

# FONDAMENTI DI INFORMATICA

Alma Artis  
Francesca Pratesi (ISTI, CNR)

HTLM e Pagine Web

# LIBRI E RIFERIMENTI



- [w3schools.com](https://www.w3schools.com)
- [html.it](https://www.html.it)
- [grafigata.com](https://www.grafigata.com)
- [Fontsinuse.com](https://www.fontsinuse.com)





**HTLM**

# HYPertext MARKUP LANGUAGE

- HTML è l'acronimo di HyperText Markup Language
- Non è un linguaggio di programmazione, ma di **markup**
  - Permette di indicare come disporre gli elementi all'interno di una pagina
- È quindi un **linguaggio dichiarativo**, che serve appunto ad indicare cosa deve apparire sullo schermo (testi, immagini, suoni, etc.), dove e in che sequenza. Non possiamo specificare algoritmi precisi con "strutture di controllo", come facciamo con JavaScript
- Queste indicazioni vengono date attraverso **tag** (etichette)
- È uno standard, stabilito dal W3C (World Wide Web Consortium)
- Ne esistono diverse versioni (adesso siamo alla quinta)



# TAG

- I tag sono racchiusi tra parentesi angolate
  - Es. <img>
- Per ogni tag aperto deve esserci il corrispondente tag chiuso
  - Es. </img>
  - Tranne rare eccezioni
- Possiamo incapsulare tag diversi, l'importante è chiuderli in ordine inverso
  - Es.
    - <p>
    - <div>
    - </div>
    - </p>
- L'indentazione aiuta moltissimo la leggibilità



# ELEMENTI FONDAMENTALI

- HTML
  - serve a definire quali sono gli elementi in gioco, stabilire collegamenti (link) tra le pagine e l'importanza (non la forma o il colore) che hanno i testi, creare form per gli utenti, fissare titoli, caricare immagini, video, etc.
- CSS
  - o "fogli di stile". Si tratta di una serie di regole che permettono di definire l'aspetto (lo stile) che devono assumere gli elementi sulla pagina. Dimensioni, colori, animazioni, ogni caratteristica visuale può essere manipolata.
- JavaScript
  - essendo un linguaggio di programmazione, consente di manipolare veramente qualunque cosa nella pagina HTML: lo stile, i contenuti della pagina, ma soprattutto l'interazione con l'utente.



# JAVASCRIPT NELLE PAGINE HTML

- Ci permette di creare la logica dell'interfaccia utente (o anche di una applicazione) e di sfruttare le API messe a disposizione dal browser:
  - gestione del mouse / touchpad
  - manipolazione delle immagini
  - richieste di dati dinamiche
  - controllo di consistenza dei dati
  - gestione di dati in locale



# I FOGLI DI STILE

- Ad un certo punto nella storia del web, si è sentita l'esigenza di separare lo stile di presentazione di un contenuto (ad esempio, un testo) che dà la forma del layout della pagina html dal contenuto stesso
- I fogli di stile (o css, cascading style sheet) consentono di separare il contenuto dalla presentazione del contenuto medesimo
- In un file .html ci occuperemo quindi di cosa mettere (il contenuto), mentre nel file .css ci occuperemo di come presentarlo





# FILE HTML

- Un file che contiene codice html è un normale file di testo con estensione .html
- Per leggerlo/scriverlo possiamo usare un qualunque **editor** (es. Notepad++ e Atom.io)
  - Ma non Word e simili, che sono word processor più evoluti e aggiungerebbero materiale che andrebbe a «sporcare» il nostro file
  - Questi editor consentono:
    - Di evidenziare la sintassi
    - L'autocompletamento
    - La chiusura automatica dei tag
- Per visualizzarlo (interpretarlo!) usiamo un qualsiasi **browser**
  - Anche se i browser non sono tutti uguali, ci torneremo più avanti



# LINK E «IPERTESTI»

- La potenza di Internet consiste nell'essere un insieme esteso di contenuti connessi tra loro
- Il linguaggio HTML e i link sono alla base di questo meccanismo che consente di muoversi tra testi, immagini, video, applicazioni e quant'altro creando percorsi di navigazione in totale autonomia
- Per accedere rapidamente a questa enorme mole di informazioni sono nati i motori di ricerca, che si basano su una serie di elementi "semantici" per rispondere adeguatamente alle richieste degli utenti
- Si parte dall'analisi del testo (o del contenuto) e dei collegamenti, fino a sfruttare i comportamenti degli utenti per catalogare le pagine e stabilire il livello di rilevanza (ranking) che hanno a seconda della singola query di ricerca
- L'ottimizzazione dei contenuti per favorire il posizionamento della pagina per specifiche ricerche si dice SEO (Search Engine Optimization)



# IL NOSTRO PRIMO FILE

- Apriamo un editor di testi e scriviamo qualcosa:  
    <h1>Ciao Mondo!</h1>  
    <p>Questa è la nostra prima pagina HTML!</p>
- Salviamo il file con il nome «hello.html»
- Facciamo doppio click sul file per aprirlo
  - Di default il browser preferito è il programma predefinito



# LA STRUTTURA DI UN FILE HTML

```
<!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>
```



# COSA TROVIAMO IN UNA PAGINA

- `<!doctype html>` Serve semplicemente a dire che il file è una pagina HTML e in particolare che si tratta di un documento secondo lo standard HTML5.
- `<html>` È il tag che racchiude tutta la pagina e (opzionale) ci permette ad esempio di definire quale sia la lingua della pagina
- `<head>` Racchiude una serie di informazioni utili per la gestione della pagina, come il titolo che apparirà sui motori di ricerca e sulle linguette del browser
- `<body>` 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.



# TAG FONDAMENTALI: <P>

- Il paragrafo <p> è un elemento contenitore che al suo interno prevede l'inserimento di testo e di altri tag.
- Sono tag di tipo **block**, cioè il comportamento di base sarà quello di andare a capo al termine del paragrafo e che siano stabiliti margini per il testo



# TAG FONDAMENTALI: <P>

`<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 primo paragrafo di questa trattazione, ci occuperemo dell'importanza del testo nel Web. Grazie ad un semplice esempio possiamo sperimentare molte cose.




MARGINE

Per default il browser manderà a capo il contenuto di questo secondo paragrafo.



# TAG FONDAMENTALI: <BR>

- Il tag <br> serve per andare a capo
- È uno dei pochi tag che non necessita di chiusura



L'html non è case-sensitive, tuttavia lo standard impone di usare sempre il minuscolo per i tag



# TAG FONDAMENTALI: TITOLI O HEADING

- Sono 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
- 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.



# TAG FONDAMENTALI: TITOLI O HEADING

`<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** ← h1

**Acquistare la casa** ← h2

**Le pratiche per l'acquisto di casa** ← h3

**Agenzie immobiliari, quali scegliere**

**Arredare la casa**

**Come scegliere la cucina per la nuova casa**

**Lampadari, tipologie e differenze**



# TAG FONDAMENTALI: GRASSETTO E CORSIVO

- Il tag `<b>` racchiude un testo che vogliamo visualizzare in grassetto (bold)
- Il tag `<i>` racchiude un testo che vogliamo visualizzare in corsivo (italic)
- Negli ultimi tempi però per enfatizzare un testo si preferiscono tag relativi alla **semantica** del testo, più che alla sua forma:
  - `<strong>` attribuisce al testo una forte importanza, serietà o urgenza (la resa grafica solitamente è il grassetto)
  - `<em>` (emphasis) serve a rappresentare un testo o una frase che si pronuncia in modo differente dal resto al testo



# TAG FONDAMENTALI: GLI ELENCHI

- Gli **elenchi puntati** devono essere dichiarati dal tag `<ul>`
- Ogni elemento è racchiuso dal tag `<li>`

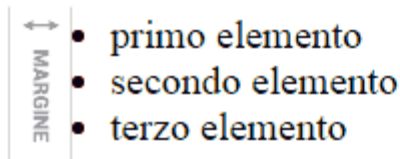
```
<ul>
```

```
  <li>primo elemento</li>
```

```
  <li>secondo elemento</li>
```

```
  <li>terzo elemento</li>
```

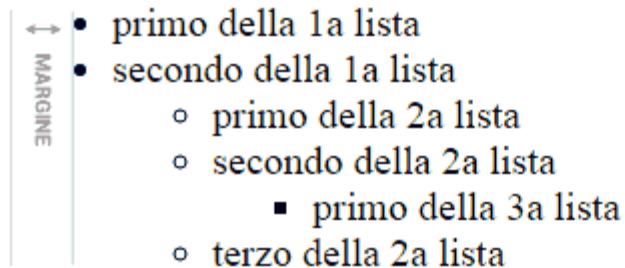
```
</ul>
```



# TAG FONDAMENTALI: GLI ELENCHI (2)

- E' possibile annidare più elenchi

```
<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>
```



# TAG FONDAMENTALI: GLI ELENCHI (3)

- Con il tag `<ol>` avremo invece **liste ordinate**

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

↑  
↓  
MARGINE

1. primo elemento

2. secondo elemento

3. terzo elemento

Testo che segue la lista

# TAG FONDAMENTALI: <DIV>

- I tag <div> identificano elementi blocco
- Definiscono la struttura (layout) della pagina

```
<div>  
  <p>  
    testo  
  </p>  
  <div>  
    <ul>  
      <li>elenco1</li>  
      <li>elenco2</li>  
    </ul>  
  </div>  
</div>
```



# TAG FONDAMENTALI: <TABLE>

- I tag <table> identificano una tabella
- I tag <tr> una riga della tabella
- I tag <td> una colonna della tabella

```
<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

- **Le tabelle nelle pagine HTML sono fondamentali per strutturare una pagina!**
  - Non pensatele sempre con i bordi





# ALTRI TAG UTILI

- `<del>` Descrive un contenuto che vogliamo togliere dal documento, tipicamente traccia una barra sul testo
- `<mark>` Evidenzia il testo
- `<ins>` Può essere utile per definire degli aggiornamenti al documento, assegnando una data specifica a una porzione del testo



# COMMENTI

- Un commento è racchiuso tra i tag `<!--` e `-->`  
`<!-- questo è un commento -->`
- Un commento in HTML è multiriga  
`<!--`  
anche questo  
è un commento  
`<p>` questo testo verrà `<strong>` completamente `</strong>`  
ignorato!`</p>`  
`-->`



# GLI ATTRIBUTI DEI TAG

- 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.  
`<tag attributo1="valore1" attributo2="valore2">`
- Sono coppie chiave-valore separate dal carattere = (uguale);
- I valori sono tipicamente racchiusi tra virgolette "", ma è possibile anche utilizzare gli apici ';
- Si scrivono lasciando almeno uno spazio dopo il nome dell'elemento nel tag di apertura



# ALCUNI ATTRIBUTI

lang	Specifica la lingua dell'elemento (è un attributo core, che non modifica in nessun modo la rappresentazione grafica dell'elemento o il suo stile)
id	Serve ad associare un identificatore univoco ad un elemento
class	Serve a stabilire che l'elemento appartiene ad una serie di "classi", cioè elementi con proprietà simili. Possiamo inserire quante classi vogliamo, tutte separate da uno spazio. Es. <code>&lt;p class="saluto testo-chiaro"&gt;ciao&lt;/p&gt;</code> , ha due classi: "saluto" e "testo-chiaro".
style	Serve ad assegnare delle proprietà grafiche (stili CSS) all'elemento; ne parleremo in seguito
draggable	Può assumere i valori true o false e serve a stabilire se un elemento sia trascinabile per una operazione di drag-n-drop



# ATTRIBUTI PER LA GESTIONE DI EVENTI

onclick	Rileva il click (o il tap) effettuato sull'elemento
onmouseover	Associa un comportamento al passaggio del mouse sopra l'elemento
onload	Associa un comportamento alla conclusione del caricamento dell'elemento
onscroll	Attiva un comportamento correlato allo scrolling della pagina
ondrag	Si attiva quando iniziamo a trascinare un elemento che abbiamo indicato come draggable="true"



# ATTRIBUTI TYPE

- Alcuni tag hanno la possibilità di definire uno stile attraverso l'attributo type
- Es: l'attributo type applicato al tag <ol> serve per specificare il tipo di enumerazione che vogliamo applicare

type="1"      numeri interi "arabi (valore di default)

type="a"      numeri alfabeto minuscolo

type="A"      numeri alfabeto maiuscolo

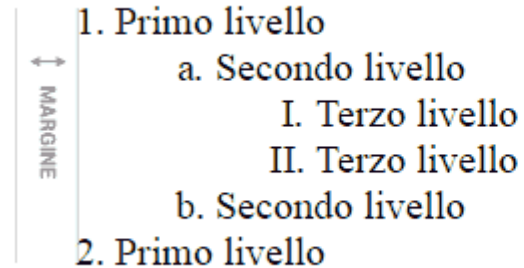
type="i"      numeri numeri romani minuscoli

type="I"      numeri numeri romani maiuscoli



# ATTRIBUTI TYPE - ESEMPIO

```
<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>
```



# TAG FONDAMENTALI: <TABLE> (REPRISE)

- Possiamo raggruppare le celle della tabella anche per colonna grazie al tag `<colgroup>`
- 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>
<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>
</table>
```

head	head	head	head	head	head
cell	cell	cell	cell	cell	cell
cell	cell	cell	cell	cell	cell
cell	cell	cell	cell	cell	cell
COLGROUP (id="a")					



# TAG FONDAMENTALI: <TABLE> (REPRISE - 2)

- Per ottenere una cella che si estenda su più di una riga utilizziamo l'attributo <rowspan> in modo del tutto simile a quanto fatto con <colspan>

```
<table>
  <thead>
    <tr><th>head</th><th>head</th><th>head</th></tr>
  </thead>
  <tbody>
    <tr><td rowspan="2">double row cell</td>
      <td>cell</td><td>cell</td></tr>
    <tr><td>cell</td><td>cell</td></tr>
    <tr><td>cell</td><td>cell</td><td>cell</td></tr>
  </tbody>
</table>
```

head	head	head
double row cell	cell	cell
	cell	cell
cell	cell	cell

# LINK

- I link sono "il ponte" che consente di passare da un testo all'altro. In quanto tali, i link sono formati da due componenti:
- Il **contenuto**:
  - è la parte visibile del link, e proprio per questo l'utente deve essere sempre in grado di capire quali sono i collegamenti da cliccare all'interno della pagina
  - "nasconde" il collegamento (non importa se si tratta di testo o di immagine)
- La **risorsa**:
  - si tratta di un'altra pagina (sullo stesso server o su un server diverso), oppure è un collegamento interno a un punto della pagina stessa
  - Indica la direzione cui punta il collegamento
- I link si indicano con il tag **<a>** comprensivo dell'attributo href

`<a href="https://github.com/prafra/AlmaArtis2023">Visitate il sito del Corso!</a>`



# LINK – LE POSSIBILI DESTINAZIONI

- È indifferente che la destinazione del link sia una pagina HTML di un sito, un'immagine, un file pdf, un file zip, 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:
  - Immagini: vengono visualizzate nel browser
  - Doc e pdf: vengono visualizzati nel browser (se l'utente non ha il programma giusto per aprirlo verrà interpellato)
  - Eseguiibile: sconsigliato! spesso il download viene bloccato da antivirus e firewall
  - Mail: viene aperto il client di posta (se disponibile)

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



# LINK – UN SITO WEB COME RISORSA

- Solitamente i link puntano a pagine web
- Di default la pagina viene aperta nella stessa scheda del contenuto
- È possibile specificare un comportamento diverso con l'attributo **target**

```
<a href=https://github.com/prafra/AlmaArtis2023 target="_blank">Visitate il sito del Corso!</a>
```

- Se la risorsa è nello stesso spazio del sito che stiamo costruendo è possibile inserire sia percorsi assoluti (iniziano con http o https) sia **percorsi relativi**, che fanno riferimento alla posizione degli altri file rispetto al documento in cui ci si trova in quel momento

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

collegamento alla pagina da linkare nella stessa directory della pagina presente

```
</a>
```

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

collegamento alla pagina da linkare nella directory superiore rispetto alla pagina presente

```
</a>
```

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

collegamento alla pagina da linkare in una directory sorella rispetto alla pagina presente

```
</a>
```



# LINK – RISORSA NELLA STESSA PAGINA

- Possiamo sfruttare il meccanismo dei link anche per creare un indice interno al documento
- In questo caso la risorsa viene solitamente chiamata **ancora**
- Ciascuna ancora può avere infatti un nome:

```
<a name="primo">Questo è un normale testo nel documento</a>
```

- È possibile far riferimento all'ancora presente all'interno del documento attraverso un link che punti ad essa
- Il cancelletto indica che il collegamento deve cercare un ancora chiamata con un certo nome all'interno della pagina stessa:

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

- 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):

```
<a href="#">link vuoto: vai ad inizio pagina</a>
```



# IMMAGINI

- Il tag `<img>` rappresenta il principale elemento per inserire un'immagine in una pagina HTML:

```

```

- Il tag `<img>` è un tag senza un contenuto, per questo non ha un elemento `</img>` di chiusura
- Questi sono gli elementi minimi da considerare:
  - `img`: è il nome del tag, abbreviazione di image (immagine)
  - `src`: sta per source (origine), è l'indirizzo (URL) del file che vogliamo mostrare
  - `alt`: è il testo alternativo, ovvero il testo che appare se, per qualche motivo, il client non riesce a mostrare l'immagine. Possiamo anche omettere questo attributo, ma risulta utile per l'accessibilità (es. screen-reader per ipovedenti) e per i motori di ricerca
- Il tag `<img>` è un elemento di tipo "inline", ovvero, quando inseriamo un'immagine in un testo, il browser in genere la mostra a piena risoluzione e la considera in linea con il testo, ovvero come parte del testo stesso
- Per separare l'immagine dal testo possiamo distinguere le righe scrivendo il nostro testo in paragrafi diversi

# I FORMATI DI IMMAGINI

- Attraverso l'attributo src possiamo caricare diversi tipi di immagine, sia bitmap (o raster), sia vettoriale
  - JPG/JPEG: ottimi per ridurre lo spazio occupato da foto di grandi dimensioni. L'algoritmo è di tipo lossy, cioè a "perdita di informazione": per ottenere file leggeri in termini di KB si effettuano dei campionamenti e si rinuncia alla qualità dell'immagine. Offre un buon compromesso tra peso e qualità dell'immagine
  - GIF: permettono di ottimizzare il peso dei file riducendo ancora il numero di colori (256 max). Consente di impostare trasparenze e di riprodurre piccoli video (GIF animate)
  - PNG: quasi equivalente al GIF (supporta il canale alfa) ma open, e per questo preferibile
  - SVG: in questo caso le dimensioni delle immagini non dipendono dallo spazio che occupano sullo schermo ma dalla loro complessità. Sono ideali per loghi o piccole icone



# IMMAGINI E LINK

- Possiamo inserire un'immagine direttamente in un link per renderne più chiaro il significato o più accattivante il link stesso

```
<a href="https://www.mioSito.it">  
      
</a>
```

- Per il percorso del file valgono le stesse regole viste per i link
  - si possono specificare sia percorsi assoluti che relativi





# IMMAGINI: DIMENSIONI

- Gli attributi **width** e **height** permettono di impostare rispettivamente la larghezza e l'altezza di un'immagine
- Di solito, però, è preferibile usare l'attributo **style** per impostare questi valori
  - Per prevenire che i fogli di stile cambino la dimensione dell'immagine

```
<!DOCTYPE html>
<html>
  <head>
    <style>
      <!-- questo comando seta la larghezza di tutte le immagini al 100% dello spazio disponibile: -->
      img {
        width: 100%;
      }
    </style>
  </head>
  <body>
    <h2>Width/Height Attributes or Style?</h2>
    <p>In questa immagine vorremmo settare la larghezza a 128 pixel con l'attributo width ma lo stile nella sezione head
sovrascrive questo settaggio. L'immagine sarà quindi larga quanto il testo.</p>

    <p>Qui l'attributo style dell'elemento ha invece priorità rispetto allo stile definito nella sezione head. </p>

    
  </body>
</html>
```

# IMMAGINI: FAVICON

- Una favicon è un'icona speciale, che viene visualizzata dai browser nelle schede
- Dal momento che è piccolo, deve essere semplice e con un alto contrasto
- I principali browser (Edge, Chrome, Firefox, Opera, Safari) supportano file .ico, .png, .gif, .jpg e .svg
- Potete crearne di personali su siti come <https://www.favicon.cc>
- Per aggiungere una favicon, salvarla nella root directory (la cartella principale) o in una cartella images
  - Di solito con il nome di "favicon.ico"
- A quell punto, usare il tag <link> nella sezione head del vostro file principale, subito dopo il tag after the <title>

```
<head>
  <title>My Page Title</title>
  <link rel="icon" type="image/x-icon" href="/images/favicon.ico">
</head>
```



# AUDIO

- È possibile anche includere anche file audio

```
<audio>  
  <source src="musica.mp3" type="audio/mp3">  
</audio>
```

- Alcuni attributi possibili sono:

- **controls**: per garantire la presenza dei controlli utente per gestire la riproduzione del file audio. Omettere l'attributo controls ci permette, di fatto, di inserire un file audio in background
- **autoplay**: forza l'audio ad essere eseguito automaticamente
- **loop**: fa sì che il file venga automaticamente riprodotto a ciclo continuo

- È inoltre possibile specificare più sorgenti, qualora ci fossero problemi di compatibilità (non tutti i tipi di file sono supportati da tutti i browser)

– In questo caso il secondo file viene riprodotto solo se il primo non è supportato

```
<audio controls>  
  <source src="musica.mp3" type="audio/mp3">  
  <source src="musica2.ogg" type="audio/ogg">  
</audio>
```



# VIDEO

- In modo del tutto analogo agli audio, è possibile includere video

```
<video width="320" height="240" controls>  
  <source src="video.mp4" type="video/mp4">  
</video>
```

- Come si vede dall'esempio, in questo caso si può specificare anche la dimensione del video
- Anche in questo caso è possibile specificare più sorgenti



# SEPARARE IL LAYOUT DAL CONTENUTO

- L'HTML in origine è nato come linguaggio per formattare i documenti presenti sul Web. Proprio per questo motivo il contenuto (ad esempio `<p>qui il mio testo</p>`) e i tag che indicano uno stile o una colorazione del contenuto (ad esempio `<font color="red">`, che colora il testo di rosso) si trovavano mischiati allo stesso livello
- Tuttavia è nata l'esigenza di separare il contenuto dalla presentazione del contenuto medesimo
- Se per esempio io avessi tutti i titoli del mio documento in rosso e in grassetto, e a un certo punto decidessi di trasformarli in verde e in corsivo, con l'HTML classico (cioè l'HTML 3.2) dovrei andare a modificarmi a mano ogni tag contente le indicazioni della formattazione



# SEPARARE IL LAYOUT DAL CONTENUTO (2)

- Titoli in rosso e in grassetto (HTML classico)

```
<p>  
  <font color="red">  
    <b>titolo 1</b>  
  </font>  
</p>
```

- Titoli in verde e in corsivo (HTML classico)

```
<p>  
  <font color="green">  
    <i>titolo 1</i>  
  </font>  
</p>
```

- Ovviamente dovrei andare a cambiare **tutti** i titoli (in **tutte** le pagine)
- Non c'è niente di più brutto di un sito web non consistente



# I FOGLI DI STILE (CSS)

- Per separare il contenuto (la scritta "titolo 1") dalla formattazione (il colore rosso e il grassetto) è necessario utilizzare i fogli di stile

```
<p class="formattaTitoli">  
  titolo 1  
</p>
```

- L'aspetto del testo (colorazione, enfasi, ma anche la dimensione e lo stile del font) viene quindi affidata alla classe "formattaTitoli", descritta altrove del documento, o su un file separato
- Basta quindi modificare la classe "formattaTitoli" per cambiare l'aspetto di tutti i titoli
- È ovviamente possibile specificare un comportamento diverso per ogni livello della gerarchia dei titoli (h1-h6)



# I FOGLI DI STILE (2)

- Di solito si parte da uno (o più) file css separato, dove definire:
  - I colori usati
  - I font (o le famiglie di font)
  - Margini (padding e spacing)
- Si aggiunge poi il riferimento a questo file nella sezione head

```
<head>  
  <title>Titolo</title>  
  <link rel="icon" type="image/x-icon" href="favicon.png"/>  
  <link href="css/styles.css" rel="stylesheet"/>  
  <link href="css/styles2.css" rel="stylesheet"/>  
</head>
```

- Se si aggiungono più file il secondo sovrascrive le impostazioni del primo
  - Nel caso in esempio se style.css e style2.css hanno clausole potenzialmente in conflitto, valgono quelle di style2.css





# I FOGLI DI STILE (3)

- Altri dettagli (locali ad un singolo file o ad uno specifico elemento) possono essere scritti direttamente nel nostro file .html
  - Le direttive locali, essendo specificate a valle del file .css hanno la precedenza su quelle del file
  - Ma questo lo avevamo visto anche quando parlavamo delle immagini

```

```

Come vedete, questa è la sintassi da usare se ci sono più proprietà nello stesso attributo

# I TRE MODI PER DEFINIRE GLI STILI

- I CSS possono essere aggiunti ad un document in tre modi:
  - **Inline** – usando l'attributo style dentro il tag html
  - **Internal** – usando un elemento <style> nella sezione <head>
  - **External** – usando un elemento <link> per riferire un file CSS nel sito
- Come detto, il modo più comune è l'aggiunta di un file esterno; alcune modifiche locali ad un file si possono fare internamente nell'head, ma solitamente si sovrascrive una singola direttiva direttamente inline



# IL COLORE DI SFONDO

- Il colore di sfondo è una caratteristica estetica della pagina







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

- È possibile definirlo con un nome, oppure con una codifica esadecimale

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



# I COLORI DI BASE

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

# I COLORI PERSONALIZZATI

- La notazione esadecimale (base 16) permette di creare il nostro punto di colore preferito
  - Le cifre nella notazione esadecimale corrispondono a: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

# 00 00 FF  
R G B

- È dunque possibile cambiare le singole coppie con valori da 00 (assenza di colore) a FF (colore massimo)
  - Trovate una bella lista di colori qua:  
<https://www.web-link.it/colori-html.html>

#ff00ff

#0000ff

#0E75F2

#0DF3F3

#0EF285

# COLORE DEL TESTO

- Allo stesso modo (e con la stessa notazione) possiamo definire anche il colore del testo!

```
<body text="blue">
```

- La parola **cascading** in CSS significa che lo stile applicato ad un element verrà applicato anche ai suoi figli
- Quindi, se settiamo il colore del test nel body a "blue", anche tutti i titoli, paragrafi e altri elementi testuali avranno lo stesso colore
  - A meno ovviamente di non specificarne altri



# COLORE DEI LINK

- I link possono avere aspetti diversi a seconda del loro stato

Stato	Significato	Descrizione	Default
Link	A riposto	Viene evidenziato in qualche modo per far sì che sia facile per l'utente individuarlo	Blu, sottolineato
Visited	Visitato	L'URL della pagina compare nella cronologia dell'utente	Viola, sottolineato
Hover	Al passaggio del mouse	È possibile creare un effetto di visualizzazione per invogliare l'utente a seguire il link	-
Active	Attivo	Storicamente era utile per indicare che un link quando aveva il focus, adesso è poco utile	-



# COLORE DEI LINK - ESEMPIO

```
<head>
  <style>
    a:link {
      color: green;
      background-color: transparent;
      text-decoration: none;
    }
    a:visited {
      color: pink;
      background-color: transparent;
      text-decoration: none;
    }
    a:hover {
      color: red;
      background-color: green;
      text-decoration: underline;
    }
    a:active {
      color: yellow;
      background-color: transparent;
      text-decoration: underline;
    }
  </style>
</head>
```





# IMMAGINI DI SFONDO

- È possibile inserire un'immagine come sfondo della pagina  
`<body background="imgSfondo.gif">`
- L'immagine di sfondo verrà ripetuta in orizzontale e in verticale
  - Ma è possibile evitare che questo accada con l'attributo `background-repeat: no-repeat`
- È anche possibile combinare i due attributi appena visti, in modo che mentre l'immagine di sfondo viene caricata, venga comunque visualizzata una colorazione della pagina:  
`<body bgcolor="#0000ff" background="imgSfondo.gif">`
- È importante definire sempre un colore di sfondo (anche se bianco), perché altrimenti il browser assegna alla pagina il colore di sfondo che l'utente ha impostato nel sistema operativo (che potrebbe essere nero)



# MARGINI

Ci sono due tipologie di margini in HTML:

- L'attributo **margin** è lo spazio vuoto che circonda l'elemento
- L'attributo **padding** rappresenta lo spazio che l'element ha al suo interno e che lo distanzia dal contenuto vero e proprio



# MARGINI (2)

- Il **margin** informa il browser dello spazio che deve essere lasciato tra gli elementi indipendenti e il margine esterno della pagina
  - Può essere impostato per regolare la posizione dei diversi elementi, mantenendoli ad una distanza uguale
  - Oppure, al contrario, per sovrapporli, impostando valori negativi
- Il **padding** è importante per creare uno spazio aggiuntivo all'interno degli elementi
  - È utile ad esempio per separare caselle di testo ed immagini assicurandosi che restino allineate
- Sia margin che padding possono essere impostati in pixel (px) e sia per l'elemento nella sua totalità che singolarmente nelle quattro dimensioni (alto, basso, sinistra, destra)



# MARGINI (3)

- Alcuni esempi:

```
<body leftmargin="0" topmargin="0">
```

```
<body margin="0">
```

Direttamente  
nell'HTML:  
sconsigliato!

```
#maincontent { margin: 10px; }
```

```
#maincontent { margin-bottom: 10px; }
```



# FONT

- I font (o caratteri) rappresentano, in ambito tipografico prima e in ambito informatico poi, “l'insieme completo dei caratteri di uno stesso tipo”
- Sono una delle caratteristiche più importanti per imporre uno stile al vostro sito
  - Può cambiare **totalmente** il messaggio che volete trasmettere!



*Sto arrivando da te*

STO ARRIVANDO DA TE

# SCELTA DEI FONT - OBIETTIVO

- Qual è il vostro obiettivo?
- Decidere la reazione che vogliamo trasmettere
  - Alcuni font più leziosi vanno bene per argomenti leggeri o sono affini ad una grafica naturalistica
  - Altri font giocosi (ma per favore evitate sempre il *Comic Sans*!) sono perfetti per progetti con bambini
  - Altri più minimalisti per progetti di grafica
  - Font più spessi sono solitamente associati alla sfera maschile







VS



# SCELTA DEI FONT - GRAZIE



**CARATTERE  
CON GRAZIE**  
(inglese: Serif)  
in colore rosso le grazie



**CARATTERE  
SENZA GRAZIE**  
(inglese: Sans Serif)

# SOTTOCLASSIFICAZIONE DEI FONT

- Storicamente esistono due grandi famiglie di font
  - Con grazie (Serif)
    - Es. Times New Roman, Courier New, Merriweather
  - Senza grazie (Sans Serif)
    - Es. Arial, Calibri, Roboto
- Ci sono però anche delle sotto categorie
  - Slab Serif (font dove le grazie sono molto marcate e geometriche)
    - Es. **Bevan**, **BioRhyme**, Josefin Slab, Arvo, **Peralta**
  - Monospace (font in cui tutti i caratteri occupano lo stesso spazio)
    - Es. Courier New, Lucida Console, Aptos Mono
  - Calligrafici (Handwriting o Script, che imitano la scrittura a mano)
    - Es. *Lucida Handwriting*, *Palace Script*, **Brush Script**, *Vivaldi*, *Delight Sunset*
  - Gotici, decorativi o ornati
    - Es. **BLADE RUNNER**, **Klibi**, Chiller, **Matura**, **Curly**, **Magneto**, Old English, Playbill, **Purrelike**, **ROSEWOOD STD**, **STCaigun**

# LOGO VS TESTO

- Un logo deve essere di impatto e facile da identificare
- Un titolo deve funzionare bene per grandi dimensioni (font display, es. **Helvetica**))
- Un testo deve essere leggibile
  - La leggibilità dipende anche dal mezzo: per la carta stampata di solito si usano caratteri con grazie (es. Times New Roman e Garamond)
- Ci sono degli aspetti oggettivi da considerare (leggibilità), ma dobbiamo considerare anche aspetti soggettivi, come l'estetica e l'adeguatezza
  - Il font rispetta le aspettative estetiche delle persone con cui dovrà interfacciarsi? Es. font per una pubblicità di un'agenzia assicurativa o un'agenzia di viaggi



# LEGGIBILITÀ

- La leggibilità di un carattere (**legibility**) ha a che fare con la struttura grafica del singolo carattere tipografico (quanto è facile distinguere un carattere dall'altro?) come:
  - lo spessore,
  - la presenza o meno di grazie,
  - la crenatura (kerning),
  - la spaziatura (tracking),
  - l'interlinea (leading).
- Leggibilità di un testo (**readability**)



# CRENATURA

- La crenatura indica una riduzione dello spazio in eccesso fra coppie specifiche di caratteri, attuata al fine di diminuire spazi bianchi antiestetici e dare un aspetto più omogeneo al testo

AV Wa  
Senza crenatura

AV Wa  
Con crenatura



# TRACKING

- Il tracking (avvicinamento) indica l'ampliamento o la riduzione della spaziatura tra i caratteri nel testo selezionato o in un blocco intero di testo

Applied to a block

We ran into a fog bank on  
the second day out from  
Boston, and for seven days  
thereafter

Applied to a block

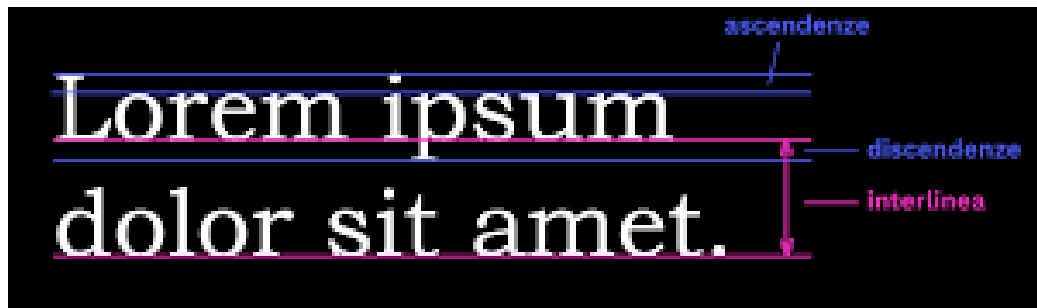
We ran into a fog bank on  
the second day out from  
Boston, and for seven  
days thereafter

Applied to a block

We ran into a fog bank  
on the second day out  
from Boston, and for  
seven days thereafter

# INTERLINEA

- Per interlinea intendiamo lo spazio verticale che separa due righe di testo. Ancora meglio, lo spazio che separa le due linee di base su cui poggiano due righe successive.
  - La linea di base è la linea invisibile su cui si poggiano le lettere senza (parti) discendenti





# READABILITY

- Analisi di tutte le proprietà appena viste (compreso il colore e altre proprietà), che collaborano nel creare un aspetto complessivo

## Frutiger

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa.

Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus.

Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem.

Nulla consequat massa quis enim.

Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis



# READABILITY (2)

- Scegliere font progettati per l'uso di cui avete bisogno
- Allineare il testo a bandiera, e possibilmente a sinistra
- Dividere il testo in paragrafi
- Utilizzare in modo corretto l'interlinea (di solito, per i testi si usa un'interlinea pari al 120% del corpo del font)
- Utilizzare nel modo corretto peso e dimensioni del testo



# READABILITY (2)

- Scegliere font progettati per l'uso di cui avete bisogno
- Allineare il testo a bandiera, e possibilmente a sinistra
- Dividere il testo in paragrafi
- Utilizzare in modo corretto l'interlinea (di solito, per i testi si usa un'interlinea pari al 120% del corpo del font)
- Utilizzare nel modo corretto peso e dimensioni del testo



# UMORE

- L'umore di un font è qualcosa di fortemente condizionato dal **messaggio** che viene percepito. Ad esempio un font può trasmettere delle sensazioni di eccitamento, di panico o di rilassatezza ma, durante la lettura di quello che è effettivamente scritto, la percezione può andare ad un altro livello e l'umore generale cambia.

Vag Rounded

**KICK BOX  
E RELAX**

Impact

**KICKBOX  
E RELAX**



# UMORE

- L'umore di un font è qualcosa di fortemente condizionato dal **messaggio** che viene percepito. Ad esempio un font può trasmettere delle sensazioni di eccitamento, di panico o di rilassatezza ma, durante la lettura di quello che è effettivamente scritto, la percezione può andare ad un altro livello e l'umore generale cambia.

Vag Rounded

**KICK BOX  
E RELAX**

Impact

**KICKBOX  
E RELAX**

**KICKBOX  
E RELAX**



# ALCUNI SUGGERIMENTI

- Decidere di quanti font abbiamo bisogno
- Guardare quello che è già stato fatto
- Fare esperimenti (graduali)
- Evitare i cliché
- Utilizzare le famiglie di font
- Partire dai font più popolari e dalle combinazioni più popolari



# QUANTI FONT DIVERSI?

- Cercare di capire quanto contenuto, quanto testo dobbiamo scrivere con i font selezionati e decidere di quanti caratteri abbiamo bisogno per scrivere quel testo
  - Ad esempio, possiamo decidere che serve un font per i titoli, uno per le didascalie e uno per i testi descrittivi
- Dobbiamo domandarci se basta un font che, tramite le sue varianti di peso o inclinazione, permette un soddisfacente risultato, oppure se pensiamo che sia meglio selezionare due o tre (meglio non più di tre) per garantire una maggior differenziazione



# GUARDARE QUELLO CHE È GIÀ STATO FATTO

- Cercare di capire che cosa ha già funzionato in passato per progetti simili al nostro e **prendere spunto e ispirazione**
  - Non c'è il copyright sulle combinazioni di font, quindi se vediamo in un progetto un'accoppiata vincente come la classica Georgia + Verdana perché non riutilizzarla?
- Il sito [Fontsinuse.com](https://fontsinuse.com) mostra numerosi esempi di come i font sono stati utilizzati da altri designer e progettisti





# FARE ESPERIMENTI GRADUALI

- Provare a variare uno spessore o un'inclinazione e osservare il risultato per capire cosa funziona e cosa no
- Cercare di fare una sola modifica alla volta per evitare di perdere possibili combinazioni perfette nella fretta di modificare



# EVITARE I CLICHÉ

- Cioè le scelte troppo banali e scontate:
  - Non usare il *Papyrus* solamente perché stiamo lavorando su qualcosa che abbia a che fare con “l’antichità” (magari non usarlo proprio)
  - Non usare il **Comic Sans MS** solamente perché stiamo lavorando su qualcosa di giocoso o divertente (magari non usarlo proprio, parte II)
  - Non usare il **TRAJAN PRO** solamente perché stiamo lavorando su qualcosa che ha a che fare con il mondo latino o romano



# USARE LE FAMIGLIE DI FONT

- Se abbiamo bisogno di più di un font ma contemporaneamente di ordine e di un unico umore, prova a ricercare famiglie di font, cioè **font con grandi varianti di peso e larghezza**
  - ad esempio la famiglia Meta progettata nel 2003 da Erik Spiekermann che comprende 28 pesi e larghezze diversi o l'Univers di Adrian Frutiger del 1956 che ne aveva 24.
    - Univers
    - Univers Light
    - **Univers Bold**
    - Univers Condensed
    - Univers Condensed Light



# USARE (O COMINCIATE DA) FONT POPOLARI

- Helvetica (uno dei più usati, forse troppo)
- Futura (logo Volkswagen e la targa lasciata sulla Luna)
- Frutiger
- Gotham (campagna Yes we can! Di Obama)
- Baskerville (creato nel 1757, molto elegante)
- Akzidenz-Grotesk
- Gill Sans (BBC e British Railways)
- Garamond
- Bodoni



# E COMBINAZIONI DI FONT CHE FUNZIONANO

- Di solito funziona bene una combinazione di un font con grazie e uno senza grazie (es. Verdana + Georgia , Futura + Bodoni o Myriad + Minion)
- I font devono avere lo stesso umore
- Ma non essere troppo simili!
- **Non più di tre font diversi**





# GOOGLE FONT

- Un ultimo consiglio utile è quello di usare i Google Font
  - ampia scelta di font, per tutte le occasioni
  - per massimizzare la compatibilità con i vari browser
- Su <https://fonts.google.com/> trovate tutti i font
  - Potete filtrare per varie categorie (es. solo senza grazie, famiglie di font che abbiano almeno X varianti), e ordinare i font per popolarità
- Potete usarli direttamente nel vostro sito

```
<link href="https://fonts.googleapis.com/css?family=Merriweather+Sans:400,700" rel="stylesheet"/>
```



# FONT: ULTIMA RACCOMANDAZIONE

- Cercate di indicare più font, in caso quello scelto non fosse disponibile
- Il modo più semplice è con la proprietà font-family, attraverso la quale è possibile indicare diversi font, separati da una virgola
  - I primi nell'ordine, se disponibili nel sistema dell'utente, avranno la precedenza
- È comunque opportuno, a fine dichiarazione, **indicare la famiglia generica dei font nel caso nessuno dei font indicati fosse disponibile**
- In caso il vostro font fosse composto da più parole, va indicato tra virgolette
- Ecco un esempio, che imposta il font dell'intera pagina con caratteri sans-serif (senza grazie):

```
body{font-family: "Trebuchet MS",Arial,Helvetica,sans-serif}
```





# FORM

- Il tag form è utile per **interagire con l'utente**, raccogliendo input (es. i dati di contatto)
- L'attributo "name" serve per indicare il nome del form, "action" indica l'URL del programma o della pagina di risposta che processerà i dati
- Grazie all'action è anche possibile far sì che i dati vengano inviati in e-mail al webmaster (si tratta infatti a tutti gli effetti di un riferimento a un URL)

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

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



# GET E POST

- Sono due modi diversi per inviare i dati
- Con il metodo GET la pagina di risposta viene contattata e i dati vengono inviati in un unico step.
  - Nell'URL della pagina di risposta potremo allora vedere tutti i parametri nella barra degli indirizzi (più precisamente nella "query string") :
  - Es. `paginaRisposta.php?nome=Francesca&cognome=Pratesi&datiInviati=prova+invio`
- Alcuni server hanno delle limitazioni per quel che riguarda il metodo GET e non consentono di inviare form con valori superiori a 255 caratteri complessivi.
- Nel metodo POST invece l'invio dei dati avviene in due step distinti: prima viene contattata la pagina sul server che deve processare i dati, e poi vengono inviati i dati stessi
  - Per questo motivo i parametri non compaiono nella query string (dunque se non si desidera che i parametri siano mostrati all'utente questo metodo è preferibile)
- In questo caso non ci sono limiti sulla lunghezza dei caratteri



# ENCTYPE

- Indica il **tipo di codifica**
- Una codifica è sempre necessaria, anche per i semplici testi (per cui, ad esempio gli spazi vengono convertiti in "+").
- Normalmente non è necessario specificare come si vuole effettuare la codifica dei dati, perché è sottinteso l'invio di semplice testo.
- A volte però, come quando è necessario inviare un'immagine, è tuttavia indispensabile dichiarare espressamente quali dati vogliamo inviare

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



# FIELDSET E LEGEND

- Grazie al tag <fieldset> possiamo creare delle **macro-aree all'interno dei form**, e grazie al tag <legend>, possiamo indicare il nome di ciascun amacro-area.
  - Poniamo ad esempio di dover raccogliere i dati di un utente, raccogliendo dati anagrafici, residenza, domicilio e reperibilità sul lavoro.
  - Possiamo farlo con la seguente sintassi:

```
<form action=ìì>  
  <fieldset>  
    <legend>Dati anagrafici</legend>  
    <br><br><br>  
  </fieldset>  
  <fieldset>  
    <legend>Residenza</legend>  
    <br><br><br>  
  </fieldset>  
</form>
```



# IL TAG LABEL

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

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

Dati anagrafici

Anno di nascita:



# INPUT

- Il tag più utilizzato nelle form è l'<input>
  - È un tag senza chiusura.
- Per specificare un determinato tipo di campo è sufficiente indicare il tipo di input con l'attributo "type"
- Per creare una casella di testo:  
`<input type="text">`
- I vari <input> sono dotati di attributi che consentono di indicare il tipo di campo, il nome (ad esempio per interagire con JavaScript), e il valore (per lo più il testo visualizzato).

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

che dà:

qui il tuo testo

# VARI TIPI DI INPUT

- `<input type="text">` → per creare una casella di testo
- `<input type="button">` → per creare un bottone (è possibile disattivarli con l'attributo `disabled="disabled"`)
- `<input type="submit" value="invia i dati">` → per creare un bottone per l'invio dei dati della form
- `<input type="image" src="invia.gif">` → speciale bottone di submit con un'immagine al posto del testo; è possibile specificare tutti gli attributi visti per `img` (es. `alt`, `width`, `height`)
- `<input type="password">` → casella di testo in cui il testo inserito viene offuscato



# VARI TIPI DI INPUT - LE SCELTE MULTIPLE

- È possibile anche guidare l'input dell'utente, facendolo scegliere tra una serie predefinita di valori
- Ci sono tre modi per fare questo:
  - checklist: consentono all'utente di selezionare più valori
    - Lo standard di visualizzazione è un quadrato ☐ ☒
  - radio button: consentono all'utente di effettuare una scelta esclusiva
    - Lo standard di visualizzazione è un cerchio ☐ ☒
  - menù a tendina: di default consentono di selezionare un solo valore, ma è possibile specificare la possibilità di selezione multipla





# CHECKBOX

```
<form action="">
  <fieldset>
    <legend>Linguaggi conosciuti</legend><br>
    <input type="checkbox" name="html" value="html"/> html<br />
    <input type="checkbox" name="css" value="css"/> css<br />
    <input type="checkbox" name="javascript" value="javascript"/> JavaScript
  </fieldset>
</form>
```



# RADIO BUTTON

```
<form action="">
  <fieldset>
    <legend>Linguaggio più usato</legend>
    HTML <input type="radio" name="linguaggio" value="html"/>
    CSS <input type="radio" name="linguaggio" value="css"/>
    JavaScript <input type="radio" name="linguaggio" value="javascript"/>
  </fieldset>
</form>
```



# MENÙ A TENDINA (SELECT)

```
<form>
  <fieldset>
    <legend>Siti per webmaster</legend>
    <select name="siti" >
      <option value="http://www.html.it" selected="selected">www.html.it
</option>
      <option value="http://freephp.html.it">frephp.html.it </option>
      <option value="http://freasp.html.it">freasp.html.it </option>
    </select>
  </fieldset>
</form>
```



# ESERCIZIO

- Pensate ad un sito web
  - Può essere un sito su un vostro hobby, o una pagina personale su di voi
- Iniziate a raccogliere idee sul mood che volete trasmettere
  - Fate una prima ricerca dei font e di colori di base

