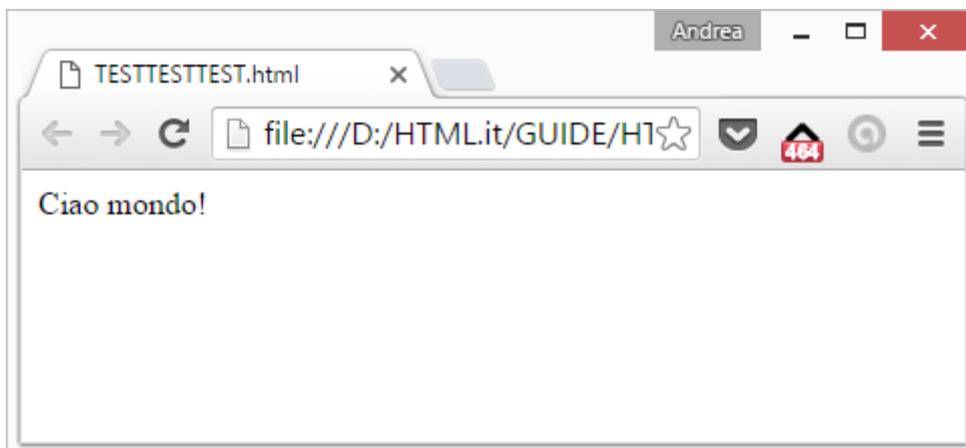
Fatto questo salviamo il file con estensione .html ad esempio “hello.html“. Per i meno esperti, Windows a volte nasconde le estensioni e potrebbe risultare difficile salvare correttamente i nostri file. In questi casi possiamo abilitare le estensioni grazie alle opzioni di visualizzazione delle cartelle.



Notare il formato **.html** e il Salva come: Hyper Text Markup Language file.

### 3.3 Apriamo il file HTML

Una volta assegnata questa estensione, cliccando sul file si aprirà automaticamente il browser predefinito di sistema e mostrerà una schermata simile alla seguente:



Niente di straordinario dunque: viene mostrato il testo a video! Ma vediamo cosa succede se aggiungiamo qualcosa al nostro markup:

```
<h1>Ciao Mondo!</h1>  
<p>Questa è la nostra prima pagina HTML!</p>
```

Una volta aggiunto un titolo (Heading) di primo livello (<h1>) e un paragrafo (<p>) proviamo ad **aggiornare la pagina** (in genere basta digitare F5) il testo viene modificato in questo modo:



Il browser mostra il nostro titolo con un carattere grande e in grassetto. Mentre il contenuto del paragrafo (quello compreso tra <p></p>) è rappresentato con una dimensione più piccola.

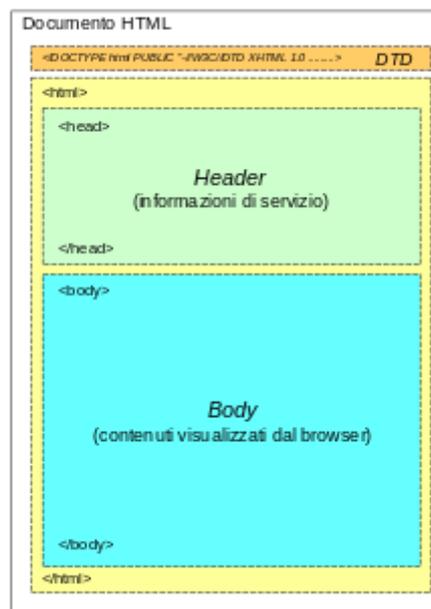
In effetti non abbiamo detto nulla sullo **stile degli elementi** rappresentati, ci siamo limitati a definire un titolo e un paragrafo e, come ci ritroveremo a dire diverse altre volte, questo è il compito del

markup HTML. Tali definizioni infatti hanno un'importanza che va oltre la sola rappresentazione grafica.

Le scelte di stile sono comunque importanti ma accessorie, in questo caso vediamo applicate quelle preconfezionate nel browser e vedremo poi come modificarle a nostro piacimento.

### 3.4 Struttura generale di una pagina HTML

La struttura generale di una pagina HTML è la seguente



Per dichiarare un documento che utilizzerà lo standard dell'Html5 si andrà ad utilizzare questo tag:

```
<!DOCTYPE html>
```

Dopo la dichiarazione del tipo di documento, il documento HTML presenta una struttura ad albero annidato, composta da sezioni delimitate da tag opportuni che al loro interno contengono a loro volta sottosezioni più piccole, sempre delimitate da tag.

La struttura più esterna è quella che delimita l'intero documento, eccetto la DTD, ed è compresa tra i tag `<html>` e `</html>`.

All'interno dei tag `<html>` lo standard prevede sempre la definizione di due sezioni ben distinte e disposte in sequenza ordinata:

- la sezione di intestazione o *header*, delimitata tra i tag `<head>` e `</head>`, che contiene informazioni di controllo normalmente non visualizzate dal browser, con l'eccezione di alcuni elementi
- la sezione del corpo o *body*, delimitata tra i tag `<body>` e `</body>`, che contiene la parte informativa vera e propria, ossia il testo, le immagini e i collegamenti che costituiscono la parte visualizzata dal browser.

Al di sotto di questa suddivisione generale, lo standard non prevede particolari obblighi per quanto riguarda l'ordine e il posizionamento delle ulteriori sottosezioni all'interno dell'*header* o del *body*, a parte l'indicazione del rispetto dei corretti annidamenti (le sottosezioni non si devono sovrapporre, ossia ogni sottosezione deve essere chiusa prima di iniziare la sottosezione successiva), lasciando così completa libertà allo sviluppatore o al progettista per quanto riguarda la strutturazione e l'organizzazione successive.

### 3.5 I tag `<head>` e `<body>`: la struttura minimale della pagina HTML

Per fare le cose un po' meglio e dirci soddisfatti della nostra prima pagina torniamo sul nostro editor e modifichiamo il markup in questo modo:

```
<!doctype html>
<html lang="it">
  <head><title>Ciao Mondo!</title></head>
  <body>
    <h1>Ciao Mondo!</h1>
    <p>Questa è la nostra prima pagina HTML!</p>
  </body>
</html>
```

Aggiornando la pagina con F5 non osserveremo sostanziali cambiamenti, ma più avanti vedremo quanto sia importante definire per bene la struttura della pagina.

Tag	Descrizione
<code>&lt;!DOCTYPE html&gt;</code>	Serve semplicemente a dire che il file è una pagina HTML e in particolare che si tratta di un documento secondo lo standard HTML5. Lo si mette punto e basta.
<code>&lt;html&gt;</code>	È il tag che racchiude tutta la pagina e ci permette ad esempio di definire quale sia la lingua della pagina, come nel nostro caso in cui abbiamo definito che il documento è in Italiano grazie all'attributo <code>lang="it"</code>
<code>&lt;head&gt;</code>	Questo tag contiene una serie di informazioni utili per la gestione della pagina, nel nostro caso abbiamo definito ad esempio quale sarà il titolo che apparirà sui motori di ricerca e sulle linguette del browser, grazie al tag <code>&lt;title&gt;</code> , ma questa sezione è destinata a contenere molto di più: dall'inserimento di librerie alla definizione dell'insieme di caratteri. Vedremo tutto in seguito.
<code>&lt;body&gt;</code>	Contiene gli elementi della pagina, tutto il contenuto e tutti i relativi tag che saranno poi resi a video vengono inseriti qui dentro, come abbiamo fatto nel nostro caso con il titolo e il paragrafo.

### 3.6 Sbirciare l'HTML nelle pagine Web

Abbiamo visto come completare con soddisfazione una prima pagina HTML, ma una delle cose più interessanti di questo linguaggio è che il markup è scritto come un testo, perciò è possibile **leggere codice scritto da altri**.

Tutti i browser infatti danno la possibilità di **mostrare il codice HTML delle pagine** e quasi tutti permettono di farlo con la combinazione di tasti `CTRL+U`.

## 4 Elementi e tag in HTML

In una pagina HTML tutti gli elementi sono connotati da **tag** (letteralmente “etichette”). Si tratta di marcatori che evidenziano non tanto l’aspetto, quanto il senso, il ruolo, o l’organizzazione che vogliamo assegnare ai contenuti. Ad esempio se indichiamo un titolo con `<h1>` per noi le parole in quel titolo rappresentano il tema principale della pagina, al di là del modo in cui saranno visualizzate.

- Questa **visione “semantica”** degli elementi è fondamentale, ci aiuta a non perdere di vista il fatto che una pagina deve essere un luogo ordinato di informazioni, soprattutto sul Web. Se pensiamo ad esempio al classico contesto delle ricerche online, risulta naturale comprendere che tanto più una pagina sarà associabile a una keyword o a un tema, tanto più facile sarà trovarla. Per questo sarà importante curare cose come l’organizzazione del testo, la definizione dei titoli, dei link dei grassetto.
- Tuttavia in HTML possiamo anche definire lo scheletro dell’interfaccia utente di una app, e in questo frangente i tag diventano utili come supporto all’**organizzazione del layout** o alla definizione di aree specifiche per l’esperienza utente.

In tutti e due i casi continuiamo a non parlare di “grafica”, ma di struttura.

### 4.1 Tag della head

I tag utilizzati nella sezione *head* sono tipicamente di tipo diverso da quelli utilizzati nella sezione *body*, essendo destinati a scopi differenti. I tag utilizzati nella sezione *head* normalmente non vengono visualizzati dal browser ma servono come informazioni di controllo e di servizio quali:

- *metadata* per convogliare informazioni utili ad applicazioni esterne (es. motori di ricerca) o al browser (es. codifica dei caratteri, utile per la visualizzazione di alfabeti non latini)
- *metadata* di tipo *http-equiv* per controllare informazioni aggiuntive nel protocollo HTTP
- collegamenti verso file di servizio esterni (CSS, script, icone visualizzabili nella barra degli indirizzi del browser)
- inserimento di script (codice eseguibile) utilizzati dal documento
- informazioni di stile (CSS locali)
- il titolo associato alla pagina e visualizzato nella finestra principale del browser

## 4.2 Tag del body

All'interno della sezione di body, che racchiude la parte visualizzabile del documento, si utilizzano i tag specifici previsti per la formattazione dei contenuti accessibili all'utente finale, ossia per il controllo di:

1. intestazioni (titoli di capitoli, di paragrafi eccetera)
2. strutture di testo (testo indentato, paragrafi, eccetera)
3. aspetto del testo (grassetto, corsivo, eccetera)
4. elenchi e liste (numerate, generiche, di definizione)
5. tabelle
6. moduli elettronici (campi compilabili dall'utente, campi selezionabili, menu a tendina, pulsanti eccetera)
7. collegamenti ipertestuali e ancore
8. layout generico del documento
9. inserimento di immagini
10. inserimento di contenuti multimediali (audio, video, animazioni eccetera)
11. inserimento di contenuti interattivi (script, applicazioni esterne)

Tra quelli elencati sopra, il tag `<a>` è quello che gioca un ruolo chiave in quanto descrive un collegamento (o *link*) a un altro documento ipertestuale consentendo, con un click del mouse o con un'operazione da tastiera, di abbandonare la pagina o il sito che si sta visualizzando e caricare quella indicata dal link, realizzando così la funzione di navigazione tipica dell'uso di Internet.

## 4.3 Come è fatto un tag

Un tag è una keyword del linguaggio racchiusa tra parentesi angolari (`<>`). Esempi di tag sono `<html>`, `<body>` e `<h1>`, che abbiamo già incontrato nelle lezioni precedenti.

I tag HTML *non sono "case sensitive"* ciò significa che scrivere `<head>` o `<HEAD>` è esattamente la stessa cosa. In ogni caso la consuetudine è quella di **scrivere i tag in minuscolo** (era una raccomandazione in HTML4 e una richiesta in XHTML).

## 4.4 Elementi contenitori e autonomi, il DOM

Molti elementi in HTML servono per descrivere porzioni di pagina, aree, o contenuti. Ad esempio `<body>` descrive il contenuto di tutta la pagina, `<h1>` racchiude un titolo e `<p>` denota un paragrafo nel testo.

Pertanto **un elemento HTML è quasi sempre un contenitore** e il suo contenuto è delimitato da:

tag di apertura (es. `<p>`);

tag di chiusura (es. `</p>`).

```
<p>Lorem ipsum, quia dolor sit, amet, consectetur, adipisci velit,  
sed quia non numquam eius modi tempora incidunt, ut labore et dolore  
magnam aliquam quaerat voluptatem.</p>
```

## 4.5 Gli attributi

I tag HTML possono essere corredati di uno o più attributi, che servono per meglio specificare la funzione o la tipologia dell'elemento, per memorizzare dati o per arricchire di significato il contenuto.

Un tag con attributi si scrive in questo modo:

```
<tag attributo1="valore1" attributo2="valore2">
```

Vedremo più avanti degli esempi pratici

## 5 Titoli, paragrafi e testi

HTML è nato per la realizzazione di contenuti ipertestuali ed è naturale che la maggior parte dei tag siano stati creati per l'**organizzazione dei testi**, quindi per la definizione di titoli, paragrafi sezioni, articoli, etc.

Questa organizzazione ha poco a che vedere con la rappresentazione grafica della pagina, quanto con la sua tematizzazione e con la sua semantica. Nel corso del tempo *i tag HTML che definivano aspetti grafici sono stati via via deprecati* e sono rimasti quelli che aiutano a identificare sezioni e porzioni significative del testo.

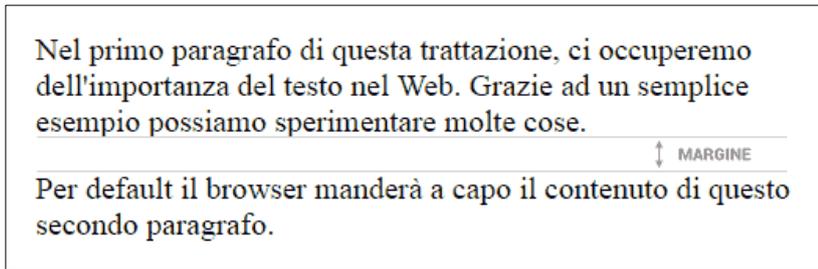
È anche vero che i browser utilizzano questi tag anche per dare maggior chiarezza al testo assegnando loro degli stili preconfezionati.

### 5.1 I paragrafi <p>

Il paragrafo `<p>`, già visto in precedenza, è un elemento contenitore che al suo interno prevede l'inserimento di testo e di altri tag. Ecco un esempio:

```
<p>Nel primo paragrafo di questa trattazione, ci occuperemo dell'importanza del testo nel Web. Grazie ad un semplice esempio possiamo sperimentare molte cose. </p>  
<p>Per default il browser manderà a capo il contenuto di questo secondo paragrafo. </p>
```

Nel Browser si avrà:



## 5.2 Andare a capo, <br>

Il tag **<br>** sta per *break line* e serve per andare a capo nel bel mezzo di un testo. Come è facile notare infatti, anche se andiamo a capo all'interno dell'editor, il testo nel browser non va a capo se non alla fine dello spazio disponibile per il suo contenitore.

```
Questo è un testo che va a capo  
spesso all'interno dell'editor  
tuttavia i browser ignoreranno  
questo fatto e considereranno <br>  
il testo come una singola  
riga continua, che va a capo  
solo quando incontra il limite  
del proprio contenitore
```

Nel Browser si avrà:

Questo è un testo che va a capo spesso all'interno dell'editor tuttavia i browser ignoreranno questo fatto e considereranno il testo come una singola riga continua, che va a capo solo quando incontra il limite del proprio contenitore

## 5.3 Headings, i titoli h1, h2, h3, etc.

I titoli (**headings**) sono i tag che ci aiutano a definire il tema della pagina. In genere sono rappresentati in grassetto e con una dimensione del testo ingrandita. Anche questi sono elementi di tipo *block* e sia le dimensioni sia il margine applicato per default dal browser sono proporzionali all'importanza del titolo.

### 5.3.1 Gerarchia degli headings

Come si intuisce dai numeri **i titoli sono gerarchici**. Questo significa che *h1* sarà il titolo principale della pagina. Se ci serviranno altri titoli per specializzare alcune sezioni importanti del testo (le “main sections”) è consigliato scegliere prima *h2*, per le sottosezioni utilizzeremo *h3* e così via con una suddivisione sempre più granulare fino ad utilizzare l'*h6*.

```
<h1>Casa</h1>  
<h2>Acquistare la casa</h2>  
<h3>Le pratiche per l'acquisto di casa</h3>  
<h3>Agenzie immobiliari, quali scegliere</h3>  
<h2>Arredare la casa</h2>  
<h3>Come scegliere la cucina per la nuova casa</h3>  
<h3>Lampadari, tipologie e differenze</h3>
```

# Casa

## Acquistare la casa

### Le pratiche per l'acquisto di casa

### Agenzie immobiliari, quali scegliere

## Arredare la casa

### Come scegliere la cucina per la nuova casa

### Lampadari, tipologie e differenze

La gerarchia impatta anche sul peso che gli headings hanno in termini di tematizzazione della pagina. In genere si consiglia di inserire nell'*h1* le parole chiave principali per definire il contenuto e negli *h2* parole chiave che sono accessorie.

### 5.3.2 Gerarchia degli headings nelle home page

Se stiamo parlando della home page è più indicato utilizzare l'*h1* per il nome del sito, l'*h2* per le categorie e *h3* per i titoli degli ultimi contenuti pubblicati (o i più importanti). L'*h4* può essere utile

per i titoli di altri contenuti correlati e h5 per cose meno importanti, meno tematizzanti o meno correlate con il tema della pagina.

## 5.4 Grassetto e corsivi

Fin dalle prime versioni di HTML, perché un testo apparisse in grassetto, è stato sufficiente inserirlo in un tag `<b>`, così come per mettere in corsivo una frase si è utilizzato il tag `<i>`.

Per una costruzione dei documenti più orientata alla semantica e con le versioni più recenti del linguaggio, è stato consigliato di utilizzare `<em>` per il corsivo e `<strong>` per i grassetto. Rispettivamente i due tag stavano per *emphasis* (enfasi) e *strong emphasis* (forte enfasi) e i browser assegnano loro come stili proprio il corsivo e il grassetto.

Questa seconda scelta certamente sembra più in linea con il principio della **separazione tra contenuto e grafica**, secondo cui i tag non dovrebbero definire l'aspetto del testo.

D'altra parte in questo modo si è posto un **limite alla presenza dei grassetto** perché destinati a poche parole molto importanti. Se questo aspetto può avere un impatto positivo anche su un piano di ritmo tipografico, può tendere a omogeneizzare la stesura dei testi.

TAG	DESCRIZIONE	RENDERING DI BASE
<code>&lt;strong&gt;</code>	Attribuisce al testo una forte importanza, serietà o urgenza (ora <i>strong</i> sta per <i>importance</i> ). Aiuta a tematizzare la pagina e può essere utilizzato anche per strategie SEO, può essere utilizzato anche all'interno di headings per indicare la parte più importante di un titolo (es. <code>&lt;h1&gt;Capitolo 1. &lt;strong&gt;La casa&lt;/strong&gt;&lt;/h1&gt;</code> ).	Grassetto
<code>&lt;b&gt;</code>	Offre una differenza stilistica rispetto al resto del contenuto, senza attribuire un'importanza specifica al testo ( <b>nota:</b> che sia bold o no non importa).	Grassetto
<code>&lt;em&gt;</code>	Simile a <i>strong</i> , serve a rappresentare un testo o una frase che si pronuncia in modo differente dal resto al testo. (da <i>emphasis</i> diventa <i>stress emphasis</i> ).	Corsivo
<code>&lt;i&gt;</code>	Serve a rappresentare testo che esprima un tono, uno stato d'animo o qualcosa che si discosti dal resto del contenuto, senza aggiungere ulteriori significati o importanza.	Corsivo

### 5.4.1 Esempi di utilizzo dei tag u, em, b e i

Sottolineiamo la parola chiave “cucina” che è la più importante della frase.

```
La <u>cucina</u> è la parte più importante della casa...
```

Qui poniamo l’accento sul momento in cui completare i lavori a casa.

```
La ristrutturazione? Sarebbe ideale avere già tutto pronto <em>prima del trasloco</em>.
```

Qui utilizziamo il tag <i> per indicare un termine tecnico.

```
Anche per le piastrelle del bagno una buona scelta è quella del <i>gres porcellanato</i>
```

In questo esempio differenziamo lo stile delle frasi pronunciate da Arnold Schwarzenegger in una celebre scena di “Commando”.

```
<b>Ricordi quando ti ho detto che ti avrei ammazzato per ultimo?</b>
```

```
Sì, l'hai detto! L'hai detto!
```

```
<b>Ti ho mentito.</b>
```

Ecco il risultato nel browser:

Sottolineiamo la parola chiave “cucina” che è la più importante della frase.

La cucina è la parte più importante della casa...

Qui poniamo l’accento sul momento in cui completare i lavori a casa.

La ristrutturazione? Sarebbe ideale avere già tutto pronto *prima del trasloco*.

Qui utilizziamo il corsivo per indicare un termine tecnico.

Anche per le piastrelle del bagno una buona scelta è quella del *gres porcellanato*

In questo esempio differenziamo lo stile delle frasi pronunciate da Arnold Schwarzenegger in una celebre scena di “Commando”.

**Ricordi quando ti ho detto che ti avrei ammazzato per ultimo?**

Sì, l'hai detto! L'hai detto!

**Ti ho mentito.**

## 5.5 Altri tag per enfatizzare il testo

### 5.5.1 Il tag `<s>` e `<strike>` - barrato

È utilizzato per applicare lo stile barrato ad un testo:

```
Applico lo stile <s>barrato</s>
```

### 5.5.2 i tag `<big>` e `<small>`

Questi tag sono utilizzati per scrivere un testo utilizzando una dimensione grande (big) o piccola (small) del font specificato:

```
Scrivo in <big>grande</big>...  
...e ora in <small>piccolo</small>
```

### 5.5.3 I tag `<sup>` e `<sub>`

Questi tag sono utilizzati per scrivere, rispettivamente, un testo in apice (sup) o in pedice (sub). Vediamo due semplici esempi:

```
E = mc<sup>2</sup>
```

Output: E = mc<sup>2</sup>

```
H<sub>2</sub>O
```

Output: H<sub>2</sub>O

## 5.6 Aggiungere un commento in HTML

Per aggiungere un commento la sintassi è piuttosto semplice, si racchiude il testo tra `<!--` e `-->`:

```
<!-- questo è un commento -->
```

Possiamo racchiudere in un commento anche molte righe e, spesso, i commenti sono utilizzati per disattivare alcune parti del markup: inserendo un tag in un commento lasciamo infatti che esso sia ignorato dal browser. Ad esempio:

```
<p>Questo paragrafo si vede</p>
<!--
<p>Questo paragrafo non si vede</p>
-->
```

## 5.7 La dimensione e lo stile dei caratteri il tag `<font>`

Utilizzeremo il tag `<font>` ed alcuni dei suoi attributi per specificare alcune proprietà del carattere come il colore, il tipo di font e la dimensione.

Attributo	Valore	Descrizione
color	<i>#xxxxxx oppure nomecolore</i>	Specifica il colore del testo
face	<i>font_family</i>	Specifica il tipo di font
size	<i>numero</i>	Specifica la dimensione

Una volta scelto il carattere con cui scrivere il nostro testo possiamo scegliere il colore, ecco il codice secondo il markup HTML:

```
<font color="yellow">testo giallo</font>
<br/>
ovvero:
<br/>
<font color="#0000FF">testo blu</font>
```

## 5.8 I colori

### 5.8.1 Colore di sfondo a tinta unita in HTML

Incominciamo col vedere come ottenere la nostra prima pagina HTML nel modo in cui desideriamo visualizzarla.

In HTML possiamo impostare un colore di sfondo è utilizzando il relativo attributo del tag body. Così:

```
<body bgcolor="blue">
```

Il colore in HTML : Background-color

**bgcolor** sta per “background color”, cioè “colore di sfondo”. Molti colori sono disponibili utilizzando le corrispondenti parole chiave in inglese.

In alternativa è preferibile utilizzare lo stile CSS, in quanto il colore di sfondo è una caratteristica estetica della pagina:

```
<body style="background-color:blue">
```

Tuttavia non è consigliabile inserire la notazione del colore facendo riferimento a questo tipo di sintassi, dal momento che non possiamo sapere esattamente a quale tonalità di colore corrisponda il blu del computer dell’utente. È preferibile in molti casi utilizzare la corrispondente codifica esadecimale del colore, che ci permette – tra le altre cose – di scegliere anche tonalità di colore non standard. Con la notazione esadecimale il nostro esempio diventa:

```
<body bgcolor="#0000FF">
```

### 5.8.2 Colore del testo in HTML

Come già visto, una volta scelto il carattere con cui scrivere il nostro testo possiamo scegliere il colore, ecco il codice secondo il markup HTML:

```
<font color="yellow">testo blu</font>
```

```
<br/>
```

ovvero:

```
<br/>
```

```
<font color="#0000FF">testo blu</font>
```

### 5.8.3 Esempio sull'uso dei colori

Proviamo il seguente esempio:

```
<!doctype html>
<html lang="it">

  <head><title>Colori</title></head>

  <body bgcolor="blue">

    <h1>Ciao!</h1>
    <p>Ne vedremo di tutti i colori e di tutti i tipi</p>

    <font size="5" color="red">Un po' di testo colorato</font>
    <font size="2" color="yellow">Un po' di testo colorato</font>
    <font size="4" face="verdana" color="green">Un po' di testo
colorato</font>

  </body>

</html>
```

**Ciao!**

Ne vedremo di tutti i colori

Un po' di testo colorato Un po' di testo colorato Un po' di testo colorato

### 5.8.4 Tabella colori HTML

Ecco una tabella con la notazione di alcuni colori (molti di essi sono disponibili anche nelle varianti “dark” e “light”, ad esempio: “darkblue”, “lightblue”):

colore	parola chiave	notazione esadecimale
arancione	orange	#ffa500
blu	blue	#0000ff
bianco	white	#ffffff
giallo	yellow	#ffff00
grigio	gray	#808080
marrone	brown	#a52a2a
nero	black	#000000
rosso	red	#ff0000
verde	green	#008000
viola	violet	#ee82ee

## 6 I link in HTML, l'ipertesto

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

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

Il tag di riferimento è:

```
<a> </a>
```

### 6.1 Link che puntano ad altri documenti

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

```
I contenuti per sviluppatori sono su  
<a href="https://www.html.it/">HTML.it</a>.
```

Che produce come risultato:

I contenuti per sviluppatori sono su [HTML.it](https://www.html.it/).

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

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

Tipologia	Comportamento standard
<b>Immagine .gif, .jpg, .png</b>	Viene visualizzata nel browser
<b>Documento .html, .pdf, .doc</b>	La pagina è visualizzata nel browser. Nel caso dei documenti come <code>.doc</code> e <code>.pdf</code> l'utente potrebbe aver bisogno di installare un apposito plugin, anche se quasi sempre questi formati sono interpretati dal browser stesso (nel caso peggiore è sufficiente avere installati rispettivamente Microsoft Word e Adobe Acrobat Reader). Se non è installato il plugin il sistema chiederà all'utente se salvare il file.

<b>File .zip, file .exe</b>	Viene chiesto all'utente di scaricare il file
-----------------------------	---

È anche possibile specificare un **indirizzo email come link**. In questo caso si aprirà direttamente il client di posta dell'utente con l'indirizzo email pre-impostato. La sintassi è la seguente:

```
<a href="mailto:tuaMail@nomeTuoSito.it">  
Invia una email  
</a>.
```

Che dà come risultato:

[Invia una email](#).

Finché le dimensioni dei progetti (specie siti web) sono ridotte e contano poche pagine possiamo anche ipotizzare di lasciare tutti i file in una medesima cartella. È evidente però che - man mano che il progetto cresce - avremo bisogno di un maggior ordine.

Si presenterà allora l'esigenza di inserire le immagini in cartelle diverse (in modo da averle tutte nella medesima locazione) e forse sarà opportuno dividere il sito in varie sezioni, in modo da avere tutti i documenti dello stesso tipo all'interno di un contesto omogeneo.

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

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

Per farlo possiamo utilizzare due modalità:

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

### 6.1.1 Web link HTML: Percorsi assoluti

Descrivere un **percorso assoluto** significa indicare per esteso l'indirizzo di un certo documento, elencando tutte le directory e sottodirectory che dobbiamo attraversare per raggiungerlo. Per comprendere meglio esaminiamo questo indirizzo:

```
<a href="https://www.html.it/css/index.html">fogli di stile</a>
```

Il link indica un percorso assoluto e fa riferimento ad una particolare directory. Nella fattispecie:

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

## 6.1.2 Link HTML con Percorsi relativi

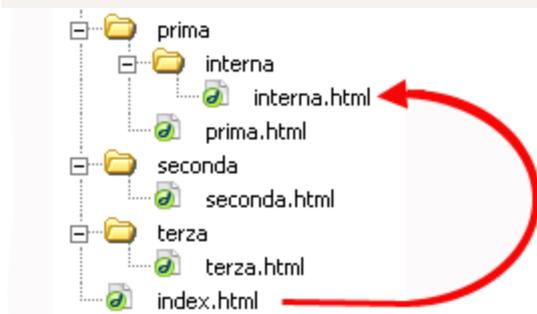
Spesso ci capita tuttavia a fare riferimento a documenti situati all'interno del progetto (o del sito), e - sviluppando sul nostro computer (cioè "in locale") - magari non abbiamo ancora un indirizzo web e non sappiamo di conseguenza come impostare i percorsi. È utile allora capire come funzionano i percorsi relativi.

I **percorsi relativi** fanno riferimento alla posizione degli altri file rispetto al documento in cui ci si trova in quel momento. Per linkare due pagine che si trovano all'interno della stessa directory è sufficiente scrivere:

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

collegamento alla pagina da linkare nella stessa directory della pagina presente

```
</a>
```

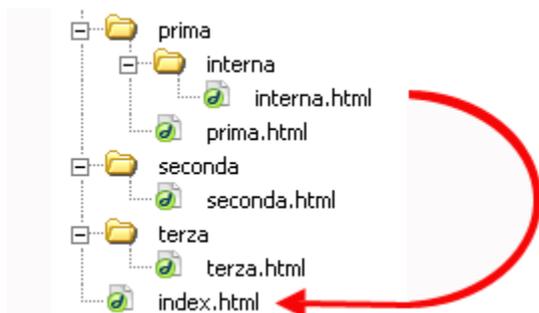


Dalla pagina `index.html` vogliamo cioè far riferimento al file `interna.html`, che si trova nella directory `interna`, che a sua volta si trova all'interno della directory `prima`.

La sintassi è la seguente:

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

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



La sintassi è la seguente:

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

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

1. Per far riferimento a un file che si trovi all'interno della stessa directory basta linkare il nome del file:

```
<a href="paginaDaLinkare.html">collegamento alla pagina</a>
```

2. Per far riferimento a un file contenuto in una cartella di livello inferiore alla posizione corrente, basta nominare la cartella seguita dal carattere / (slash), e poi il nome del file. Secondo la formula: cartella/nomeFile.htm:

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

3. Per tornare su di un livello, è sufficiente utilizzare la notazione: ../nomeFile.html

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

Grazie a questi accorgimenti possiamo agevolmente navigare tra le directory del progetto o del sito: per fare un ulteriore esempio si può tornare su di un livello rispetto alla posizione del file, scegliere un'altra cartella, e poi scegliere un altro file in questo modo:

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

## 6.2 Href: index.html

È possibile incontrare una notazione come la seguente:

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

Se il progetto è su un server Unix (ma la sintassi funziona anche in sistemi Windows, basta che non siano in locale), questa notazione non deve stupirvi: il carattere ‘/’, messo all’inizio dell’indirizzo, indica la directory principale del sito, altrimenti detta **root**. Dunque `<a href="/css/index.html">` è un altro modo di esprimere i percorsi assoluti all’interno del proprio sito.

Un’altra cosa importante da sapere è che una volta che il sito è online, l’indicazione dell’index all’interno di una directory è quasi sempre facoltativa. Al posto di questo:

```
https://www.html.it/css/index.html
```

è sufficiente indicare la directory:

```
https://www.html.it/css/
```

Bisogna verificare con il gestore dello spazio web (cioè “hosting”), se le pagine index della directory devono avere forma `index.html`, `index.htm`, `index.asp`, `index.php`, `home.asp`, o altro.

## 6.3 Consigli per i nomi dei file

Come noto esistono due famiglie di sistemi operativi: Windows e Unix (Linux). Questi due sistemi operativi utilizzano differenti modi per gestire i file, dunque alcuni accorgimenti sono necessari:

è consigliabile non lasciare spazi vuoti nei nomi dei file (gli spazi vuoti non sempre vengono interpretati correttamente), meglio ovviare a questa necessità con un “trattino basso” (cioè “\_”). Ad esempio: `mio_file.html`

maiuscole e minuscole possono fare la differenza (in ambiente Unix spesso la fanno), quindi meglio controllare il modo in cui sono scritti i file

Inoltre nel creare un collegamento bisogna fare attenzione a non avere una notazione simile a questa:

```
<a href="file:///C|percorsonomeFile.html">testo</a>
```

che in genere rappresenta un riferimento (assoluto) al nostro computer e non funzionerebbe una volta messo online il progetto.

## 6.4 I link interni o ancore in HTML

Nel **markup HTML**, possiamo sfruttare il meccanismo dei link anche per creare un indice interno al documento, utilizzando le àncore. Ciascuna ancora può avere infatti un nome:

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

### 6.4.1 HTML anchor senza href

In **assenza dell'attributo href**, che indica il collegamento ipertestuale le àncore non sono viste come link, ma la loro formattazione è indistinguibile dal “normale” testo.

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

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

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

### 6.4.2 Anchor link a inizio pagina

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

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

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

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

È bene non confondere le due fasi.

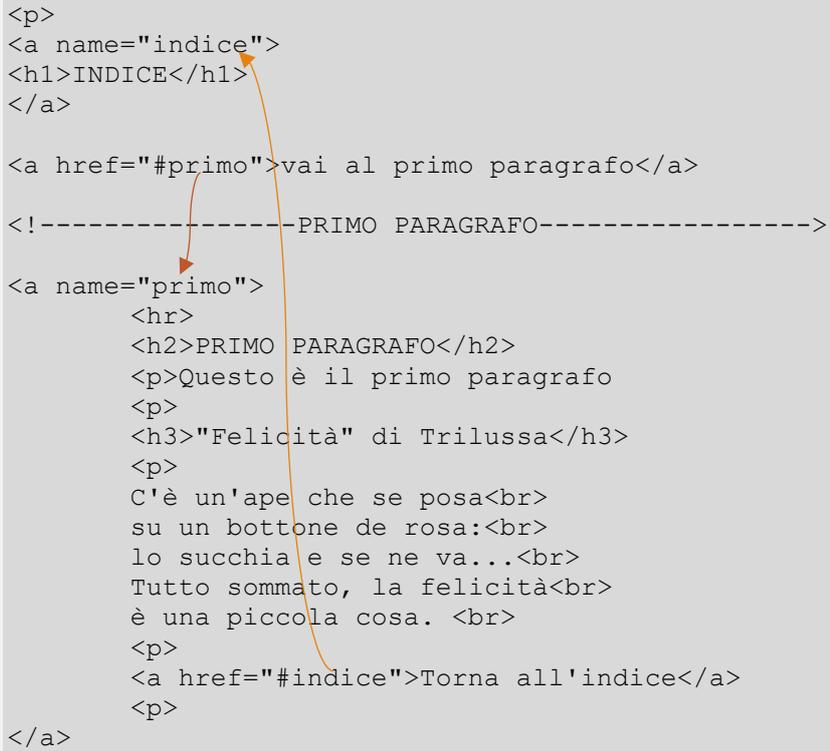
## 6.5 Esempio di uso delle ancore all'interno dello stesso documento

```
<p>
<a name="indice">
<h1>INDICE</h1>
</a>

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

<!-------PRIMO PARAGRAFO----->

<a name="primo">
  <hr>
  <h2>PRIMO PARAGRAFO</h2>
  <p>Questo è il primo paragrafo
  <p>
  <h3>"Felicità" di Trilussa</h3>
  <p>
  C'è un'ape che se posa<br>
  su un bottone de rosa:<br>
  lo succhia e se ne va...<br>
  Tutto sommato, la felicità<br>
  è una piccola cosa. <br>
  <p>
  <a href="#indice">Torna all'indice</a>
  <p>
</a>
```



Provare l'esempio "9 - I link interni o ancore in HTML"

## 6.6 Esempio di uso dei percorsi relativi

In un percorso relativo dobbiamo spiegare al browser quale strada dovrà seguire per poter andare a raggiungere la pagina web destinazione, partendo dal punto in cui si trova quella di partenza.

Per riferirmi all'esempio descritto sopra, cosa potrebbe succedere se volessi incontrare una persona che abita nella mia stessa via, usando il metodo descritto per il percorso assoluto?

Dovrei di nuovo uscire dalla città per poi rientrarci: inutile, no?

Se la persona abita nella mia stessa via, mi segno soltanto il numero civico e:

1. esco di casa
2. vado al palazzo di destinazione
3. cerco il cognome sul citofono.

Ovviamente questo procedimento non va bene se la persona si trova in un'altro paese.

Facciamo ora un esempio.

Il nostro sito contiene la seguente struttura:

www.sito.it:

- homepage.html
- cartella1/
  - [partenza.html](#)
  - topolino.html
  - sottocartella/
    - pluto.html
- cartella2/

pippo.html

Immaginiamo di essere sulla pagina **partenza.html** e di voler raggiungere una delle altre pagine con questo collegamento:

```
<a href="">Vai alla destinazione</a>
```

### **Caso 1 – Destinazione = topolino.html**

La pagina di destinazione del collegamento è nella stessa cartella che ospita la pagina di origine (cartella1).

Non mi muovo da lì, perché la pagina partenza vede perfettamente la pagina topolino (sono nella stessa cartella), quindi mi basta chiamare la pagina di destinazione col suo nome:

```
<a href="topolino.html">Vai alla destinazione</a>
```

### **Caso 2 – Destinazione = pluto.html**

La pagina di destinazione del collegamento non è nella stessa cartella ma è immediatamente raggiungibile entrando in una cartella che riusciamo a vedere e scrivendo il nome della pagina:

```
<a href="sottocartella/pluto.html">Vai alla destinazione</a>
```

### **Caso 3 – Destinazione = homepage.html**

La pagina di destinazione del collegamento è immediatamente fuori dalla cartella che ospita la pagina di origine (cartella1).

Se la pagina non è nella stessa cartella, devo “uscire” da dove mi trovo ora usando:

```
../
```

e poi digito il nome del file:

```
<a href="../homepage.html">Vai alla destinazione</a>
```

### **Caso 4 – Destinazione = pippo.html**

La pagina di destinazione del collegamento non è nella stessa cartella (quindi usciamo con ../), ma non riusciamo ancora a vederla, perché si trova in un'altra cartella.

Scriviamo quindi il nome della cartella e, quindi, il nome del file:

```
<a href="../cartella2/pippo.html">Vai alla destinazione</a>
```

Torniamo all'esempio di prima. Questa volta organizziamo il lavoro nel seguente modo: una pagina per creare l'indice e 3 differenti pagine in cui reperire i contenuti

```
<!doctype html>
<html lang="it">

<head><title>I link Relativi in HTML</title></head>

<body>
  <p>
    <h1>INDICE</h1>
    <a href="../ESEMPI/paragrafi/primo paragrafo.html">vai al primo paragrafo</a>
  <p>
    <a href="../ESEMPI/paragrafi/secondo paragrafo.html">vai al secondo
paragrafo</a>
  <p>
    <a href="../ESEMPI/paragrafi/terzo paragrafo.html">vai al terzo paragrafo</a>
</body>

</html>
```

Il link punta al file: **primo paragrafo.html**  
presente nella sottocartella relativa **../paragrafi**

*10 - I link relativi.html*

```
<!doctype html>
<html lang="it">

<head>
  <title>I link Relativi in HTML</title>
</head>

<body>

  <h2>PRIMO PARAGRAFO</h2>
  <p>Questo è il primo paragrafo
  <p>
    <h3>"Felicità" di Trilussa</h3>
    <p>
      C'è un'ape che se posa<br>
      su un bottone de rosa:<br>
      lo succhia e se ne va...<br>
      Tutto sommato, la felicità<br>
      è una piccola cosa. <br>
    <p>
      <a href="../10 - I link relativi.html">Torna all'indice</a>
    <p>

</body>

</html>
```

Torna indietro all'indice

*primo paragrafo.html*

Il cui risultato è:

## INDICE

[vai al primo paragrafo](#)

[vai al secondo paragrafo](#)

[vai al terzo paragrafo](#)

## PRIMO PARAGRAFO

Questo è il primo paragrafo

### "Felicità" di Trilussa

C'è un'ape che se posa  
su un bottone de rosa:  
lo succhia e se ne va...  
Tutto sommato, la felicità  
è una piccola cosa.

[Torna all'indice](#)

## 6.7 Esempio di percorso assoluto

```
<!doctype html>
<html lang="it">

  <head><title>I link Assoluti in HTML</title></head>

  <body>
    In questo caso si punta a risorse esterne
  <p>
  <a name="indice">
  <h1>INDICE</h1>
  </a>
  <a href="https://www.poesiedautore.it/trilussa/felicita-di-trilussa">vai al
  primo paragrafo</a>
  <p>
  <a href="https://www.poesiedautore.it/giacomo-leopardi/l-infinito">vai al
  secondo paragrafo</a>
  <p>
  <a href="https://www.poesiedautore.it/wislawa-szymborska/le-tre-parole-piu-
  strane">vai al terzo paragrafo</a>
```

Il cui risultato è:



## 6.8 Gli attributi dei link

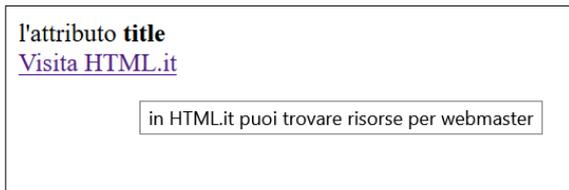
Gli attributi dei link servono ad aggiungere valore semantico o funzionale ai collegamenti. Vediamo di cosa si tratta.

### 6.8.1 title

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

```
l'attributo <b>title</b> <br>  
<a title="in HTML.it puoi trovare risorse per webmaster"  
href="https://www.html.it/" target="_blank" >  
Visita HTML.it  
</a>
```

Il cui risultato è:



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

### 6.8.2 hreflang, rel

L'attributo `rel="nofollow"` è un valore assegnabile all'attributo `rel` dell'elemento HTML a per evidenziare pagine non inerenti al proprio sito web.

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

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

### 6.8.3 target

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

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

`target="_blank"` indica che il documento linkato verrà aperto in una nuova finestra.

## visita HTML.IT

### 6.8.4 accesskey

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

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

cioè:

### Torna all'home page di HTML.it

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

## 6.9 Il tag <base>

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

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

indichiamo che d’ora in poi tutti i percorsi relativi faranno riferimento al percorso indicato. Nella pagina possiamo quindi scrivere:

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

certi che farà riferimento a:

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

Si tratta di una caratteristica particolarmente utile quando bisogna mandare ad esempio delle mailing list in formato HTML: possiamo infatti utilizzare i percorsi relativi per sviluppare la pagina della mailing list in locale, e mantenerli inalterati grazie all’utilizzo di questo tag. Grazie ad esso siamo

infatti sicuri che anche l'utente che riceverà la mail potrà visualizzare le immagini e i link con un percorso corretto.

## 7 Elenchi puntati e numerati

Gli elenchi giocano un ruolo molto importante nei testi, ma anche nella definizione di elementi strutturali delle pagine HTML.

Dal punto di vista dell'**organizzazione del testo** sono utili ad esempio quando occorre sintetizzare concetti per punti, indicare in sequenza le fasi di un processo oppure stilare un elenco di definizioni o un glossario.

Nella organizzazione della **struttura della pagina** sono spesso utilizzati per definire menu ad esempio, cosa che anche semanticamente ha una sua valenza, perché si tratta sempre di liste.

Sul piano del rendering, tipicamente **spezzano il ritmo tipografico** e rendono il testo più gradevole e più leggibile.

Vediamo dunque quali sono le tre tipologie di elenco disponibili in HTML:

- Elenchi **ordinati**
- Elenchi **non ordinati**
- Elenchi di **definizioni**

Tutti e tre i tipi di elenchi funzionano nel medesimo modo: si apre il tag, si elencano i vari elementi della lista (ciascuno con il proprio tag), si chiude il tag dell'elenco. La sintassi ha quindi questa forma:

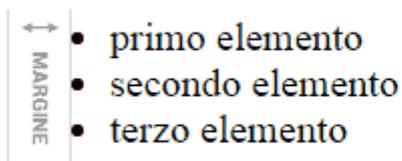
```
<elenco>
  <elemento>primo elemento<elemento>
  <elemento>secondo elemento<elemento>

  <elemento>N-esimo elemento<elemento>
</elenco>
```

### 7.1.1 UL, gli elenchi non ordinati (o elenchi puntati)

L'elenco non ordinato (*unordered list*) è forse il più usato e si descrive utilizzando il tag `<ul>`. Al suo interno possiamo inserire gli elementi della lista (*list item*) utilizzando il tag `<li>`. Ecco un semplice esempio

```
<ul>
  <li>primo elemento</li>
  <li>secondo elemento</li>
  <li>terzo elemento</li>
</ul>
```



Anche se la resa grafica dipende dal browser, il risultato in genere è quello che otteniamo quando utilizziamo gli elenchi puntati nei programmi di videoscrittura: otteniamo gli elementi uno sotto l'altro con un margine a sinistra e il classico "pallino pieno" per ogni punto.

Possiamo scegliere altri caratteri o simboli, ma vedremo in seguito come farlo sfruttando i CSS, per ora ci basti dire che l'attributo **"type"** che era previsto a questo scopo è stato deprecato da HTML 5.

### 7.1.2 Annidare le liste non numerate

In generale possiamo inserire diversi livelli all'interno delle liste, creando delle strutture "ad albero", utili a definire oggetti come menu o tassonomie. Per farlo è sufficiente inserire un nuovo elenco all'interno di un elemento:

```
<ul>
  <li>primo della 1a lista</li>
  <li>secondo della 1a lista
    <ul>
      <li>primo della 2a lista</li>
      <li>secondo della 2a lista
        <ul>
          <li>primo della 3a lista</li>
        </ul>
      </li>
      <li>terzo della 2a lista</li>
    </ul>
  </li>
</ul>
```

Interessante notare che il rendering di base in genere mostra simboli diversi per i punti a seconda del livello dell'elemento.

- 
- primo della 1a lista
  - secondo della 1a lista
    - primo della 2a lista
    - secondo della 2a lista
      - primo della 3a lista
    - terzo della 2a lista

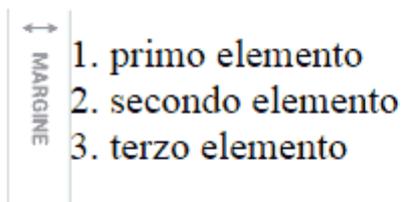
Interessante notare che il rendering di base in genere mostra simboli diversi per i punti a seconda del livello dell'elemento.

## 7.2 OL, gli elenchi ordinati (o elenchi numerati)

Gli elenchi ordinati (*ordered list*) sono contraddistinti dall'enumerazione degli elementi che compongono la lista. Il tag da utilizzare per aprire un elenco ordinato è `<ol>` e anche in questo caso gli elementi sono individuati dal tag `<li>`:

```
Testo che precede la lista
<ol>
  <li>primo elemento</li>
  <li>secondo elemento</li>
  <li>terzo elemento</li>
</ol>
Testo che segue la lista
```

Testo che precede la lista



Testo che segue la lista

Avremo quindi una serie progressiva ordinata e individuata da lettere o numeri. Per riprendere il parallelo con i software di videoscrittura, sarebbero i cosiddetti **elenchi numerati**.

### 7.2.1 Attributo `type`, per descrivere il tipo di elenco

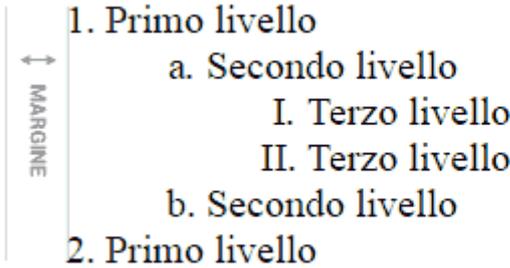
L'attributo **type** applicato al tag `ol` serve per specificare il tipo di enumerazione che vogliamo applicare. Lo stile di enumerazione visualizzato di default dal browser è quello numerico, ma possiamo indicare se le voci di elenco seguono punti definiti come numeri, come lettere dell'alfabeto o come numeri romani.

Valore di <code>type</code>	Descrizione
<code>type="1"</code>	numeri interi "arabi" (valore di default)
<code>type="a"</code>	numeri alfabeto minuscolo

type="A"	numeri alfabeto maiuscolo
type="i"	numeri numeri romani minuscoli
type="I"	numeri numeri romani maiuscoli

Proviamo quindi a vedere come rendere un elenco annidato con diversi tipi di enumerazione:

```
<ol type="1"><!-- numeri -->
  <li>Primo livello
    <ol type="a"><!-- lettere -->
      <li>Secondo livello
        <ol type="I"><!-- numeri romani -->
          <li>Terzo livello</li>
          <li>Terzo livello</li>
        </ol>
      </li>
      <li>Secondo livello</li>
    </ol>
  </li>
  <li>Primo livello</li>
</ol>
```

- 
- 1. Primo livello
    - a. Secondo livello
      - I. Terzo livello
      - II. Terzo livello
    - b. Secondo livello
  - 2. Primo livello

Poiché questi aspetti hanno rilevanza semantica, non è stato necessario deprecare l'attributo come nel caso di *ul*.

Nel file 12 - Elenchi.html troverete tutte le tipologie

## 8 Le tabelle

La creazione di una pagina Web dall'aspetto gradevole e ben strutturato si basava, fino a qualche tempo fa, sull'utilizzo delle **tabelle HTML**, ovvero griglie in cui venivano inseriti i contenuti della pagina in modo da poterli disporre affiancati o a cascata a seconda dei propri gusti.

Con l'evolversi delle tecniche di Web Design, tuttavia, l'utilizzo delle tabelle si è ridotto notevolmente in quanto, per la creazione di layout è andata affermandosi la tecnica *tableless* basata sull'utilizzo di semplici DIV formattati con apposite proprietà CSS attraverso i quali è possibile riprodurre, con un minor dispendio di markup, i risultati ottenibili con design di tipo tabellare.

Fatta questa doverosa premessa, resta il fatto che l'utilizzo di tabelle in HTML non può e non deve essere considerato obsoleto: l'utilizzo di una tabella HTML, infatti, resta essenziale per la presentazione di taluni contenuti come, ad esempio, elenchi oppure report di dati.

## 8.1 I tag HTML per la creazione di tabelle

In **HTML** si utilizzano diversi tag per realizzare una **tabella**:

- **table** - crea una tabella;
- **tr** - crea una riga di una tabella;
- **td** - crea una cella di una tabella.

Ma anche:

- **thead** - crea l'intestazione di una tabella;
- **th** - crea una cella per l'intestazione di una tabella;
- **tbody** - crea il corpo di una tabella;
- **tfoot** - crea la chiusura di una tabella ma deve essere inserito prima di tbody.

In effetti i primi tre tag sono i più utilizzati (in quanto indispensabili), mentre gli altri molto spesso vengono omessi. I tag `thead`, `tbody` e `tfoot` servono per raggruppare righe e definirne il contenuto: `thead` definisce l'intestazione della tabella, `tfoot` la chiusura e `tbody` le righe di dati.

per completezza vedremo esempi per tutti i tag, dando però precedenza ai primi tre tag: `<table>`, `<tr>` e `<td>`.

Vediamo un primo esempio di tabella:

```
<table>
  <tr>
    <td>A</td>
    <td>B</td>
  </tr>
  <tr>
    <td>C</td>
    <td>D</td>
  </tr>
</table>
```

<b>A</b>	<b>B</b>
<b>C</b>	<b>D</b>

## 8.2 Il tag <table>

Il tag **table**, che apre e chiude la tabella e funge da contenitore di tutti gli altri tag, suporta i seguenti principali attributi:

- **align** - allinea l'intera tabella a destra, sinistra o al centro (ma se ne sconsiglia l'utilizzo a favore dei CSS). Può assumere i valori center, left e right;
- **width** - imposta la larghezza di una tabella in pixel o in percentuale rispetto al suo contenitore (oggi poco utilizzato a favore dei CSS);
- **border** - imposta il bordo di una tabella (oggi poco utilizzato a favore dei CSS);
- **cellpadding** - imposta la spaziatura tra il contenuto della cella ed i margini della cella stessa;
- **cellspacing** - imposta la distanza tra le diverse celle della tabella.

vediamo un esempio:

```
<!doctype html>
<html lang="it">

  <head><title>Tabelle</title></head>

  <body>

<h1>Tabelle Generiche</h1>

<table align="center" border="1" width="500" cellpadding="5"
cellspacing="0">
  <tr>
    <td>A</td>
    <td>B</td>
  </tr>
</table>

</body>

</html>
```

### **Tabelle Generiche**

A	B
---	---

### 8.3 Il tag <td>: gli attributi colspan e rowspan

Vediamo adesso due importanti attributi del tag **td**:

- **colspan** - unisce due o più celle in orizzontale;
- **rowspan** - unisce due o più celle in verticale.

Vediamo un esempio di utilizzo di **colspan** che utilizzeremo per unire due celle della prima riga della tabella vista nel primo esempio di questa lezione:

```
<table border="1">
  <tr>
    <td colspan="2">A</td>
  </tr>
  <tr>
    <td>B</td>
    <td>C</td>
  </tr>
</table>
```

A	
B	C

Vediamo adesso un esempio di utilizzo di **rowspan** per unire verticalmente la prima cella di entrambe le righe di cui è composta la nostra tabella:

```
<table border="1">
  <tr>
    <td rowspan="2">A</td>
    <td>B</td>
  </tr>
  <tr>
    <td>C</td>
  </tr>
```

```
</table>
```

<b>A</b>	<b>B</b>
	<b>C</b>

## 8.4 Altri attributi del tag <td>

Si ricordi che **td** supporta anche altri attributi, i più importanti sono: **width**, **height** (larghezza e altezza), **align** e **valign** (utilizzati per gestire l'allineamento dei contenuti della cella). In merito agli attributi *align* e *valign* rimandiamo il lettore alla prossima lezione dedicata alla gestione dell'allineamento dei contenuti.

## 8.5 Il tag <tr>

Il tag <tr> è utilizzato per definire una linea e, pertanto, svolge il ruolo di contenitore per i tag <td> (contenuti) o <th> (intestazioni).

Il tag in questione è utilizzato, di solito, senza attributi, tuttavia è possibile usare **align** e **valign** per gestire l'allineamento dei contenuti all'interno della linea.

## 8.6 Un esempio completo di tabella HTML

Concludiamo con un esempio completo di tabella:

```
<table border="1">

  <thead>
    <tr>
      <th>Nome</th>
      <th>Cognome</th>
    </tr>
  </thead>

  <tbody>
    <tr>
      <td>Giuseppe</td>
      <td>Verdi</td>
    </tr>
    <tr>
      <td>Antonio</td>
      <td>Bianchi</td>
    </tr>
    <tr>
      <td>Mario</td>
      <td>Rossi</td>
    </tr>
  </tbody>

  <tfoot>
    <tr>
      <td colspan="2">Ho mostrato 3 utenti</td>
    </tr>
  </tfoot>

</table>
```

Questo il nostro risultato visivo:

Nome	Cognome
Giuseppe	Verdi
Antonio	Bianchi
Mario	Rossi
Ho mostrato 3 utenti	

## 8.7 Gestire l'allineamento con HTML: il tag `center` e gli attributi `align` e `valign`

Attraverso il markup HTML è possibile definire l'allineamento degli elementi costitutivi della pagina (testo, immagini, tabelle, ecc.) per farlo si ricorre al tag `<center>` e agli attributi **align** e **valign**. E' bene precisare, tuttavia, che ad oggi si tratta di tecniche di allineamento deprecate in quanto la presentazione dei contenuti dovrebbe essere gestita esclusivamente attraverso i CSS.

### 8.7.1 Il tag `<center>`

Come è facile immaginare questo tag veniva utilizzato per centrare del testo o degli elementi della pagina come immagini e tabelle. Vediamo un esempio di tabella centrata utilizzando il tag `<center>`:

```
<center>
  <table border="0">
    ...
  </table>
</center>
```

Come potete vedere l'elemento che abbiamo voluto centrare è compreso tra il tag `<center>` e `</center>`.

Ad oggi questo tag è uscito dalle specifiche del linguaggio tant'è che HTML5 nè ha escluso il supporto ed il suo utilizzo è vivamente sconsigliato.

### 8.7.2 Gli attributi **align** e **valign**

Questi attributi possono essere applicati a diversi tag HTML. Nello specifico l'attributo *align* riguarda l'allineamento orizzontale mentre *valign* quello verticale.

I valori possibili per **align** sono "left", "center" e "right" (il valore di default è "left"). Questo attributo veniva solitamente applicato ai tag <p>, <div> e <td> per definire l'allineamento dei rispettivi contenuti. Veniva spesso applicato anche al tag <img> per definire l'allineamento dell'immagine stessa in relazione ad eventuali elementi adiacenti.

```

```

L'attributo **valign**, come detto, definisce l'allineamento verticale e può avere uno di questi valori: "top", "middle", "bottom" e "baseline". Questo attributo veniva utilizzato frequentemente nelle tabelle per allineare verticalmente il contenuto di righe (<tr>) o singole celle (<td>).

## 8.8 Approfondimenti sulle tabelle

Per definire una **tabella “minimale”**, che sia comunque interpretata correttamente dai browser possiamo riprodurre un esempio simile al seguente:

```
<table>
<tr><td>Colonna 1</td><td>Colonna 2</td></tr>
<tr><td>Dato 1,1</td><td>Dato 1,2</td></tr>
<tr><td>Dato 2,1</td><td>Dato 2,2</td></tr>
<tr><td>Dato 3,1</td><td>Dato 3,2</td></tr>
</table>
```

Colonna 1	Colonna 2
Dato 1,1	Dato 1,2
Dato 2,1	Dato 2,2
Dato 3,1	Dato 3,2

Nell’esempio riusciamo a definire una griglia formata da righe e colonne, il risultato è piuttosto povero a livello semantico (non distinguiamo ad esempio il nome di una colonna dai suoi valori), ma ci aiuta a presentare i tag di base:

Tag	Descrizione
<b>&lt;table&gt;</b>	È il contenitore di tutta la tabella e la definisce
<b>&lt;tr&gt;</b>	“ <i>table row</i> ” Contiene una riga della tabella
<b>&lt;td&gt;</b>	“ <i>table data</i> ” Una cella che contiene i valori all’interno di una riga

Questo tipo di tabella era un elemento predominante nel “vecchio Web” (ed è ancora presente in quello attuale), perché funzionale anche alla suddivisione degli spazi, oltre che alla rappresentazione dei dati. Con il consolidarsi di pratiche più specifiche per il markup dei layout, questo utilizzo delle tabelle è caduto in disuso.

Per ottenere una visualizzazione come quella in figura dobbiamo forzare un po’ lo stile della tabella utilizzando le seguenti regole CSS:

```
<style>
table {
border-collapse:collapse
}
td, th {
border:1px solid #ddd;
padding:8px;
}
</style>
```

Le riportiamo solo per completezza senza commentarle in quanto non aggiungono nulla alla comprensione delle tabelle in questa fase.

## 8.9 Il template di una tabella

### 8.9.1 caption, thead, tbody, tfoot

Per scrivere una tabella che fornisca una rappresentazione più chiara dei dati introduciamo un template leggermente più ricco:

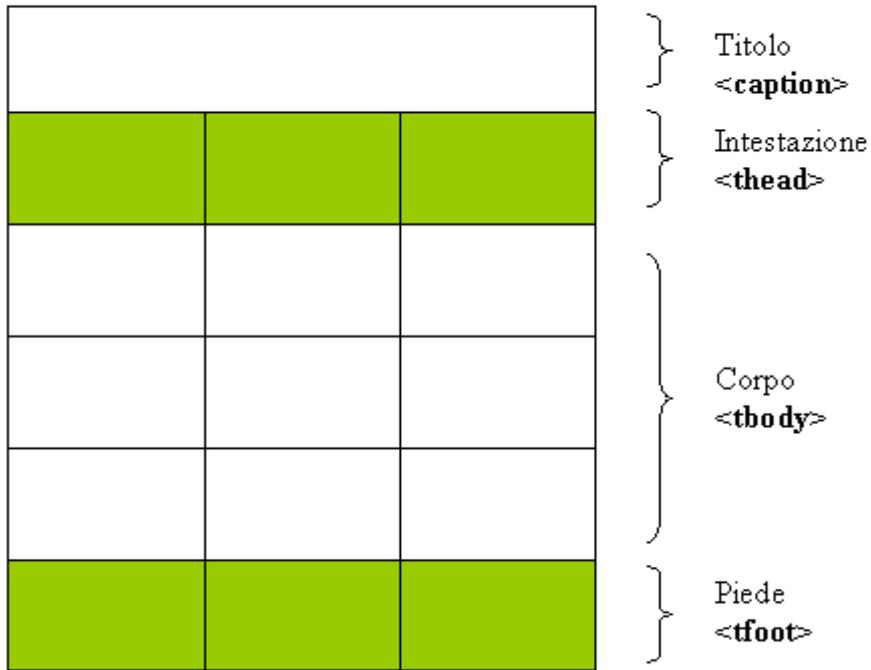
```
<table>
<caption>
<p>I miei dati</p>
</caption>
<thead>
<tr><th>Colonna 1</th><th>Colonna 2</th></tr>
</thead>
<tfoot>
<tr><td>Totale 1</td><td>Totale 2</td></tr>
</tfoot>
<tbody>
<tr><td>Dato 1,1</td><td>Dato 1,2</td></tr>
<tr><td>Dato 2,1</td><td>Dato 2,2</td></tr>
<tr><td>Dato 3,1</td><td>Dato 3,2</td></tr>
</tbody>
</table>
```

<b>I miei dati</b>		← CAPTION
<b>Colonna 1</b>	<b>Colonna 2</b>	← THEAD
Dato 1,1	Dato 1,2	> TBODY
Dato 2,1	Dato 2,2	
Dato 3,1	Dato 3,2	
Totale 1	Totale 2	← TFOOT

I tag introdotti sono tutti definiti come “facoltativi”, nel senso che la tabella non si “rompe” se manca uno di essi. Tuttavia si tratta di elementi fondamentali per la semantica delle tabelle. Esaminiamoli:

Tag	Descrizione
<b>&lt;caption&gt;</b>	È una didascalia che ci permette di dare una contestualizzazione ai dati e rendere più chiaro il loro significato.
<b>&lt;thead&gt;</b>	Serve per raggruppare le righe che rappresentano l’intestazione della tabella.
<b>&lt;th&gt;</b>	“ <i>table header</i> ” Indica una cella che contiene un’intestazione (ad esempio il titolo di una colonna o di una riga) e serve a dare una definizione dei dati cui si riferisce.
<b>&lt;tbody&gt;</b>	Raggruppa le righe che contengono il corpo della tabella, spesso con i dati veri e propri.
<b>&lt;tfoot&gt;</b>	Racchiude le righe che utilizziamo come footer della tabella, in cui si possono inserire dei dati di riepilogo, somme, medie, etc.

### Schema di una tabella



## 8.9.2 colgroup e col, raggruppare le colonne

Possiamo raggruppare le celle della tabella anche per colonna grazie al tag **<colgroup>**, che deve essere inserito tra il *caption* e ogni possibile raggruppamento per riga (*thead*, *tbody*, *tfoot*).

All'interno di *colgroup* definiamo le colonne che vogliamo includere usando il tag **<col>**.

Con *colgroup* i gruppi vengono creati prendendo le colonne da sinistra verso destra. Il numero di colonne da considerare lo indichiamo grazie all'**attributo span**.

```
<table>
<colgroup id="a" span="3"></colgroup>
<colgroup id="a">
<col id="b1">
<col id="b2" span="2">
</colgroup>
<thead>
<tr><th>head</th><th>head</th><th>head</th><th>head</th><th>head</th><th>head</th></tr>
</thead>
<tbody>
<tr><td>cell</td><td>cell</td><td>cell</td><td>cell</td><td>cell</td><td>cell</td></tr>
<tr><td>cell</td><td>cell</td><td>cell</td><td>cell</td><td>cell</td><td>cell</td></tr>
<tr><td>cell</td><td>cell</td><td>cell</td><td>cell</td><td>cell</td><td>cell</td></tr>
</tbody>
```

## 8.10 Esempi di tabelle che utilizzano i fogli di stile

Si prenda questo esempio. Utilizzando i fogli di stile `<style>` è possibile generare tabelle, e non solo, dall'aspetto più professionale. Si noti come `<style>` sia incapsulato nel tag `<head>`.

```
<!DOCTYPE html>
<html>
<head>
<style>
table {
  font-family: arial, sans-serif;
  border-collapse: collapse;
  width: 100%;
}

td, th {
  border: 1px solid #dddddd;
  text-align: left;
  padding: 8px;
}

tr:nth-child(even) {
  background-color: #dddddd;
}
</style>
</head>
<body>

<h2>HTML Table</h2>

<table>
  <tr>
    <th>Company</th>
    <th>Contact</th>
    <th>Country</th>
  </tr>
  <tr>
    <td>Alfreds Futterkiste</td>
    <td>Maria Anders</td>
    <td>Germany</td>
  </tr>
  <tr>
    <td>Centro comercial Moctezuma</td>
    <td>Francisco Chang</td>
    <td>Mexico</td>
  </tr>
  <tr>
    <td>Ernst Handel</td>
    <td>Roland Mendel</td>
    <td>Austria</td>
  </tr>
  <tr>
    <td>Island Trading</td>
    <td>Helen Bennett</td>
    <td>UK</td>
  </tr>
  <tr>
    <td>Laughing Bacchus Winecellars</td>
```

```

        <td>Yoshi Tannamuri</td>
        <td>Canada</td>
    </tr>
    <tr>
        <td>Magazzini Alimentari Riuniti</td>
        <td>Giovanni Rovelli</td>
        <td>Italy</td>
    </tr>
</table>

</body>
</html>

```

Una tabella con stile

<b>HTML Table</b>		
<b>Company</b>	<b>Contact</b>	<b>Country</b>
Alfreds Futterkiste	Maria Anders	Germany
Centro comercial Moctezuma	Francisco Chang	Mexico
Ernst Handel	Roland Mendel	Austria
Island Trading	Helen Bennett	UK
Laughing Bacchus Winecellars	Yoshi Tannamuri	Canada
Magazzini Alimentari Riuniti	Giovanni Rovelli	Italy

Stessa tabella senza stile

<b>HTML Table</b>		
<b>Company</b>	<b>Contact</b>	<b>Country</b>
Alfreds Futterkiste	Maria Anders	Germany
Centro comercial Moctezuma	Francisco Chang	Mexico
Ernst Handel	Roland Mendel	Austria
Island Trading	Helen Bennett	UK
Laughing Bacchus Winecellars	Yoshi Tannamuri	Canada
Magazzini Alimentari Riuniti	Giovanni Rovelli	Italy

## 9 Le immagini

Una delle caratteristiche del linguaggio **HTML** è, come insito nel suo nome, l'essere ipertestuale. Con l'HTML, quindi, si supera il testo ed i documenti si arricchiscono di altri elementi come immagini ed elementi multimediali. In questa lezione vedremo **come inserire un'immagine in un documento HTML**. Prima di vedere qual è il markup HTML necessario per inserire un'immagine in una pagina web, tuttavia, è bene fare qualche premessa circa i file d'immagine e le caratteristiche che devono avere.

### 9.1 I formati delle immagini per il Web

I principali e più diffusi formati immagine utilizzati sul Web sono:

- **GIF** (adatto a loghi, bottoni e banner, e possono essere immagini animate, quindi in movimento)
- **JPG** (adatto alle foto, dato che si tratta di un formato ad alta risoluzione)
- **PNG** (molto portabile su tutte le piattaforme e meglio trattabile attraverso software di grafica).

Inutile utilizzare altri formati d'immagine in quanto il risultato potrebbe non essere quello aspettato: i browser, infatti, sono in grado di gestire solo i formati grafici per il Web. Non possono essere utilizzati, quindi, file nei formati PSD, BMP, TIFF, ecc.

Altra raccomandazione importante riguarda il "peso" delle immagini. E' consigliabile, infatti, utilizzare immagini non troppo pesanti in quanto potrebbero rallentare notevolmente il caricamento della pagina. Prima di pubblicare un'immagine all'interno di una pagina web, quindi, è consigliabile gestire con attenzione risoluzione e dimensione del file al fine di trovare un compromesso tra la qualità dell'immagine ed il suo peso.

Un'altra raccomandazione importante riguarda il numero di immagini presenti all'interno di una pagina web: se è possibile inserire infinite immagini (non ci sono limiti in tal senso) è altrettanto vero che, di prassi, si cerca di non esagerare col numero di immagini presenti in un documento HTML. Anche in questo caso valgono le osservazioni già fatte circa i tempi di caricamento: un numero elevato di immagini, infatti, rischierebbe di rallentare la navigazione infastidendo l'utente (ed anche i motori di ricerca).

## 9.2 Il tag <img>

Al fine di inserire **immagini** all'interno delle pagine Web è necessario utilizzare il tag IMG.

Il tag <img> si dice vuoto, quindi non prevede l'omonimo tag di chiusura.

L'attributo principale del tag IMG è **src** che è utilizzato per definire la URL dell'immagine che si vuole incorporare nella pagina. Altro attributo fondamentale è **alt** il quale è utilizzato per impostare un testo alternativo, ovvero un'etichetta dell'immagine.

La sintassi di base del tag <img>, quindi, è la seguente:

```

```

## 9.3 L'attributo src

Per quanto riguarda la URL dell'immagine valgono le considerazioni già fatte nella lezione dedicata ai link circa le URL relative ed assolute. Nulla vieta, infatti, di inserire all'interno delle nostre pagine delle immagini reperibili su altri siti web semplicemente indicandone la URL all'interno dell'attributo src:

```
img src="http://www.sito.ext/file-remoto.gif"  
alt="breve descrizione dell'immagine">
```

Il rischio di una simile prassi, tuttavia, è che l'immagine remota venga sostituita o cancellata con conseguenze spiacevoli (non solo da un punto di vista estetico) per la nostra pagina web.

## 9.4 L'attributo alt

Per quanto riguarda l'attributo *alt* è bene sottolinearne l'importanza: mediante questo attributo, infatti, viene definito un testo alternativo che sarà utile ogni qualvolta l'immagine - per una qualsiasi ragione - non possa essere visualizzata (ad esempio perché rimossa o momentaneamente non disponibile per un problema di connessione). L'attributo *alt*, inoltre, riveste una grande importanza dal punto di vista dell'accessibilità della pagina web da parte di utenti non vedenti nonché, cosa non secondaria, per una

corretta ottimizzazione SEO dei contenuti (il valore dell'attributo *alt* è considerato, a tutti gli effetti, contenuto della pagina e come tale spiderizzabile dai motori di ricerca). Per maggiori informazioni sull'attributo *alt* si invita alla lettura di questo approfondimento.

## 9.5 Altri attributi del tag `<img>`

Altri attributi del tag **img** (oltre a *src* e *alt* già visti in precedenza) sono:

- **border** - imposta la dimensione (in pixel) dell'eventuale bordo all'immagine (0 = nessun bordo);
- **width** - imposta in modo esplicito la larghezza in pixel dell'immagine (facoltativo);
- **height** - imposta in modo esplicito l'altezza in pixel dell'immagine (facoltativo);
- **hspace** - imposta la distanza in orizzontale tra l'immagine ed altri elementi;
- **vspace** - imposta la distanza in verticale tra l'immagine ed altri elementi;
- **align** - imposta un allineamento dell'immagine;
- **longdesc** - consente di specificare una URL di un file HTML o TXT contenente la descrizione esaustiva dell'immagine (questo attributo completa la funzione di *alt* che, come detto, consente di inserire una sintetica descrizione dell'immagine ma è scarsamente supportato e poco utilizzato);

## 9.6 Considerazioni sugli attributi `width` e `height` del tag `<img>`

Questi attributi, come detto, sono facoltativi in quanto il browser è in grado di rilevare automaticamente le dimensioni delle immagini inserite nel codice HTML. Tuttavia, affinché il browser possa conoscere le dimensioni dell'immagine è necessario che questa sia stata caricata completamente: sino a quel momento, quindi, l'immagine "non ha una dimensione".

Da un punto di vista visivo, quindi, potremmo vedere spostarsi gli elementi della pagina man mano che vengono caricate le immagini. Al fine di evitare questo fastidioso effetto è possibile dichiarare (mediante gli attributi *width* e *height*) le dimensioni delle immagini in modo tale che queste occupino, fin da subito, il loro giusto ingombro.

```

```



Questi attributi, infine, consentono anche di **ridimensionare** al volo un'immagine senza modificare il file originale. Si supponga ad esempio di avere una foto di 400 x 200 pixel ma di volerne pubblicare una miniatura. In questo caso le strade sono due: o creiamo una versione ridotta della foto oppure, tramite `width` e `height`, impostiamo dimensioni ridotte al file (le dimensioni esplicitate tramite i due attributi prevalgono sulle dimensioni reali dell'immagine).

## 9.7 Immagini con link

Un'immagine, ovviamente, può essere usata come link, ad esempio per creare dei bottoni grafici all'interno di un menu oppure per creare dei banner pubblicitari. La sintassi per collegare un [link HTML](#) ad un'immagine è la seguente:

```
<a href="pippo.html">  
    
</a>
```

In questo caso si consiglia di impostare il bordo dell'immagine a zero per evitare che il link faccia comparire un antiestetico e poco professionale bordo blu intorno all'immagine.

## 9.8 Link ad un'immagine

Può capitare di voler visualizzare sulla propria pagina web un'immagine che si trova su un altro sito.

In questo caso si può procedere così

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

<h2> immagine presente sul mio PC </h2>


<h2> immagine non presente sul mio PC ma su un altro sito</h2>



</body>
</html>
```

### **immagine non presente sul mio PC ma su un altro sito**



## 10 I Form

L'**HTML** mette a disposizione una serie di interessantissimi tag il cui scopo è quello di creare **moduli** con tanto di **caselle di testo**, **opzioni**, **bottoni di invio** ed altro ancora.

Il termine tecnico che riassume tutti questi elementi è **form** ed il loro scopo è quello di creare interazione e scambio dati tra gli utenti ed il sito. Più precisamente, attraverso un **form HTML** (o modulo che dir si voglia) è possibile "raccogliere" gli input degli utenti per poi inviarli ad un qualche sistema di elaborazione (normalmente uno script lato server).

I **form HTML**, infatti, permettono la realizzazione dell'interfaccia del modulo ma da soli non offrono il minimo controllo e la minima gestione sui dati inseriti dagli utenti. Per far ciò, come detto, sarà necessario utilizzare degli script in grado di ricevere i dati raccolti dal form per poi eseguirne una qualche elaborazione (ad esempio inserirli all'interno di un database oppure utilizzarli per inviare una mail).

Vediamo ora di capire come si creano form completi, quali tag ed elementi in generale sono coinvolti e quali sono le loro specifiche caratteristiche.

### 10.1 Il tag form

Per la creazione di un form utilizziamo l'omonimo tag **form**, come nell'esempio che segue:

```
<form method="" action="" target="">
...
</form>
```

Scopo di questo tag è di fare da contenitore ad una serie di tag (che vedremo tra poco) che costituiranno gli specifici controlli del modulo.

I principali attributi del **tag form**, come visto nell'esempio, sono i seguenti:

- **method** - specifica il metodo di invio dei dati ed accetta i valori **get** o **post**;
- **action** - specifica lo script che riceverà, controllerà ed elaborerà i dati.
- **target** - specifica se il frutto dell'elaborazione (effettuata dallo script indicato in *action*) verrà mostrata nella stessa finestra oppure in un'altra.

#### 10.1.1 I tag utilizzati per creare moduli HTML

Una volta definito un form mediante l'omonimo tag sarà necessario "popolarlo" con una serie di tag annidati al suo interno. Attraverso i singoli tag (che andremo a vedere tra poco), infatti, sarà possibile

creare i vari elementi per l'interazione con l'utente come, ad esempio, caselle di testo o menu di selezione.

Passiamo adesso in rassegna i singoli tag che generano gli elementi dei form HTML:

- **input** - genera la maggior parte degli elementi dei form HTML, a seconda del **type** specificato. Gli input più utilizzati sono:
  - **text** - è utilizzato per creare *caselle di testo* in cui l'utente può scrivere del contenuto su "singola linea";
  - **file** - è utilizzato per creare *caselle di selezione di file in locale* al fine di poterli trasmettere al server remoto;
  - **radio** - permette di creare un gruppo di opzioni al cui interno deve essere fatta una scelta (non ammette scelte multiple);
  - **checkbox** - permette di creare un gruppo di opzioni al cui interno devono essere fatta delle scelte (ammette scelte multiple);
  - **button** - permette di creare bottoni "neutri" ai quali, cioè, può essere associata un'azione mediante Javascript;
  - **submit** - permette di creare bottoni di invio attraverso i quali viene, appunto inviato e processato il form;
  - **image** - permette di inserire immagini "attive" all'interno del modulo che fungeranno da bottoni;
  - **reset** - permette di creare bottoni per il reset del form (in sostanza vengono cancellate le scelte effettuate dall'utente ed il modulo torna al suo stato iniziale).
- **select** - crea una casella di riepilogo a scorrimento, chiamata in gergo **selectbox**;
- **textarea** - genera un'area di testo in cui è possibile andare a capo e viene utilizzata per permettere di inserire descrizioni, commenti o comunque testi piuttosto lunghi.

## 10.2 Esempio di Form

Vediamo un esempio di un ipotetico form HTML per la raccolta di dati anagrafici:

```
<form method="post" action="esegui.php">

  <!-- CASELLE DI TESTO -->
  Nome<br>
  <input type="text" name="nome"><br>
  Cognome<br>
  <input type="text" name="cognome"><br>

  <!-- SELECTBOX -->
  Paese<br>
  <select name="paese">
  <option value="I">Italia</option>
  <option value="E">Estero</option>
  </select><br>

  <!-- RADIO -->
  Sesso<br>
  <input type="radio" name="sesso" value="M"> M<br>
  <input type="radio" name="sesso" value="F"> F<br>

  <!-- CHECKBOX -->
  Hobby<br>
  <input type="checkbox" name="hobby" value="S"> Sport<br>
  <input type="checkbox" name="hobby" value="L"> Lettura<br>
  <input type="checkbox" name="hobby" value="C"> Cinema<br>
  <input type="checkbox" name="hobby" value="I"> Internet<br>

  <!-- TEXTAREA -->
  Commento<br>
  <textarea name="commento" rows="5" cols="30"></textarea>
  <br><br>
```

```
<!-- SUBMIT -->
```

```
<input type="submit" name="invia" value="Invia i dati">
```

```
</form>
```

Il risultato visivo (al netto di eventuali stilizzazioni CSS) è il seguente:



The image shows a web form with the following elements:

- Nome: A single-line text input field.
- Cognome: A single-line text input field.
- Paese: A dropdown menu with "Italia" selected.
- Sesso: Two radio buttons labeled "M" and "F".
- Hobby: Four checkboxes labeled "Sport", "Lettura", "Cinema", and "Internet".
- Commento: A large multi-line text area.
- Invia i dati: A button at the bottom left.

Notare che i dati vengono inviati tramite l'attributo action su un file esegui.php

```
action="esegui.php"
```

Non ha affatto senso utilizzare i form se non si utilizza il php o l'asp.

### 10.3 Alcune osservazioni in merito ai tag per la creazione di moduli

- ogni elemento dei form HTML ha una caratteristica in comune, ovvero la specifica del **name** che dev'essere diverso da elemento a elemento ed uguale solo per gruppi di elementi (come nel caso dei radiobutton e dei checkbox);
- i tag **input** prevedono (oltre a *name*) due attributi principali:
  - *type*: che definisce il tipo di input;
  - *value*: definisce il valore dell'input; nei campi "text" è facoltativo e, se previsto, rappresenta una sorta di valore predefinito che gli utenti potranno modificare (a meno che il campo non sia stato impostato come *read only*).
- gli input **radio** e **checkbox** ammettono l'attributo *checked* qualora il campo debba comparire come selezionato al momento del caricamento della pagina;
- il tag **select** prevede, oltre all'attributo *name*, gli attributi *size* (che ne definisce le dimensioni in termini di linee mostrate, di default 1) e *multiple* (da specificare nel caso siano ammesse scelte multiple);
- all'interno del tag *select* vengono annidati una sequenza di tag **option** contenenti le varie opzioni disponibili all'interno della selectbox; questi tag hanno, normalmente, un attributo *value* con il valore da inviare nel caso l'opzione venga selezionata;

- i tag *option* ammettono l'attributo facoltativo *selected* attraverso il quale effettuare una selezione di default al momento del caricamento del modulo;
- il tag **textarea** prevede, oltre all'attributo *name*, gli attributi *rows* e *cols* che ne definiscono le dimensioni in termini di linee e colonne.

## 10.4 Inviare file attraverso un modulo

Come accennato poco sopra, tra i vari tipi di input esiste *file*: si tratta di un particolare campo attraverso il quale è possibile selezionare un file, presente in locale sul nostro computer, per poi inviarlo ad un server remoto. Vediamo un esempio di codice per un form HTML:

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

  <!-- FILE -->
  Carica una foto:<br>
  <input type="file" name="foto">
  <br><br>

  <!-- SUBMIT -->
  <input type="submit" name="invia" value="Invia i dati">

</form>
```

## 10.5 I form e l'interazione con l'utente

In un sito Web, per l'invio di informazioni da parte dell'utente al server, tramite il browser, si usano i **form**, che in HTML sono rappresentati dal tag `<form> . . . </form>`.

All'interno del form si possono acquisire i dati o le scelte dell'utente. I dati sono inseriti in appositi spazi, detti **campi**.

I componenti del form per inserire un valore o effettuare una scelta vengono aggiunti utilizzando il tag `<input>`. L'attributo **type** contiene la specifica del tipo di input.

```
<form action="URL-script" method="post">  
NOME <input type="text" name="nomepersona" size="25"><br>  
COGNOME <input type="text" name="cognomepersona" size="30"><br>  
<input type="submit" name="invia" value="Invia">  
<input type="reset" name="annulla" value="Cancella i dati">  
</form>
```



The image shows a rendered HTML form. It contains two text input fields. The first is labeled 'NOME' and has a width of 25 characters. The second is labeled 'COGNOME' and has a width of 30 characters. Below the input fields are two buttons: 'Invia' (a submit button) and 'Cancella i dati' (a reset button).

L'attributo **name** identifica un oggetto all'interno del form, mentre l'attributo **value** contiene la scritta che compare sul pulsante. Quando l'utente preme sul pulsante **Invia** (di tipo **submit**) i dati vengono inviati al server (metodo **post**) e viene eseguito lo script il cui indirizzo sul server (URL) è indicato come valore del parametro **action**; il codice dello script è normalmente scritto con un linguaggio lato server (per esempio **PHP** o **ASP.NET**). Il pulsante **Cancella i dati** (di tipo **reset**) svuota le caselle del form.

## 10.6 Un esempio

```
<!DOCTYPE html>
<html>
<head>
</head>
<body>
<!DOCTYPE html>
<h2>HTML Forms</h2>

<form action="https://www.w3schools.com/action_page.php">
  <label for="fname">Nome:</label><br>
  <input type="text" id="fname" name="fname" ><br>
  <label for="lname">Cognome:</label><br>
  <input type="text" id="lname" name="lname" ><br><br>
  <input type="submit" value="Submit">
</form>

<p>Se clicchi "Submit" la pagina verrà elaborata da
https://www.w3schools.com/action_page.php".</p>

</body>
</html>
```

Il Form...

## HTML Forms

Nome:

Cognome:

Se clicchi "Submit" la pagina verrà elaborata da [https://www.w3schools.com/action\\_page.php](https://www.w3schools.com/action_page.php)

... e la sua elaborazione sul Server che fa girare il motore PHP.

## Submitted Form Data

Your input was received as:

The server has processed your input and returned this answer.

**Note:** This tutorial will not teach you how servers are processing input. Processing input is explained in our [PHP tutorial](#).

## 10.7 I tag utilizzabili in un form

- **Casella password (type="password")**

È simile al tipo testo e viene utilizzata per campi speciali contenenti una password.

```
<input type="password" name="pwd" size="20" maxlength="12">
```

In questo caso i caratteri inseriti dall'utente sono sostituiti durante la digitazione da una sequenza di asterischi o di punti.

- **Campo nascosto (type="hidden")**

È un campo nascosto che non viene visualizzato. Viene utilizzato per inviare i dati al server in maniera trasparente all'utente:

```
<input type="hidden" name="codicepost" value="143354_2">
```

- **Pulsante di comando (type="button")**

Indica un pulsante generico inserito in un form:

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

Il pulsante cliccabile <button> si differenzia dal precedente perché il contenuto del bottone può essere un testo o anche un'immagine.

```
<button type="button">fai clic qui</button>
```

- **Pulsanti di opzione (type="radio")**

È utilizzato per effettuare una scelta all'interno di un elenco di voci indicate con piccoli cerchi. L'eventuale elemento preselezionato è indicato con l'attributo checked. Tutti gli oggetti dell'elenco devono avere lo stesso valore per l'attributo name.

```
<form>  
Scegli la carta di credito<br>  
<input type="radio" name="carta" value="visa" checked>Visa<br>  
<input type="radio" name="carta" value="mastercard">Mastercard<br>  
</form>
```

- **Casella di controllo (type="checkbox")**

È utilizzato per selezionare anche più di una voce contemporaneamente in un elenco di voci indicate con piccoli quadrati.

```
<form>
Lingue conosciute<br>
<input type="checkbox" name="cb1" value="ita" checked>italiano<br>
<input type="checkbox" name="cb2" value="ing">inglese<br>
<input type="checkbox" name="cb3" value="fra">francese<br>
</form>
```

- **Area di testo (tag <textarea>)**

Permette di creare un'area nella quale inserire testo, specificando il numero di righe (attributo rows) e di colonne (attributo cols).

```
<textarea name="richiesta" rows="10" cols="40">scrivi qui la tua domanda
</textarea>
```

- **Casella combinata (tag <select>)**

Rappresenta una casella combinata per effettuare una scelta da un elenco di voci. L'attributo selected del tag <option> specifica un'eventuale voce preselezionata.

```
<form>
<select name="regione">
<option value="1" selected>nord</option>
<option value="2">centro</option>
<option value="3">sud e isole</option>
</select>
</form>
```

Per consentire all'utente di effettuare scelte multiple all'interno della casella combinata (lista di selezione multipla), occorre aggiungere il parametro multiple subito dopo il tag select:

```
<select multiple name="scelte">
....
</select>
```

## 10.8 La validazione dei dati nei form

Alcuni tipi per il tag `<input>` del linguaggio HTML permettono un controllo implicito e immediato sui dati immessi, prima di inviarli al server (validazione lato client).

Per esempio, se si definisce una casella per l’inserimento di un indirizzo di posta elettronica con il tag `<input>` e l’attributo `type` uguale a `email`, il browser effettua una validazione sulla stringa inserita e segnala un messaggio di errore nel caso in cui l’indirizzo non abbia il formato corretto di un indirizzo e-mail (nome@dominio):

```
<label for="email">Indirizzo email</label>
<input type="email" name="email">
```

Nel caso in cui il browser non supporti il tipo `email`, la casella viene considerata come una normale casella di testo.

Per tutti i tipi presentati si può rendere obbligatorio un campo aggiungendo l’attributo `required`.

```
<input type="email" name="email" required>
```

Il mancato inserimento del dato richiesto provoca la visualizzazione del messaggio “Compila questo campo”.

Per verificare l’inserimento corretto dell’indirizzo di un sito Web è necessario indicare url come valore dell’attributo `type`, come nell’esempio seguente.

```
<label for="sito">Sito Web</label>
<input type="url" name="sito">
```

Altri tipi per il tag `<input>` sono:

- `tel`, per controllare i numeri di telefono,
- `date`, per l’inserimento di una data,
- `range` per visualizzare un selettore (slider) che permette di impostare un valore compreso in un intervallo.

Per esempio, impostando `date` come valore di `type`, viene visualizzato, sotto la casella di input, un calendario dal quale si può selezionare la data desiderata.

```
<input type="date" name="datanascita"><br>
```

La seguente riga di codice HTML utilizza il valore `range` dell'attributo `type` per consentire all'utente di selezionare una percentuale nell'intervallo (0, 100) attraverso uno slider.

```
0% <input type="range" name="perc" min=0 max=100 step="1"> 100%
```

## **11 I fogli di stile CSS**

**TO BE DONE**

## **12 I contenitori HTML e i selettori CSS**

**TO BE DONE**

## **13 L'accessibilità e l'usabilità**

**TO BE DONE**