esci

Basi per html5, fogli di stile css3 e 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
- Si può settare cliccando su→ textedit→ preferenze→ plain text
- Si può settare cliccando su→ formato → make plain text e su → Edit/modifica→ ignora rich text
- Salvare come .html (nella finestra di dialogo che si apre)
- https://www.youtube.com/watch?v=Ju12fK6jm6M

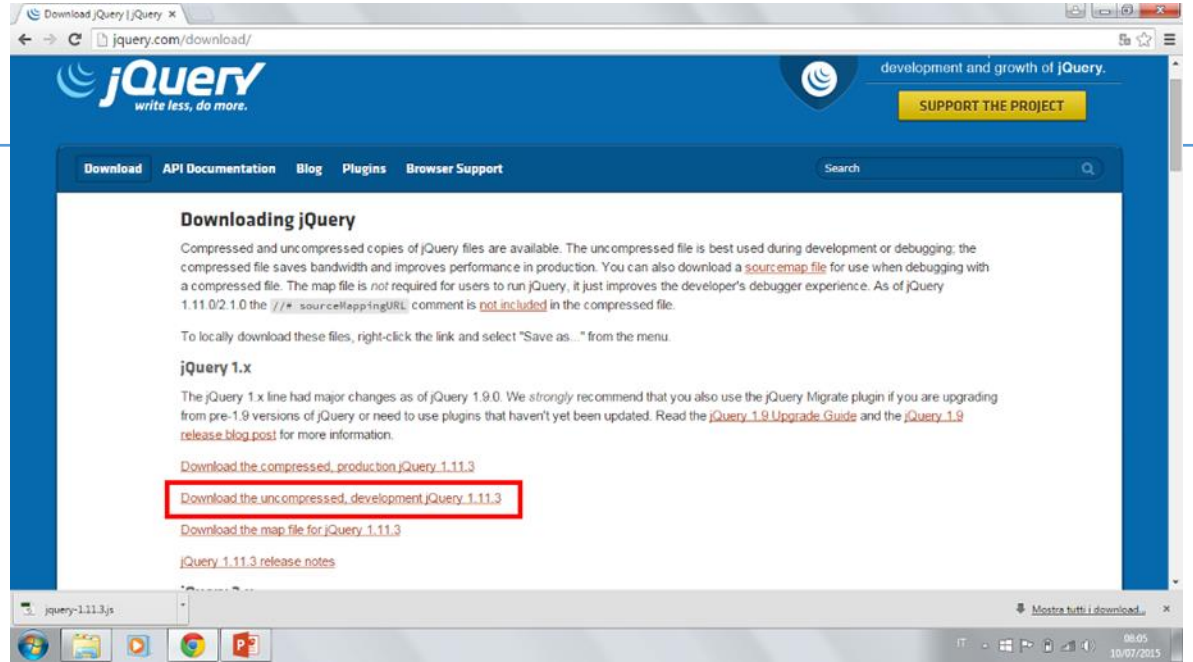
Link utili

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):

Per realizzazione di file in html5→ Scaricare la versione uncompressed development jQuery 1.11.3 **http://jquery.com/download/**



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

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-1.11.4.custom

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

.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

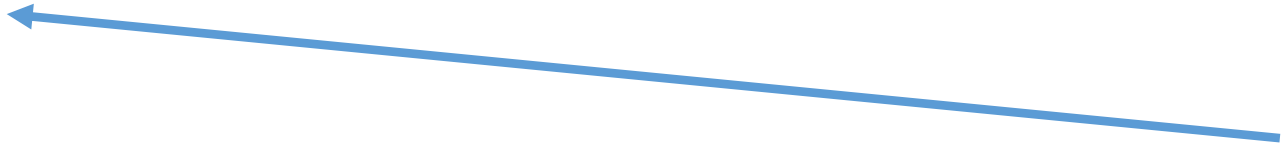
.css e .js

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 document-file
</body>
</html>
```

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

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>
```

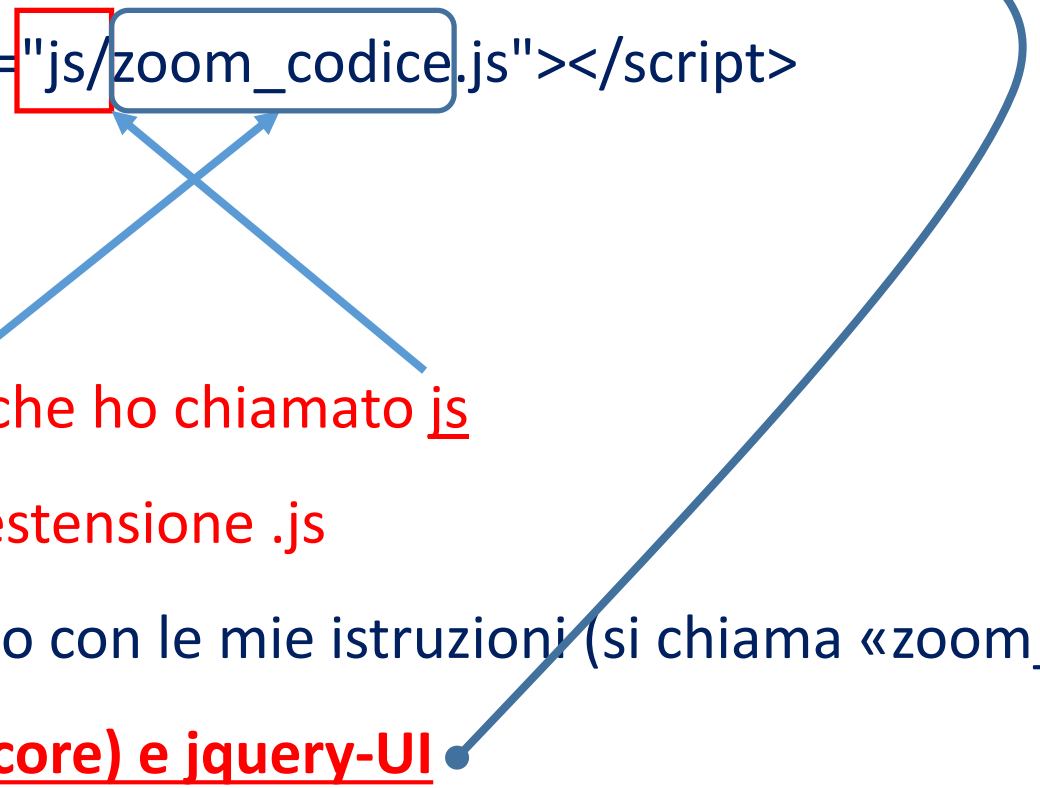
Richiamo dei file js con il percorso

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
 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

Principali tag all'interno del body

Principali tag «block»

<p> <table>   <h1>… <h6>

Elenchi puntati-numerati

Principali tag «inline»

 <img> <a>

immagine **collegamento**

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

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

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

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]

font-style: italic;

color: red;

letter-spacing: 1px;

word-spacing: 5px;

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

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

text-decoration-color: red;

text-indent: 50px;

CSS e gestione Bordi

border: dotted;

border: dashed;

border: solid;

border: double;

border: groove;

border: ridge;

border: inset;

border: outset;

border-style values:

none: Defines no border

dotted: Defines a dotted border

dashed: Defines a dashed border

solid: Defines a solid border

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

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

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

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

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)

i «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

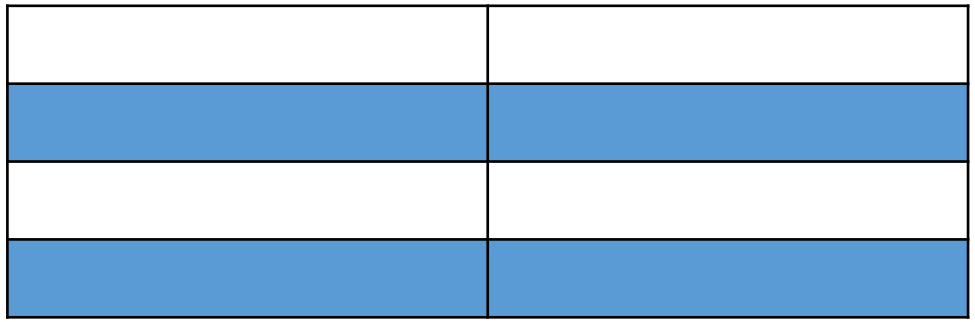
Interessante uso di PSEUDO CLASSI:

- 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



Id Class e altri selettori

Interessante uso di PSEUDO CLASSI:

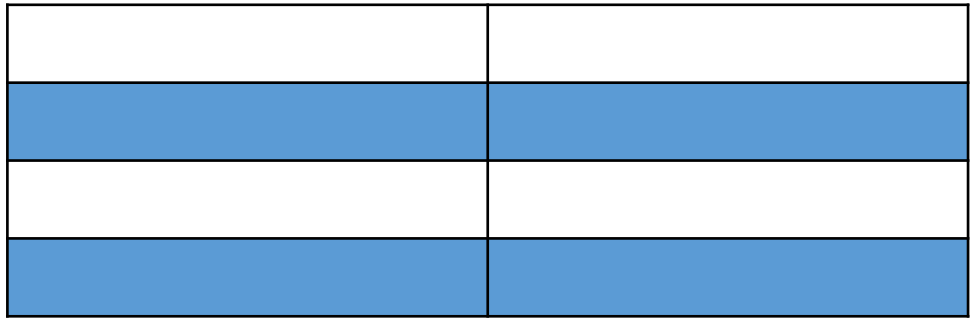
- Nel file .html c'è una tabella (ricordarsi che <tr> è una riga)

```
<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



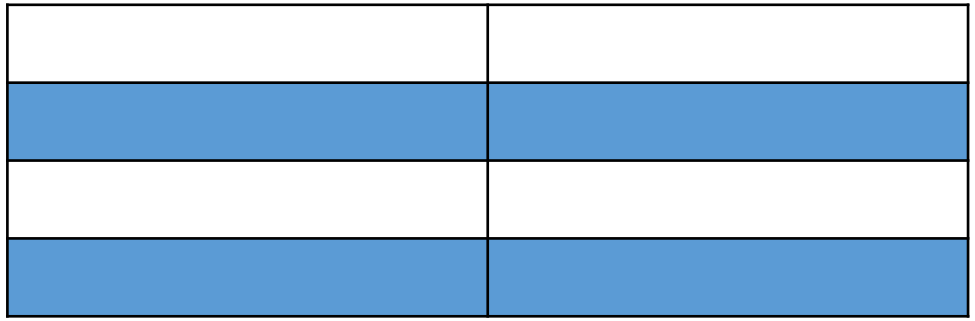
Id Class e altri selettori

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