HTML

Rev. 4.5 del 11/09/2024

HTML

HTML 5 : Media Query e Content Model

Il Linguaggio HTML

HTML significa **HyperText Markup Language** cioè linguaggio per la formattazione degli ipertesti, utilizzato per codificare le pagine web. In realtà si tratta di un **metalinguaggio** nel senso che non è un vero e proprio linguaggio di programmazione, ma è un linguaggio **descrittivo** costituito da una serie di marcatori (**TAG**) che vengono **interpretati** dal browser in modo opportuno.

- I Tag Html sono indipendenti dalla piattaforma su cui vengono scritti / utilizzati (windows, unix, linux).
- **Il Browser è** semplicemente **un interprete** (visualizzatore) **di documenti HTML**

Un **TAG Html** è un comando che specifica sostanzialmente quali effetti applicare ad esempio ad un testo o ad un altro elemento multimediale. Un tag è sempre racchiuso tra i caratteri < >.

Dal browser è possibile visualizzare il codice html di una qualsiasi pagina web (comando Visualizza/html oppure tasto destro oppure F12). Editor WYSIWYG significa "What You See Is What You Got".

Regole Base per la scrittura dei tag HTML

- **HTML non è case sensitive**, anche se da HTML 4, è vivamente raccomandata la scrittura dei tag in **minuscolo**.
- Tutti i <tag> tranne qualche rara eccezione, devono essere "chiusi" dal corrispondente </tag>. La chiusura dei tag deve essere gestita in modo annidato come le scatole cinesi. L'ultimo tag ad essere stato aperto dovrà essere il primo ad essere chiuso.
- Se un tag non prevede la chiusura, si usa la **chiusura abbreviata** />. Es: **
**
- Alcuni tag sono accompagnati da **ATTRIBUTI** e VALORI, sostanzialmente parametri che indicano come deve essere perseguito lo scopo del tag. I valori degli attributi devono *sempre* essere **racchiusi tra virgolette** (anche quelli numerici)
- All'interno degli ATTRIBUTI occorre **SEMPRE OMETTERE l'unità di misura**. (usata solo nei CSS). Ogni attributo può avere UNA SOLA UNITA' di MISURA, cha va sempre omessa, qualunque essa sia.

Elenco dei principali TAG

Qualunque pagina Html inizia col tag **<html>** e termina col tag **</html>**

<HTML> </HTML> Inizio e fine di una singola pagina html
<HEAD> </HEAD> Intestazione della pagina, entro i quali si scrivono meta tag, titolo, scripts.
<BODY> </BODY> Contenuto vero e proprio della pagina web.
<TITLE> </TITLE> Titolo della pagina web che sarà visualizzato nella barra del titolo del browser
Utilizzato anche per aggiungere la pagina sulla barra dei preferiti. Max 64 chr

Esempio :

```
<!DOCTYPE html>        <!--  documento html5  -->
<html lang="it">
   <head>
        <meta charset="UTF-8" />
        <title>  Esercizio 01  </title>
   </head>
   <body>
        ……
   </body>
</html>
```

Attributi del tag BODY [deprecati]
BGCOLOR colore di sfondo della pagina. Colori fondamentali: white, black, red, green, blue, magenta, yellow, cyan
TEXT colore di default per il testo della pagina.
BACKGROUND immagine di sfondo che viene ripetuta fino al completo riempimento della pagina
LeftMargin, TopMargin Margini sinistro/desto, superiore/inferiore espressi in pixel.
RightMargin, BottomMargin Margini desto e inferiore in pixel. Per default uguali a LeftMargin e TopMargin.

Formattazione del testo

Il browser ignora gli "spazi multipli" e gli "a capo". Gli "a capo" scritti in fase di editazione, vengono ignorati. Gli spazi multipli vengono compattati in un unico spazio.

<P> testo </P> Inizio e Fine paragrafo. Caratteristiche:

- A fine paragrafo il testo va a capo.
- Sopra e sotto il paragrafo viene lasciato un certo margine che lo separa dall'elemento precedente e successivo.
- Accetta al suo interno soltanto tag inline (<A>, <IMG>, <SPAN>,
). Non è quindi consentito inserire un paragrafo all'interno di un altro paragrafo.
- Non riconosce gli attributi html **WIDTH** e **HEIGHT** e quindi occupa sempre l'intera riga
- Dispone di un attributo **ALIGN** che indica l'allineamento del testo interno:
  Esempio : <**P ALIGN** = "LEFT" - "CENTER" -"RIGHT" >

<FONT> testo </FONT> Specifica il tipo di Font da utilizzare nel paragrafo. Viene utilizzato in abbinamento a <p>, all'**esterno** o all'**interno** del paragrafo o applicato anche a singole parti di testo. Dispone dei seguenti attributi

<**FONT FACE** = "*Times New Roman*"> i nomi disponibili sono gli **Screen.Fonts** presenti sul client. Font particolari possono essere "**allegati**" alla pagina mediante l'utilizzo dei CSS.

<**FONT COLOR** = "RED" opp "#FF0000">
Non è riconosciuto l'attributo **ALIGN** utilizzabile eventualmente all'interno del tag <P>

<**FONT SIZE** = "3"> Dimensione del carattere espresso in **PUNTI WEB** da **1 a 7**. **Il default è 3**
All'interno di SIZE, al posto del valore assoluto, si può specificare un **incremento / decremento** della dimensione rispetto genitore. Es: FONT SIZE = "+1" equivale a FONT SIZE = "4"

Oltre ai punti web, esistono diverse altre unità di misura che però **non sono utilizzabili in HTML tradizionale** ma solo all'interno dei **CSS.** Di seguito una tabella indicante la reale dimensione dei punti web:

1	8 pt
2	10 pt
3	**12 pt** (valore di default)
4	14 pt
5	18 pt
6	24 pt
7	36 pt

**
** consente di andare a capo all'interno di un paragrafo senza lasciare righe vuote.
<blockquote> testo **</blockquote>** Testo a margini rientrati. E' possibile l'annidamento.
<B> testo **</B>** Grassetto (analogo a <**strong**>)
<I> testo **</I>** Corsivo (analogo a <**em**>)
<U> testo **</U>** Sottolineato **<STRIKE>** testo **</STRIKE>** Barrato
<TT> testo **</TT>** Testo Teletype a spaziatura fissa, stile macchina da scrivere.
<**SUP**> testo </**SUP**> Apice **_{** testo **}** Pedice

<HR/> Inserisce una linea orizzontale per l'intera larghezza della pagina. L'attributo **SIZE** indica l'altezza in pixel. Es: <HR SIZE = "10" ALIGN = "CENTER" WIDTH = "80%" COLOR = "RED" />

TITOLI *(approfondimenti a pag 15 del modulo CSS: "Note su tag H e font size")*
Per i titoli sono stati definiti appositi TAG **H** che vanno da **H1** fino ad **H6.** I tag **<h>** sono visualizzati in **grassetto** e modificano il **font-size** (dimensione) ed il **padding** (spaziatura) del testo contenuto al loro interno rispetto ai valori di default ereditati dal tag genitore, cioè font-size **12 pt** e padding **16px**. In particolare

- **H1** raddoppia entrambi i valori (font **24pt** e padding **32px**)
- **H2** applica un aumento del 50% (font **18pt** e padding **24px**)
- **H3** applica un aumento del 16,5% (metà di 33) (font 14pt e padding19px)
- **H4** applica al font lo stesso font-size e lo stesso padding del genitore
- **H5** applica una riduzione di 16,5% (font 10pt padding 13px)
- **H6** applica una riduzione di 33% (font 8pt padding 10px)

Se il tag <H> viene 'avvolto' all'interno di un tag <FONT>, assume il *font-size* indicato nel tag <FONT>

<SPAN> testo </SPAN> simile a <p> però:

- a differenza di <p>, è **inline** cioè non va a capo
- non crea spaziature intorno.
- Può contenere al suo interno soltanto tag inline (<A>, <IMG>, <SPAN>,
).

<DIV> testo </DIV> E' un contenitore simile a <p>. Caratteristiche:

- E' un contenitore generico in grado di contenere qualsiasi altro tag.
- NON riconosce **nessun** attributo.
- Va a capo ma non crea spaziatura prima e dopo

<CENTER>testo</CENTER> Equivalente a <P ALIGN = "CENTER"> </P> - Deprecato

Caratteri Particolari

Impostando il meta charset="UTF-8" i browser dovrebbero riconoscere caratteri accentati e similari (file salvato in formato **UTF-8** senza BOM). Rimane però il problema dello **spazio** (il browser compatta gli spazi), dei segni di **maggiore** e **minore** usati come tag. Cioè < p> con lo spazio davanti a p viene visualizzato così com'è, mentre <p> senza spazi viene interpretato come tag e non visualizzato.

&**nbsp**;	Spazio non compattabile		
<	Segno di minore <	>	Segno di maggiore >
'	Apice singolo	"	Apice doppio
è	è	±	±
€	€	&	&
©	© Copyright	®	® Marchio depositato

&#code; **HTML character code** di un qualsiasi carattere, tra 0 e 65535. **'** equivale all'apice.

Elenchi puntati e numerati

<UL> Elenco Voci </UL> Unordered List. **Elenco puntato**. L'attributo TYPE definisce quale punto elenco utilizzare

<UL TYPE = "disc"> cerchio pieno (default) <UL TYPE = "square"> quadratino
<UL TYPE = "circle"> cerchio vuoto <UL TYPE = "none"> nessun punto elenco

<OL> Elenco Voci </OL> Ordered List. **Elenco numerato**.

<OL TYPE = "1" START="50"> numeri decimali (default) a partire da 50.
<OL TYPE = "A"> lettere maiuscole <OL TYPE = "a"> lettere minuscole
<OL TYPE = "I"> numeri romani <OL TYPE = "i"> numeri romani minuscoli

<LI> singola voce </LI> List Item. Definisce un **singola voce** di un elenco puntato UL o numerato OL. Esempio

```
<ul>
   <li> Roberto Mana   </li>
   <li> Oscar Cambieri </li>
</ul>
```

<DL> Elenco Voci </DL> Definition List. Elenco a chiavi.
Consente di definire elenchi a due livelli di annidamento senza l'utilizzo di puntini / numerazione.
Per ciascuna voce sono disponibili due tag: uno per il titolo ed uno per la descrizione

<**dt**> = **definition Tag** consente di assegnare un titolo al Paragrafo
<**dd**> = **definition descriptor** consente di assegnare una descrizione visualizzata indentata a destra rispetto al titolo

```
<dl>    <dt>autori:</dt>
          <dd> Roberto Mana</dd>
          <dd> Oscar Cambieri</dd>
        <dt>collaboratori </dt>
          <dd> Gli studenti della 3 inf B </dd>          </dl>
```

Global attributes

Attributi applicabili a tutti i tag della pagina HTML.

TITLE Rappresenta un **tooltip** visualizzato in corrispondenza del mouseOver. Breve descrizione dell'elemento.

CLASS e **STYLE** attributi x la gestione dei CSS. Ognuno può essere usato UNA SOLA VOLTA all'interno del tag

NAME e ID consentono entrambi di identificare un tag HTML. La differenza è la seguente:

- **ID** consente di identificare **univocamente** ciascun tag ed è il principale strumento di accesso da JavaScript
- **NAME** non è univoco (possono esistere più tag con lo stesso NAME) ed è usato principalmente come identificatore per i valori da restituire al server nel caso di pagine dinamiche. E' usato anche per identificare un'ancora a cui punta un tag A oppure per identificare una mappa immagine.

In entrambi i casi il valore dell'attributo è sempre case sensitive !

Collegamenti ipertestuali

<A> </A> **A = ancora** Il testo contenuto nel tag diventa un link alla URL indicata.

- Si tratta di un tag **INLINE**, cioè che non va a capo ma viene visualizzato in linea con il testo
- L'attributo **HREF** (**HyperText Reference**) definisce la risorsa da caricare.

Ogni link può puntare **:**

- ad una altra pagina dello stesso sito,
  `<A HREF = "pagina2.htm"> Vai a Pagina 2 </A>.`
- ad una pagina di un alto sito. In questo caso occorre anteporre http://
  `<A HREF = "http://www.vallauri.edu"> sito Vallauri </A>`
- ad un'ancora della stessa pagina o di un'altra pagina.
  I link ad un'ancora provocano uno scroll all'elemento rappresentato dall'ancora stessa.
  ```
  <A HREF = "#Sezione2"> vai a Sezione2 </A>
  <P ID = "Sezione2"> </P> // ancora
  ```
  Il simbolo # indica un riferimento interno alla pagina L'attributo ***ID*** di un qualsiasi controllo definisce una ancora di collegamento a cui può puntare il tag <A HREF>.
  E' anche possibile definire link ad un'ancora contenuta su una pagina diversa:
  <A HREF = "Pag9.htm#Sezione5">

  HREF = "#" **Link fittizio**. **ritorna in cima alla pagina** senza ricaricare la pagina dal server
  HREF = "" **ricarica la pagina dal server**.
- ad un file esterno (immagine, eseguibile o altro). Il browser controlla dapprima il formato del file da aprire.
  - Se il formato è gestibile dal browser (es jpg) il file viene aperto in una nuova scheda
  - Se il formato non è gestibile dal browser (es exe**)** viene chiesto dove scaricarlo

  `<A HREF = "documenti/mioFile.pdf"> open PDF file ..</A>.`
  Nelle opzioni dei browser è possibile configurare, per ogni formato, l'azione di default da intraprendere.
- ad un indirizzo di posta elettronica **<A HREF = "mailto:dest@gmail.com">**
  Apre il client di posta predefinito impostando come destinatario l'indirizzo indicato. Come parametri dopo il ? si possono subject, cc, bcc, body. Esistono alcune utility (es GMail Notifier) che consentono di ridirigere la richiesta verso una web mail piuttosto che verso il client di posta predefinito.

L'attributo TARGET controlla la modalità di apertura della nuova pagina a cui punta il collegamento ipertestuale Può assumere i valori **"*_self*" "*_blank*" "*_top*" "*_parent*"** (frames annidati), "***Nome di un frame***".

L'attributo LINK del tag <body>
<BODY **LINK** = "colore1" **VLINK** = "colore2" **ALINK** = "colore3" > consente di impostare il colore dei collegamenti ipertestuali quando sono **Unvisited**, **Visited**, e **Active** (dove Active significa attivo, cioè quello attualmente cliccato). Se non si specifica questo attributo, il browser utilizza le impostazioni di default (Opzioni Internet / Generale / Colori): Unvisited=Blu, Visited=Viola, onMouseOver=Rosso.
Per riportare ad **Unvisited** un link **Visited** occorre ripulire la cache del browser.

Immagini

<IMG SRC = "filename.jpg"/> Immagini GIF, JPEG, PNG. Non ha tag di chiusura. Attributi:

SRC definisce il nome e percorso dell'immagine (percorso relativo alla cartella corrente oppure web http://sito.it/img.jpg)
WIDTH e **HEIGHT** larghezza e altezza a cui **ridimensionare** l'immagine
Il valore può essere assoluto (in pixel) oppure in % rispetto alle dimensioni del genitore. Impostando soltanto uno dei due, (width o height) l'altro viene calcolato automaticamente mantenendo le proporzioni dell'immagine.

ALIGN indica l'allineamento dell'immagine rispetto al testo circostante. E' riferito sia all'allineamento orizzontale sia all'allineamento verticale, che dunque non possono mai essere settate contemporaneamente.

Per l'**Allineamento Verticale** i valori possibili sono BOTTOM, CENTER/MIDDLE e TOP. Se fa parte di un paragrafo P, l'immagine viene inserita **inline** nella riga esattamente dove si trova il tag IMG e viene espansa verso l'alto per la sua intera altezza. A fianco rimane però tutto spazio bianco. Utilizzando ALIGN = "TOP" l'immagine viene estesa verso il basso ma rimane lo stesso problema. Utilizzando ALIGN = "CENTER", la riga si trova a metà

Per l'**Allineamento Orizzontale** i valori possibili sono **LEFT** e **RIGHT**. Utilizzando ALIGN = "**LEFT**", l'immagine diventa fluttuante e viene ancorata a sinistra mentre il testo la avvolge sulla destra. Sono ammessi anche *paragrafi multipli* fino all'occupazione dell'intero spazio. ALIGN = "RIGHT" forza allineamento a destra

Per terminare il testo a fianco dell'immagine e riprendere a scrivere sotto l'immagine, occorre impostare **<BR CLEAR = "LEFT/RIGHT/ALL">**.
Notare che l'attributo CLEAR (peraltro deprecato in HTML5) è riconosciuto dal tag **BR** ma non da DIV, P, H.

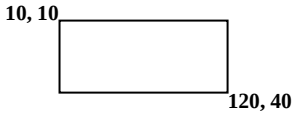
HSPACE definisce lo spazio orizzontale destro e sinistro che separano l'immagine dal testo
VSPACE definisce lo spazio verticale superiore e inferiore che separano l'immagine dal testo
ALT rappresenta il testo alternativo da visualizzare se il caricamento dell'immagine fallisce.
Riconosciuto dagli screenReader per la lettura vocale del testo ai non vedenti.
BORDER definisce l'ampiezza di un eventuale bordo di colore fisso pari ad un link non ancora visitato. Lo spessore del bordo vale per default 0 (1 nel caso di IE).

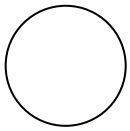
Per applicare un collegamento ipertestuale ad una immagine, occorre inserire il tag dell'immagine all'interno del tag di link: **<A HREF = "http://www.vallauri.edu"> <IMG SRC = "img.jpg"> </A>**

Mappe Immagine

Consentono di creare collegamenti ipertestuali multipli a partire da una unica immagine, suddivisa in diverse aree sensibili. L'angolo superiore sinistro dell'immagine rappresenta il punto di coordinate (0,0). L'asse delle Y è rivolto verso il basso dell'immagine. . Sono consentite aree sensibili di forma:

- Rettangolare: occorre specificare le coordinate dell'angolo superiore sinistro **e** dell'angolo inferiore destro
- Ovale: occorre specificare le coordinate del centro ed il raggio
- Poligonale generica: occorre specificare le coordinate di ciascun angolo della poligonale.





C = 20, 65 px
R 10 px
RMIN = 10 px

Per definire una mappa immagine occorre specificare l'attributo **USEMAP** all'interno del tag IMG:
<**IMG** SRC = "filename.jpg" **USEMAP = "#mappa"**>
in cui mappa è un target impostabile mediante l'attributo **NAME** del tag **MAP**:

<**MAP** **NAME = "mappa"** TITLE="Tooltip replicato su tutte le aree sensibili della mappa">
<AREA SHAPE = "RECT" COORDS = "10,10,120,40" HREF = "pag1.htm">
<AREA SHAPE = "CIRCLE" COORDS = "20,65,10" HREF = "pag2.htm" >
<AREA SHAPE = "POLY" COORDS = "78,309,183,255," TITLE="Tooltip relativo alla singola area">
</**MAP**>
Per creare rapidamente aree sensibili si può utilizzare il sito **image-map**

NB Quando si usano le mappe immagine andando a leggere la posizione delle mappe su Paint, poi **NON** bisogna impostare una width all'immagine tramite html, altrimenti le mappe vanno fuori posizione.

Tabelle

<TABLE>	**</TABLE>**	Inizio e Fine di una tabella. All'interno saranno racchiuse tutte i tag sulla tabella
<TR>	**</TR>**	Inizio e Fine di una riga. Una riga può contenere celle di dati o celle di intestazione
<TH>	**</TH>**	Inizio e Fine di una cella di intestazione con testo grassetto e centrato.
<TD>	**</TD>**	Inizio e Fine di una cella di dati con testo normale allineato a sinistra

Attributi di Table:

WIDTH	Larghezza complessiva della tabella, compresi BORDER e CELLSPACING (omettere **px**) Il WIDTH viene solitamente espresso come % rispetto alla larghezza della pagina: WIDTH = "80%". Sono ammessi anche valori con la virgola: WIDTH ="**33.3**"
HEIGHT	Altezza complessiva della tabella. L'altezza delle tabella non viene quasi mai impostata, ma risulterà dalla somma dell'altezza delle singole righe
ALIGN	Allineamento della tabella (**LEFT**, CENTER, RIGHT) rispetto alla pagina
BORDER	Spessore del bordo **esterno** della tabella mentre il bordo **interno** ha spessore fisso " 1 Il default è **BORDER="0"** con il quale spariscono **sia** il bordo esterno **sia** quelli interni.
CELLPADDING	Crea una spaziatura interna alla cella tutto intorno al testo. Il valore di CellPadding funge in pratica da margine superiore, inferiore, destro e sinistro del testo all'interno della cella.
CELLSPACING	Spaziatura in pixel fra le celle
COLS	Numero di colonne della tabella
BGCOLOR	Colore di sfondo dell'intera tabella

Non esiste un attributo per settare il font-color di una intera tabella, ma occorre racchiudere la tabella dentro un tag <FONT>. Nulla si può fare sull'intera riga. E' invece possibile impostare un <FONT> dentro una singola cella.

Attributi delle Righe TR *(Il tag TR **non** dispone dell'attributo WIDTH)*

ALIGN	Allineamento del testo in tutte le celle costituenti la riga (LEFT, CENTER, RIGHT, JUSTIFY)
VALIGN	Allineamento verticale del testo nella riga (TOP, MIDDLE, BOTTOM)
HEIGHT	Altezza della riga. Può essere una % della height della tabella. ES HEIGHT = "25%"
BGCOLOR	Colore di sfondo della riga

Attributi delle Celle TH e TD

WITDH	Larghezza della singola cella. Può essere scritta come % rispetto alla width della tabella
HEIGHT	Altezza della singola cella. Può essere una % della height della tabella. HEIGHT = "25%"
ALIGN	Allineamento del testo nella singola cella (LEFT, CENTER, RIGHT, JUSTIFY)
VALIGN	Allineamento verticale della singola cella (TOP, MIDDLE, BOTTOM)
BGCOLOR	Colore di sfondo della singola cella
BACKGROUND	Immagine di sfondo della singola cella.
COLSPAN	Numero di colonne su cui estendere la cella.
ROWSPAN	Numero di righe su cui estendere la cella. Utili per intestazioni di righe o di colonne.
NoWrap	Il testo all'interno della cella non va a capo automaticamente.

Note

- **Un tag <TABLE> può contenere al suo interno SOLTANTO tag <TR>**
- **Un tag <TR> può contenere al suo interno SOLTANTO tag <TD> o <TH>**
- **Una tabella deve sempre mantenere una struttura regolare** (stesso numero di celle pero ogni riga)
- **Non sono ammesse su una stessa colonna celle a larghezza differente** (tutte le celle di una colonna devono avere la **stessa larghezza**, al più possono estendersi su più colonne o più righe.
- La ripartizione dello spazio fra e varie colonne viene fatta sulla base del testo interno.
  Meglio sempre specificare su ogni cella <**TH**> una larghezza percentuale ben definita a somma 100.
- Se non ci specifica la larghezza dell'ultima cella, questa si estenderà in automatico.
- Se si specificano altezze diverse per celle appartenenti alla stessa riga, viene applicata l'altezza della prima cella

***Ulteriori NOTE*:**

1) Width e Height della tabella sono prioritarie rispetto alle dimensioni delle singole celle. Se si imposta Width sulla tabella, la larghezza delle singole celle viene automaticamente dimensionata in modo da coprire l'intera tabella. Se si imposta una larghezza specifica per ogni singola cella, la somma deve coincidere con la larghezza della tabella, altrimenti una o più celle vengono automaticamente ridimensionate.
2) E' possibile inserire una immagine all'interno di una singola cella usando il tag IMG. **In questo caso, la larghezza della cella non potrà MAI diventare inferiore alla larghezza dell'immagine. Anche riducendo le dimensioni della finestra la cella continuerà a mantenere la sua dimensione minima.**

Esempio 1: utilizzo di COLSPAN e ROWSPAN

Promossi	**Voti**	
	Mario Rossi	29
	Giuseppe Verdi	28
Respinti	Matteo Bianchi	16
	Marco Galli	14

```
<table width="60%" align="center" border="1" >
     <tr>
            <th rowspan="3">Promossi</th>
            <th colspan="2">  Voti  </th>
     </tr>
     <tr>
            <td> <a href="#">Mario Rossi</a> </td>
            <td> 29 </td>
     </tr>
     <tr>
            <td>Giuseppe Verdi </td>
            <td> 28 </td>
     </tr>
     <tr>
            <th rowspan="2">Respinti</th>
            <td>Matteo Bianchi</td>
            <td>16</td>
     </tr>
     <tr>
            <td>Marco Galli</td>
            <td>14</td>
     </tr>
</table>
```

Esempio 2 di Creazione di una barra di navigazione

Il tag <table> può essere sfruttato per allineare degli oggetti, ad esempio una barra di navigazione:

```
<table width="300" align="center" border="1" >
     <tr align="center">
          <td width="33%"> <a HREF ="index.html">   Home  </a> </td>
          <td width="33%"> <a HREF ="pagina2.html">  Pagina 2  </a> </td>
          <td width="33%"> <a HREF ="pagina3.html">  Pagina 3  </a> </td>
     </tr>
</table>
```

Template completo di una tabella

All'interno di una tabella bisognerebbe sempre utilizzare due tag intermedi

- **thead** per righe di intestazione
- **tbody** per righe relative ai dati

Sintassi completa:

```
<table>
    <caption>
        <p>I miei dati</p>
    </caption>
    <thead>
        <tr><th>Colonna 1</th><th>Colonna 2</th></tr>
    </thead>
    <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>
    <tfoot>   <!-- L'ordine è comunque irrilevante -->
        <tr><td>Totale 1</td><td>Totale 2</td></tr>
    </tfoot>
</table>
```

I miei dati

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

CAPTION
THEAD
TBODY
TFOOT

In caso di stampa di tabelle di grandi dimensioni, **`header`** e **`footer`** vengono ripetuti all'inizio e alla fine di ogni singola pagina. **`caption`** invece viene stampata SOLTANTO sulla prima pagina

Il tag <pre>

Consente di inserire testo Preformattato, ovvero il testo viene visualizzato dal browser esattamente come è scritto all'interno della pagina HTML. **Sente cioè gli "spazi" e gli "a capo"**.
Al suo interno i tag html possono essere utilizzati nel modo usuale.

```
<pre>
    Uno
    Due
    Tre
</pre>
```

```
<pre>   o
      /|\
       |
      / \  </pre>
```

Moduli e Controlli

Scopo dei **controlli** è quello di trasmettere dei dati ad un server.
Vengono di solito inseriti all'interno di un apposito tag Form. **<FORM>** **</FORM>**

La maggior parte dei controlli viene gestita tramite il tag **<INPUT>** che è un tag INLINE e non ha tag di chiusura.
L'attributo **TYPE** identifica il tipo di controllo:

`<input type="text">`	Casella di testo (**Text Box**) // **default**
`<input type="password">`	Casella di testo il cui contenuto viene visualizzato tramite pallini neri
`<input type="number">`	Casella di testo che accetta solo numeri. E' possibile delimitare il range di inserimento mediante gli attributi MIN e MAX
`<input type="checkbox">`	Check Box a scelta multipla
`<input type="radio">`	Radio Button mutuamente esclusivi (se hanno lo stesso name)
`<input type="file">`	Casella di testo che consente di selezionare un file utilizzando il tipico pulsante **SFOGLIA** di windows
`<input type="hidden">`	Casella di testo nascosta. Utile per passare dati al server senza visualizzarli
`<input type="reset">`	Button che cancella il contenuto dei controlli presenti nella form corrente
`<input type="submit">`	Button che invia al server il contenuto dei controlli presenti nella form
`<input type="button">`	Button per interagire con la pagina e richiamare procedure javascript
`<input type="image">`	Pulsante submit di tipo grafico. **SRC**="img.jpg"

Attributi comuni ai diversi controlli

NAME: Identifica la chiave con cui i valori verranno trasmessi al server
VALUE: rappresenta il **contenuto del controllo** che verrà trasmesso al server al momento del submit

disabled
Booleano. Consente di disabilitare un controllo, nel senso che viene modificate la sua impostazione grafica (diventa grigio) e non risponde più agli eventi. Vale **SOLO** per i controlli

required (html5)
Attributo booleano che **rende obbligatoria la compilazione dell'elemento a cui è applicato**.
Il controllo viene eseguito al momento del submit. Se il campo non è stato compilato verrà segnalato un errore. Si applica SOLO sui tag INPUT e NON sulla textArea, sulla quale NON sortisce effetti.

placeholder (html5)
Testo visualizzato in grigio all'interno di un tag input, o di una textarea, **fino a quando il campo è vuoto e non riceve il focus** (tramite il click o spostamento tramite tasto Tab).
Rappresenta un aiuto a ciò che l'utente dovrà scrivere nel campo.
Il placeholder ricompare nel momento in cui il contenuto del campo viene cancellato.

min max step (html5)
min e **max** descrivono **il valore minimo e massimo consentito** all'interno di un input type=number. Il valore di max deve essere maggiore del valore di min (se indicato).
step definisce la distanza tra un valore e il successivo, cioè la granularità dei valori permessi (a partire dallo zero). Il valore di step deve essere un valore positivo non nullo.
Questi tre attributi si applicano sia ai numeri (number e range) sia alle date

```
<input type="number" name="age" min="18" max="100" step="1">
```

Nota sugli attributi booleani

Per impostare un attributo **Booleano** occorre scrivere soltanto il nome dell'attributo (es CHECKED), omettendo il valore (= TRUE) (html5) oppure ripetere dentro il valore il nome dell'attributo (html4)
Es **checked="checked**". Se il nome dell'attributo viene omesso, l'attributo si intende impostato a FALSE

autocomplete (html5)

Attiva l'auto completamento del campo al valore precedente in caso di ricaricamento della pagina.
Può assumere i seguenti valori:

- **on**: indica che il browser può ricompilare il campo in modo automatico (va bene per campi non particolarmente sensibili)
- **off**: indica che il valore è particolarmente sensibile e che quindi l'utente deve reinserirlo **manualmente** ogni volta
- **in assenza di una assegnazione esplicita** viene utilizzato il valore di default del browser **off** nel caso di Chrome**, on** nel caso di Firefox.
  Per cui Firefox, quando si esegue il refresh di una pagina, mantiene in cache lo stato di tutti i controlli, visualizzando gli stessi valori anche dopo il refresh. Lo scopo è quello di facilitare l'autocomplete ed evitare di dover reinserire i valori già inseriti all'interno di una form di registrazione. Questo effetto può essere abbastanza fastidioso negli esercizi javascript.
  Per disabilitarlo occorre impostare **autocomplete="off"** su tutti i tag input.

INPUT TYPE = "TEXT"

Tipico campo di testo
READONLY True/False. Consente di utilizzare il Text Box in sola lettura
MAXLENGTH numero max di caratteri digitabili. Vale SOLO per input[type=text] e **NON** per input[type=number]
MINLENGTH numero minimo di caratteri richiesti.
SIZE Specifica la **larghezza** del controllo espressa come numero di caratteri.
Un campo può avere dimensioni inferiori (o comunque diverse) rispetto al max numero di caratteri inseribili.
Se la larghezza del campo è inferiore al numero di caratteri digitabili, viene automaticamente attivato lo scorrimento

INPUT TYPE = "PASSWORD"

Uguale identico al precedente però, anziché visualizzare ciò che viene digitato, ogni carattere viene visualizzato mediante un pallino nero

INPUT TYPE = "NUMBER" (html5)

Il tag **input type ="number"** consente di creare **un campo destinato all'inserimento di sole cifre numeriche**. Due freccine laterali consentono di incrementare un numero tra un valor MIN e ed un valore MAX esprimibili mediante gli omonimi attributi. L'utente in realtà, digitando direttamente all'interno del controllo, può inserire ciò che vuole (anche caratteri letterali). Il controllo verrà però eseguito in corrispondenza del submit e, in caso di numero non valido, verrà segnalato un errore che impedirà l'invio dei dati al server.

INPUT TYPE = "CHECKBOX"

CHECKED Se specificato il text box risulterà inizialmente selezionato.
VALUE: Valore trasmesso al server in corrispondenza del submit.
Es <INPUT TYPE = "checkbox" NAME = "chkMaggiorenne" VALUE = "si"> Maggiorenne.
Se l'opzione è selezionata verrà restituita la stringa "chkMaggiorenne = si", altrimenti stringa vuota.
Se il checkbox non ha un value, in corrispondenza dell'invio viene inviato un valore di default pari a **"on"**
REQUIRED Fa si che il singolo check box debba essere obbligatoriamente selezionato (ad esempio per accettazioni delle condizioni di utilizzo).

NOTA: In caso di più checkbox attinenti uno stesso tema (es hobbies), si assegna solitamente **lo stesso nome** a tutti i check box. Nel momento dell'invio al server compariranno nella url più parametri distinti tutti con lo stesso nome. Se in coda al nome si aggiungono delle parentesi quadre, queste NON modificano l'invio dei dati (cioè sarà sempre presente un parametro distinto per ogni controllo) però facilitano la gestione lato server che aggregherà i valori all'interno un unico vettore enumerativo

INPUT TYPE = "RADIO" (radio buttons)

I "radio buttons" con lo stesso name vengono aggregati all'interno di un unico gruppo e diventano **mutuamente esclusivi**
CHECKED Se specificato il radio button risulta inizialmente selezionato.
VALUE: Valore trasmesso in corrispondenza del submit.
REQUIRED messo su un qualunque radio obbliga l'utente a selezionare almeno uno dei radio buttons appartenenti a quel gruppo

Se il checkbox non ha un value, in corrispondenza dell'invio viene inviato un valore di default pari a **"on"**

```
Es   <p> Avete un computer?
     <input type="radio"   name="optComputer"  value="si">   si
     <input type="radio"   name="optComputer"  value="no">   no
     </p>
```

Verrà restituito "**optComputer=si**" oppure "**optComputer=no**"

Il tag LABEL

Tag inline. Applica una etichetta ad un controllo associato ed identificato tramite l'attributo **ID**.
L'attributo **FOR** della Label deve contenere l'ID del controllo associato
In questo modo in corrispondenza del click sull'etichetta **il focus viene automaticamente spostato sul controllo.**

```
<label for="txtNome"> Nome: </label>
<input Type="text"  id="txtNome" />
```

Utilizzato anche per:

- garantire l'utilizzo del form e fornire lettura vocale a utenti non vedenti che navigano tramite screen reader
- strutturare pagine multilingua con testo letto da database

I tag FIELDSET e LEGEND

Il tag **<fieldset>** (già HTML4) è un Panel utilizzato per raggruppare elementi simili all'interno di una form. In visualizzazione il gruppo viene racchiuso all'interno di un rettangolo bordato

Il tag **<legend>** Inserito subito dopo **Fieldset** consente di inserire una Caption al Panel di raggruppamento.

<TEXTAREA> testo da visualizzare </TEXTAREA>

Text Box a righe multiple. All'interno del testo vengono accettati tutti i caratteri: spazi, a capo, segni minore e maggiore. Riconosce il **placeholder** ma **NON riconosce a livello di html l'attributo value**. Un eventuale testo iniziale deve essere scritto all'interno del tag. Attenzione che se il tag di chiusura viene fatto su una nuova riga, questa spaziatura viene interpretata come testo e pertanto impedisce la visualizzazione del placeholder.

COLS: Numero di colonne. TextArea usa un carattere teletype a spaziatura fissa, quindi COLS rappresenta in pratica il numero di caratteri utilizzabili per ogni riga.
ROWS: Numero di righe visibili all'interno della TextArea. Se si eccede compare la barra di scorrimento verticale
WRAP: "on" / "off". Se viene impostato wrap = "off" i caratteri possono estendersi oltre il valore di COLS con comparsa automatica della barra di scorrimento orizzontale (fino a quando non viene premuto INVIO):

Nota: Il testo scritto all'interno del tag rappresenta il **defaultState** del controllo ed è ciò che verrà visualizzato.
Il **value** invece rappresenta il corrispondente **currentState** dinamico ed è utilizzabile **SOLO** dinamicamente da javascript. Se usato nel file html NON sortisce nessun effetto, come se non ci fosse.
In conclusione, nell'html **NON** bisogna utilizzare il value ma solo il valore scritto all'interno del tag.

<SELECT> opzioni </SELECT> List Box / Combo Box

All'interno dei tag SELECT occorre inserire uno o più tag OPTION.

```
<SELECT name="lstColori"
   <OPTION>  Rosso  </OPTION>
   <OPTION SELECTED>  Verde  </OPTION>
</SELECT>
```

Al contrario di checkbox e radio buttons, se la option selezionata non presenta un value, in corrispondenza del submit verrà inviato al server il testo visualizzato all'interno della option ("verde")

Attributi del tag SELECT

PLACEHOLDER: **Non** è riconosciuto.

SIZE: Numero di righe da visualizzare all'interno del List Box (in pratica altezza del List Box).
Utilizzando il valore **1 (default)**, il List Box diventa simile ad un Combo Box con un pulsante di apertura, ma **NON** consente la scrittura diretta. Nella casella di testo viene visualizzato il primo elemento della lista.

MULTIPLE: Consente di selezionare (mediante i tasti Shift e Ctrl) più opzioni all'interno del List Box.
Es <SELECT Name = LstColori MULTIPLE> ……. </SELECT>.
Ogni voce scelta verrà aggiunta alla stringa SUBMIT nel formato LstColori = Colore1 & LstColori = Colore2.
Aggiungendo delle quadre al fondo del name il server restituirà un vettore enumerativo di stringhe (mentre invece il formato della querystrng rimane sempre il medesimo con replica delle voci).

SelectedIndex: Indice della voce attualmente selezionata all'interno del List Box (partendo da 0).
Rappresenta il ***currentState*** dinamico del controllo, pertanto è utilizzabile soltanto in javascript.
Impostando da javascript selectedIndex=-1 compare in testa una riga vuota che scompare automaticamente in corrispondenza della selezione di una voce sul ListBox.

Attributi dei tag OPTION

SELECTED: L'opzione indicata sarà preselezionata all'avvio. Se nessuna opzione viene impostata SELCTED, non ci saranno opzioni preselezionate, e, nel caso del combo, verrà visualizzata la prima opzione della lista.

DISABLED: La singola opzione risulterà disabilitata

VALUE: Valore che verrà restituito nel caso in cui, al momento del SUBMIT, l'opzione risulti selezionata.

Il tag SELECT (a differenza di checkbox e radiobutton) **dispone anche di un attributo `VALUE` riassuntivo che contiene in ogni momento il value della voce attualmente selezionata** (molto comodo).

All'interno del tag SELECT è anche possibile suddividere le OPTION in gruppi differenti:

```
<OPTGROUP label="gruppo1">
   <OPTION>  Rosso  </OPTION>
   <OPTION>  Verde  </OPTION>
</OPTGROUP>
```

Concetto di submit

- Scopo del pulsante di submit è quello di trasmettere al server il contenuto di tutti i controlli presenti all'interno della form.
- In corrispondenza del submit il browser provvede a richiedere al server la pagina html indicata all'interno dell'attributo **action** del tag form. Es `<form action="pagina2.html">`
- Contemporaneamente trasmette al server (**in coda alla url**) il contento di tutti i controlli presenti all'interno del tag form. Il valore dei controlli viene 'passato' al server nel formato **name=value**, per cui tutti i controlli DEVONO avere un **name** e NON essere disabled.
- Come **value**, nel caso degli `<input type=text">` si intende il contenuto inserito dall'utente. In tutti gli altri casi l'attributo **value** deve essere esplicitato direttamente all'interno del tag.
- All'interno del tag `form` è anche possibile specificare un attributo **target** che indica la finestra di visualizzazione dell'Action

Il tag <button> </button>

Da HTML4 in avanti è stato aggiunto anche un TAG **<button>** analogo a **<input type="button">.**

Questo tag però, a differenza del tradizionale tag `<input type="button">` **non** dispone dell'attributo **value**, ma il testo del pulsante va scritto fra il tag di apertura ed il tag di chiusura.

Inoltre il tag <button> presenta due comportamenti di default a seconda di dove viene scritto.

- se il tag è inserito all'interno di una form, equivale a **INPUT TYPE=SUBMIT**
- se il tag **NON** è inserito all'interno di una form, equivale a **INPUT TYPE=BUTTON**

IFRAME

IFRAME è un frame senza FrameSet. Rappresenta in pratica un contenitore **inline** all'interno del quale può essere caricata una seconda pagina HTML indipendente dalla prima. Presenta soltanto gli attributi indicati nell'esempio. Utilizzato per inserire all'interno di una pagina un filmato YouTube o una Google Map.

```
<iframe   name="frame1"
          src="pagina2.html"
          width="80%"
          height="380"
          frameborder="1"
          allowfullscreen        // significativo solo nel caso dei video
          scrolling="no">
</iframe>
```

La pagina indicata all'interno dell'attributo src è quella che viene visualizzata all'apertura della pagina principale.

Per fare in modo che un collegamento ipertestuale apra la risorsa di destinazione all'interno dell'IFRAME occorre indicare all'interno dell'attributo target il nome dell'IFRAME:

```
<a href="pagina2.html.html" target="frame1">
     Apri il link corrente all'interno del frame frame1
</a>
```

Per inserire un filmato you tube all'interno dell'iframe occorre, sul filmato, cliccare sul pulsante **CONDIVIDI**, copiare il codice del filmato, ed inserirlo all'interno dell'attributo **src** dell'iframe nel modo seguente:

```
<iframe width="640" height="420"
        src="http://www.youtube.com/embed/o_cHvtPB2dY"
        frameborder="0"
        allowfullscreen>
</iframe>
```

I META TAGS

I META TAG servono per memorizzare nella sezione di head **informazioni relative al contenuto della pagina**, come ad esempio le informazioni per l'indicizzazione dei motori di ricerca. Non hanno in genere tag di chiusura

Il meta-tag META - due attributi principali :

- **HTTP-EQUIV,** utilizzato per impostare le intestazioni relative alle HTTP Request
- **NAME,** utilizzato per memorizzare contenuti descrittivi della pagina non presenti nell'intestazione HTTP.. Alcuni server però ignorano gli HTTP-EQUIV per cui, allo stato attuale, i due attributi sono utilizzati abbastanza indifferentemente. In entrambi i casi l'attributo CONTENT memorizza il contenuto del meta tag.
- <META **NAME**="*tipo di attributo*" **CONTENT**="*descrizione*">
- <META **HTTP-EQUIV**=" *tipo di attributo* " **CONTENT**=" *descrizione* ">

Elenco dei principali META TAG di tipo NAME

"Keywords" Content="gatti, gatto, cani, cane, addestramento"
Elenco di parole chiave utilizzabili dai motori per aggiornare i loro cataloghi. E' raccomandato il minuscolo.
"Description" Content="Sito che tratta dell'addestramento di cani e gatti"
Descrizione che verrà visualizzata dal motore in caso di esito positivo della ricerca. Max 120 chr
"Author" Content="Nome dell'Autore" Colui che ha creato il sito.

Es

```
<meta name="keywords"   content="gatti, gatto, cani, cane, addestramento">
<meta name="description" content=" Sito che tratta di cani e gatti">
```

Elenco dei principali META TAG di tipo HTTP-EQUIV

"Content-Type" Content="text/html" E' il tag che dreamweaver aggiunge automaticamente a tutte le pagine create. Indica il formato della pagina espresso nel tipico formato MIME.
"Content-Language" Content="it-IT" Linguaggio utilizzato nel contenuto della pagina: italiano - Italia
"Expires" Content="data ora"
Data Ora assoluta o relativa oltre la quale la pagina è da considerare obsoleta. In corrispondenza della scadenza i proxy elimineranno la pagina dalla cache. Il valore 0 significa in pratica no-cache.
"Pragma" Content="**no-cache**" Avvisa i client (proxy e browser) che la pagina non deve essere messa in cache
"Cache-Control" Content="Private | Public | No-cache | No-store" E' la versione HTTP1.1 del precedente
Private = La pagina può solo essere salvata in cache private
Public = La pagina può solo essere salvata in cache pubbliche
No-cache = non può essere messa in cache
No-store = può essere messa in cache ma non archiviata.
"Refresh" Content="4" *oppure* Content="4; url=newPage.htm"
Provoca un refresh della pagina ogni 4 secondi (come se si cliccasse sul pulsante Aggiorna", oppure, in corrispondenza del Refresh, esegue un Redirect sulla nuova Pagina che viene caricata al posto della corrente.

Es

```
<meta http-equiv="Cache-control" content="no-cache">
<meta http-equiv="Pragma" content="no-cache">
```

Il meta-tag LINK

Consente di collegare la pagina corrente d una risorsa esterna. Utilizzato soprattutto per collegare la pagina con il relativo **foglio di stile**. Due attributi obbligatori : **rel** indica il tipo di risorsa a cui sta accedendo, **href** indica la destinazione: Esempio :

```
<link rel="stylesheet" href="mioFile.css">
<link rel="icon"  href="myIcon.ico" type="image/ico" size="16x16">
```

Gli attributi `type` e `size` sono facoltativi. E' supportato anche il formato png

Introduzione ad HTML 5

Specifiche ufficiali definite dal w3c [World Wide Web Consortium] del **17 dicembre 2012**
http://www.whatwg.org/specs/web-apps/current-work/multipage/elements.html
http://www.w3schools.com – *Reference Completo di tutti i tag HTML 5 conformi al w3c*

Fra le caratteristiche più importanti di HTML5 si possono elencare:

1. introduzione di nuovi **input type** per il l'inserimento dei dati all'interno di una form
2. introduzione delle **Media Query** per la realizzazione di un responsive design
3. introduzione di elementi specifici per il controllo di **contenuti multimediali** (tag <**audio**> e <**video**>) in forma nativa (cioè tramite codice scritto direttamente all'interno del browser)
4. un nuovo **Content Model** per la **strutturazione della pagina** mediante nuovi tag.
   HTML4 non fornisce nessun strumento adatto a consentire una corretta gestione e classificazione del contenuto obbligando gli sviluppatori a ripiegare su strutture anonime, quali e , arricchite di valore semantico con l'utilizzo di classi e attributi
   HTML5 introduce invece una serie di tag che assumono un valore semantico, consentendo di strutturare meglio la pagina e creare una suddivisione sempre più netta fra **contenuti** e **aspetto grafico** demandato ai CSS

- introduzione del tag **Canvas** che permette di utilizzare javaScript per creare animazioni e grafica vettoriale (anche 3D)
- introduzione di **Html5storage** per la memorizzazione locale di **piccole quantità di dati** gestiti dal browser all'interno del disco locale.

- introduzione di supporto alla **geolocazione**, dovuto alla forte espansione di sistemi mobili (cioè strumenti per accedere alla posizione geografica del device dell'utente;
- rinvigorimento delle **API JavaScript** che vengono estese per supportare tutte le funzionalità di cui una applicazione moderna potrebbe aver bisogno

Sintassi semplificata per la scrittura dei TAG

La sintassi HTML5 si caratterizza per una spiccata **flessibilità e semplicità** di implementazione. Sotto questo aspetto HTML 5 è più vicino ad HTML 4 che non ad XHTML, per cui cadono gran parte delle restrizioni imposte da XHTML che, per una migliore leggibilità, restano comunque consigliate:

- sostituzione del lungo e complesso doctype, con un semplice **<!doctype html>**
- Per i tag che non hanno tag di chiusura (es **img**) non c'è l'obbligo della chiusura abbreviata />.
- **Attributi booleani**: non è più necessario definire un valore per gli attributi booleani, basta il nome. L'assenza dell'attributo significa FALSE.

```
XHTML: <input checked="checked" />
HTML5: <input checked />
```

- Quando il browser può determinare univocamente i limiti di operatività di un tag, non è obbligatorio scrivere il tag di chiusura (es) e talvolta neppure quello di apertura (head, html, e body diventano facoltativi)
- Non è obbligatorio racchiudere i valori degli attributi tra virgolette, Il valore dell'attributo può essere scritto senza virgolette, con virgolette singole oppure con virgolette doppie

Tag soppressi

HTML 5 riduce drasticamente il numero di tag e attributi mirati alla sola rappresentazione grafica dei contenuti, rappresentazione che passa completamente "in carico" ai CSS.
Questi tag e attributi deprecati mantengono validità formale solo per preservare la retrocompatibilità di pagine datate, ma sono espressamente sconsigliati nella creazione di nuovi documenti.

I principali tag deprecati in HTML 5 sono:

- **font**
- **center**
- **u** (underline) **s** (barrato) CSS equivalenti: `text-decoration: underline/line-through`
- **strike** (uguale a s), **big e small** (incrementa / decrementa di 1 il font rispetto al genitore
- **tt** (teletype a spaziatura fissa).
- Tutti i tag relativi ai frames: **frame, frameset e noframes**.
- Il tag **acronym** (sostituito dal più comune **abbr**) e **applet** (sostituito da **object**),

Sono invece stati mantenuti i tag come i e b; raccomandandone però l'uso solo come ultima risorsa.

I principali attributi deprecati in HTML 5 sono:
align, valign, background, bgcolor, border, cellpadding, cellspacing.

1. Nuovi INPUT TYPE e relativi Attributi

HTML5 introduce nuovi controlli di tipo **input** che, se inseriti all'interno di una form, consentono di eseguire la validazione del campo unicamente tramite HTML, senza dover ricorrere a Javascript

- nuovi **input type** : number, **date, color, range, email**,
- nuovi **attributi** : **required, placeholder, min, max**, step, **autocomplete, pattern, autofocus, multiple**

pattern

L'attributo pattern, se specificato, deve contenere una **espressione regolare valida**.
Nel solo caso dell'attributo pattern, in caso di errore viene visualizzato il contenuto dell'attributo **title** (tool tip). In tutti gli altri casi il messaggio visualizzato è fisso. Esempio:

```
<input type="text" name="nickname" pattern="[A-Z][a-zA-Z_]{7,11}">
```

Stringa di lunghezza compresa tra 8 e 12 caratteri avente iniziale maiuscola ed i rimanenti 7/11 caratteri costituiti da lettere maiuscole, oppure lettere minuscole, oppure underscore.

autofocus

attributo booleano serve **a** impostare il focus su uno specifico elemento **del form** quando la pagina viene caricata. Esempio tipico è la home page di Google: quando viene caricata il focus è automaticamente impostato sul campo per la ricerca. Equivale alla seguente riga JavaScript:

```
document.getElementById("myid").focus();
```

novalidate Attributo booleano che si applica SOLO al tag form e permette di disabilitare tutte le validazioni dei tag interni alla form.

multiple

L'attributo multiple è un booleano che consente di **inserire più valori per lo stesso input,** ad esempio consente di inserire più indirizzi mail all'interno di un unico Text Box con la **virgola** come separatore (la virgola è un carattere non valido all'interno di una mail)

Nuovi Input type

color

Il tag **`input type="color"`** dovrebbe creare un color picker, cioè un widget utile per la **selezione di un colore a partire da una palette di colori**.
Una volta selezionato il colore, il campo passa alla nostra pagina di ricezione un colore RGB esadecimale composto da 6 cifre. Nei Browser che non lo supportano, viene generato un normale input di testo. `<input type="color" name="txtColore">`

datetime

I tag di tipo `datetime` sono 6.

`<input type="date" min="2020-01-01" max="2020-12-31"`
Consente di scegliere la data tramite l'apertura di un apposito widget.
E' possibile limitare la scelta fra un valore iniziale ed un valore finale

`<input type="datetime">`
Consente l'inserimento di data e ora in un solo colpo. Vengono aperti due widget, uno di tipo datepicker per la selezione del giorno ed un altro per la selezione dell'ora. **Deprecato**

`<input type="datetime-local">`
E' simile a datetime, ma l'ora viene considerata sul fuso orario locale. Non accetta min e max

`<input type="time" >`
Consente l'inserimento della sola ora senza la data

`<input type="month" >`
Consente la scelta del mese

`<input type="week" >`
Consente la selezione di una determinata settimana dell'anno (numero di settimana da 1 a 53).

Datalist

Il tag **<datalist>** realizza una lista di voci che, abbinata ad un `<input type=text>`, consente di realizzare un **auto completamento del testo** inserito. All'interno del `datalist` si possono definire dei tag **`option`**, esattamente come per il listbox. Esempio:

```
<input type="text" name="mood" list="lstStati">
<datalist id="lstStati">
      <option value="triste">  </option>
      <option value="annoiato">  </option>
      <option value="felice">  </option>
</datalist>
```

Il `datalist` realizza di fatto un vero **combo Box** (il tag **`select`** consente sì la scelta fra le voci di una lista, ma NON consente all'utente la possibilità di inserire un nuovo valore non presente nella lista).

Può essere abbinato al value di qualunque input type, ad es `<input type range>`

range

Semanticamente simile all'input type=number, questo nuovo tipo di input permette agli utenti di **inserire un numero tramite uno slider**.

```
<input type="range" name="voto"  min="0" max="5" step="1">
```

email

Il tag **`input type ="email"`** consente di creare **un campo per inserire un indirizzo mail**.
Una fondamentale condizione di validità, dunque, sarà rappresentata dalla presenza del simbolo @. Nel caso in cui il valore inserito non sia valido viene sollevata una eccezione.

tel

Il tag **`input type="tel"`** consente di creare un campo adatto all'inserimento di numero di telefono. A differenza degli input di tipo email e url, questo tipo **non impone un particolare formato**, questo perché, a livello intenzionale, i numeri possono essere scritti in diversi modi.
È comunque possibile usare l'attributo pattern per forzare un determinato formato.

url

Il tag **`input type="url"`** consente di creare **un campo destinato all'inserimento di una url** assoluta, ovvero nel formato http://www.sito.com/etc.... https://www.sito.com/etc
Nel caso in cui il valore inserito non sia valido, viene sollevata una eccezione.

search

Il tag **`input type="search"`** consente di creare un campo di ricerca. Questo campo è, ovviamente, un campo libero nel senso che non impone nessun pattern. In alcuni browser (Safari) il campo di ricerca viene visualizzato con bordi arrotondati. Es:

```
<label> Parola chiave:
   <input type="search" autocomplete="on" name="keyword"
                required maxlength="50" placeholder="keyword da cercare">
</label>
<input type="submit" value="ricerca">
```

command

Attualmente non supportato. Definisce un pulsante di comando che può essere un **command** button, **radio** button o **checkbox**.

```
<command type="command" onclick="save()">Save</command>
<command type="radio" radiogroup="alignment" >Left</command>
```

L'attributo **`radiogroup`** consente di creare gruppi indipendenti di radio button
L'attributo **`label`**, nel caso del command, sembra sostituire l'attributo **`name`**

`<mark>` `Testo evidenziato a sfondo giallo / Altro colore` **`</mark>`**
Tag inline equivalente a **span** a cui viene applicato un `background-color giallo`.
Utilizzato ad esempio per evidenziare la parola chiave cercata tramite un motore di ricerca.

`<progress>` // Tag inline che realizza una progress bar. Il valor minimo non è modificabile e vale 0. Se non si imposta value assume automaticamente una animazione di avanti / indietro (chrome)

```
<progress max="10" value="6"  >
      Testo visualizzato dai browser che non supportano il tag
</progress>
```

`<meter>` Simile alla precedente. Presenta MIN, MAX e altri due attributi che sono LOW e HEIGHT che rappresentano le soglie di un range predefinito. All'interno di questo range la barra di scorrimento assume un colore differente. Ad esempio per l'altezza di una persona si può importare un range di normalità compreso tra 1,60 e 1,90. Al di fuori di questo range l'altezza è da considerarsi anomala.

```
<meter min="100" max="220" value="6" low="160" high="190">
      Testo visualizzato dai browser che non supportano il tag
</meter>
```

`<figure>` E' un contenitore in cui possiamo racchiudere dei contenuti aggiuntivi / indipendenti rispetto al contenuto principale, ad es per annotare illustrazioni, schemi, foto, elenchi di codice, etc.
Il coda al contenitore è possibile, opzionalmente, aggiungere una didascalia (<figcaption>).

```
<figure>
   <img src="foto.jpg">
   <figcaption> Foto dimostrativa © Diritti riservati </figcaption>
</figure>
```

<output> Tag inline che restituisce il risultato di un calcolo

```
<form onInput="x.value=parseInt(a.value)+parseInt(b.value)">
   <input type="number" name="a" value="50">+
   <input type="number" name="b" value="50">=
   <output name="x" ></output>
```

<address> serve **per identificare un indirizzo email.** Ad esempio per racchiudere le informazioni di contatto dell'autore del documento. Il suo contenuto viene visualizzato dai browser in corsivo.

```
<address> <a href="mailto:xxx@gmail.com">xxxxx </a> </address>
```

<time> **A**ttualmente non supportato dai browser. Serve per **identificare una data/ora** di un certo blocco di testo (ad es data / ora di pubblicazione di un articolo). E' previsto anche il suo utilizzo da parte dei motori di ricerca per il riconoscimento temporale degli articoli (più recenti maggior peso).

```
<time datetime="2012-11-22" > Lunedì 22 Novembre </time>
<p>sarò lì per le <time> 15:00 </time> </p>
```

<details> E' un contenitore in cui possono essere inserite informazioni supplementari o controlli aggiuntivi. I contenuti dell'elemento <details> possono essere mostrati o meno dal browser grazie all'attributo **open**, di tipo booleano. Attualmente non supportato dai browser.

<keygen> Generatore di chiavi pubbliche/private all'interno di una form. Quando si effettua l'invio di un form contenente il tag <keygen>, la chiave privata viene memorizzata nel keystore locale e la chiave pubblica viene confezionato e inviata al server.

<menu> Attualmente non supportato dai browser. Lo scopo è quello di creare menù tipo desktop.

```
<menu type="toolbar">
  <li>
    <menu label="File">
       <button type="button" onclick="file_new()">New...</button>
       <button type="button" onclick="file_open()">Open...</button>
    </menu>
  </li>
  <li>
    <menu label="Edit">  ……  </menu>
  </li>
```

<wbr> Inserito all'interno di una parola piuttosto lunga, indica la posizione migliore in cui andare a capo

Nuovi Attributi Globali

HTML5 aggiunge diversi **attributi globali** (cioè che possono essere applicati a qualsiasi tag HTML) fra cui:

accesskey specifica una combinazione di tasti veloci per spostare il focus sul tag

contentEditable Interessante nuovo attributo globale di HTML5. Se impostato a true *su un qualsiasi tag* lo rende editabile runtime dal browser. Lo stesso destino subiscono tutti gli elementi in esso contenuti a meno che non espongano un esplicito *contenteditable=false*. Esempio:

```
<p contenteditable="true"> bla bla bla</p> <br>
<a href="#" onClick="document.execCommand('bold');"> Neretto </a>
```

Il comando **document.execCommand**('bold') consente di applicare un effetto al **Testo Selezionato**

2. Media Query e Responsive Design

In HTML 5 / CSS3 è possibile utilizzare stili differenti a seconda del dispositivo in cui la pagina verrà visualizzata. Si parla in questo caso di **Responsive Layout**.

A tal fine sono disponibili due sintassi equivalenti:

- L'attributo HTML5 `<link rel="stylesheet" href="screen.css" media="screen" >`
- La @rule CSS3 `@media screen { p {color:blue} }`

che presentano la stessa identica sintassi e consentono di realizzare le cosiddette "media query".
Il primo consente di definire un file CSS diverso per ogni tipo di dispositivo,
Il secondo consente di scrivere tutto all'interno di uno stesso file .css senza dover riscrivere le stesse cose più volte. Molto più comodo. In entrambi i casi le proprietà CSS vengono applicate solo se le condizioni dell'espressione **media** risultano vere.

I Media Type base

All'attributo **media** è possibile assegnare un cosiddetto **Media Type base** che si scrive **senza parentesi tonde** e può assumere uno dei seguenti valori:

`all` (default, equivale a tutti i dispositivi di visualizzazione)
`print, screen, speech` (sintesi vocale)

I Media Features

Oltre ai media type precedenti, all'interno dell'attributo **media** è possibile utilizzare anche i cosiddetti **Media Features** i cui valori devono essere scritti **con le parentesi tonde** in formato nome:valore.
Es **(max-width:480px)**

Pricipali Media Features

width	larghezza della **finestra** corrente
height	altezza della **finestra** corrente
aspect-ratio	rapporto tra larghezza e altezza della finestra
device-width	larghezza dello **schermo** fisico
device-height	altezza dello **schermo** fisico
device-aspect-ratio	rapporto tra larghezza e altezza dello schermo
orientation	**landscape**(orizzontale) e **portrait** (verticale)

Nel caso di smartphone e tablet i due gruppi coincidono. Nel caso invece dei computer può invece verificarsi che la **width** della finestra corrente sia inferiore rispetto alla **device-width** dello schermo.

Con il termine **width** si intende sempre la dimensione orizzontale del dispositivo, mentre con il termine **height** si intende sempre la dimensione verticale.
Se **width** > **height** il telefono avrà una orientation **landscape**.
Se **height** > **width** il telefono telefono avrà una orientation **portrait**.

Differenze:

- device-width e device-height sono in genere utilizzate quando l'applicazione viene visualizzata via web su PC, tablet, e smartphone. **Per i test si può restringere la finestra del browser**.
- width e height invece sono in genere utilizzate per realizzare le app ibride (sono utilizzate ad esempio da **cordova** per il build di una applicazione).
  Per i test si può utilizzare la 'Device Toolbar' all'interno dell'inspector del browser.

Chrome Device Toolbar

Il 2° pulsantino dell'inspector, denominato appunto **device toolbar,** consente di verificare come l'applicazione verrà visualizzata su vari tipi di smartphone

Per verificare la piena responsività di una applicazione occorre eseguire i test:

- sia ridimensionando la finestra del browser (test per PC)
- sia utilizzando la Device Toolbar per emulare il funzionamento su vari telefoni

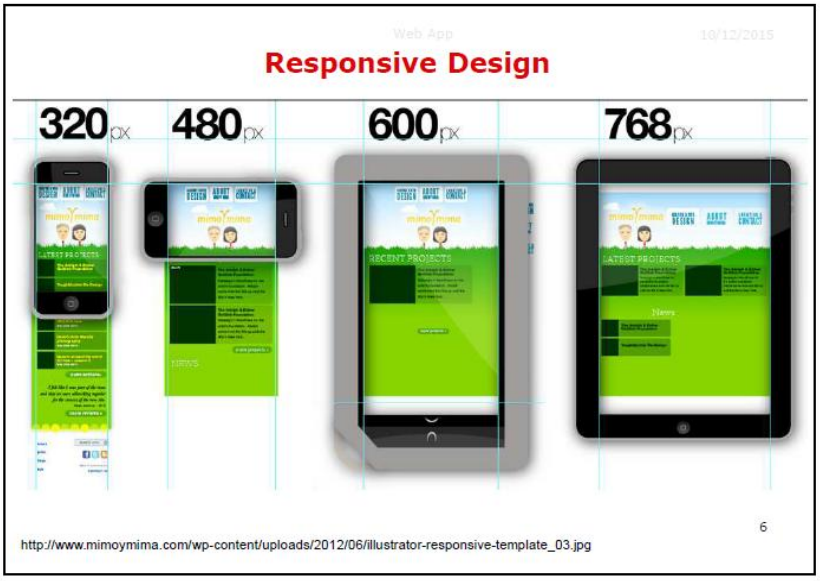
Sintassi delle media query

Poiché le proprietà di stile possono essere solo assegnate (es `width:800px`) e non possono essere utilizzate con gli operatori > e <, sono state definite altre featurs più flessibili quali:

`max-width`, `min-width`, `max-height`, `min-height`
`max-device-width` `min- device-width` `max-device-height` `min-device-height`

- **`max-width:600px`** significa **`finestre <= 600px`**
- **`min-width:200px`** significa **`finestre >= 200px`**

Si supponga di voler realizzare una app che utilizzi la seguente visualizzazione a seconda dei contesti:



Impostazioni CSS

Si utilizza la direttiva **@media** (scritta senza il segno di uguale e senza virgolette):

```
@media screen and (min-width:768px) {  }   // finestre > 768
@media screen and (max-width:768px) {  }   // tablet
@media screen and (max-width:600px) {  }   // tablet piccoli
@media screen and (max-width:480px) and (orientation: landscape){} //tel h
@media screen and (max-width:320px) and (orientation:portrait){}   //tel v
```

- Per la scrittura si parte sempre dai dispositivi più grandi andando via via verso quelli più piccoli **L'impostazione successiva maschera quella precedente**
- All'interno della stessa Media Query è possibile impostare più condizioni tramite **and** e **not**.
- Più Media Query possono essere scritte all'interno di una stessa riga e suddivise tramite una **virgola** che ha il significato di **OR.** Non è possibile intersecare le Media Query suddivise da virgola con le and. La virgola consente soltanto di impostare più Media Query su una stessa riga. Es:
  `@media screen and (max-width:600px), screen and (max-device-width:600px) {  }`

Nota: *Se si lasciano dei "buchi" cioè se ad esempio si definisce una Media Query per dispositivi con* ***width>1000*** *ed un'altra per i dispositivi con* ***width<800,*** *per i dispositivi di dimensione intermedia (width tra 800 e 1000) non verrà applicato nessun stile specifico.*

Impostazioni HTML

```
<link rel="stylesheet" href="index.css">  // impostazioni comuni
<link media="screen and (min-width:768px)"
    rel="stylesheet" href="pc.css">      // impostazioni aggiuntive per PC
<link media="screen and (max-width:768px)"
    rel="stylesheet" href="tablet.css"> // impostazioni aggiuntive per tablet
<link media="screen and (max-width:600px)"
    rel="stylesheet" href="phones.css">  // tablet piccoli
<link media="screen and (max-width:480px)  and  (orientation:landscape)"
    rel="stylesheet" href="horizontalPhones.css" >
<link media="screen and (max-width:320px)  and  (orientation:portrait)"
    rel="stylesheet" href="smallPhones.css" >
```

Altre possibili impostazioni

```
<link media="print and (min-resolution: 1000dpi)" ……..>
<link media="screen and (monochrome)" ……..>
<link media="screen and (not color) and (device-aspect-ratio: 16/9)" ……..>
<link media="screen and (color), print and (color)" ……..>
```

Il meta tag viewport

Consente di impostare come verrà aperta la pagina sul dispositivo. Presenta i seguenti attributi :

- **width** che consente di impostare le dimensioni che dovrà avere la finestra del browser in corrispondenza dell'apertura della pagina. Si può impostare un valore in pixel oppure utilizzare lo speciale valore **device-width** che rappresenta la larghezza effettiva dello schermo dell'utente.
- **initial-scale** definisce il livello di zoom con cui la pagina verrà caricata la prima volta. Il valore 1 significa a tutto schermo senza ridimensionamenti.
  `<meta name="viewport"  content="width=device-width, initial-scale=1">`
  Peraltro `device-width` e `1` dovrebbero essere entrambi valori di default.

3. Il tag audio

Permette il caricamento di un file audio (in formato mp3 wav ogg) direttamente dal codice html della pagina senza rivolgersi a plugin esterni. Esempio

```
<audio src="myFile.mp3" type="audio/mp3" controls autoplay muted loop>
    Your browser does not support the audio tag
</audio>
```

src è il riferimento al contenuto multimediale da riprodurre

type rappresenta il mime type del file da riprodurre

controls consente di visualizzare i pulsanti standard per il controllo dell'audio.
In assenza di control non viene visualizzato nulla.

autoplay consente di avviare l'audio automaticamente al caricamento della pagina.
Nelle versioni più recenti, i browser non consentono più l'autoplay di contenuti multimediali, che:

- Nel caso di **Firefox** è consentito aggiungendo l'opzione **muted** (disattivazione audio) in coda ad autoplay
- Nel caso di **Chrome** è consentito soltanto per i video (sempre aggiungendo l'opzione **muted**)

loop il brano verrà ripetuto automaticamente dopo ogni terminazione. Oppure loop=3

preload="none"/"auto" Il valore auto forza il browser a scaricare l'intero file prima di iniziare la riproduzione. In caso di impostazione dell'attributo *autoplay*, l'attributo *preload* viene ignorato.

All'interno del tag si può indicare un testo da visualizzare nel caso il browser non supporti il tag audio.

Il tag <source>

Il tag <audio> supporta anche la possibilità di gestire più di un sorgente. in tal caso si omette l'attributo src e si annidano più tag <source> all'interno del tag <audio>. Il tag **<source>** consente di definire risorse alternative per gli elementi multimediali in base alla capacità di riproduzione del browser.

Il browser cercherà di riprodurre il primo file audio. Se questo per qualche motivo dovesse fallire o fosse in un formato non supportato dal browser, automaticamente passerà al file audio successivo.
L'attributo **type** indica il MIME type del documento. Es `type="audio/mpeg"`.

```
<audio controls>
   <source src="horse.ogg" type="audio/ogg">
   <source src="horse.mp3" type="audio/mp3">
   Your browser does not support the audio tag.
</audio>
```

Formati supportati dai principali browser

Browser	Ogg Vorbis	MP3	WAV
FireFox 3.6+	✓		✓
Safari 5+		✓	✓
Chrome 6	✓	✓	
Opera 10.5+	✓		✓
Internet Explorer 9 (beta)		✓	✓

Perchè Firefox non supporta MP3?

The MP3-compression algorithm is patent-protected by the Frauenhofer Institute IIS (iis.fraunhofer.de). If they would do that, they could no longer distribute Firefox for free. The better question is: Why don't Apple & Microsoft support ogg vorbis, which is (and always has been and always will be) a completely free file format, with quality and compression just as good as mp3, if not better...

In realtà Firefox ha iniziato a supportare mp3/mp4 a partire dalla versione **71** (3 dicembre 2019)

PlayList possono essere create soltanto tramite javaScript.

Il tag video

Sintassi simile al tag `<audio>`. Permette il caricamento di un file video (in formato mp4 webM ogg) direttamente dal codice html della pagina senza rivolgersi a plugin esterni. Esempio

```
<video width="320" height="240" controls autoplay muted loop >
   <source src="movie.mp4" type="video/mp4">
   <source src="movie.ogg" type="video/ogg">
</video>
```

Attributi:
src, **controls**, **autoplay**, **muted**, **loop**, **preload** sono gli stessi del tag AUDIO,
width e **height** consentono di impostare le dimensioni di visualizzazione del filmato.
Questi attributi sono vivamente consigliati, altrimenti il browser, non potendo sapere le dimensioni del filmato, andrà ad occupare tutto lo spazio necessario modificando il layout della pagina.
poster="url" Indica una immagine temporanea da visualizzare nel caso in cui il video non venga trovato oppure mentre il video viene scaricato o fino a quando l'utente non preme il tasto di avvio di riproduzione del video.

All'interno del tag si può indicare un testo da visualizzare nel caso il browser non supporti il tag video.

Note

- Come **src** si potrebbe inserire il link "embed" di un filmato youtube. Però mentre youtube consente questi collegamenti all'interno di un iframe, non sono invece consentiti all'interno del tag video.
- Come detto Firefox ha iniziato a supportare mp3/mp4 a partire dalla versione 71 (3 dicembre 2019)
- WMV non è invece supportato perché considerato troppo pesante
- Per i formati non supportati è possibile installare dei plug-in esterni.

Elenco dei principali mime Type per i file audio / video:

- `"audio/mp3"  o  "audio/mpeg3"`
- `"audio/x-wav"`
- `"audio/x-mid"`
- `"audio/x-ms-wma"`
- `"video/x-ms-wmv"`
- `"video/mp4"`
- `"audio/basic"  // au`
- `"application/ogg"`

Il tag object

In HTML4 i browser non contenevano un codec nativo per i file audio, ma la loro riproduzione era possibile tramite plug-in installati sulla macchina. Inizialmente il tag utilizzato per l'inclusione di un file audio era il tag **BGSOUND** presto abbandonato a favore del più generale tag **OBJECT** utilizzato per richiamare qualsiasi plug-in esterno. Ad esempio

```
<object type="audio/x-mpeg" data="mySong.mp3" width="200" height="20"  autoplay="true" >
      <param name="src" value=" mySong.mp3" />
      <param name="controller" value="true" />
      <param name="autostart" value="true" />
      <param name="autoplay" value="true" />
      <param name="loop" value="true"> </object>
```

Sulla base del **mime Type** il browser decideva automaticamente quale plug-in utilizzare. Nelle versioni più vecchie occorreva esplicitare direttamente quale player utilizzare, specificando il suo **classid** ed anche l'indirizzo web di dove andare a scaricarlo nel caso in cui non fosse presente sulla macchina.

Gli **attributi** devono essere scritti 2 volte:

- all'interno del tab **obiect (**ed indirizzati al browser da cui verranno interpretati)
- all'interno dei **Parameters** da passare al player, i quali dipendono dal player utilizzato.
  Ad esempio `autoplay` è utilizzato da QuickTime per i video, mentre i player audio di solito utilizzano `autostart`. Eventuali parametri non supportati vengono semplicemente ignorati per cui, in caso di incertezza, è meglio inserirne più di uno. `Controller = true` abilita la visualizzazione della barra per il controllo del player.

nota: Il tag `OBJECT` deve essere scritto in modo indipendente rispetto al tab `AUDIO` e NON al suo interno. Se fosse inserito all'interno verrebbe eseguito SOLO dai browser che NON supportano Html5 (cioè mai).

Il tag `OBJECT` viene ancora utilizzato per aprire ad esempio un `pdf` all'interno della pagina:

```
<object data="data/pagina pdf di prova.pdf" type="application/pdf"
    style="width:300px; height:500px;">
 </object>
```

Il nuovo tag embed

Il nuovo tag **embed** definisce un contenitore preposto a contenere dei dati interpretabili solo tramite un **plug-in esterno**. E' simile al tag `object` ma è più semplice e non ha tag di chiusura.
Ha solo 4 attributi: `src, type, width e height`

```
<embed type="application/x-shockwave-flash" src="x.swf"
       style="width:300px; height:500px;" />
```

Il tag canvas

Il Canvas (in italiano **TELA**) è un contenitore su cui è possibile disegnare elementi grafici al volo tramite apposite funzioni **Java Script**. L'oggetto Canvas dispone di pennelli e diversi metodi di disegno.
Ha solo gli attributi `width e height.`

```
<canvas id="myCanvas" width="600" height="400"
        style="border:1px solid #d3d3d3;">
</canvas>
```

4. Il nuovo CONTENT Model di HTML5 [Modello di organizzazione dei contenuti]

All'inizio l'unico modo per strutturare i contenuti era l'utilizzo delle **Tabelle**. I dati venivano spezzati all'interno di una griglia fatta da infiniti **<tr>** e **<td>:** un attività noiosa, resa ancora peggiore dalla scarsa qualità e flessibilità del risultato.

In HTML 4.0 arrivarono il **tag ** ed i CSS e ci fu un buon miglioramento. Finalmente un modello di costruzione del documento pensato per separare in modo chiaro i contenuti della pagina.
Le pagine HTML diventarono molto più eleganti e leggibili come nel seguente esempio:

```
<div id="header">
  --- Titolo e Testata ---
</div>
<div id="body">
   <div id="menu">
   --- Elenco dei post presenti nella pagina ---
   </div>
   <div id="main_content">
      <div class="post">
      --- Un Post ---
      </div>
      <div class="post">
      --- Un altro Post ---
      </div>
  </div>
</div>
```

Con gli anni però anche questo modello ha cominciato a mostrare le proprie debolezze: né i browser né i motori di ricerca avrebbero mai potuto beneficiare di questa divisione semantica, per colpa di quell'arbitrarietà che permetteva a milioni di siti web (es i blog) di essere organizzati in **strutture simili** ma sempre **leggermente diverse** tra loro e per questo non raggruppabili secondo schemi automatici. Emerse in questo modo uno dei più grandi problemi dell'HTML4: **l'incapacità di descrivere il significato delle informazioni di una pagina web in un formato interpretabile da altri software**.

Il **Content Model di HTML5** invece i utilizzare tutti tag DIV ognuno con un proprio ID o una propria CLASSE, è articolato mediante i cosiddetti **Tag di Contenuto**, cioè tag il cui nome contiene una descrizione del tipo di contenuto previsto.
In HTML 5 il codice precedente potrebbe essere riscritto nel modo seguente:

```
<header>
    --- Titolo e Testata ---
</header>

<section>
    <nav>
    --- Elenco dei post presenti nella pagina ---
    </nav>
    <article>
    --- Un Post ---
    </article>
    <article>
    --- Un altro Post ---
    </article>
</section>
```

i nuovi tag relativi al Sectionig Content
Studiati per ospitare contenuti **atomici** e semanticamente ordinati.

- **header**: intestazione visualizzata all'inizio della pagina o di una sezione
- **section**: corrisponderebbe ad un capitolo di un libro o ad una sezione ben precisa
- **article**: un testo "indipendente", quale ad es un messaggio scritto in un blog, un articolo ecc.
- **footer**: classico piè di pagina, nel quale inserire l'indirizzo email per i contatti, copyright ecc.
- **nav:** racchiude una serie di link ad altre pagine attinenti interne o esterne al sito
- **aside**: rappresentare una nota, un suggerimento, una barra laterale o qualcosa che si trova solitamente **al di fuori** del flusso principale di un articolo.

Sono tutti BLOCK TAG simili a DIV
Questo approccio risolve in modo elegante :

- sia il problema dell'utilizzo degli attributi class / ID con valore semantico,
- sia la riconoscibilità delle singole aree del documento da parte di un browser.

Outliner

L'outliner è un nuovo plug-in HTML5 che può essere aggiunto ad un browser per mostrare l'**outline** (cioè la struttura o sommario) del documento. L'outliner identifica correttamente i vari livelli di profondità, e per ognuno di essi riesce anche recuperare il titolo adeguato.

Per ogni Sectioning-Tag legge e visualizza i tag H interni
I Sectioing Tag riconosciuti dall'Outliner sono:

- **Section**
- **Article**
- **Nav**
- **Aside**

In questo ambito è importante utilizzare sempre almeno un tag appartenente alla categoria ***heading content*** (h1 – h6) all'interno di ogni Sectioning Tag, in modo da non avere un outline (sommario) popolato da voci senza titolo. L'inserimento i un titolo non è comunque obbligatorio.
Le sezioni senza titolo rimangono tali ma non generano errori di validazione.

All'interno dei **`sectioning tag`** l'utilizzo di un **`<h1>`** è considerato **relativo alla sezione in cui si trova**. Cioè h1 di un **`Article`** interno ad una **`Section`** avrà automaticamente size inferiore.
Questo vale però SOLO per `<h1>`. Non vale per h2, h3, h4 etc.

Nota*: Nel caso di chrome **HTML 5 Outliner** è una estensione scaricabile all'indirizzo :*
*https://**chrome.google.com/webstore**/detail/afoibpobokebhgfnknfndkgemglggomo*
Dopo l'installazione l'icona viene mostrata in alto a destra dopo la barra di navigazione, ma solo per le pagine scaricate via http.

Il tag hgroup [deprecato]

*Definisce l'area di intestazione di una sezione o un documento. Viene utilizzato per racchiudere i tags da <h1> a <h6> quando il titolo ha più livelli, come sottotitoli, titoli alternativi, etc. Racchiudendo più tag H in un unico **hgroup**, l'outliner riconosce come un titolo solamente l'heading con il valore **più alto** (h1). Senza <hgroup> tutti i vari titoli verrebbero invece mostrati dall'outliner come titoli dell' <`article`>.*

Il tag header

Serve a rappresentare "un gruppo di ausili *introduttivi* o di navigazione", relativamente a:

- una intera pagina
- una sezione
- un articolo.

Raccomandazioni
- È un contenitore per altri elementi.
- Non va confuso con quello che era l'header frame, cioè la testata/intestazione principale di un documento. Come detto un header può essere riferito ad una pagina, una sezione o un articolo
- Non introduce una nuova sezione e quindi non è rilevante per l'**outliner**.

Il tag section

Per definire la sezione che ospita i contenuti principali della pagina, cioè i post, si usa il tag **<section>**
L'elemento <section> rappresenta un **contenitore di sezionamento dei contenuti,** cioè individua **un raggruppamento tematico di contenuti**,che in genere contiene un titolo h1 introduttivo.

Raccomandazioni:
- Il tag **<section> non è** sostitutivo del per impostare graficamente la pagina.
- Il tag **<section>** rappresenta un elemento che viene considerato una sezione a sé stante dall'**outliner** e quindi un blocco con dei contenuti univoci che necessitano di un titolo (<h_N>).
- l'elemento **<section>** e l'elemento **<article>** non sono indipendenti ne esclusivi: possiamo avere sia un **<article>** all interno di un **<section>** che viceversa.

Il tag article

Il tag **<article>** rappresenta una **sezione autonoma in un documento,** che è potenzialmente ridistribuibile o riutilizzabile, e quindi ripubblicabile in parte o interamente in diverse pagine.
Esso può identificare :
- un post di un forum,
- l'articolo di un blog,
- un articolo di una rivista o di un giornale,
- un commento, un widget interattivo,
- qualsiasi cosa che abbia un contenuto indipendente

Raccomandazioni:
- quando gli elementi <article> sono nidificati, gli <article> interni rappresentano gli articoli che sono in linea di principio relativi al contenuto dell'<article> esterno. Ad esempio, un blog che accetta commenti dagli utenti potrebbe rappresentarli come <article> figli annidati all'interno dell'elemento padre <article>
- le informazioni relative all'autore dell'<article> non devono essere replicate all'interno degli elementi nidificati all'interno dello stesso;
- l'elemento <time> definisce la data di pubblicazione dell'<article>;

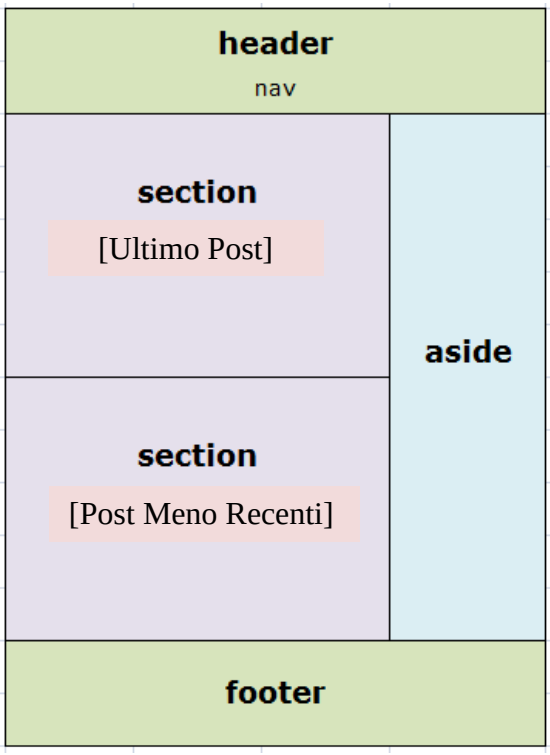
Il tag footer

L'elemento **<footer>** contiene **informazioni sulla sezione che lo contiene** come ad esempio:
- I dati di chi ha scritto i contenuti;
- collegamenti ai documenti correlati;
- i dati di copyright;

Raccomandazioni:
- Non necessariamente deve essere inserito solo alla fine di un documento.
- Non introduce una nuova sezione e quindi non è rilevante per l'**outliner**.
- All'interno di una pagina web possono essere presenti diversi <footer> anche più di uno per lo stesso elemento.

Esempio di organizzazione logica di un Blog

Struttura:



La pagina è suddiviso in due grandi macrosezioni del blog: "l'ultimo post" e "i post meno recenti" contenuti in due `<section>` successive.

Ciascun post, a livello strutturale, è rappresentato da un tag **`<article>`**:

Ciascun post (articolo) avrà un titolo e una data. Questi campi sono certamente ausili introduttivi all'articolo, per cui è consigliato racchiuderli in un tag **`<header>`.**

```
<body>
<header>
  <div id="hgroup">
     <h1>3 INF A - NEW BLOG</h1>
     <h2>Approfittiamo dei vantaggi delle specifiche HTML5! </h2>
  </div>

  <nav>
     <h1> Navigazione </h1>
          <ul>
               <li> <a href="#"> Home </a> </li>
               <li> <a href="#"> Gli autori </a> </li>
               <li> <a href="#"> Archivio </a> </li>
          </ul>
  </nav>
</header>
```

Il tag nav

Rappresenta una sezione di una pagina che contiene dei **link** (collegamenti) di navigazione verso altre pagine o parti interne dello stesso documento**.**

Non necessariamente deve essere inserito nell'`<header>`**.** Esistono molti tipi di layout in cui il menu di navigazione può essere facilmente slegato dagli elementi introduttivi di intestazione del documento.

Raccomandazioni:

- solo sezioni con blocchi di navigazione 'importanti' sono appropriati per l'elemento <nav>;
- i link a pie' di pagina e le breadcumb non devono essere inseriti in una sezione <nav> (Le breadcumb sono"briciole di pane" Rappresentano cioè un "sentiero" composto di link utili agli utenti per tornare indietro alla pagina iniziale del sito web o a pagine visitate in precedenza per arrivare all'attuale) ;
- l'elemento <nav> non sostituisce i link inseriti in elementi come <ul> o <ol> ma deve racchiuderli

All'interno del Content **<nav>** utilizzato nell'esempio del blog, da notare la presenza del titolo <h1> che serve a specificare più dettagliatamente la funzione del blocco e a conferirgli un titolo valido anche per l'**outliner** che considera il tag **<nav>** come una sezione a sé stante, cioè un blocco con dei contenuti univoci che necessitano di un titolo che li riassuma.

L'ultimo Posto: Un `article` con due `section`

All'interno dell'articolo ULTIMO POST vengono inserite due sezioni aggiuntive :

- una nuova `<section>` relativa ai commenti. Questa `<section>` rende i commenti **semanticamente separati dal contenuto principale**. **In questo modo il contenuto dell'<article>** (cioè il post vero e proprio) **può essere citato o ripubblicato in altri blog indipendentemente dai commenti che ha ricevuto.** Si ottiene così una netta separazione tra i commenti (che sono una sezione aggiuntiva eventualmente anche eliminabile) e l'argomento principale trattato all'interno dell'articolo.
- una nuova `<section>` per l'inserimento di nuovi commenti

Per tutti i **Titoli** delle varie section / article, Invece di `<h3> <h4>` etc è raccomandato di usare sempre **<h1>** le cui dimensioni vengono automaticamente "riscalate" in base al livello di annidamento in cui h1 si trova. `<h2> <h3> >` etc sono da utilizzarsi soltanto per i sottotitoli di quelle sezioni che già contengono un h1.

```
<section>    <!—     ULTIMO POST     -->
  <h1> L'ultimo Post</h1>
  <article>
     <header>
          <time datetime="2012-11-22" > Lunedì 22 Novembre</time>
          <h1>Nuove scoperte sul tag video!</h1>
     </header>
     <p>
         [Corpo dell'articolo: contenuto del post sulle scoperte del tag
     video]
     </p>
     <footer>
          <p> [Informazioni riguardo l'autore] </p>
     </footer>
```

```
      <section> <!-- Commenti all'Ultimo Post -->
            <h1>Commenti all'ultimo Post</h1>
            <article>
                  [commento1...] ogni commento può avere header, corpo, footer
            </article>
            <article>
                  [commento2...] ogni commento può avere header, corpo, footer
            </article>
            <article>
                  [commento3...] ogni commento può avere header, corpo, footer
            </article>
      </section>

      <section>  <!—Form per i Commenti all'Ultimo Post -->
            <h1>Inserimento di un Nuovo Commento</h1>
            <form name="frmCommenti" id="frmCommenti">
                  ……
            </form>
      </section>
  </article>
</section>

<section>    <!--     Post Precedenti     -->
   <h1>Post meno recenti</h1>
   <article>
      [testo penultimo post articolato sempre cin <header> <p> <footer>]
   </article>
   <article>
      [testo terzultimo post articolato sempre cin <header> <p> <footer>]
   </article>
</section>

<footer>   <!-- Info su Creatore e Amministratore del Blog-->
      <dl>
         <dt>Creato da</dt>
         <dd><address><a href="mailto:xx">Sandro Paganotti</a></address></dd>
         <dt>Ultimo aggiornamento</dt>
         <dd><time datetime="2012-11-23" >Ma 23 Novembre</time></dd>
      </dl>
</footer>  </body>
```

Il tag aside

L'elemento **<aside>** rappresenta una sezione di una pagina costituita da **informazioni marginali** rispetto al contenuto della pagina che lo contiene. Si tratta tipicamente di un **contenitore di approfondimenti** in cui si possono inserire gruppi di link, pubblicità, bookmark etc.

Nel blog si potrebbe ad esempio aggiungere il seguente blocco **<aside>** posizionato o subito dopo l'**<header>** (prima delle due **<section>**) oppure dopo le due <section> e prima del <footer>.

> Questa sezione **<aside>** contiene una form per la ricerca di parole chiave ed una barra con i link alle pagine correlate. Non essendo informazioni particolarmente rilevanti per il contenuto centrale del blog, possiamo separarli con l'elemento <aside> che li qualifica come **contenuti marginali**.

```
<aside>
   <h1>Sidebar</h1>
   <section>
      <h2>Ricerca di parole chiave all'interno del sito </h2>
      <form name="ricerca" method="post" action="/search">
               [form di ricerca]
      </form>
   </section>

   <nav>
        <h2>Categorie trattate all'interno del blog </h2>
        <ul>
            <li><a href="/categoria/multimedia">multimedia</a></li>
            <li><a href="/categoria/form">forms</a></li>
        </ul>
   </nav>

   <section>
       <h2>Pubblicità</h2>
       <img src="img/Koala.jpg" width="200">
   </section>

</aside>
```

Anche il tag **<aside>** appartiene alla categoria dei "contenitori di sezionamento dei contenuti" validi per l'**outliner**. Necessita pertanto di un titolo che riassuma i propri contenuti.