Introduzione alla programmazione lato client

Il linguaggio Javascript

Sviluppo di siti web – UD14

prof. Giovanni Borga

Cos'è la programmazione

La programmazione, in informatica, è un'attività con cui possiamo far svolgere ad un computer delle azioni connesse ad un particolare obiettivo.

Non tutti i «listati» di codice sono programmi

L'HTML come abbiamo visto non è un linguaggio di programmazione in quanto non permette di far compiere delle scelte al computer.

Gli elementi di base della programmazione sono sostanzialmente 4:

- **COMANDI**
- **COSTANTI**
- **VARIABILI**
- **FUNZIONI**

I comandi

I comandi, anche detti istruzioni, sono:

UN SET DI PAROLE CHIAVE FINITO (come i tag HTML)

messo a disposizione dal tipo di linguaggio scelto per programmare.

I comandi hanno dei **nomi univoci** che non vanno mai utilizzati per altri scopi (es. per variabili o funzioni).

Per **utilizzare un comando** è sufficiente scrivere il suo nome nel punto del flusso in cui vogliamo che accada l'effetto per cui il comando è stato creato.

Molti comandi:

a) Accettano dei dati in ingresso
b) Restituiscono dati in uscita
c) Fanno entrambe le cose

Costanti e variabili

Costanti e variabili sono dei «contenitori di dati» dotati:

a) di un proprio nome
b) di un valore (ovvero di un dato)

(come gli attributi dei tag HTML)

Le **costanti** sono caratterizzate dal fatto che il valore ad esse associato non è modificabile durante l'esecuzione del programma.

Le **variabili** invece vengono espressamente utilizzate come «veicolo di trasporto» di dati e informazioni e il valore ad esse associato può essere cambiato.

Tipicamente quindi:

Per le costanti svolgeremo solo un'azione:

UTILIZZO (lettura del valore)

Per le variabili svolgeremo due azioni:

ASSEGNAZIONE (scrittura del valore)

UTILIZZO (lettura del valore)

Costanti e variabili devono inoltre essere create, ovvero DICHIARATE. La dichiarazione può avvenire in modo isolato, oppure può essere fatta contestualmente all'assegnazione del valore.

Funzioni

Le funzioni non sono altro che sequenze di comandi

Sono come dei «mini-programmi» dentro al programma principale e servono per l'appunto a svolgere più volte la stessa sequenza senza doverla riscrivere.

Per utilizzare le funzioni occorre seguire due distinte fasi:

1. **Definizione**
2. **Chiamata**

La definizione viene svolta per prima e una sola volta
serve a indicare al programma **che sequenza** eseguire

Le chiamate possono essere multiple e vengono svolte dopo la definizione
servono a indicare al programma principale **quando** eseguire la sequenza.

(come per gli stili CSS che devono essere definiti e poi applicati)

Javascript

JavaScript è un linguaggio di scripting lato-client, che viene interpretato dal browser.

Il web funziona a due livelli:

1. le pagine web vengono inviate all'utente da un web server, cioè da un programma che si trova su un computer remoto, e che per lo più non fa nient'altro che inviare le pagine a chi ne fa richiesta
2. l'utente che naviga visualizza sul proprio browser le pagine che gli vengono inviate.

Un "browser" è un programma che permette di leggere le pagine scritte in linguaggio HTML. Quando visualizziamo le pagine web ci sono dunque due computer che si parlano: il server e il client.

Alcuni linguaggi di scripting (asp, php, perl) vengono eseguiti dal web server (si chiamano appunto linguaggi server side o lato server). JavaScript, invece, viene eseguito sul nostro computer direttamente dal browser (è un linguaggio client side o lato client).

JavaScript è un linguaggio lato client. Ovvero gli script hanno validità all'interno delle singole pagine web, e non da una pagina all'altra.

Javascript

I **linguaggi di scripting** sono definiti tali perché **la sintassi degli script si può scrivere direttamente dentro la destinazione** ed eseguire **senza doverli compilare.**

Con i linguaggi di programmazione invece (come il C, il Java ecc.) la sintassi va passata ad un compilatore, che produce un file binario in cui la sintassi scompare.

I programmi di Windows ad esempio sono dei file compilati e sono riconoscibili dal formato EXE. In essi non c'è più traccia della sintassi originaria (cioè del codice "sorgente").

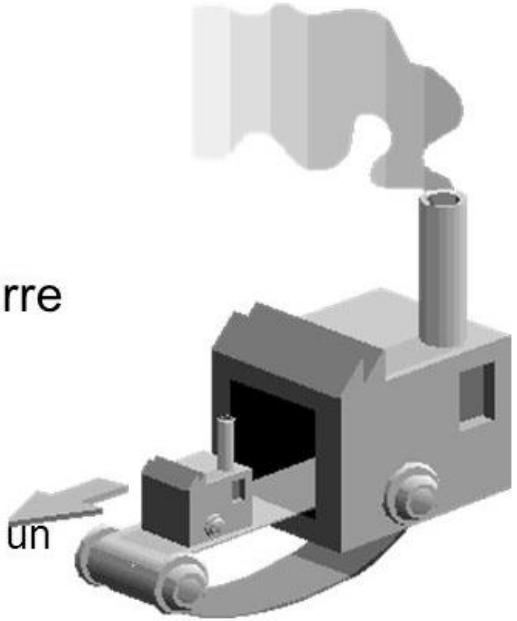
JavaScript non è compilato: è possibile quindi visualizzare in qualsiasi momento il codice di una pagina HTML e leggere le righe di codice JavaScript.

Dire che è un linguaggio di scripting sottintende anche il fatto che si tratta di un **linguaggio interpretato**: ovvero l'assenza del compilatore impone che sia lo stesso browser a «tradurre» i comandi in operazioni mediante un apposito «motore di scripting» che legge le parti di codice JavaScript.

La programmazione orientata agli oggetti (OOP): oggetti e classi

- Gli ***oggetti*** sono generati da una ***classe***
  - Si dicono anche ***istanze*** della classe

- La ***classe*** è uno schema per produrre una categoria di oggetti identici di struttura
  - La classe costituisce il *prototipo*
  - La classe descrive le caratteristiche di un oggetto
  - Una classe è una fabbrica di istanze: possiede lo schema e la tecnica di produzione



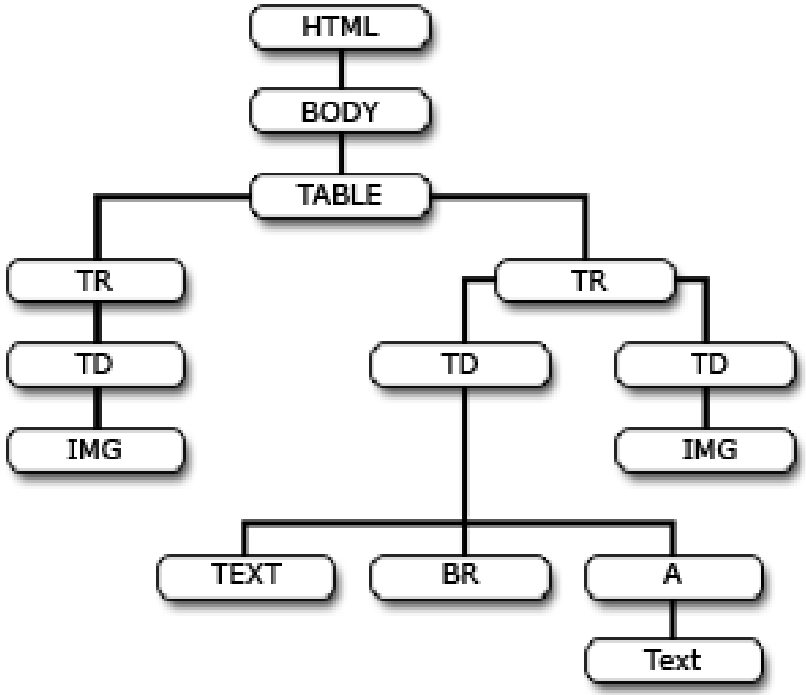
Oggetti e pagina HTML

La nostra pagina HTML è governata dal Document Object Model (DOM)

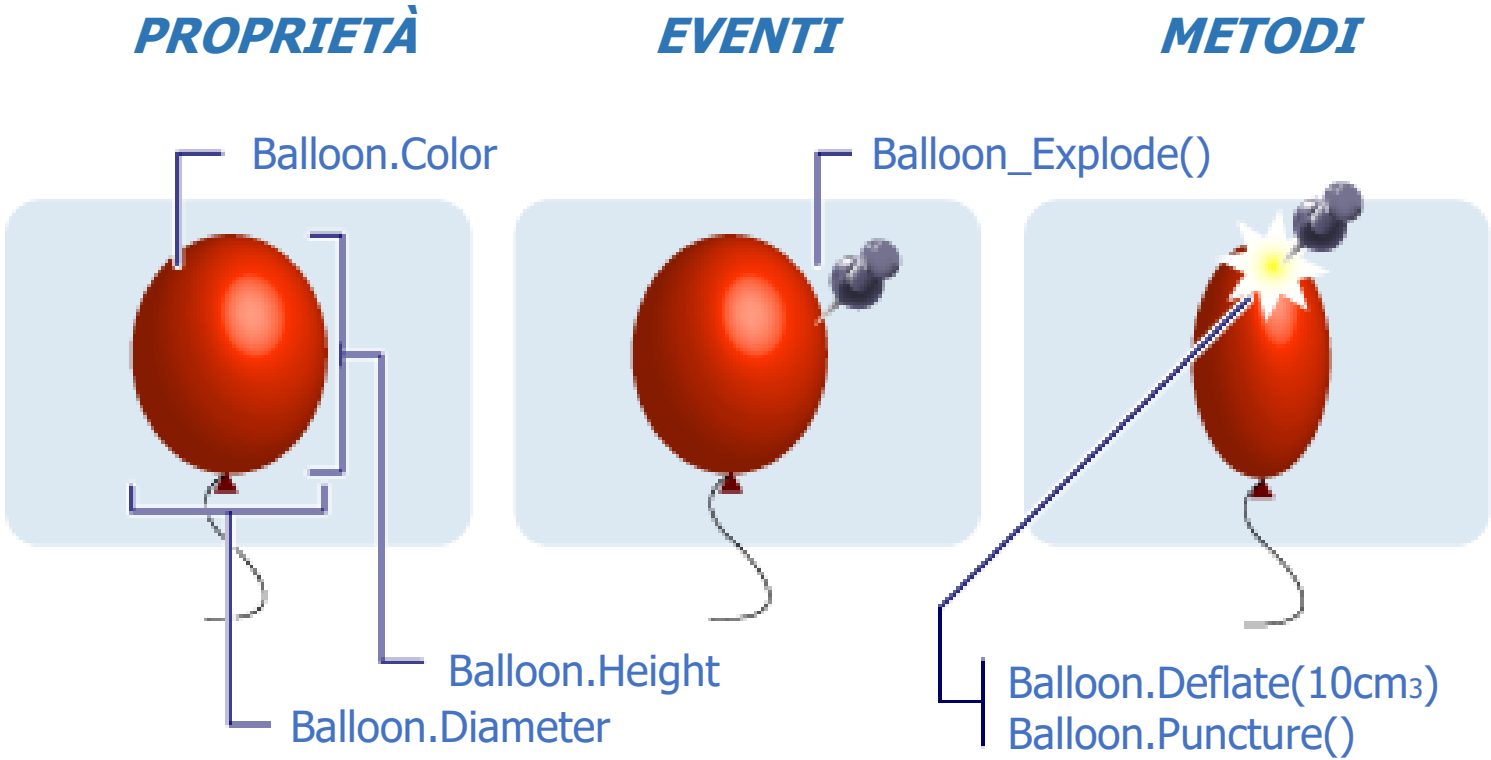
Ogni elemento HTML è un oggetto anche per la programmazione Javascript.

Tutti gli elementi HTML sono oggetti utilizzabili da Javascript principalmente tramite i rispettivi ID.

E' anche possibile far interagire gli script tramite i type selector o le classi anche se è una tecnica molto meno efficiente.



Caratteristiche degli oggetti



Utilizzo delle caratteristiche degli oggetti

PROPRIETÀ

- *Si **scrivono** (se non sono readonly)*
- *Si **leggono** (se non sono writeonly)*

METODI

- *Si **invocano** (opzionalmente inviando delle informazioni necessarie) ed eventualmente si registra l'effetto (si recupera un'informazione di ritorno)*

EVENTI

- *Si **intercettano** (e si associano ad un comando)*

Esempi di proprietà metodi ed eventi Javascript/HTML

PROPRIETÀ

- ***top***, ***bottom*** *(riferiti ad un qualsiasi elemento HTML corrispondono alle rispettive proprietà CSS)*
- ***value*** *(riferito ad una casella di testo corrisponde alla stringa contenuta nella casella)*

METODI

- ***open()*** *(invocato dall'oggetto window – il browser – apre una nuova finestra)*
- ***write()*** *(invocato dall'oggetto document – la pagina – scrive qualcosa al suo interno)*

EVENTI

- ***onclick()*** *(riferito ad esempio ad un pulsante si verifica quando esso viene premuto con il click del mouse)*

Sintassi di Javascript

Sintassi di base

Alcune informazioni di base sulla sintassi Javascript:

1. Al termine di ogni riga di codice occorre mettere un punto e virgola (**;**)

2. I tipi di dato utilizzabili sono sostanzialmente quattro:
   - **Numerico**: si utilizzano le **cifre** senza nessun delimitatore. Il separatore decimale è il punto.
   - **Stringa**: si utilizzano tutti i **caratteri** delimitandolo con doppio o singolo apice.
   - **Booleano**: il classico valore binario. Si utilizza scrivendo ***True*** o ***False*** senza alcun altro carattere o delimitatore
   - **Nullo**: significa assenza di dato. Si utilizza scrivendo ***Null*** senza alcun altro carattere o delimitatore.

3. I commenti:

```
// commento di una sola riga
/* commento su più righe */
```

Sintassi di base

4. Per assegnare un valore ad una variabile si usa il simbolo di uguale (**=**)

5. Quando il simbolo di uguale si utilizza come operatore di comparazione in una espressione si deve scrivere die volte. Es:

```
if (x == 5) { …. }
```

6. Le **parentesi tonde** () si utilizzano per lo più per racchiudere gli argomenti delle funzioni mentre quelle graffe il codice da eseguire. Es:

```
function saluta (nome) { alert(nome) } ;
```

7. Le **parentesi graffe** {} si usano anche per i blocchi di codice del costrutto IF – ELSE - END IF

8. Le **parentesi quadre** [] si usano per gli indici di vettori e matrici. Es:

```
alunni [4]
```

Lavorare con le variabili

Creare (dichiarare) una variabile:

```
var mioNome;
```

Attribuire il valore ad una variabile:

```
mioNome = "pippo";
var mioNome = "pippo";
```
(in questo caso faccio contemporaneamente la dichiarazione)

Utilizzare una variabile (ad esempio usarne il valore per modificare il titolo della pagina):

```
document.title = mioNome;
```

Posizione del codice Javascript

Il codice Javascript può essere scritto in qualunque punto all'interno del tag HEAD o del tag BODY.

La cosa fondamentale è che sia contenuto nel suo tag <script>

```
<script>
    alert("ciao");
</script>
```

NB: è opportuno che

- ***nel tag HEAD vengano inserite le definizioni di funzioni e le inclusioni di librerie esterne***
- ***nel tag BODY tutti gli altri script***

Inclusione di script esterni

La logica di inclusione degli script è esattamente la stessa di quella dei fogni di stile.

Come i CSS i file javascript sono semplici file di testo salvati con una propria estensione: **.js**

All'interno di un file js deve essere scritto **solo codice con sintassi Javascript.**

Per includere un file javascript si inserisce dentro all'HEAD un tag **<script>** con la seguente sintassi:

```
<script src="scripts/mioscript.js"></script>
```

Con questa tecnica è possibile **utilizzare librerie anche molto complesse** dentro alle nostre pagine web avendo a disposizione numerosissime funzioni pronte per gli scopi più diversi.

Molte librerie javascript servono per creare effetti ed animazioni e consentono di migliorare sensibilmente l'usabilità e l'esperienza utente delle pagine web.

L'importante è che queste librerie siano costantemente aggiornate e ben documentate.

La libreria JQuery

jQuery

What is jQuery?

jQuery is a fast, small, and feature-rich JavaScript library. It makes things like HTML document traversal and manipulation, event handling, animation, and Ajax much simpler with an easy-to-use API that works across a multitude of browsers. With a combination of versatility and extensibility, jQuery has changed the way that millions of people write JavaScript.

Concetti di base per l'utilizzo di jQuery

Tutti i comandi di jQuery iniziano con il simbolo $

$ è in pratica un alias di jQuery

Tutto il codice di utilizzo della libreria è associato all'evento READY del documento

Questa tecnica prevede semplicemente che il nostro codice sia contenuto nella seguente frase:

```
$(document).ready ( function() {
        …
        …
});
```

Concetti di base per l'utilizzo di jQuery

Tutti gli elementi della pagina HTML sono indirizzati tramite i selettori CSS

Di fatto con jQuery si selezionano (query) elementi HTML e si eseguono delle azioni su di essi.

La sintassi di base è: **$(selettore).azione()**

Esempi:

$(this).hide()	nasconde l'elemento corrente.
$("p").hide()	nasconde tutti gli elementi <p>.
$(".test").hide()	nasconde tutti gli elementi con class="test".
$("#test").hide()	nasconde l'elemento con id="test".

NB: notare che tutti i selettori CSS sono racchiusi da doppi apici.

Concetti di base per l'utilizzo di jQuery

Molte azioni sono associate agli eventi:

Mouse Events	Keyboard Events	Form Events	Document/Window Events
click	keypress	submit	load
dblclick	keydown	change	resize
mouseenter	keyup	focus	scroll
mouseleave		blur	unload

Esempio di codice da eseguire quando si clicca sopra un qualsiasi paragrafo:

```
$("p").click(function() {
  …
  … codice da eseguire …
  …

});
```

Documentazione

I principali riferimenti per l'utilizzo di jQuery sono:

- **La documentazione principale:** http://api.jquery.com
- **Il portale per l'autoformazione:** http://learn.jquery.com
- **Il tutorial fornito dal W3C:** http://www.w3schools.com/jquery