esci

Basi per html5, fogli di stile css3 e Javascript

Editor

Link utili

.css e .js

Struttura della pagina

Principali tag

Css3 e formattazione

Id- Class- selettori

Audio e video

Posizionamento (position-float-colonne)

Collegamenti

Tabelle

Elenchi

Resize- zoom

Impostare un menù

Finestre con tab

Esempio con Javascript

Editor html

Per scrivere pagine html5 utilizziamo la modalità di scrivere codice direttamente sugli editor di testo presenti su computer.

Editor free del tipo wysiwyg esistono ovviamente ma ci impediscono di apprendere la sintassi e le regole, inoltre hanno sempre necessità di intervento diretto sul codice.

Blocco note→ Windows (Avvio→ programmi→ accessori→ blocco note)

Textedit→ Mac (cartella delle Applicazioni)

Su **Linux** gli editor più gettonati sono *Gedit* (predefinito per Gnome), *Kate* o *KWrite* (predefiniti per KDE), e sono sempre inclusi nel sistema base

Editor

- Blocco note→ dopo aver aperto il blocco note→ scrivere le istruzioni e quindi salvare…

  l'estensione del file DEVE essere .html→ esempio di file salvato: **prova.html**
- Il file così salvato sarà visualizzato con icona del browser presente sul tuo pc
- Per aprire e visualizzare il file basta fare doppio click
- Per modificare il file si deve fare click destro→ apri con→ blocco note

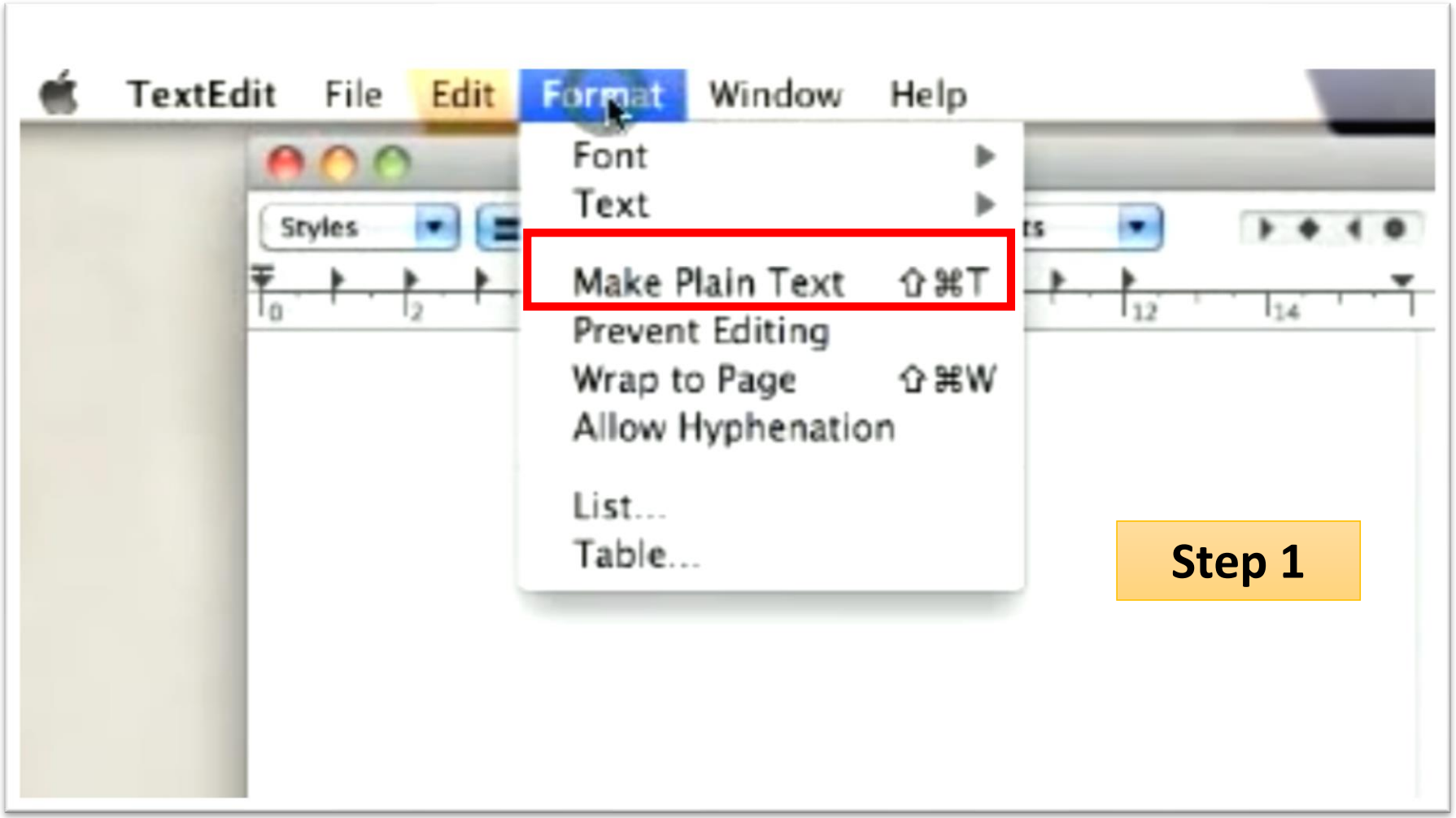
Editor

- Textedit→ richiede che il testo sia formattato come→ plain text **(STEP 1)**
- Clicca su→ textedit→ preferenze→ plain text **(STEP 2)**

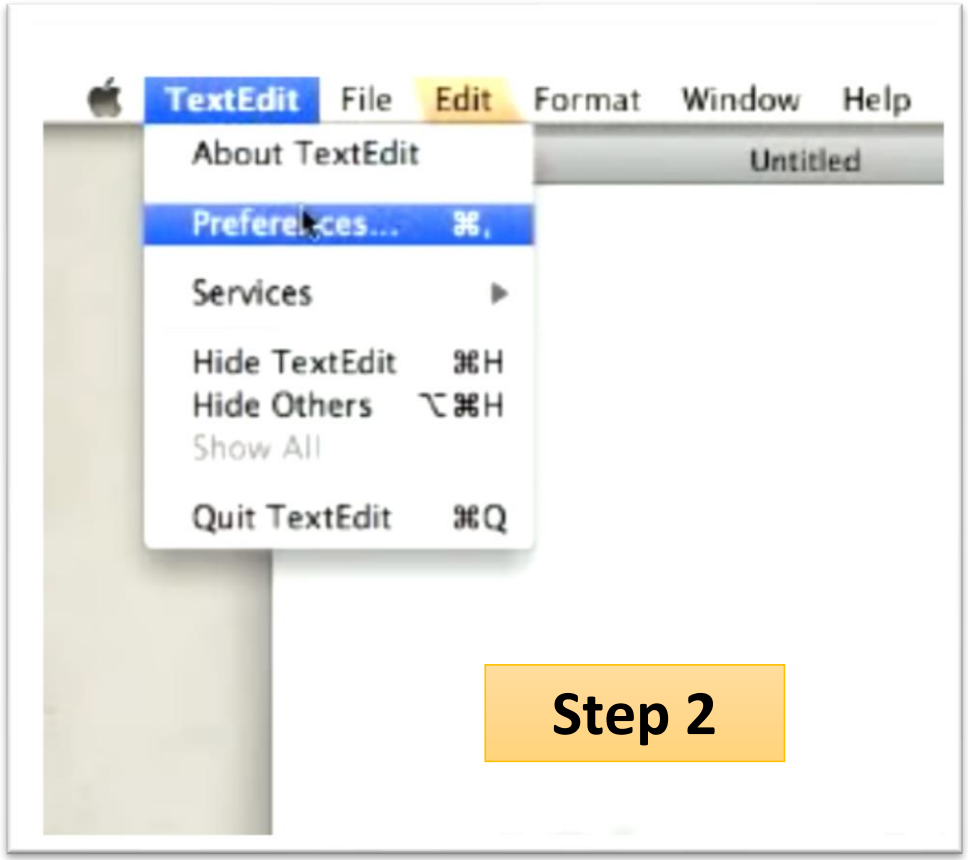
- Clicca su→ formato → make plain text e su → Edit/modifica→ ignora rich text **(STEP 3-4)**
- Salvare come .html (nella finestra di dialogo che si apre)→ **(STEP 5-6)**

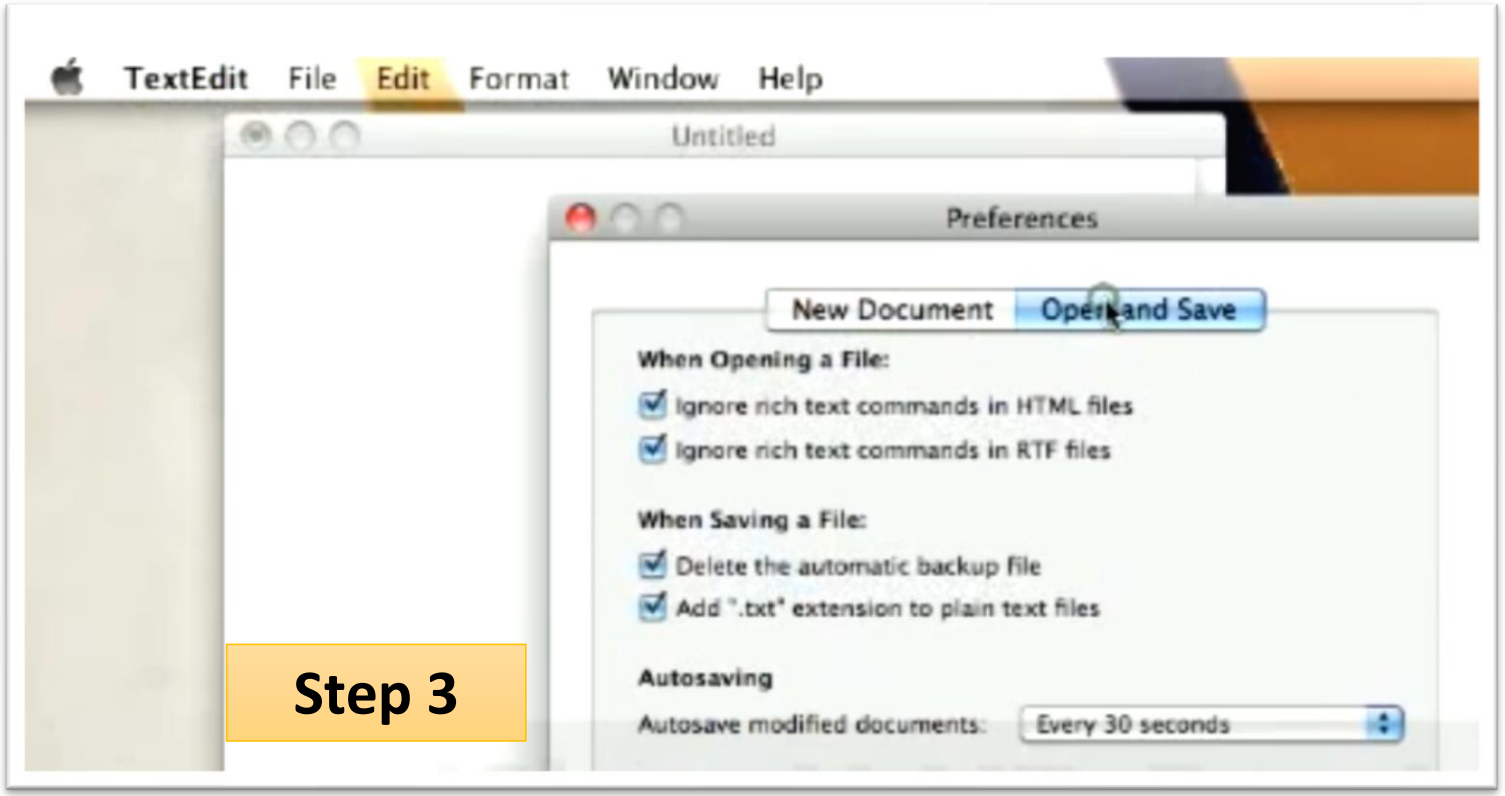
- https://www.youtube.com/watch?v=Ju12fK6jm6M

Editor

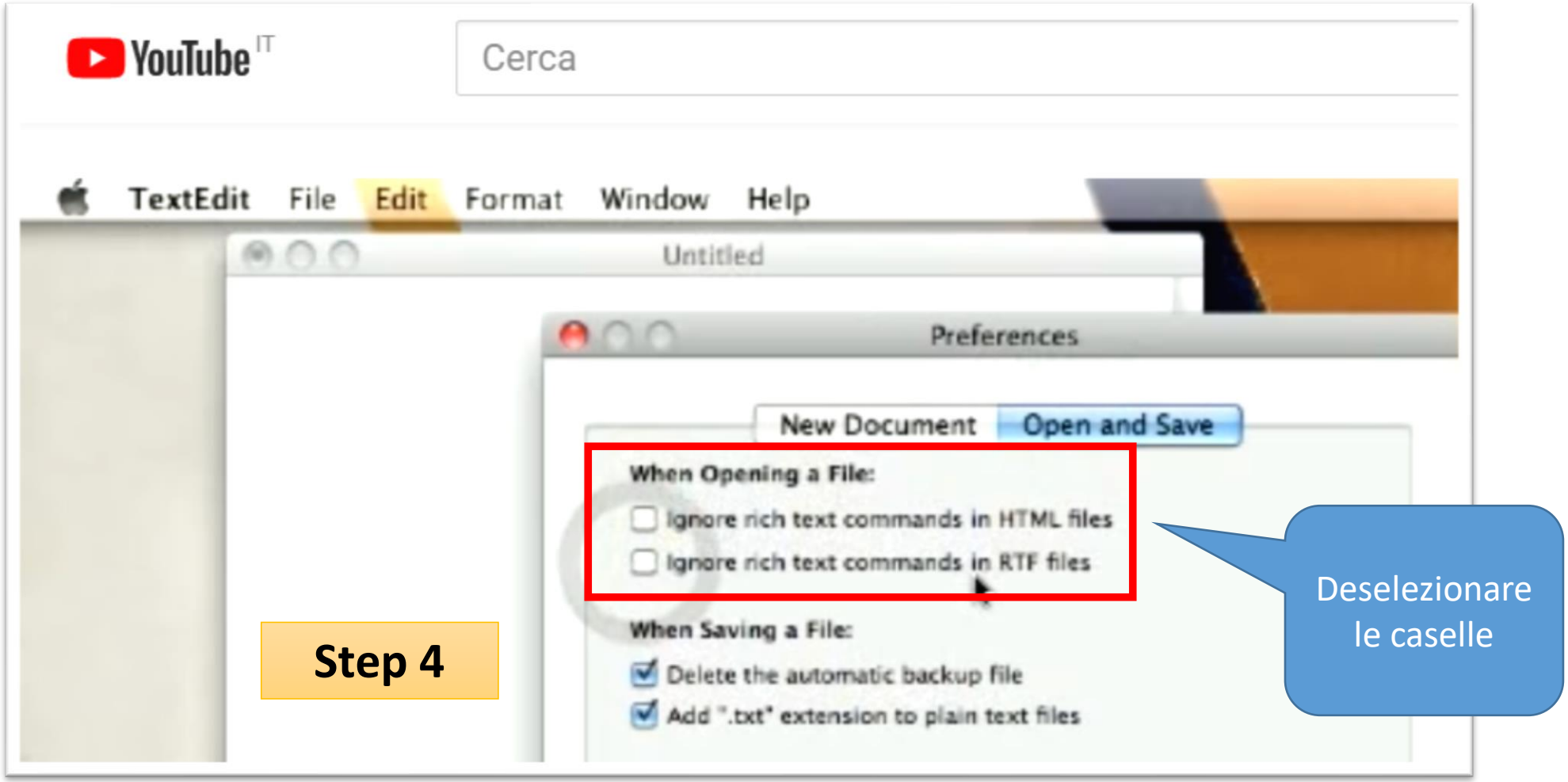

TextEdit
File
Edit
Format
Window
Help
Font
Text
Make Plain Text ⇧⌘T
Prevent Editing
Wrap to Page ⇧⌘W
Allow Hyphenation
List...
Table...
Styles
Step 1

Editor

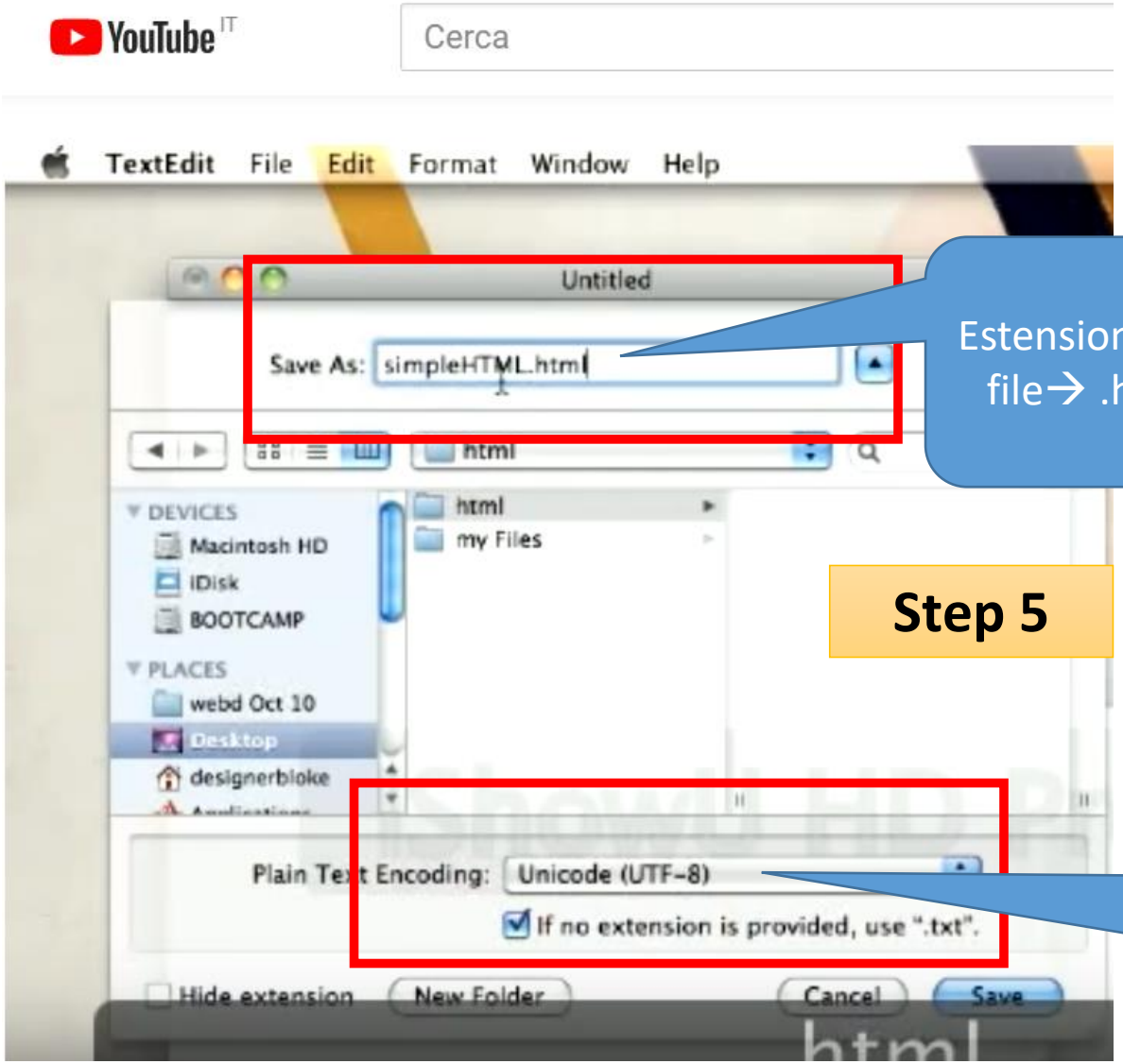

TextEdit
File
Edit
Format
Window
Help
About TextEdit
Preferences... ⌘,
Services
Hide TextEdit ⌘H
Hide Others ⌥⌘H
Show All
Quit TextEdit ⌘Q
Untitled
Step 2


TextEdit
File
Edit
Format
Window
Help
Untitled
Preferences
New Document
Open and Save
When Opening a File:
Ignore rich text commands in HTML files
Ignore rich text commands in RTF files
When Saving a File:
Delete the automatic backup file
Add ".txt" extension to plain text files
Autosaving
Autosave modified documents:
Every 30 seconds
Step 3

Editor


YouTube IT
Cerca
TextEdit File Edit Format Window Help
Untitled
Preferences
New Document
Open and Save
When Opening a File:
Ignore rich text commands in HTML files
Ignore rich text commands in RTF files
When Saving a File:
Delete the automatic backup file
Add ".txt" extension to plain text files
Step 4
Deselezionare le caselle

Editor

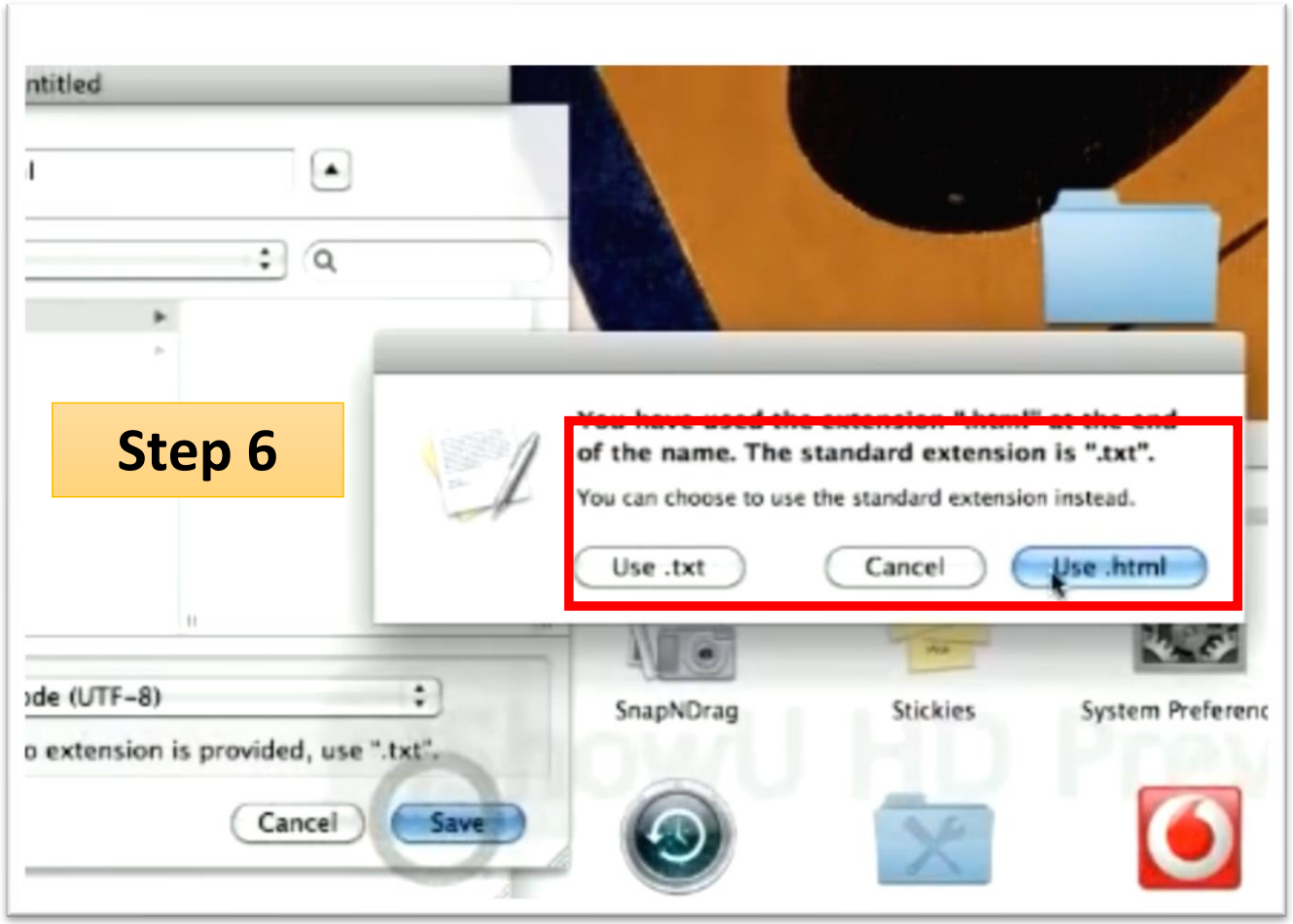


Estensione del file→ .html

Step 5

Unicode (UTF-8)

Editor


Step 6
of the name. The standard extension is ".txt".
You can choose to use the standard extension instead.
Use .txt
Cancel
Use .html
de (UTF-8)
o extension is provided, use ".txt".
Cancel
Save
SnapNDrag
Stickies
System Preferenc

Link utili

Per esempi e corretta sintassi di HTML- CSS3 e JQUERY:

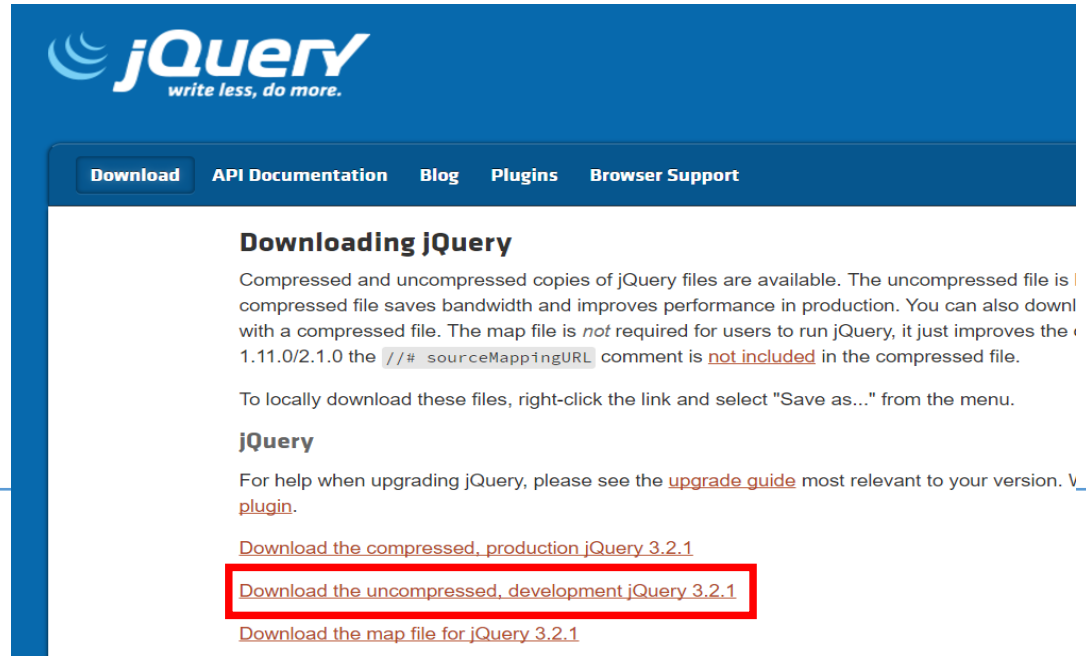
http://www.w3schools.com/html/default.asp

Per scaricare le librerie dei file di JQUERY (core)→ funzionamento base:

Per realizzazione di file in html5→ Scaricare la versione **uncompressed development**

jQuery 3.2.1 http://jquery.com/download/

per l'esempio con i «Tab» inseriti in questa Presentazione io ho utilizzato la vecchia versione 1.11.3



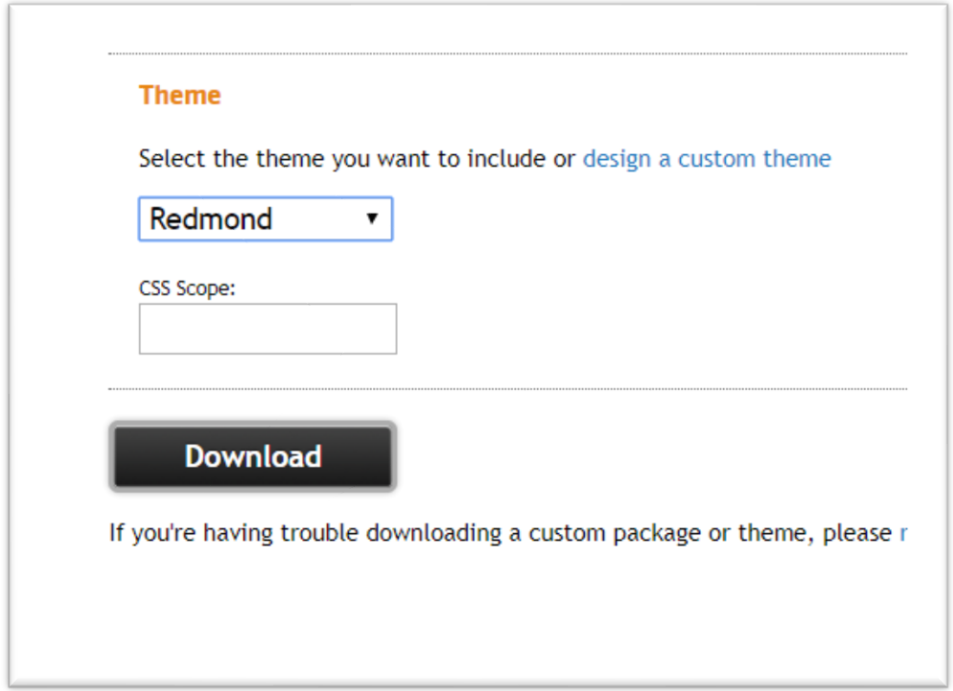
Link utili

Scaricare e scompattare (UNZIPPARE) la cartella di Jquery-UI dal sito **http://jqueryui.com/download/**

Attenzione: sul sito cliccare sull'etichetta DOWNLOAD→ version 1.11.4 (adesso siamo arrivati alla 1.12.1)

Lasciare selezionati tutti i componenti→ scendere in fondo alla pagina… scegliere il layout preferito (io credo di aver scelto REDMOND)…

cliccare su DOWNLOAD→ verrà scaricata una cartella zippata dal nome jquery-ui-1.12.1.custom ma con la versione di layout prescelto (cambiano i colori e il look)

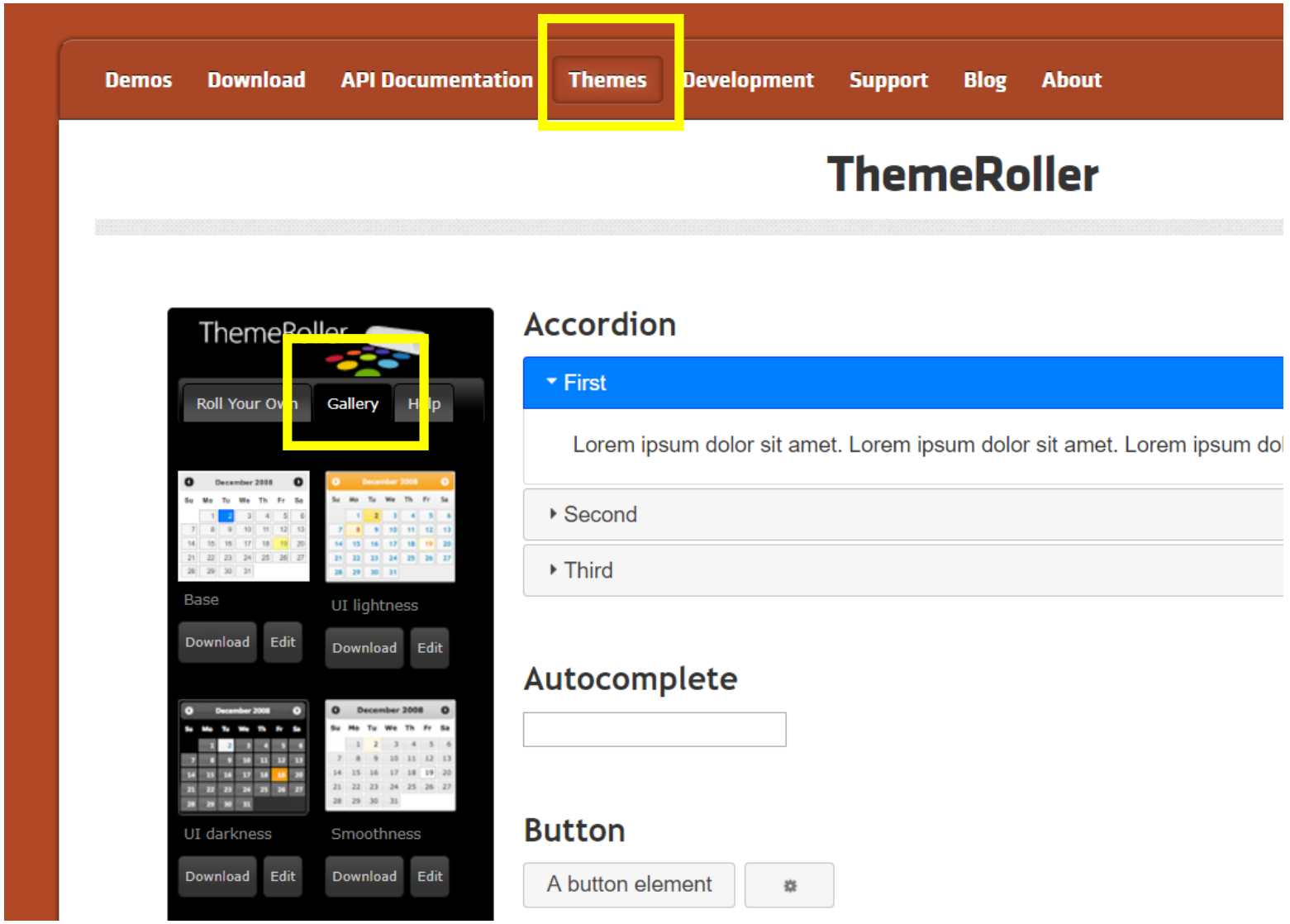


Questo invece è **il secondo file di Jquery** che **serve principalmente per l'aspetto** (colori e forme)

Link utili

OVVIAMENTE, prima di scaricare un «layout» bisogna vedere come è fatto e se piace, quindi…

Per visualizzare e scegliere i diversi stili cliccare prima su etichetta **THEMES** e sulla sinistra **utilizzare il THEME ROLLER-GALLERY**



Link utili

Se non avete notato, **si scaricano due file diversi di JQUERY**

Uno serve per il funzionamento «base» **jQuery 3.2.1** **http://jquery.com/download/**

Mentre l'altro file serve per caratterizzare il layout (il look→ colori e forme)

http://jqueryui.com/download/ (adesso siamo arrivati alla versione 1.12.1)

Questa versione infatti si chiama **UI→ user interface**

.css e .js

I **contenuti** (TESTO- IMMAGINI- VIDEO- AUDIO, etc.) vengono inseriti tramite codice nei file .html

La **formattazione e il layout** vengono ottenuti tramite la realizzazione di un file parallelo che si salva con estensione .css

Anche questo file si realizza con BLOCCO NOTE.

L'**interattività e i comportamenti degli oggetti** vengono ottenuti tramite la realizzazione di un file parallelo che si salva con estensione .js

Anche questo file si realizza con BLOCCO NOTE→ si tratta del linguaggio di programmazione in javascript

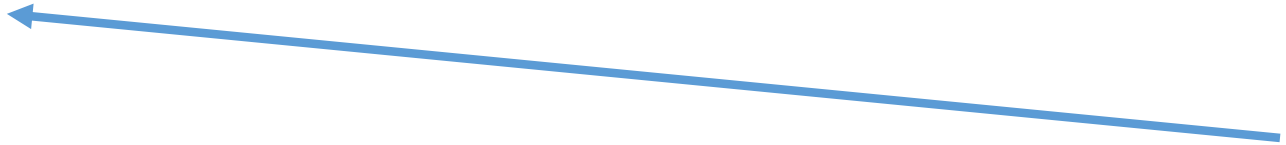
.css e .js

Alla fine, quindi, il progetto di solito è costituito da tre file:

.html

.css

.js



Dentro al file html si richiamano con un collegamento **gli altri due file**.

Oltre ai tre file di cui abbiamo parlato, solitamente si inserisce anche un file di **Jquery**, che è una libreria con dei comandi già predisposti da una comunità di programmatori e che semplifica il nostro lavoro di programmazione.

Inoltre si inserisce un file di **Jquery-UI**, che è un'altra libreria che possiede dei layout/comportamenti già predisposti (menù, pulsanti, slideshow, etc.).
UI= User Interface

.css e .js

Sia Jquery (core) che Jquery-UI hanno i loro file .css collegati, quindi alla fine il nostro progetto, se formato da una sola pagina, in verità avrà i seguenti file:

File1.html (fatto da noi con i contenuti)

File2.css (fatto da noi con il layout e la formattazione)

File3.js (fatto da noi con istruzioni in javascript)

Realizzati da noi

File4.js + file5.css (forniti da Jquery e da richiamare dentro al progetto)

File6.js + file7.css (forniti da jquery-UI e da richiamare dentro al progetto)

Jquery

Come richiamare questi file lo vediamo nelle slide dedicate alla struttura della pagina HTML

Struttura della pagina

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title> Titolo del documento-file </title>
</head>
<body>
Contenuto del documento-file
</body>
</html>
```

Ogni tag deve essere aperto e chiuso→
<head>
</head>

Tag <title> è opzionale

Tra i tag di apertura e chiusura di HEAD si inseriscono istruzioni che NON vengono visualizzate ma che sono molto importanti… qui si inseriscono anche i riferimenti ai file .css e .js

Tra i tag di apertura e chiusura di BODY si inserisce tutto il contenuto che si vedrà a video

Struttura della pagina→ codice tra i tag HEAD

```
<head>
   <meta charset=utf-8">
   <title>zoom & resize</title>
   <script type="text/javascript" src="js/jquery-1.11.3.js"></script>
   <script type="text/javascript" src="js/jquery-ui.js"></script>
   <script type="text/javascript" src="js/zoom_codice.js"></script>
   <link rel="stylesheet" href="css/zoom_stili.css" type="text/css" />
   <link rel="stylesheet" href="css/jquery-ui.css" type="text/css" />
</head>
```

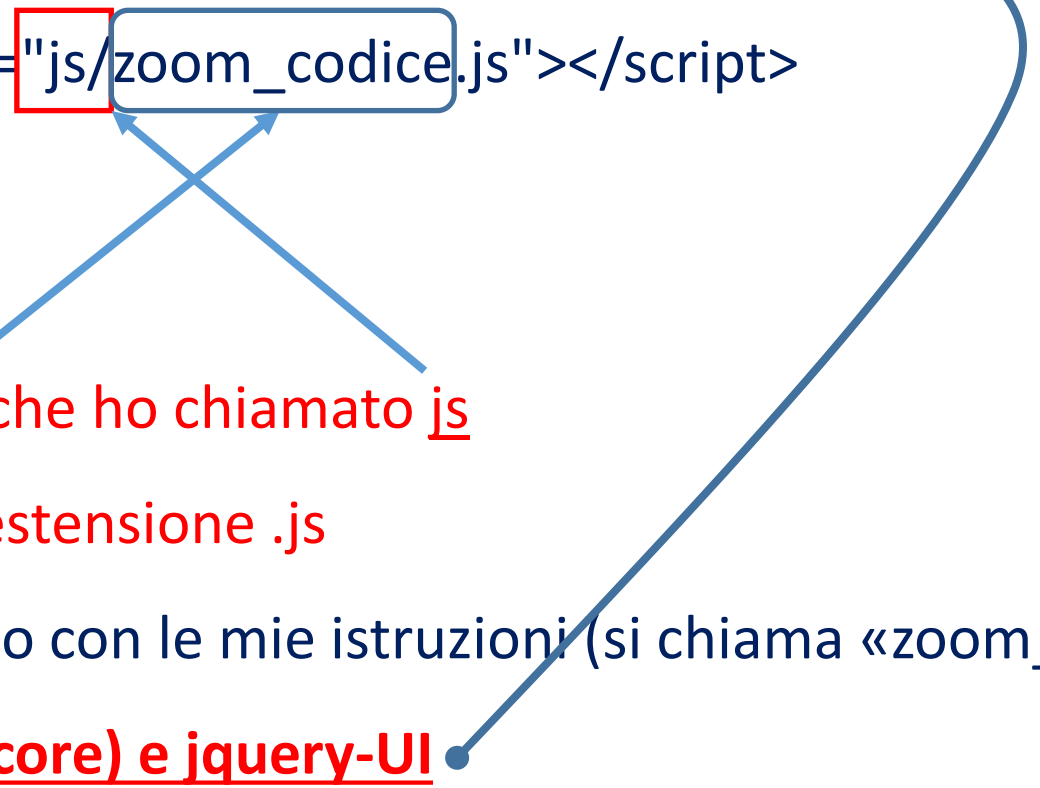
Richiamo dei file js con il percorso

Richiamo dei file css con il percorso

Prossima pagina analizziamo la porzione di codice

Struttura della pagina→ analisi dei file javascript

```
<head>
<script type="text/javascript" src="js/jquery-1.11.3.js"></script>
<script type="text/javascript" src="js/jquery-ui.js"></script>
<script type="text/javascript" src="js/zoom_codice.js"></script>
</head>
```



I file sono inseriti in una **cartella** che ho chiamato js

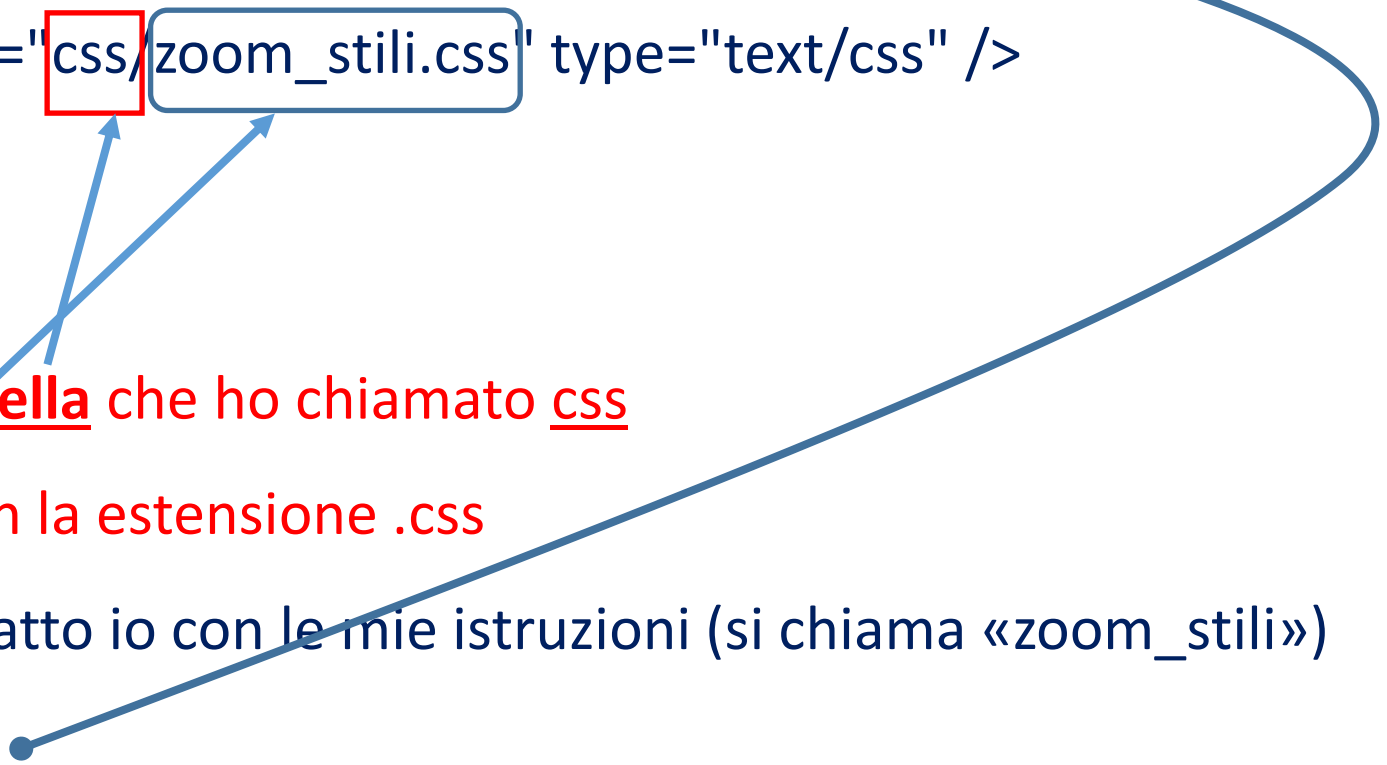
Poi segue il **nome del file** con la estensione .js

Come vedete→ un file l'ho fatto io con le mie istruzioni (si chiama «zoom_codice»)

Gli altri due sono i file **di jquery (core) e jquery-UI**

Struttura della pagina→ analisi dei file css

```
<head>
<link rel="stylesheet" href="css/jquery-ui.css" type="text/css" />
   <link rel="stylesheet" href="css/zoom_stili.css" type="text/css" />
</head>
```



I file sono inseriti in una **cartella** che ho chiamato css

Poi segue il **nome del file** con la estensione .css

Come vedete→ un file l'ho fatto io con le mie istruzioni (si chiama «zoom_stili»)

L'altro è un file **di jquery-UI**

Struttura della pagina→ title e charset

```
<head>
<meta charset="UTF-8">
<title> Titolo del documento-file </title>
</head>
```

Il title **è visibile** MA non dentro al documento quanto piuttosto **nella barra del browser**.

Per quanto riguarda il <meta charset="UTF-8"> specifica solo l'ENCODING dei caratteri per html5 ed **evita che vi siano dei caratteri non visualizzati a schermo**, come a volte accade…

Avete mai visto un sito con alcuni caratteri sostituiti da questa immagine?

Il charset non era stato dichiarato o dichiarato in modo errato.

Principali tag all'interno del body

NOTA BENE: tutto il codice viene letto e interpretato (sia quello html che quello Javascript) dall'alto verso il basso in modo sequenziale

TAG

 testo testo testo il tag inserisce un paragrafo e questo produce un'interlinea maggiore sia prima che dopo il testo→ si tratta di **un elemento «Block»…** non ammette un altro elemento in linea, a fianco di sè stesso

TAG

testo testo testo
  il tag <br> non richiede chiusura **e inserisce un «a capo»** che altrimenti viene ignorato dai browser→ se noi scriviamo una frase andando a capo nel BLOCCO NOTE, ciò NON si vede nel browser… serve istruzione

Principali tag all'interno del body

TAG <hr>

testo testo testo<hr> il tag <hr> non richiede chiusura **e inserisce una linea orizzontale di separazione**

TAG <h1><h2><h3><h4><h5><h6>

Sono i Tag che definiscono i titoli con dimensioni prestabilite (h1 il più grande→ h6 il più piccolo)→ questa modalità dà la possibilità al browser di interpretare la gerarchia del testo

Principali tag all'interno del body

TAG

È un tag «contenitore» che serve per raggruppare del contenuto e così poterlo controllare con le istruzioni CSS

È un tag di tipo «block»→ inserisce una interlinea maggiore prima e dopo di sé e può essere controllato per dimensione in larghezza con istruzioni CSS… non ammette un altro elemento in linea, a fianco di sè stesso

TAG

È un Tag che rende un testo «enfatizzato»→ praticamente mette in corsivo il testo compreso tra *testo testo testo*

Principali tag all'interno del body

TAG <mark>

È un Tag che rende un testo «evidenziato» → come fosse evidenziato → di default in giallo

TAG

Individua una porzione di testo, che poi può essere formattato con gli stili CSS →

testo testo testo **** *testo testo testo* **** testo testo testo

NOTA BENE… non è un Tag ma si utilizza a volte per risolvere **problemi di impaginazione**

** ** serve per inserire uno spazio bianco perché il **linguaggio html NON legge gli spazi bianchi,** cioè anche se ce ne sono due/tre ne legge sempre UNO SOLO

Principali tag all'interno del body

Principali tag «block»

<p> <table>   <h1>… <h6>

Elenchi puntati-numerati

Principali tag «inline»

 <img> <a> <b> <u> <i> <sup> <sub> <cite>

immagine **collegamento** **Formattazione «in linea»**

Principali tag all'interno del body

FORMATTAZIONE «INLINE»

Dentro al tag…

Style=«property: value; property: value;»

Esempio

<p style=«color: green; font-size: 24px;»> testo

** <u> <i>** si possono utilizzare all'interno del linguaggio di markup html

Esempio

Questo è <b style=«color: red»><u>un esempio</u>

Questo è **un esempio**

Principali tag all'interno del body

Differenze tra tag «block» e tag «inline»:

I **Tag «block»** non ammettono un elemento accanto a sé (a meno che non si intervenga con una istruzione «float»)… **si controllano con width** (larghezza)

I **Tag «inline»** ammettono un elemento al proprio fianco, **NON accettano proprietà width**

I **Tag «inline-block»** ammettono un elemento al proprio fianco MA **possono essere controllati attraverso l'imposizione di una larghezza specifica→ WIDTH** (larghezza)

Si può modificare l'impostazione di default intervenendo con i fogli di stile e l'istruzione…

Display: (block) (inline) (inline-block) (none)→ **con disply «none» l'elemento diventa invisibile e, anche se presente, non occupa fisicamente posto**

http://www.w3schools.com/css/tryit.asp?filename=trycss_display_none

Box model

http://www.w3schools.com/css/tryit.asp?filename=trycss_boxmodel

Per **calcolare la larghezza** di un elemento di tipo Block si devono tenere in considerazione 4 parametri:

Width (larghezza solo del contenuto)

Padding (distanza contenuto-bordo)

Border (larghezza del bordo)

Margin (distanza tra un elemento e l'altro

Esempio→ width=300px; padding= 15px; border= 2.5px; margin=7.5px;

Totale larghezza=

300+15+15+2.5+2.5+7.5+7.5=350px

CSS e gestione Font

Font-family:

Family name	Generic family
Times New Roman- Georgia	Serif (grazie)
Arial- Helvetica- Verdana- Tahoma	Sans serif (senza grazie)
Courier-Lucida Console	Monospace (dimensioni uguali delle lettere)

Si inserisce anche un «generic family» nel caso non venga trovato il font specificato

CSS e gestione Font

font-family: arial, verdana, sans-serif;

font-size: 16px; [dimensione di default nei browser]

font-weight: bold; [normal→ impostazione di default]→ si possono anche utilizzare dei numeri (da 100 a 900)→ font-weight: 700;

Nei browser più diffusi «normal» equivale a 400, «bold» a 700

font-style: italic;

font-height: altezza della riga (interlinea)→ in pixel o misura relativa

color: red; normalmente si utilizzano i colori esadecimali→ #ff0000 (colore rosso)

letter-spacing: 1px; (si possono anche utilizzare le misure relative→ vedi slide successive)

word-spacing: 5px;

CSS e gestione Font

```
text-decoration: underline; [overline-   line-through]

-moz-text-decoration-color: red; /* Code for Firefox */

text-decoration-color: red;

text-indent: 50px;
```

<h1> 32px

<h2> 24px

<h3> 19px (18,72)

<h4> 16px

<h5> 13px (13,28)

<h6> 11px (10,72)

Valori di default nei più comuni browser

Sono dei titoli e quindi hanno associato solitamente anche l'impostazione «grasseto»→ bold

https://www.w3schools.com/tags/tag_hn.asp

CSS e dimensioni relative

body {font-size: 100%} **spesso si usa questo codice perché…**

… permette di settare il font con dimensione pari a quella decisa dal browser→

solitamente **per default= 16px**

Oltre alle percentuali, spesso si utilizzano gli «em» e i «rem»

em= dimensione relativa alla dimensione del font «genitore»

rem= dimensione relativa alla dimensione del font «stabilito nella root» (solitamente in body)→ vedi esempio (file «rem-em.html»)

CSS e gestione Bordi

border-style values:

none: Defines no border

border: dotted;

dotted: Defines a dotted border

border: dashed;

dashed: Defines a dashed border

border: solid;

solid: Defines a solid border

border: double;

double: Defines two borders. The width of the two borders are the same as the border-width value

border: groove;

groove: Defines a 3D grooved border. The effect depends on the border-color value

border: ridge;

ridge: Defines a 3D ridged border. The effect depends on the border-color value

border: inset;

inset: Defines a 3D inset border. The effect depends on the border-color value

border: outset;

outset: Defines a 3D outset border. The effect depends on the border-color value

Id Class e altri selettori

Abbiamo detto che attraverso i fogli di stile CSS si può formattare tutto il documento HTML

Per far questo però si deve essere in grado di selezionare parte del codice in modo che il foglio di stile «riconosca» su quale elemento applicare le istruzioni.

Ogni tag del codice può essere individuato con un «marcatore»→

si possono applicare più marcatori allo stesso tag, ma questo si fa solitamente quando si vogliono inserire controlli e formattazioni piuttosto complessi (nel nostro caso non lo faremo)

Id Class e altri selettori

Id → si tratta dell'abbreviazione per «identificatore»

class → si tratta dell'abbreviazione per «classe»

Pseudo classi → si tratta di una classe che si applica solo a elementi che si trovano in uno «stato», caso più comune di utilizzo sui link (:hover- :visited- :active)

Pseudo elementi → si tratta di un controllo specifico che si può applicare a «prima lettera» e «prima riga» di un elemento

Selettore :not → si tratta di un selettore per «raffinare» la selezione→ comprende un tipo di tag/elemento MA esclude altri

Id Class e altri selettori

Si crea un file CSS con «blocco note»→ basta salvare il file con estensione .css

Si richiama il file CSS dentro al file html→ tra i tag <head> e </head>

<link rel=«stylesheet» href=«percorso_file.css» type=«text/css» />

Il primo modo è quello di selezionare un tag «generico» e assegnare le istruzioni… in questo modo le istruzioni si applicheranno a tutti i tag di quel tipo→ esempio

p { ……. (istruzioni)→ esempio→

font-family: arial, verdana, sans serif;

font-size: 20px;

color: red;

}

ATTENZIONE

Istruzioni separate da **«punto e virgola»** altrimenti non funziona!!!

In questo caso tutti i paragrafi «p» del documento avranno la stessa impostazione grafica

Id Class e altri selettori

Le istruzioni possono essere date contemporaneamente a più tag→

p, div, span { ……. (istruzioni)→ esempio→

font-family: arial, verdana, sans serif;

font-size: 20px;

color: red;

}

Il secondo modo è quello di selezionare un tag «specifico» e assegnare le istruzioni… in questo modo le istruzioni si applicheranno solo e solamente a quel tag→ per identificare un tag specifico gli si deve assegnare un ID oppure una CLASSE→ esempio

<p id=«nome_inventato»>

Id Class e altri selettori

Qualora vi siano istruzioni che vanno in conflitto, esiste una gerarchia:

Id

Class

Pseudo classi

Nei file CSS

gli «id» si richiamano con questa sintassi→ #nome (cancelletto)

le «class» si richiamano con questa sintassi→ .nome (punto)

le «pseudo class» si richiamano con questa sintassi→ :nome (due punti)

gli «pseudo-elementi» si richiamano con questa sintassi→ ::nome (due punti-due punti)

Id Class e altri selettori

esempio:

- Nel file .html   …….
- Nel file .css

```
#daniele, .gino {
font-family: arial, verdana, sans serif;
font-size: 20px;
color: red;
}
```

In questo caso le istruzioni NON si applicano a tutti i tag MA SOLO al tag che si chiama «daniele» e al con nome classe «gino»

Id Class e altri selettori

esempio:

- Nel file .html <p id=«daniele»>
- Nel file .css

```
p:NOT(#daniele) {
font-family: arial, verdana, sans serif;
font-size: 20px;
color: red;
}
```

In questo caso le istruzioni si applicano a tutti i tag <p> MA NON al tag <p> che si chiama «daniele»

Id Class e altri selettori

Esempio (pseudo elementi):

- Nel file .html <p id=«daniele»>
- Nel file .css

```
#daniele::first-line {
font-family: arial, verdana, sans serif;
font-size: 20px;
color: red;
}
```

ATTENZIONE

::first-letter→ solo prima lettera

::before→ inserisce contenuto prima dell'elemento

::after→ inserisce contenuto dopo l'elemento

In questo caso le istruzioni si applicano al tag «daniele» SOLO per la prima riga

Id Class e altri selettori

Interessante uso di PSEUDO CLASSI:

- Nel file .html c'è una tabella (la vedremo meglio in seguito)

```
<table>
<tr>      si tratta della riga 1
<td> contenuto </td>     si tratta di una cella
<td> contenuto </td>     si tratta di una cella
</tr>
<tr>      si tratta della riga 2
<td> contenuto </td>     si tratta di una cella
<td> contenuto </td>     si tratta di una cella
</tr>
</table>
```

È risultata una tabella di due colonne e due righe… non avevo spazio… ma si poteva proseguire ed avere il caso di una tabella di 4 righe

Id Class e altri selettori

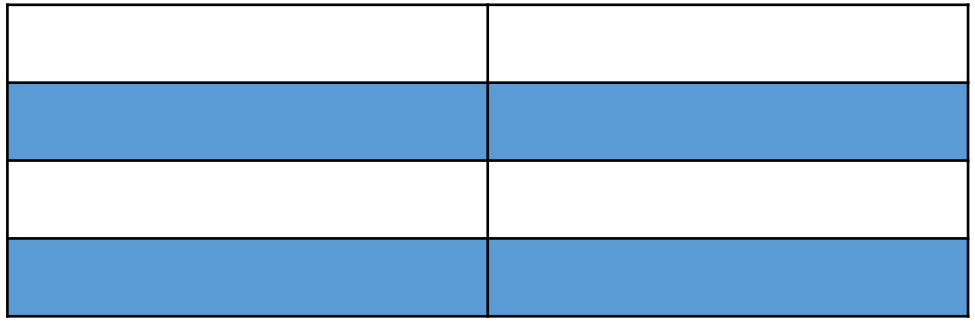
Interessante uso di PSEUDO CLASSI:

- Nel file .html c'è una tabella (la vedremo meglio in seguito)

```
<table>
<tr>    si tratta della riga 1
<td> contenuto </td>    si tratta di una cella
<td> contenuto </td>    si tratta di una cella
</tr>
<tr>    si tratta della riga 2
<td> contenuto </td>    si tratta di una cella
<td> contenuto </td>    si tratta di una cella
</tr>
<tr>    si tratta della riga 3
<td> contenuto </td>    si tratta di una cella
<td> contenuto </td>    si tratta di una cella
</tr>
<tr>    si tratta della riga 4
<td> contenuto </td>    si tratta di una cella
<td> contenuto </td>    si tratta di una cella
</tr>
</table>
```

È risultata una tabella di due colonne e quattro righe…

ADESSO VOGLIO
che le righe siano colorate in modo alternato
Come qui sotto



Id Class e altri selettori

Interessante uso di PSEUDO CLASSI:

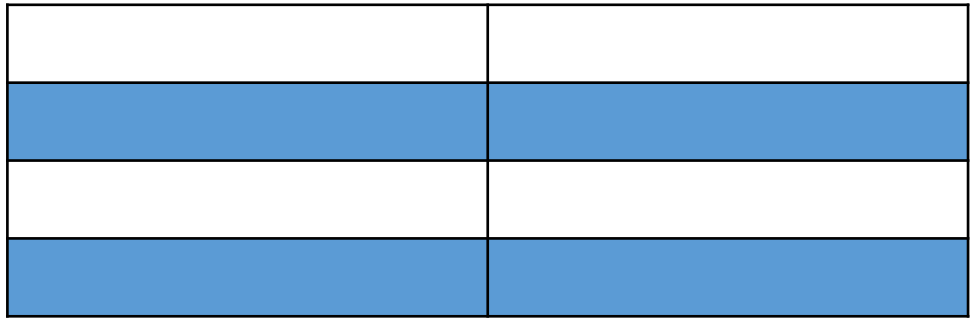
- Nel file .html c'è una tabella (ricordarsi che <tr> è una riga)

```
<table>
<tr>    si tratta della riga 1
<td> contenuto </td>    si tratta di una cella
<td> contenuto </td>    si tratta di una cella
</tr>
<tr>    si tratta della riga 2
<td> contenuto </td>    si tratta di una cella
<td> contenuto </td>    si tratta di una cella
</tr>
<tr>    si tratta della riga 3
<td> contenuto </td>    si tratta di una cella
<td> contenuto </td>    si tratta di una cella
</tr>
<tr>    si tratta della riga 4
<td> contenuto </td>    si tratta di una cella
<td> contenuto </td>    si tratta di una cella
</tr>
</table>
```

Istruzioni nel file CSS

```
tr:nth-child(even){
background: #0000FF;
}
```

https://it.wikipedia.org/wiki/Lista_dei_colori



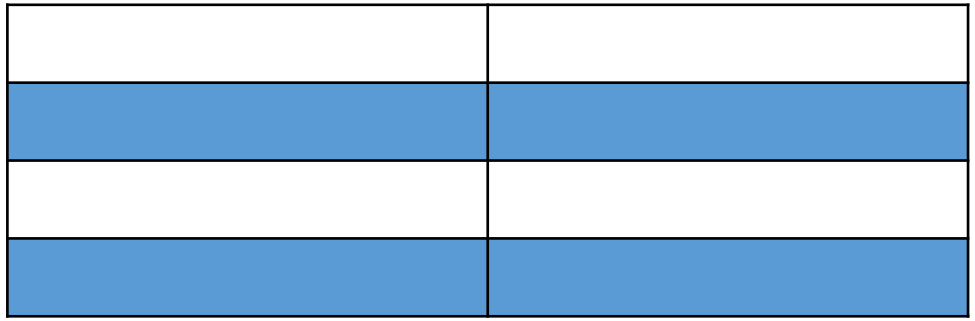
Id Class e altri selettori

Interessante uso di PSEUDO CLASSI:

:nth-child(n) → questa istruzione (che abbiamo utilizzato prima) serve a individuare l'ennesimo figlio di un altro elemento… al posto di «n» ovviamente si può mettere il numero, così individua specificatamente quell'elemento posizionato al numero…

noi abbiamo usato «even» che in Inglese significa «pari»
Esiste anche «odd» che in Inglese significa «dispari»

http://www.w3schools.com/cssref/tryit.asp?filename=trycss3_nth-child



Id Class e altri selettori

Per visualizzare la sintassi di tutti i selettori:

http://www.w3schools.com/cssref/css_selectors.asp

Ultima annotazione sulla sintassi precedente

:nth-child(n) → questa istruzione (che abbiamo utilizzato prima) serve a individuare l'ennesimo figlio di un altro elemento…

Noi in effetti avevamo questa struttura… quindi <tr> era «figlio» di <table>

```
<table>
        <tr>

        </tr>
</table>
```

Audio e video

```
<video width="480" height="360" controls >
  <source src="immagini_html/intervista.mp4«  type="video/mp4">
 Il tuo browser non supporta il video.
</video>
```

Un’altra proprietà che si può inserire è **autoplay**→ all’interno del codice, assieme oppure al posto di «controls»→ la dicitura «Il tuo browser…» compare solo se il video non viene supportato.

Le dimensioni del video possono essere impostate anche più piccole→ nell’esempio che ho trovato sul sito c’era width 320 e height 240→ comunque sempre meglio impostare una dimensione

Audio e video

Video supportati… mp4 è la scelta migliore… se si dispone di un video con altra estensione si può convertire in mp4 utilizzando il servizio online di zamzar→ http://www.zamzar.com/

Browser	MP4	WebM	Ogg
Internet Explorer	YES	NO	NO
Chrome	YES	YES	YES
Firefox	YES from Firefox 21 from Firefox 30 for Linux	YES	YES
Safari	YES	NO	NO
Opera	YES From Opera 25	YES	YES

- MP4 = MPEG 4 files with H264 video codec and AAC audio codec
- WebM = WebM files with VP8 video codec and Vorbis audio codec
- Ogg = Ogg files with Theora video codec and Vorbis audio codec

Audio e video

```
<audio controls>
    <source src="immagini_html/audio_lettura.mp3"  type="audio/mpeg">
Il tuo browser non supporta il file audio.
</audio>
```

Un'altra proprietà che si può inserire è **autoplay**→ all'interno del codice, assieme oppure al posto di «controls»→ la dicitura «Il tuo browser…» compare solo se l'audio non viene supportato.

Audio e video

Audio supportati… mp34 è la scelta migliore… se si dispone di un audio con altra estensione si può convertire in mp3 utilizzando il servizio online di zamzar→ http://www.zamzar.com/

Browser	MP3	Wav	Ogg
Internet Explorer	YES	NO	NO
Chrome	YES	YES	YES
Firefox	YES	YES	YES
Safari	YES	YES	NO
Opera	YES	YES	YES

Posizionamento: testo in colonna

column-count	Numero di colonne
column-fill	Riempimento colonne (“balance” o “auto”) supportato solo da Mozilla
column-gap	Spazio tra le colonne (in px)
column-rule	Scorciatoia per dichiarare in una sola volta il setting di COLORE- STILE- LARGHEZZA della linea divisoria
column-rule-color	LINEA DIVISORIA: colore
column-rule-style	LINEA DIVISORIA: stile
column-rule-width	LINEA DIVISORIA: larghezza
column-span	Numero colonne sulle quali si può espandere un elemento (esempio: il “titolo”)
column-width	Larghezza dele colonne
columns	Scorciatoia per dichiarare in una sola volta il setting di LARGHEZZA e NUMERO colonne

Posizionamento: immagini in colonna

La **gestione delle immagini** **all'interno delle colonne** NON è del tutto facile/intuitiva

Si utilizza la **funzione «float»** per spostare l'immagine a sinistra /destra all'interno della colonna MA funziona solo se l'immagine ha spazio per «restare» confinata nella colonna→

se l'immagine è troppo grande si sposta nella colonna adiacente e si sovrappone al testo…

quindi tre possibilità:

1) Lasciare l'immagine dove l'abbiamo posizionata in origine;
2) Posizionare l'immagine abbastanza in alto nella colonna;
3) Non utilizzare immagini troppo grandi

Posizionamento: static- absolute- relative- fixed

POSITION specifica il metodo di posizionamento utilizzato per un elemento.

Esistono quattro valori di «position»:

- ❑ static
- ❑ relative
- ❑ fixed
- ❑ absolute

Una volta stabilito il metodo, gli elementi vengono posizionati **utilizzando i valori**

top- bottom- left- right.

ATTENZIONE: questi valori **non funzionano** se prima non si è dichiarato il metodo

(static- relative- fixed- absolute).

Posizionamento: static- absolute- relative- fixed

POSITION → STATIC

Per **impostazione predefinita**, gli elementi HTML vengono posizionati staticamente.

Gli elementi posizionati statici **non vengono influenzati** dalle proprietà TOP- BOTTOM- LEFT- RIGHT

Un elemento con posizione «static» NON è posizionato in nessun modo speciale;

è sempre posizionato in base al flusso normale della pagina.

Posizionamento: static- absolute- relative- fixed

POSITION → RELATIVE

Un elemento con posizione «relative» è spostato rispetto alla sua posizione normale…

```
img{
position: relative;
top: 50px;
}
```

ESEMPIO

Questa immagine viene spostata di 50 pixel dall'alto, rispetto alla sua posizione naturale

lo spostamento in pixel viene impostato nelle **proprietà TOP-BOTTOM- LEFT-RIGHT**

Gli altri contenuti mantengono la propria posizione per cui potrebbe accadere che l'elemento spostato vada a sovrapporsi sopra gli altri…

In questo caso si dovrà utilizzare la **proprietà Z-INDEX** per regolare la profondità (in primo piano- in secondo piano etc.)

Posizionamento: static- absolute- relative- fixed

POSITION → FIXED

Un elemento con **position «fixed»** è posizionato rispetto alla finestra del browser, il che significa che rimane sempre nello stesso posto anche se la pagina viene «scrollata».

Le solite **proprietà TOP-BOTTOM- LEFT- RIGHT** vengono utilizzate per posizionare l'elemento.

Un elemento «fixed» **non lascia uno spazio vuoto nella pagina**, nel posto in cui sarebbe stato normalmente collocato, quindi quel posto viene occupato da altro testo, immagini etc.

È buona norma che un elemento «fixed» stia sempre in primo piano, utilizzando la proprietà z-index, impostata ad un numero superiore a tutti gli altri elementi.

Normalmente «fixed» viene usato per i «banner pubblicitari» o per lasciare un pulsante «torna su» sempre a disposizione anche se facciamo «scroll» nella pagina.

Posizionamento: static- absolute- relative- fixed

POSITION → ABSOLUTE

Un elemento con «position: absolute» è posizionato rispetto al suo «GENITORE» (anziché essere posizionato rispetto alla finestra del browser)... si usano sempre i soliti valori (TOP- BOTTOM- LEFT- RIGHT)

ATTENZIONE: il «GENITORE» deve però essere stato a sua volta posizionato (NON può essere «STATIC»)

Nota: un elemento "posizionato" è quello la cui posizione è **qualsiasi cosa tranne «static».**

https://www.w3schools.com/css/tryit.asp?filename=trycss_position_absolute

Posizionamento: float- clear- clearfix

FLOAT → LEFT- RIGHT

Con questa proprietà è possibile **rimuovere un elemento dal normale flusso** del documento e **spostarlo su uno dei lati (destro o sinistro)** **del suo elemento contenitore**.

Il contenuto che circonda l'elemento scorrerà intorno ad esso sul lato opposto rispetto a quello indicato come valore di float.

L'uso più comune è «far scorrere del testo attorno ad una immagine».

Altro utilizzo molto comune è quello di posizionare dei DIV uno a fianco all'altro (dentro al DIV poi si inseriscono i contenuti desiderati)→ è un modo per ottenere un «layout a colonne»

Posizionamento: float- clear- clearfix

FLOAT → LEFT- RIGHT

Gli elementi float vengono resi automaticamente **block-level**: questo significa che si può attribuire loro una **larghezza e/o un altezza** via CSS.

A differenza degli elementi block-level normali, gli elementi float oltre ad **essere auto-adattanti in altezza** (ossia assumere l'altezza necessaria al contenuto), lo sono **anche in larghezza.**

Questo significa che **un elemento float assumerà la larghezza massima per il suo contenuto**, fino ad **espandersi per tutta la larghezza del suo contenitore (a meno che non si stabiliscano valori specifici con→ width e height)**

Posizionamento: float- clear- clearfix

CLEAR → LEFT- RIGHT- BOTH

La proprietà **clear** serve a **impedire che al fianco di un elemento compaiano altri elementi** con il float. Si applica solo agli elementi blocco.

Float sposta un elemento dal flusso normale del documento ed è possibile che esso venga a trovarsi in posizioni non volute, magari a fianco di altri elementi che vogliamo tenere separati.

«Clear: left» → impedisce che un elemento tocchi con il proprio bordo sinistro un altro elemento.

«Clear: right» → impedisce che un elemento tocchi con il proprio bordo destro un altro elemento…. **«Clear: both»** si applica a entrambi i lati.

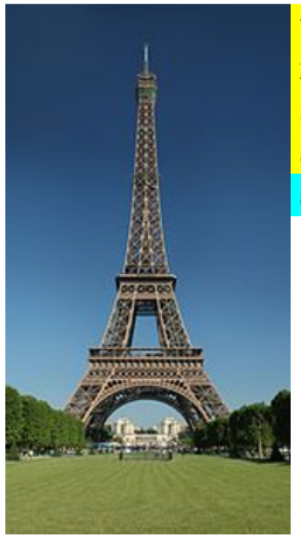
http://it.learnlayout.com/clear.html

Posizionamento: float- clear- clearfix

La proprietà **float** a volte genera un inconveniente:

Se abbiamo un contenitore DIV dentro al quale abbiamo inserito poco testo e una immagine abbastanza grande, ci accorgeremo che **l'immagine esce dal contenitore DIV (strasborda)**→ come effetto ci sarà quello indesiderato che un altro elemento si posiziona nel «buco (gap)» creato→ nell'esempio il testo in azzurro si è posizionato sotto al DIV giallo

Grandi monumenti dell'Europa



testo corto, di due sole righe, per cui senza ".clearfix" il contenitore giallo è "alto" solo due righe; (prova a togliere .clearfix)
se poi inserisco "clear" al contenitore blu allora esso si posiziona sotto gli altri, altrimenti sale a riempire il vuoto (gap)

altro testo da posizionare sotto il contenitore giallo

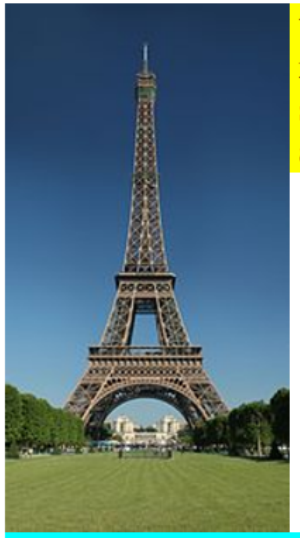
Il DIV giallo contiene sia il testo che l'immagine MA l'immagine è fuoriuscita dal DIV

Posizionamento: float- clear- clearfix

La proprietà **float** a volte genera un inconveniente:

Se **applichiamo la proprietà CLEAR al testo azzurro** risolviamo solo in parte il problema… in effetti il testo si posiziona sotto al DIV giallo, però (come in questo caso) noi volevamo avere uno sfondo nel DIV giallo che è rimasto troppo corto e così abbiamo una parte della pagina bianca→ **come fare per «allungare il DIV giallo»** in modo che contenga anche la foto?

Grandi monumenti dell'Europa



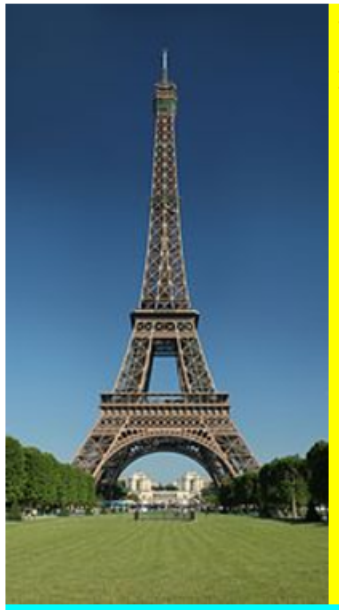
testo corto, di due sole righe, per cui senza ".clearfix" il contenitore giallo è "alto" solo due righe; (prova a togliere .clearfix)
se poi inserisco "clear" al contenitore blu allora esso si posiziona sotto gli altri, altrimenti sale a riempire il vuoto (gap)

altro testo da posizionare sotto il contenitore giallo

Il DIV giallo contiene sia il testo che l'immagine MA l'immagine è fuoriuscita dal DIV

Posizionamento: float- clear- clearfix

Grandi monumenti dell'Europa



testo corto, di due sole righe, per cui senza ".clearfix" il contenitore giallo è "alto" solo due righe; (prova a togliere .clearfix)
se poi inserisco "clear" al contenitore blu allora esso si posiziona sotto gli altri, altrimenti sale a riempire il vuoto (gap)

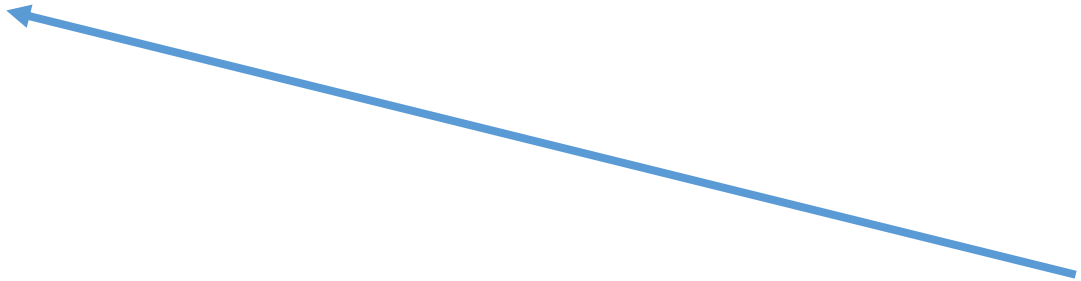
altro testo da posizionare sotto il contenitore giallo

Il DIV giallo adesso contiene sia il testo che l'immagine perché si è allungato

Posizionamento: float- clear- clearfix

Sistema più semplice (ma non sempre funziona perché potrebbe inserire una barra di scorrimento a lato del DIV)

```
#giallo{
background-color: yellow;
overflow: auto;
}
```



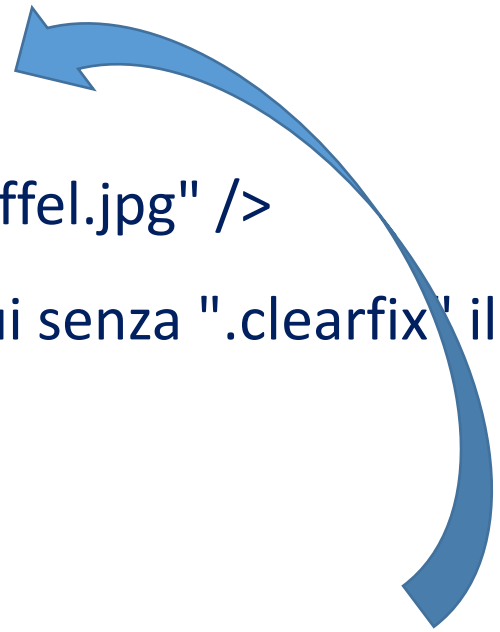
Aggiungere al «DIV giallo», che si deve allungare, **questa semplice istruzione**

Posizionamento: float- clear- clearfix

Sistema più lungo (ma più sicuro):

Aggiungere nel file html al DIV giallo una **classe specifica**, in modo che dopo nel file CSS si potranno attribuirgli delle proprietà→ il nome della classe è assolutamente libero e ognuno può scegliere il nome che vuole, comunque, per convenzione si utilizza il nome «**clearfix**»

```
<div id="giallo" class="clearfix">
<img  id="foto-eiffel" src="media/eiffel.jpg" />
testo corto, di due sole righe, per cui senza ".clearfix" il contenitore giallo è "alto" solo due righe; </div>
```



Aggiungere al «DIV giallo», che si deve allungare, **la classe «clearfix»**

Posizionamento: float- clear- clearfix

```
.clearfix:before, .clearfix:after{
content: " ";
display: table;
}
.clearfix:after{
clear: both;
}
.clearfix {
*zoom: 1;
}
```

Sistema più lungo (ma più sicuro):

Nel file CSS:

specificare **per la classe**

«clearfix» quanto riportato a lato

Collegamenti

Sintassi:

** testo da cliccare**

ATTENZIONE

Se dobbiamo collegarci ad un file allora si DEVE specificare anche l'estensione e fare attenzione alla ubicazione del file→

esempio: collegati a questo file

Se dobbiamo collegarci ad un sito allora si deve specificare l'intero indirizzo→

esempio: visita il sito e scarica

Collegamenti

collegati a "file1.html" dentro alla cartella "files"

collegati in un punto specifico della pagina all'interno dello stesso file

(si usa una cosiddetta «ancora»→ # + id→ #montagna→ uno dei paragrafi della pagina ha un id=«montagna»)

collegati a "file1.html" dentro alla cartella "files" ma in altra finestra

(si usa attributo **target="_blank«)**

Collegamenti

```
<a id="nosottolineato" href="files/file1.html" target="_blank">collegati a "file1.html"
dentro alla cartella "files"</a> <span>
```

ma il testo da cliccare NON appare sottolineato perché…

…nel File CSS

```
#nosottolineato{
text-decoration: none;
}
```

Collegamenti

collegati a "html_provvisorio.pdf" dentro alla cartella "files"

→si può collegare file di qualsiasi tipo ed estensione (in questo caso un file PDF)

collegati a un sito web

→ci si può collegare ad un sito web

Collegamenti

<a **id="look-pulsante"** href="https://www.materialididattici.org/" >collegati a un sito web

→si può abbellire il collegamento e trasformarlo in una specie di pulsante utilizzando i CSS

```
#look-pulsante{
border: solid 2px blue;
border-radius: 25px;
text-decoration: none;
background-color: aqua;
padding: 5px;
}
```

```
#look-pulsante:hover{
background-color: #a5c5f7;
}
```

Collegamenti

```
<a  href="https://www.materialididattici.org" ><img src="media/eiffel.jpg" /></a>
```

→anche un'immagine può essere un link

Tabelle

Le tabelle sono sempre introdotte dal tag **<table>** che naturalmente in fondo ha la sua chiusura **</table>**

Le tabelle sono formate da righe **<tr>** (Table Row) con chiusura **</tr>**

Ogni riga è formata da celle contenenti dati **<td>** (Table Data) con chiusura **</td>**

Le celle si adattano in larghezza al loro contenuto, a meno che non si stabilisca una larghezza predefinita utilizzando il file CSS→ width

Tabelle

La struttura base può essere arricchita inserendo delle celle speciali che sono evidenziate da GRASSETTO e diversa DIMENSIONE→ vengono indicate **con <th>** (Table Header) **</th>** in quanto fungono da descrittore della colonna o riga

	Mese	Risparmio
1	Gennaio	€100
2	Febbraio	€80
3	Marzo	€180

Tabelle

```
<table>
 <tr>
  <th></th>
   <th>Mese</th>
   <th>Risparmio</th>
 </tr>
 <tr>
 <th>1</th>
  <td>Gennaio</td>
  <td>€100</td>
 </tr>
 <tr>
  <th>2</th>
   <td>Febbraio</td>
   <td>€80</td>
 </tr>
 <tr>
  <th>3</th>
   <td>Marzo</td>
   <td>€180</td>
 </tr>
</table>
```

Prima riga… fatta tutta da <th>, cioè **intestazioni**… prima cella vuota, seconda «Mese», terza «Risparmio»

	Mese	Risparmio
1	Gennaio	€100
2	Febbraio	€80
3	Marzo	€180

Terza riga… fatta tutta da un <th>, cioè **intestazione, che racchiude il nu. 2**… seconda cella «Febbraio», terza «€80»

	Mese	Risparmio
1	Gennaio	€100
2	Febbraio	€80
3	Marzo	€180

Tabelle

```
<table>
 <tr>
 <th></th>
  <th>Mese</th>
  <th>Risparmio</th>
 </tr>
 <tr>
 <th>1</th>
  <td>Gennaio</td>
  <td>€100</td>
  <td>€200</td>
  <td>€300</td>
 </tr>
... etc.
</table>
```

È anche possibile creare tabelle con numero di celle NON omogeneo inserendo un numero maggiore di celle

	Mese	Risparmio		
1	Gennaio	€100	€200	€300
2	Febbraio	€80		
3	Marzo	€180		

Tabelle

```
<table>
 <tr>
 <th></th>
  <th>Mese</th>
  <th colspan="3">Risparmio</th>
 </tr>
 <tr>
 <th>1</th>
  <td>Gennaio</td>
  <td>€100</td>
  <td>€200</td>
  <td>€300</td>
 </tr>
…. Etc.
</table>
```

È anche possibile costringere una cella ad espandersi su più colonne… in questo caso ne occupa 3 → COLSPAN=«3»

	Mese	Risparmio		
1	Gennaio	€100	€200	€300
2	Febbraio	€80		
3	Marzo	€180		

Tabelle

```
<table>
…etc.
  <tr>
  <th>2</th>
   <td>Febbraio</td>
  <td rowspan="2">€80</td>
 </tr>
 <tr>
  <th>3</th>
   <td>Marzo</td>
 </tr>
</table>
```

È anche possibile costringere una cella ad espandersi su più righe… in questo caso ne occupa 2 → ROWSPAN=«2»

Notare che nella riga 4 abbiamo tolto una cella, sapendo che quella sopra si sarebbe «espansa» sino ad occupare quella sottostante

	Mese	Risparmio		
1	Gennaio	€100	€200	€300
2	Febbraio	€80		
3	Marzo			

Tabelle

```
<table>
…etc.
 <tr>
  <th>2</th>
   <td>Febbraio</td>
  <td rowspan="2">€80</td>
 </tr>
 <tr>
  <th>3</th>
   <td>Marzo</td>
  <td>€300</td>
 </tr>
</table>
```

È anche possibile costringere una cella ad espandersi su più righe… in questo caso ne occupa 2 → ROWSPAN=«2»

Se nella riga 4 NON avessimo tolto una cella, questo sarebbe stato il risultato

	Mese	Risparmio		
1	Gennaio	€100	€200	€300
2	Febbraio	€80		
3	Marzo		€300	

Tabelle

```
<table>
<caption>
Risparmio mensile in Italia
</caption>
 <tr>
 <th></th>
  <th>Mese</th>
  <th>Risparmio</th>
…etc.
 </tr></table>
```

Il **TAG <caption>** inserisce un TITOLO alla tabella

Risparmio mensile in Italia

	Mese	Risparmio		
1	Gennaio	€100	€200	€300
2	Febbraio	€80		
3	Marzo			

Tabelle: file CSS

```
table, th, td {
    border: 1px solid black;
    padding: 5px;
    text-align: center;
}
```

```
tr:nth-child(even){
background-color: aqua;
}
```

Seleziona le righe (TR) pari (EVEN) e attribuisce colore azzurro acqua
Quelle dispari sarebbero state individuate con tr:nth-child(odd)

```
td:hover, th:hover{
background-color: yellow;
}
```

Al passaggio del mouse (HOVER) sia le celle (TD) che le celle-intestazione (TH) assumono colore giallo

Risparmio mensile in Italia

	Mese	Risparmio		
1	Gennaio	€100	€200	€300
2	Febbraio	€80		
3	Marzo			

Tabelle: file CSS

```
td:nth-child(2){
background-color: aqua;
}
```

Seleziona le celle (TD) posizionate in secondo posto→ **td:nth-child(2)**

rispetto al «PADRE» che sarebbe (TR)
e attribuisce colore azzurro acqua

```
<tr>
 <th>1</th>
  <td>Gennaio</td>
  <td>€100</td>
</tr>
 <tr>
  <th>2</th>
  <td>Febbraio</td>
  <td>€80</td>
 </tr>
 <tr>
  <th>3</th>
  <td>Marzo</td>
  <td>€180</td>
 </tr>
</table>
```

Risparmio mensile in Italia

	Mese	Risparmio
1	Gennaio	€100
2	Febbraio	€80
3	Marzo	€180

Tabelle-bordi: file CSS

```
table, th, td {
border-collapse: collapse;
    border: 1px solid black;
}
```

Firstname	Lastname
Peter	Griffin
Lois	Griffin

```
table, th, td {
border-collapse: false;
    border: 1px solid black;
}
```

Firstname	Lastname
Peter	Griffin
Lois	Griffin

Tabelle-bordi: file CSS

```
table {
border: 1px solid black;
}
```

Firstname	Lastname
Peter	Griffin
Lois	Griffin

```
table, td {
border: 1px solid black;
}
```

Firstname	Lastname
Peter	Griffin
Lois	Griffin

```
table, td, th {
border: 1px solid black;
}
```

Firstname	Lastname
Peter	Griffin
Lois	Griffin

Tabelle: altre istruzioni CSS

border-spacing: **un valore in pixel | % | em** → (inserisce dello spazio tra le celle in modo che non si tocchino)

caption-side: **top | bottom** → (posizione del titolo)

empty-cells: **hide | show** → (la cella, anche se vuota, può avere bordi oppure scompare alla vista)

text-align: **left | center | right**

vertical-align: **top | bottom | middle**

width: **un valore in pixel | % | em** → (può essere applicato all'intera tabella- table- a una "cella titolo- th-", a una "cella normale- td-" o, ancora, a una riga o colonna)

height: **un valore in pixel | % | em** → (può essere applicato a→ vedi sopra)

padding: **un valore in pixel | % | em** → (spazio tra il testo/contenuto e il bordo)

Elenchi

Ci sono fondamentalmente **due tipi di elenchi:**

Elenchi ordinati →

**Order List → TAG → …. **

Si tratta di elenchi con numeri o lettere:

1. Frutta
2. Verdura
3. Carne

In Italiano sono chiamati «elenchi numerati»

Elenchi non ordinati →

**Unorder List → TAG → …. **

Si tratta di elenchi con simboli:

- Frutta
- Verdura
- Carne

In Italiano sono chiamati «elenchi puntati»

Elenchi

All'interno dei TAG di «apertura/chiusura», entrambi possiedono gli **elementi della «lista»** che sono individuati dai TAG ** **

Seguono dunque la seguente struttura:

Elenchi ordinati →

```
<ol>
<li>Frutta</li>
<li>Verdura</li>
<li>Carne</li>
</ol>
```

Elenchi non ordinati →

```
<ul>
<li>Frutta</li>
<li>Verdura</li>
<li>Carne</li>
</ul>
```

Elenchi

Attraverso il **file CSS** si possono **impostare i diversi tipi di numeri/simboli** → si utilizzano le espressioni:

```
ul {
list-style-type: ……. ;
}
```

oppure

```
ol {
list-style-type: ……. ;
}
```

Attraverso il **file CSS** si può impostare una lista **SENZA numeri/simboli** → si utilizza di solito quando si realizza un MENU':

```
ul {
list-style-type: none ;
}
```

oppure

```
ol {
list-style-type: none ;
}
```

Elenchi

Più comuni «list-style-type» per le liste

<OL>→ order list (numerate):

list-style-type: decimal ;

list-style-type: decimal-leading-zero ;

list-style-type: lower-alpha ;

list-style-type: upper-alpha ;

list-style-type: lower-roman ;

list-style-type: upper-roman ;

list-style-type: lower-greek ;

Più comuni «list-style-type» per le liste

<OL>→ order list (numerate):

decimal→ 1.

decimal-leading-zero→ 01. ;

lower-alpha→ a. ;

upper-alpha→ A. ;

lower-roman→ i. ;

upper-roman→ I. ;

lower-greek→ α. ;

Prova i diversi tipi
https://www.w3schools.com/cssref/playit.asp?filename=playcss_ol_list-style-type&preval=initial

Elenchi

Più comuni «list-style-type» per le liste

<UL>→ unorder list (elenchi puntati):

list-style-type: disc ;

list-style-type: square ;

list-style-type: circle ;

Più comuni «list-style-type» per le liste

<UL>→ Unorder list (elenchi puntati):

- disc
- square
- circle

Prova i diversi tipi
https://www.w3schools.com/cssref/playit.asp?filename=playcss_ul_list-style-type&preval=circle

Elenchi: utilità varie in CSS

Aumentare o diminuire spaziatura tra le righe di una lista:

```
ol{
list-style-type: upper-latin;
line-height: 1.5em; → ovviamente si mette il valore che si desidera
}
```

Elenchi: utilità varie in html

Iniziare la numerazione della lista NON dal primo numero-lettera ma da un altro, ad esempio iniziare la numerazione da «3»:

→ start=«valore» (nel file HTML)

```
<ol start="3">
<li>Frutta</li>
<li>Verdura</li>
<li>Carne</li>
</ol>
```

C. Frutta
D. Verdura
E. Carne

Come si può vedere, essendo impostato **list-style-type su «upper-alpha»** la numerazione **comincia da «C»→** la terza lettera

Elenchi: utilità varie in html

Se si volesse avere una lista «particolare» impostata su una numerazione che si incrementa in modo non lineare (esempio: 1-4-9) vi sono vari sistemi ma il più semplice è il seguente:

Mettere tre liste una sotto l'altra e farle apparire come fosse un'unica lista…

Poiché, però, le liste per default hanno un margine che le distanzia una dall'altra, bisogna azzerare i margini

Elenchi: utilità varie in html

Se si volesse avere una lista «particolare» impostata su una numerazione che si incrementa in modo non lineare (esempio: 1-4-9)…

```
<ol >
<li >Frutta</li>
</ol>
```

Lista 1

```
<ol start="4">
<li >Verdura</li>
</ol>
```

Lista 2

```
<ol start="9">
<li >Verdura</li>
</ol>
```

Lista 3

Nel file CSS azzerare I margini

```
ol{
list-style-type: decimal;
margin: 0;
}
```

Elenchi: utilità varie in CSS

Se si volesse avere una lista «particolare» con una immagine al posto degli elenchi puntati standard si può utilizzare la seguente sintassi:

```
ol{
list-style-image: url(«immagine.jpg»);
}
```

ATTENZIONE:

Il collegamento all'immagine parte dal file CSS quindi nello specificare URL (indirizzo) bisogna stare attenti… o l'immagine è posta nella stessa cartella del file CSS oppure bisogna «uscire dalla cartella» per entrare in quella dedicata alle immagini… quindi spesso l'indirizzo assume forme del genere «../media/immagine.jpg»

Elenchi: utilità varie in CSS

Se si volesse avere una lista «particolare» con una immagine al posto degli elenchi puntati standard si può utilizzare la seguente sintassi:

```
ol{
list-style-image: url(«immagine.jpg»);
}
```

ATTENZIONE:

Sono supportati tutti i tipi di file→ .png; .gif; .jpg etc.

Dimensioni consigliate per l'immagine→ da 16px a 30px

Eventualmente aumentare l'interlinea→ line-height: 2em;

Elenchi: utilità varie in CSS

Posizione dei numeri/simboli nell'elenco:

list-style-position: valore; i valori sono due: inside-outside

```
ol{
list-style-position: inside;
}
```

https://www.w3schools.com/cssref/playit.asp?filename=playcss_list-style-position&preval=inside

Elenchi: utilità varie in CSS

Posizione dei numeri/simboli nell'elenco→ personalizzazione

Si può fare in modo personalizzato agendo sul valore **«padding sinistro»** dei singoli

elementi della lista >li>

```
li{
padding-left: 20px;
}
```

Elenchi: utilità varie in CSS

Posizionare un elenco puntato in modo orizzontale

```
li{
float: left;
margin-left: 35px;
}
li:first-child{
margin-left: 0px;
}
```

Si imposta un float a tutti gli elementi

Poiché però i numeri/simboli dell'elenco si sovrappongono, si deve impostare un margine sinistro con margin-left

Al primo elemento li:first-child si imposta margin-left 0px altrimenti tutto l'elenco rientra nella pagina

Elenchi: utilità varie in CSS

Posizionare un elenco puntato in modo orizzontale spesso serve per creare un menù

https://www.w3schools.com/html/tryit.asp?filename=tryhtml_lists_menu

```
ul {
    list-style-type: none;
    margin: 0;
    padding: 0;
    overflow: hidden;
    background-color: #333333;
}

li {
    float: left;
}
```

```
li a {
    display: block;
    color: white;
    text-align: center;
    padding: 16px;
    text-decoration: none;
}

li a:hover {
    background-color: #111111;
}
```

Elenchi: utilità varie in HTML

Esiste anche la possibilità di creare «liste annidate»

```
<ol>
<li>Frutta</li>
<li>Verdura</li>
<li>Bevande</li>
        <ul>
        <li>birra</li>
        <li>vino</li>
        <li>acqua</li>
        </ul>
<li>Carne</li>
<li>Cereali</li>
</ol>
```

Il fatto che il codice sia «rientrato» NON è assolutamente obbligatorio…

è solo una questione di chiarezza nel leggere il codice

1. Frutta
2. Verdura
3. Bevande
    - birra
    - vino
    - acqua
4. Carne
5. Cereali

Elenchi: utilità varie in HTML

Creare un terzo tipo di lista: una lista di definizioni

```
<dl>
<dt>Coffee</dt>
<dd>black hot drink, it comes from Brazil;
tipical drink of Italy</dd>
<dt>Milk</dt>
<dd>white cold drink</dd>
</dl>
```

Coffee
: black hot drink, it comes from Brazil; tipical drink of Italy

Milk
: white cold drink

Attraverso i CSS si può enfatizzare il <dt> magari attribuendo del grassetto

Elenchi: utilità varie in HTML

Creare una lista con simboli speciali (tramite file CSS)

```
ul {
    list-style: none; /* Remove list bullets */
    padding: 0;
    margin: 0;
}
li {
    padding-left: 16px;
}
li:before {
    content: "•"; /* Insert content that looks like bullets */
    padding-right: 8px;
    color: blue; /* Or a color you prefer */
}
```

Al posto di questo simbolo si potrebbe inserire qualsiasi «carattere speciale»

Codice html

```
<ul>
  <li>Coffee</li>
  <li>Tea</li>
  <li>Coca Cola</li>
</ul>
```

- Coffee
- Tea
- Coca Cola

Resize

Un **DIV** (contenitore) può essere **reso RESIZE (modificabile in dimensioni)** con un semplice comando a livello di file CSS→ **i valori sono 3 (both- horizontal- vertical)**

Questo permette di modificare dimensioni di qualsiasi contenuto

```
div {
    resize: both;
    overflow: auto;
}
```

Pagina di esempio

https://www.w3schools.com/cssref/playit.asp?filename=playcss_resize&preval=both

Zoom

```
div {
border: 1px outset;
padding: 13px 0 7px 15px;
width: 20%;
resize: both;
overflow: auto;
}
img{
width: 95%;
height: 95%;
}
```

La **proprietà «resize»** può essere utilizzata per **creare uno «zoom- immagini»** molto semplice e forse un po' «grezzo» ma senza usare una linea di «codice di programmazione»

L'immagine deve essere abbastanza grande in origine, altrimenti, quando si effettua lo zoom potrebbe sgranare (1024X768 o similari

Apri la cartella «esempi-esercizi» e poi la cartella «zoom-resize»
Troverai i file di esempio

Impostare un menù

Nella cartella «**esempi-esercizi**» c'è una sottocartella che si intitola «**menù**»

All'interno c'è un file che si può modificare e adattare alle proprie esigenze…

È un menù realizzato con una lista-elenco, ma formattato con file CSS

Si tratta di una lista nidificata in un'altra lista

Se si desidera aggiungere o togliere una, o più voci, al menù

si deve MODIFICARE la LISTA </li<

https://www.w3schools.com/css/css_navbar.asp

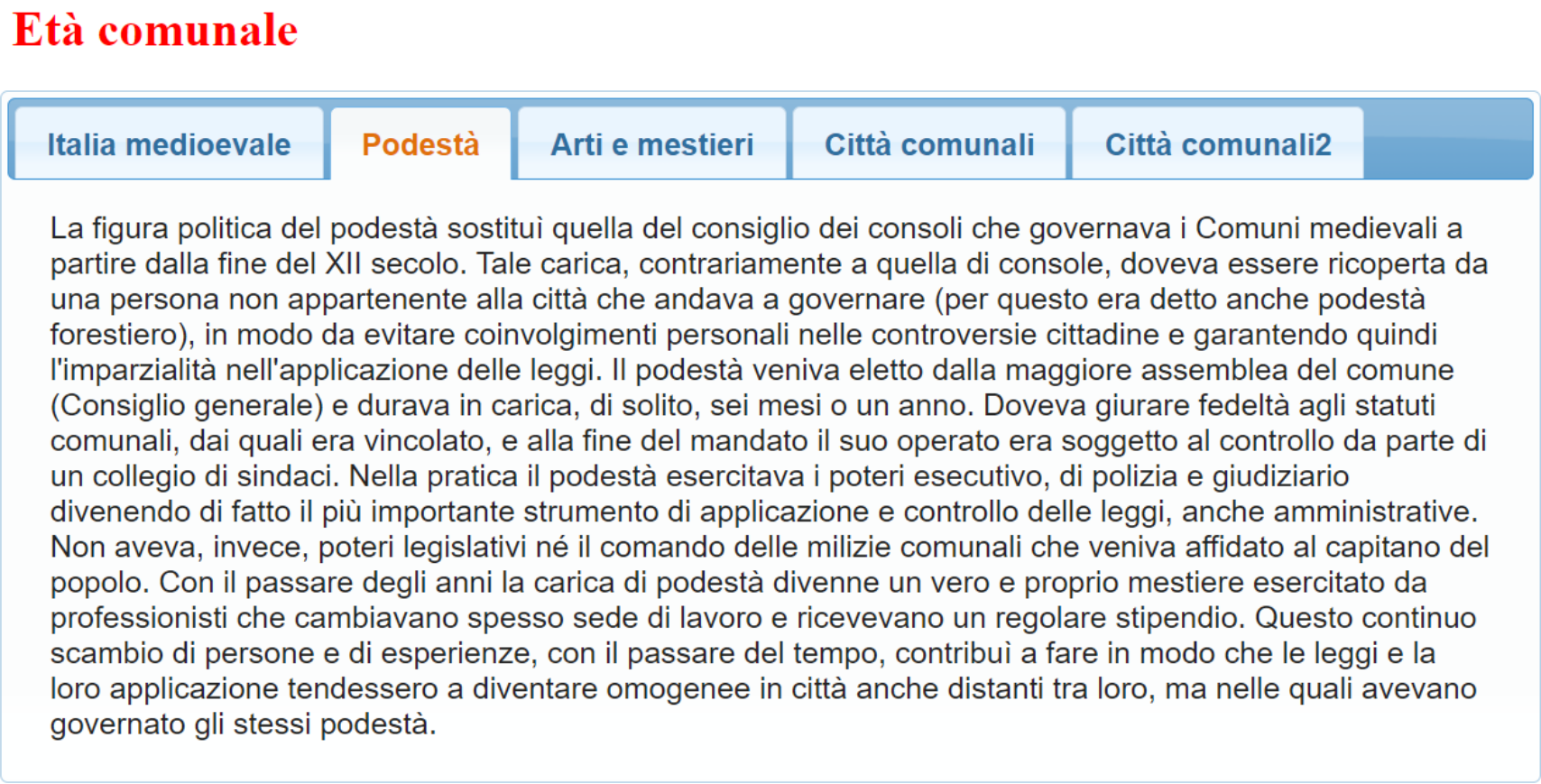
(accedendo a questo link troverai un esempio, spiegato «passo passo», per realizzare un menù)

Tab presentazione (con uso di jquery)

Nella cartella «**esempi-esercizi**» c'è una sottocartella che si intitola «**TAB**»…

puoi aprire questa cartella e trovare un esempio da adattare alle tue esigenze

Il risultato di una presentazione con finestra a «tab» è la seguente:



Tab presentazione

Vediamo come personalizzare il codice

Apri il file HTML che si intitola «Tab»

Puoi togliere o aggiungere

testo testo

```
<body>
<header><p><h1>Età comunale</h1></p></header>

<div id="tabs" >
<ul>
<li><a href="#tabs-1">Italia medioevale</a></li>
<li><a href="#tabs-2">Podestà</a></li>
<li><a href="#tabs-3">Arti e mestieri</a></li>
<li><a href="#tabs-4">Città comunali</a></li>
<li><a href="#tabs-5">Città comunali2</a></li>
</ul>
```

In questo modo aggiungerai o toglierai dei TAB nella intestazione

Tab presentazione

```
<div id="tabs-1"><img id="castello" src="media/castello.jpg" style= "float: left" />
  Se con storia del Medioevo si intende la storia dal 476, anno della deposizione dell'ultimo imperatore
A sua volta il Medioevo tradizionalmente si divide in alto Medioevo (fino all'anno 1000) e basso Medioevo
rima delle altre.
Così all'inizio del basso Medioevo, mentre in Europa si diffondono le monarchie feudali, in Italia si svi
  </div>
```

**Il vero e proprio contenuto deve essere inserito nel codice sotto alla lista tra due testo testo etc. **

ATTENZIONE: il tab deve avere un ID (cioè un nome) che è esattamente quello che compariva nella lista

```
<body>
<header><p><h1>Età comunale</h1></p></header>

<div id="tabs" >
<ul>
<li><a href="#tabs-1">Italia medioevale</a></li>
<li><a href="#tabs-2">Podestà</a></li>
<li><a href="#tabs-3">Arti e mestieri</a></li>
<li><a href="#tabs-4">Città comunali</a></li>
<li><a href="#tabs-5">Città comunali2</a></li>
</ul>
```

Tab presentazione

Come puoi notare nell'immagine:

- ❑ **Se il testo è lungo comparirà una «barra laterale di scroll»**
- ❑ **All'interno del contenuto (cioè nei div) si può inserire quello che si vuole, ovviamente anche immagini**

Età comunale

Italia medioevale | Podestà | Arti e mestieri | Città comunali | Città comunali2

Se con storia del Medioevo si intende la storia dal 476, anno della deposizione dell'ultimo imperatore d'occidente, al 1492, anno della scoperta dell'America, con Italia medievale dobbiamo forse più precisamente intendere quel periodo della storia d'Italia che va dall'invasione longobarda (568) fino alla stipulazione della Pace di Lodi, nel 1454, la quale dette inizio ai quarant'anni di equilibrio politico e che fu il periodo più importante del Rinascimento italiano. A sua volta il Medioevo tradizionalmente si divide in alto Medioevo (fino all'anno 1000) e basso Medioevo. È bene puntualizzare che tali datazioni sono semplicemente delle convenzioni per riferirsi con maggiore chiarezza ad un periodo tanto lungo quanto complesso. Infatti, spesso, dietro alle nette datazioni ci sono molteplici sfumature tipiche della storia. All'inizio dell'alto Medioevo l'Europa e l'Italia romane vengono germanizzate, con la formazione dei regni romano-barbarici. Dopo la sedentarizzazione dei nomadi germanici, è la volta di nuovi nomadi, gli Arabi che rompono l'unità del Mediterraneo. Inoltre parte dell'Italia venne occupata anche dalle truppe dell'Impero romano d'Oriente, comunemente detto Impero bizantino. Il tentativo di unire l'Europa da parte di Carlo Magno (742-814) non avrà fortuna, ma il sistema con cui organizzò la sua società, il feudalesimo,

Tab presentazione

Apri il file che si trova nella cartella «CSS» e che si intitola «tab-stili.css»

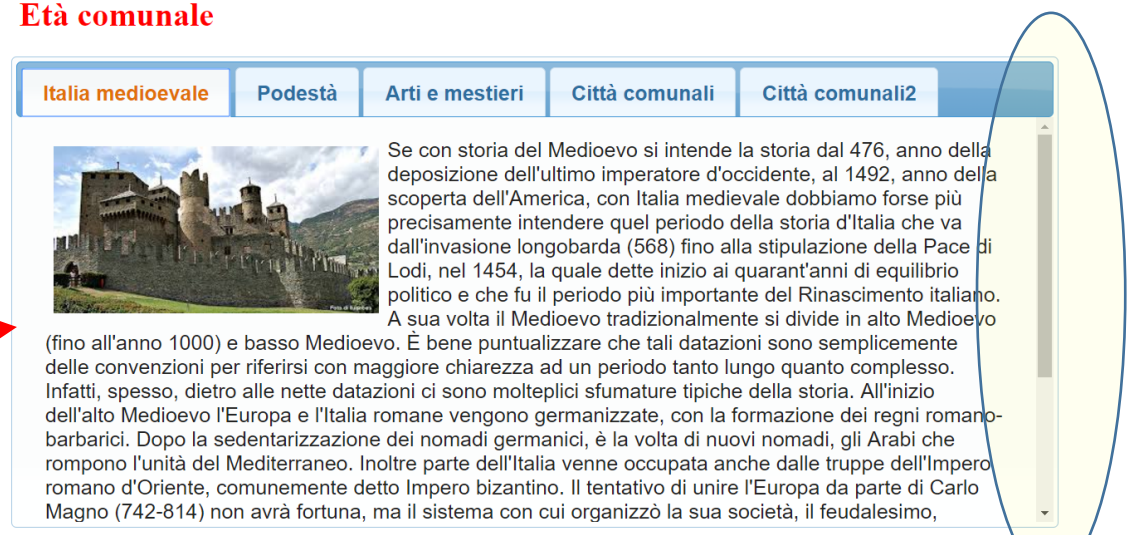
Si possono fare diversi adattamenti

```
#tabs div{
height: 350px;
overflow-y:auto;
/*altezza del contenitore del testo dei singoli tabs...
se il testo è più lungo, come appare nel tab 1 "Italia medioevale",
allora si genera una barra di scroll a lato... si può provare a settarlo a 400 o 450 px a piacere
*/
}
```

Tab presentazione

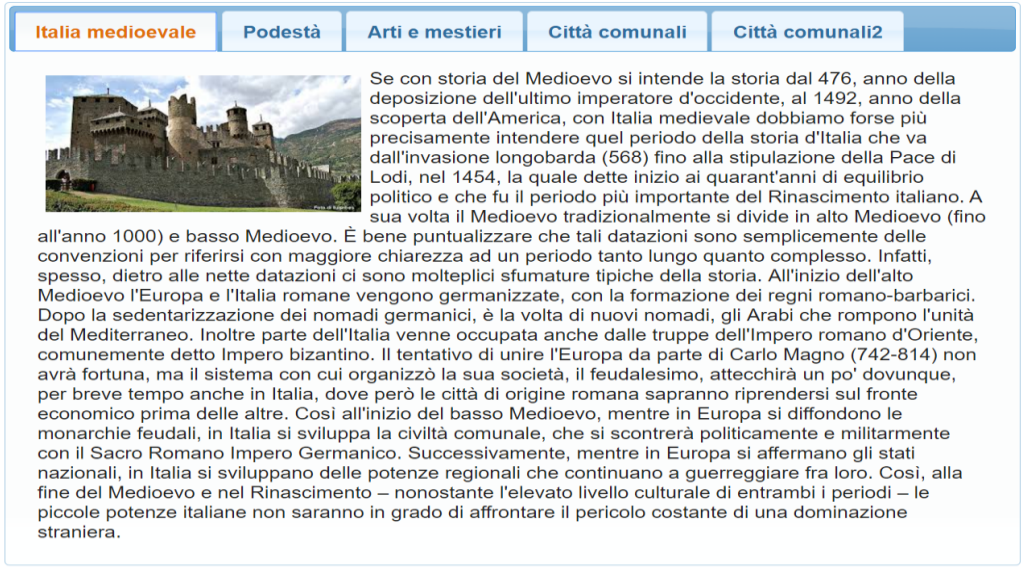
Altezza del contenitore

```
#tabs div{
height: 350px;
overflow-y:auto;
```



Altezza del contenitore

```
#tabs div{
height: 560px;
overflow-y:auto;
```



Tab presentazione

```
#tabs {
font-size: 20px;        /*dimensione del testo all'interno dei tabs*/
width: 1000px;        /*larghezza dell'intero contenitore con i 5 tabs*/
position: absolute;
left: 50px;
}
```

Tab presentazione

```
#tabs {
font-size: 20px;
width: 1000px;
position: absolute;
left: 50px;
}
```

Età comunale

Italia medioevale | Podestà | Arti e mestieri | Città comunali | Città comunali2

La figura politica del podestà sostituì quella del consiglio dei consoli che governava i Comuni medievali a partire dalla fine del XII secolo. Tale carica, contrariamente a quella di console, doveva essere ricoperta da una persona non appartenente alla città che andava a governare (per questo era detto anche podestà forestiero), in modo da evitare coinvolgimenti personali nelle controversie cittadine e garantendo quindi l'imparzialità nell'applicazione delle leggi. Il podestà veniva eletto dalla maggiore assemblea del comune (Consiglio generale) e durava in carica, di solito, sei mesi o un anno. Doveva giurare fedeltà agli statuti comunali, dai quali era vincolato, e alla fine del mandato il suo operato era soggetto al controllo da parte di un collegio di sindaci. Nella pratica il podestà esercitava i poteri esecutivo, di polizia e giudiziario divenendo di fatto il più importante strumento di applicazione e controllo delle leggi, anche amministrative. Non aveva, invece, poteri legislativi né il comando delle milizie comunali che veniva affidato al capitano del popolo. Con il passare degli anni la carica di podestà divenne un vero e proprio mestiere esercitato da professionisti che cambiavano spesso sede di lavoro e ricevevano un regolare stipendio. Questo continuo scambio di persone e di esperienze, con il passare del tempo, contribuì a fare in modo che le leggi e la loro applicazione tendessero a diventare omogenee in città anche distanti tra loro, ma nelle quali avevano governato gli stessi podestà.

```
#tabs {
font-size: 14px;
width: 1000px;
position: absolute;
left: 50px;
}
```

Età comunale

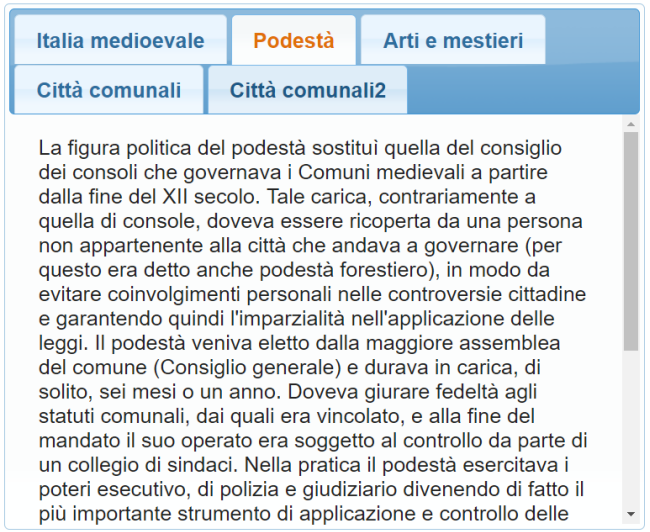
Italia medioevale | Podestà | Arti e mestieri | Città comunali | Città comunali2

La figura politica del podestà sostituì quella del consiglio dei consoli che governava i Comuni medievali a partire dalla fine del XII secolo. Tale carica, contrariamente a quella di console, doveva essere ricoperta da una persona non appartenente alla città che andava a governare (per questo era detto anche podestà forestiero), in modo da evitare coinvolgimenti personali nelle controversie cittadine e garantendo quindi l'imparzialità nell'applicazione delle leggi. Il podestà veniva eletto dalla maggiore assemblea del comune (Consiglio generale) e durava in carica, di solito, sei mesi o un anno. Doveva giurare fedeltà agli statuti comunali, dai quali era vincolato, e alla fine del mandato il suo operato era soggetto al controllo da parte di un collegio di sindaci. Nella pratica il podestà esercitava i poteri esecutivo, di polizia e giudiziario divenendo di fatto il più importante strumento di applicazione e controllo delle leggi, anche amministrative. Non aveva, invece, poteri legislativi né il comando delle milizie comunali che veniva affidato al capitano del popolo. Con il passare degli anni la carica di podestà divenne un vero e proprio mestiere esercitato da professionisti che cambiavano spesso sede di lavoro e ricevevano un regolare stipendio. Questo continuo scambio di persone e di esperienze, con il passare del tempo, contribuì a fare in modo che le leggi e la loro applicazione tendessero a diventare omogenee in città anche distanti tra loro, ma nelle quali avevano governato gli stessi podestà.

Tab presentazione

```
#tabs {
font-size: 20px;
width: 1000px;
position: absolute;
left: 50px;
}
```

```
#tabs {
font-size: 14px;
width: 600px;
position: absolute;
left: 50px;
}
```



Tenere presente che una finestra maggiore di 1000px NON si vedrebbe su una iPad/tablet→ si dovrebbe ridimensionare la schermata

Tab presentazione

Apri il file che si trova nella cartella «JS» e che si intitola «tab-codice.js»

```
$(function() {
  $("#tabs").tabs();/* attiva il tab... lo fa esistere*/
$( "#tabs" ).tabs({
  active: 1 /* quando si apre il file html, un tab risulta già selezionato (quello su cui di default
si apre la schermata);
in informatica si inizia a contare da 0
quindi 0 risulta attivo il primo tab, 1 il secondo, 2 il terzo, 3 il quarto, 4 il quinto etc.*/
});
});
```

Tab presentazione

Cambiando il valore di active: 1 si otterrà il cambiamento del tab che all'apertura risulterà di default già visibile

Età comunale

Italia medioevale | Podestà | Arti e mestieri | Città comunali | Città comunali2

La figura politica del podestà sostituì quella del consiglio dei consoli che governava i Comuni medievali a partire dalla fine del XII secolo. Tale carica, contrariamente a quella di console, doveva essere ricoperta da una persona non appartenente alla città che andava a governare (per questo era detto anche podestà forestiero), in modo da evitare coinvolgimenti personali nelle controversie cittadine e garantendo quindi l'imparzialità nell'applicazione delle leggi. Il podestà veniva eletto dalla maggiore assemblea del comune (Consiglio generale) e durava in carica, di solito, sei mesi o un anno. Doveva giurare fedeltà agli statuti comunali, dai quali era vincolato, e alla fine del mandato il suo operato era soggetto al controllo da parte di un collegio di sindaci. Nella pratica il podestà esercitava i poteri esecutivo, di polizia e giudiziario divenendo di fatto il più importante strumento di applicazione e controllo delle leggi, anche amministrative. Non aveva, invece, poteri legislativi né il comando delle milizie comunali che veniva affidato al capitano del popolo. Con il passare degli anni la carica di podestà divenne un vero e proprio mestiere esercitato da professionisti che cambiavano spesso sede di lavoro e ricevevano un regolare stipendio. Questo continuo scambio di persone e di esperienze, con il passare del tempo, contribuì a fare in modo che le leggi e la loro applicazione tendessero a diventare omogenee in città anche distanti tra loro, ma nelle quali avevano governato gli stessi podestà.

active: 1 → all'apertura ci troviamo il «tab podestà» attivo perché in informatica si comincia a contare da 0, quindi active: 1 significa aprire di default il secondo tab

Età comunale

Italia medioevale | Podestà | Arti e mestieri | Città comunali | Città comunali2

Se con storia del Medioevo si intende la storia dal 476, anno della deposizione dell'ultimo imperatore d'occidente, al 1492, anno della scoperta dell'America, con Italia medievale dobbiamo forse più precisamente intendere quel periodo della storia d'Italia che va dall'invasione longobarda (568) fino alla stipulazione della Pace di Lodi, nel 1454, la quale dette inizio ai quarant'anni di equilibrio politico e che fu il periodo più importante del Rinascimento italiano. A sua volta il Medioevo tradizionalmente si divide in alto Medioevo (fino all'anno 1000) e basso Medioevo. È bene puntualizzare che tali datazioni sono semplicemente delle convenzioni per riferirsi con maggiore chiarezza ad un periodo tanto lungo quanto complesso. Infatti, spesso, dietro alle nette datazioni ci sono molteplici sfumature tipiche della storia. All'inizio dell'alto Medioevo l'Europa e l'Italia romane vengono germanizzate, con la formazione dei regni romano-barbarici. Dopo la sedentarizzazione dei nomadi germanici, è la volta di nuovi nomadi, gli Arabi che rompono l'unità del Mediterraneo. Inoltre parte dell'Italia venne occupata anche dalle truppe dell'Impero romano d'Oriente, comunemente detto Impero bizantino. Il tentativo di unire l'Europa da parte di Carlo Magno (742-814) non avrà fortuna, ma il sistema con cui organizzò la sua società, il feudalesimo,

active: 0 → all'apertura ci troviamo il «tab Italia medioevale» attivo perché in informatica si comincia a contare da 0, quindi active: 0 significa aprire di default il primo tab

Tab presentazione

I file messi a disposizione danno la possibilità di creare abbastanza facilmente una presentazione con «tab»… non è comunque l'unico modo per ottenere questo effetto.

Ad esempio utilizzando i software «Ardora7_5» o «exelearning» si ottengono gli stessi risultati in modo semplice, quasi immediato.

ATENZIONE: per quanto riguarda il layout e i colori, essi dipendono dalla versione di Jquery che è stata scaricata… io ho scaricato il layout che si intitola «redmond»…
se uno desidera altri layout, deve scaricare le cartelle relative al layout prescelto.
Consulta in questa presentazione quanto detto nella slide num. 6 (pulsante «link utili»)

Uso di Javascript (un esempio)

Nella cartella «**esempi-esercizi**» c'è una sottocartella che si intitola «mostra-nascondi-javascript»… puoi aprire questa sottocartella e quindi nuovamente la sottocartella «nascondi-tabella-semplice»

Analizziamo il codice

Iniziamo dal codice nel file HTML→ «nascondi-tabella-semplice.html»

```
<head>
<meta charset="utf-8" />
<script type="text/javascript" src="js/nascondi-codice.js"></script>
<link rel="stylesheet" href="css/nascondi-stili.css" type="text/css" />
</head>
```

Uso di Javascript (un esempio)

In <head></head> è stato inserito **un richiamo ai file .css** (per formattare gli stili) e **.js** (per implementare comandi interattivi)

```
<head>
<meta charset="utf-8" />
<script type="text/javascript" src="js/nascondi-codice.js"></script>
<link rel="stylesheet" href="css/nascondi-stili.css" type="text/css" />
</head>
```

Uso di Javascript (un esempio)

In <body> è stata aggiunta una istruzione IMPORTANTISSIMA→ onload="start()"

Questa istruzione spiega al browser di **caricare il file .js** SOLO dopo aver caricato la pagina HTML… senza questa istruzione NULLA FUNZIONEREBBE→ il nome "start()« è inventato… potrebbe essere anche «ciao()«→ attenzione che le parentesi tonde sono obbligatorie

```
<body onload="start()">
<div id="cont1">
<table id="tabella1">
<th class="titolo"></th>
<th class="titolo">Contenuti</th>

<tr>
<td><input type="checkbox" id="pulsante1"></td>
<td id="riga1">Parigi è una città di circa 12 milioni di abit
</tr>
```

Uso di Javascript (un esempio)

Il div con id=«cont1» in questo caso potrebbe anche essere superfluo… comunque fa da contenitore a tutta la tabella… una precauzione se volessimo controllare la posizione in qualche modo attraverso i css, magari in un secondo tempo

```
<body onload="start()">
<div id="cont1">
<table id="tabella1">
<th class="titolo"></th>
<th class="titolo">Contenuti</th>

<tr>
<td><input type="checkbox" id="pulsante1"></td>
<td id="riga1">Parigi è una città di circa 12 milioni di abit
</tr>
```

Uso di Javascript (un esempio)

Inizia la tabella… **<th>** funge da intestazione…

la **class=«titolo»** servirà per controllare le dimensioni in altezza della riga di intestazione

della tabella attraverso, come al solito, il file .css

Il primo <th> </th> è vuoto, infatti tra di essi non c'è scritto nulla

Il secondo <th> </th> è «riempito» con la parola Contenuti

```
<body onload="start()">
<div id="cont1">
<table id="tabella1">
<th class="titolo"></th>
<th class="titolo">Contenuti</th>

<tr>
<td><input type="checkbox" id="pulsante1"></td>
<td id="riga1">Parigi è una città di circa 12 milioni di abit
</tr>
```

Uso di Javascript (un esempio)

Prima riga della tabella <tr> </tr>

Prima cella <td> viene inserito il pulsante di tipo «checkbox» e gli viene attribuito un nome→ id=«pulsante1»

Seconda cella <td> si inseriscono i contenuti; a questa seconda cella viene attribuito un nome→ id=«riga1»

questi nomi serviranno al codice di javascript per effettuare delle azioni su di essi

```
<body onload="start()">
<div id="cont1">
<table id="tabella1">
<th class="titolo"></th>
<th class="titolo">Contenuti</th>

<tr>
<td><input type="checkbox" id="pulsante1"></td>
<td id="riga1">Parigi è una città di circa 12 milioni di abit
</tr>
```

Uso di Javascript (un esempio)

Lo schema prosegue, inserendo altre righe… quante se ne vogliono

Cambiano i nomi→

Id=«pulsante2»

Id=«riga2»

Etc. etc. si prosegue così allungando la tabella quanto si desidera

```
<tr>
<td><input type="checkbox" id="pulsante2"></td>
<td id="riga2"><img src="media/eiffel.jpg"></img> Il monumento fr
</tr>
```

Uso di Javascript (un esempio)

Chiusura della tabella </table>

Chiusura del div

Chiusura del body </body>

```
<tr>
<td><input type="checkbox" id="pulsante3"></td>
<td id="riga3">La Senna è il fiume che attraversa Parigi;
è lunga 776 km e sfocia nel Canale della Manica presso la
La navigabilità per grandi imbarcazioni è limitata a 120 k
</tr>
</table>
</div>
</body>
```

Uso di Javascript (un esempio)

Passiamo ad aprire il file CSS dentro alla sottocartella css; il nome del file è nascondi-stili.css

```
#cont1{
width: 800px;
float: left;
}
tr, td, table{
border: solid 1px grey;
font-size: 20px;
font-family: verdana, arial, sans-serif;
}
```

Il **#cont1 è un DIV che di fatto stabilisce quanto larga sarà la tabella**, perché una tabella si allarga fino a occupare tutto lo spazio del suo contenitore

Impostazioni per la **tabella, le celle e le righe** per quanto riguarda bordi e font

Uso di Javascript (un esempio)

Passiamo ad aprire il file CSS dentro alla sottocartella css; il nome del file è nascondi-stili.css

```
th{
color: red;
}
.titolo {
height: 30px;
}
#riga1, #riga2, #riga3{
height: 120px;
}
```

th colore del titolo della tabella

La classe «titolo» imposta l'altezza solo per il titolo

Si richiamano i nomi delle righe per impostare la loro altezza→ se nella tabella si aggiungono righe, si deve aggiungere il loro nome anche qui!!!

Uso di Javascript (un esempio)

Passiamo ad aprire il **file JS** dentro alla sottocartella JS;

il nome del file è «nascondi-codice.js»

Risulterebbe un po' complicato spiegare tutto il codice per cui concentriamoci solamente

sulle modifiche da apportare→ prossima slide

```
function start(){
var testo = ["riga1", "riga2", "riga3"];
var pulsanti = ["pulsante1", "pulsante2", "pulsante3"];
function nascondi() {
var a = pulsanti.indexOf(this.id);
if(document.getElementById(this.id).checked == true){
    document.getElementById(testo[a]).style.visibility = "hidden";}
else{
document.getElementById(testo[a]).style.visibility = "visible";
}
}
document.getElementById("pulsante1").onclick = nascondi;
document.getElementById("pulsante2").onclick = nascondi;
document.getElementById("pulsante3").onclick = nascondi;
}
```

Uso di Javascript (un esempio)

Aggiungere, nell'ordine corretto, altri nomi delle righe e altri nomi dei pulsanti (se sono stati aggiunti)…

Non è obbligatorio ma si consiglia di mantenere nei nomi la modalità della numerazione→ «pulsante1» «pulsante2» «pulsante3» «pulsante4» etc.

```
function start(){
var testo = ["riga1", "riga2", "riga3"];
var pulsanti = ["pulsante1", "pulsante2", "pulsante3"];
function nascondi() {
var a = pulsanti.indexOf(this.id);
if(document.getElementById(this.id).checked == true){
    document.getElementById(testo[a]).style.visibility = "hidden";}
else{
document.getElementById(testo[a]).style.visibility = "visible";
}
}
document.getElementById("pulsante1").onclick = nascondi;
document.getElementById("pulsante2").onclick = nascondi;
document.getElementById("pulsante3").onclick = nascondi;
}
```

Uso di Javascript (un esempio)

Aggiungere (se è necessario) altro codice per attivare più pulsanti

document.getElementById("pulsante1").onclick = nascondi;

document.getElementById("pulsante2").onclick = nascondi;

document.getElementById("pulsante3").onclick = nascondi;

document.getElementById("pulsante4").onclick = nascondi; etc. etc. etc.

```
function start(){
var testo = ["riga1", "riga2", "riga3"];
var pulsanti = ["pulsante1", "pulsante2", "pulsante3"];
function nascondi() {
var a = pulsanti.indexOf(this.id);
if(document.getElementById(this.id).checked == true){
    document.getElementById(testo[a]).style.visibility = "hidden";}
else{
document.getElementById(testo[a]).style.visibility = "visible";
}
}
document.getElementById("pulsante1").onclick = nascondi;
document.getElementById("pulsante2").onclick = nascondi;
document.getElementById("pulsante3").onclick = nascondi;
}
```