ACM Rome Tor Vergata

acm.uniroma2.it

…a hour of code

Parte 1 – Il linguaggio HTML

L'HTML (*HyperText Markup Language*) è un linguaggio nato per la formattazione e l'impaginazione di documenti ipertestuali disponibili sul *web*. Attraverso l'HTML è possibile assegnare una struttura alla pagina web definendone gli elementi. Ogni elemento della pagina è limitato con l'uso di "*marcatori*", chiamati anche *tag*, secondo la seguente struttura:

```
<nome_del_tag attributo1="valore" attributo2="valore" ... >
     elemento corrispondente al tag
</nome_del_tag>
```

Gli elementi possono essere *intestazioni*, *paragrafi*, *immagini*, *collegamenti ipertestuali*, *elenchi*, *etc.*
L'utilizzo di un *tag* è costituito da una chiamata di apertura `<nome_del_tag>` e una di chiusura `<nome_del_tag/>`, proprio ad indicare il blocco che racchiude l'elemento corrispondente a quel tag. Un tag può prevedere l'uso di uno o più attributi per definire le proprietà che deve avere quell'elemento.

1.1 I TAG DELLA LEZIONE DI OGGI

1. **`<html> ... </html>`**
   Apre l'intestazione di una pagina web, segna cioè l'inizio e la fine di un documento html.

2. **`<head> ... </head>`**
   Raccoglie le informazioni descrittive del documento. Ad esempio i file esterni di cui si ha bisogno, il titolo della pagina, etc.

3. **`<title> ... </title>`**
   Rappresenta il titolo della pagina.

4. **`<link ... />`**
   Utilizzato per importare il file *.css* che definisce gli aspetti grafici della pagina web.

5. **`<script ... />`**
   Utilizzato per importare i file *.js* (*Javascript*), il cui contenuto permette di dare funzionalità logica agli elementi.

6. **`<body> ... </body>`**
   Rappresenta il corpo del documento. All'interno si definisce la struttura che deve avere la pagina web.

7. **<h1> ... </h1>**
   I tag da h1 ad h6 rappresentano lo stile topografico che vogliamo assegnare ad un titolo.
   La numerazione ne regola la dimensione.

8. ** ... **
   Utilizzato per definire un paragrafo.

9. **<button> ... </button>**
   Permette di definire un bottone cliccabile e di associare ad esso una funzione.

10. **
**
    Assegna uno spazio tra gli elementi.

11. **<input ... />**
    Definisce un *box* per l'inserimento di un input da tastiera. Il tipo di dato può essere specificato come un indirizzo *mail*, un testo, un numero, una *password*, etc.

12. ** ... **
    I tag *div* sono utilizzati per strutture gli elementi della pagina. Ad esempio è possibile raccogliere più elementi su una riga definendo un *div* di classe *row*.

13. ** ... **
    Assegnano la funzionalità di collegamento ipertestuale ad un oggetto o ad un testo.
    Il collegamento può essere ad un documento, ad una pagina *web*, ad un indirizzo *mail*, etc.

1.2 GLI ATTRIBUTI AI TAG DELLA LEZIONE DI OGGI

a) **rel**
   Specifica la relazione tra il documento HTML corrente e il documento che si sta importando. Nel nostro caso il valore "*stylesheet*" sta ad indicare un documento di tipo stile grafico.

b) **type**
   Indica il tipo di elemento che si sta utilizzando. Ad esempio per il tag *"input"* indica che il tipo di dato gestito sarà un testo.

c) **href**
   Assegna la funzionalità di collegamento ipertestuale verso l'indirizzo indicato.

d) **src**
   Permette di definire un percorso ad un file.

e) **id**
   Permette di assegnare un identificativo all'elemento dichiarato nel tag.

f) **class**
   Definisce la classe di stile grafico a cui fa riferimento quel tipo di oggetto. Esempio "*myButton*" sta ad indicare che per quel tag di tipo *button* avrà un aspetto grafico definito dalla classe "*myButton*".

g) **onclick**
   Permette di definire la funzionalità di "click" su un oggetto, come ad esempio un bottone.

Parte 2 – Il linguaggio CSS

Il CSS (*Cascading Style Sheets* – "*Fogli di stile a cascata*") è un linguaggio usato per descrivere lo stile di un documento HTML. Grazie ad esso è possibile definire come gli elementi HTML devono essere visualizzati dall'utente, ad esempio si può intervenire sulla formattazione del testo, sul posizionamento degli elementi grafici, e così via. L'idea per la quale sono stati introdotti è: separare il contenuto di un documento dalla sua presentazione.

La sintassi per definire un insieme di regole in un CSS è la seguente:

Figura 1. Sintassi CSS

2.1 – I SELETTORI

Servono a selezionare la parte o le parti di un documento HTML soggette ad una specifica regola.
Ad esempio, nel seguente HTML

```
<p id="paragraph" class="myParagraph">Words words words...and words!</p>
```
[1]

l'attributo *class* permette di definire l'aspetto grafico di cui vogliamo arricchire il nostro paragrafo.
Il valore "*myParagraph*" trova corrispondenza nel foglio di stile CSS, dove ne è definita la struttura.

Come avviene la selezione automatica dello stile?

Per selezionare uno specifico elemento dell'HTML tramite il suo attributo *class* si può usare la seguente struttura (con riferimento all'esempio in [1]):

```
.myParagraph {……}
```

`.myParagraph` → identifica tutti gli elementi dell'HTML che hanno come valore dell'attributo *class* denominato "*myParagraph*". Più elementi con attributo *class* nell'HTML possono avere lo stesso valore: in ognuno di esso saranno apportate le modifiche definite tra le parentesi graffe *{…}*.

2.2 LE PROPRIETA'

Osserva l'immagine in Figura1:

- Ogni blocco riservato alle dichiarazioni può contenere una o più dichiarazioni, separate dal *punto e virgola.*
- Ogni dichiarazione include il nome di una proprietà e un valore, separati dai *due punti,* e termina sempre con *un punto e virgola*.
- Un blocco di dichiarazioni è sempre contenuto in parentesi graffe.

Una proprietà definisce un aspetto dell'elemento dell'HTML (richiamato tramite un selettore).
Le proprietà che useremo in questa lezione sono:

- ***color*** → consente di modificare il colore di un elemento.
  I colori standard possono essere richiamati con le seguenti 16 parole chiave:

```
black | navy | blue | maroon | purple | green | red | teal | fuchsia |
olive | gray | lime | aqua | silver | yellow | white
```

Se invece si vuole utilizzare la vasta gamma dei colori con le più diverse gradazioni e sfumature, si possono utilizzare i codici con notazione esadecimale. Ad esempio, il colore nero corrisponde a #000000, il bianco a #FFFFFF, la vasta gamma di colori è facilmente disponibile con una ricerca sul web.
Nella sintassi CSS avremo qualcosa del tipo:

```
.myParagraph {
  color: #abababb;
}
```

- ***background-color*** → Imposta il colore dello sfondo di un elemento. Ad esempio, si può impostare lo sfondo di un pulsante con la dichiarazione:

  background-color: #f45c42;

- ***border-style*** → Imposta lo stile dei quattro bordi di ogni elemento. Può assumere 4 valori: ***dotted, solid, double*** e ***dashed.***

  Esempio: ***border-style: double;***

- ***border-color*** → Imposta il colore dei bordi dell'elemento in questione.

  Esempio: ***border-color: #910000;***

- ***height*** → Imposta l'altezza di un elemento. Il *valore* può essere espresso da:
  - un valore numerico con unità di misura:
    height: 50px;
  - un valore in percentuale, definito rispetto all'altezza esplicitamente dichiarata del blocco contenitore:
    height: 50%;
  - ***"auto",*** l'altezza sarà quella determinata dal contenuto, è il valore di default:
    height: auto;

- ***width*** → Imposta la larghezza dell'elemento. Il valore può essere espresso allo stesso modo della proprietà *height*.
-
- ***text-align*** → Serve ad impostare l'allineamento del testo. I valori possono essere rappresentati dalle seguenti parole chiave:
  - ***left***: allinea il testo a sinistra;
  - ***right***: allinea il testo a destra;
  - ***center***: centra il testo;
  - ***justify***: giustifica il testo;

  Esempio: ***text-align: center;***

- ***position*** → Utile per gestire la posizione degli elementi. Alcuni dei valori con cui è possibile definire la modalità di posizionamento sono:
  - ***static:*** valore di default.
  - ***relative*:** l'elemento è posizionato relativamente al suo box contenitore.
    Esempio di box contenitore (dal codice HTML)*:*

    ```
    <div id="divId" class="myDiv">
        <p>Hello World</p>
    </div>
    ```

  - ***fixed:*** l'elemento è posizionato rispetto alla finestra del *browser*.

Parte 3 – Il linguaggio JAVASCRIPT

JavaScript è il linguaggio di programmazione che permette di aggiungere dinamicità ad una pagina web, attraverso l'interazione con l'utente. I documenti realizzati fino ad adesso in HTML e CSS sono infatti statici.

In seguito vengono mostrate quattro operazioni, definite funzioni. Si noti che ciascuna operazione è incapsulata in una **function.** Ciascuna function è denominata con un termine intuitivo in riferimento all'azione che intende svolgere (Es. La funzione denominata "*changeColor*" effettuerà il cambiamento di un colore).

I nomi delle funzioni vengono tipicamente congiunti in un unico termine, come la ricorrente struttura:

eseguiAzione

Tuttavia questa scelta è puramente convenzionale in informatica e non obbligatoria.

Le convenzioni invece **obbligatorie** per la corretta esecuzione del programma sono:

- le parentesi graffe all'inizio e alla fine di ciascuna funzione;
- i punti e virgola dopo ciascuna istruzione.

Ciascuna funzione effettua le proprie operazioni come conseguenza alla manifestazione di un evento (ad esempio il "*click*" di un bottone).

3.1 FUNZIONE SHOWALERT

La funzione *showAlert* apre una finestra di dialogo con il seguente contenuto: "*What the f* this is an alert message!*"

```
function showAlert() {
      alert('What the f... this is an alert message!');
}
```

PREMESSE:

a) Il termine **var** indica una variabile. La variabile può essere pensata come un oggetto a cui si vuole associare un valore.

   Esempi: **var** numero = 1; **var** nome = "*Chiara*";

   Nota bene: va sempre inserito il punto e virgola alla fine di ciascuna operazione!

b) Nei linguaggi di programmazione il simbolo " = " viene utilizzato per effettuare un'associazione da destra verso sinistra.

   Esempi:
   var nome = "Mickey";
   Associo alla variabile *nome* il valore "Mickey".

   var cognome = "Mouse";
   Associo alla variabile cognome il valore "Mouse".

   nome = cognome;
   E ora? La variabile nome ora conterrà come valore "Mouse", non più "Mickey"!

3.1 FUNZIONE CHANGECOLOR

La funzione *changeColor,* associa il colore scelto al paragrafo.

La denominazione "`document.getElementById('inColor').value`" estrae dal documento HTML il **valore**(*value*) dell'elemento che possiede come **id** "*inColor*". Tale valore sarà un colore in esadecimale, come spiegato in precedenza.

La denominazione "`document.getElementById('inColor').style.color`" estrae il colore dell'elemento di HTML con **id** "*paragraph"*. Si associa a quest'ultimo il colore appena estratto con l'uso del simbolo "=".

```
function changeColor() {

      var color = document.getElementById('inColor').value;
      document.getElementById('paragraph').style.color = color;

}
```

3.2 FUNZIONI MOVELEFT E MOVERIGHT

La variabile *myRectangleLeft,* definita fuori dalla funzione, rappresenta, per tale motivo, una variabile "**globale**", in contrapposizione alle variabili "**locali**" che esistono soltanto entro le parentesi graffe dove sono state definite le funzioni. I valori memorizzati dalle variabili "globali" sono disponibili a tutte le funzioni di quel file, mentre i valori delle variabili locali sono disponibili per la sola funzione entro cui sono dichiarate.

La funzione *moveLeft* sposta l'elemento con **id** "*divRectangle*" a sinistra di *10pixel*, variando la posizione verso sinistra dell'elemento. Ricorda che un pixel è un'unità puntiforme con il quale si rappresentano le immagini, così come le dimensioni di un monitor.

La funzione *moveRight* sposta l'elemento con **id** "*divRectangle*" a destra di *10pixel*, variando la posizione verso destra dell'elemento.

L'uso della variabile globale *myRectangleLeft,* da parte di entrambe le funzioni, permette di poter condividere gli effetti degli spostamenti a destra e a sinistra, con risultato conseguente che un successivo spostamento avverrà dall'ultimo memorizzato. Ad esempio, partendo dalla posizione iniziale e spostandosi di 40 pixel con 4 click verso destra, un successivo spostamento a sinistra farà collocare l'oggetto in posizione 30 pixel.

```
var myRectangleLeft = 0;

function moveLeft() {

      myRectangleLeft = myRectangleLeft - 10;
      document.getElementById('divRectangle').style.left = myRectangleLeft;

}

function moveRight() {

      myRectangleLeft = myRectangleLeft + 10;
      document.getElementById('divRectangle').style.left = myRectangleLeft;

}
```

Grazie per la vostra attenzione!