Programmazione lato client

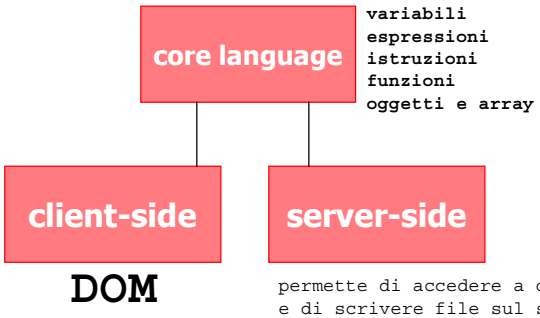
JavaScript

JavaScript

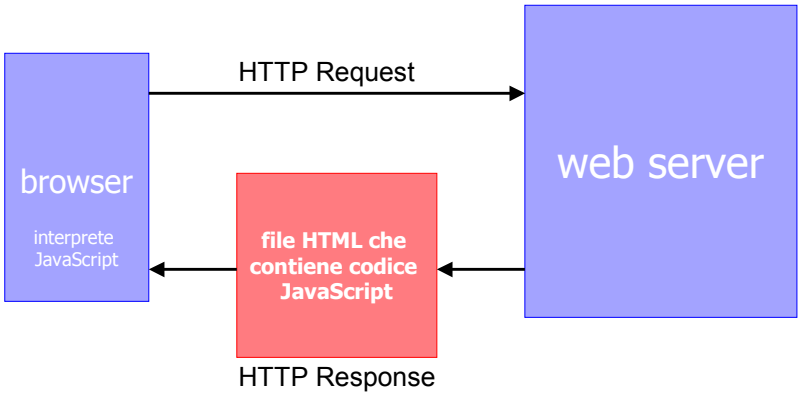
- Netscape: **JavaScript**
- Microsoft: **JScript**
- **ECMAScript** (ECMA-262)
  (European Computer Manufactures Association)

JavaScript

Linguaggio di script interpretato con alcune caratteristiche Object Oriented



JavaScript



JavaScript

- Il browser **visualizza** il documento **HTML** e interpreta (**esegue**) le istruzioni scritte in **JavaScript**

```
<script language="JavaScript">
     <!--
          codice JavaScript
     //-->
</script>
```


JavaScript

JavaScript

- Per "arricchire" le pagine HTML di codice JavaScript si può usare (**ed è opportuno farlo**) un **file esterno** con estensione **.js** che contiene funzioni JavaScript

```
<script src="myfile.js"></script>
```


JavaScript

- Molto spesso il codice JavaScript è scritto **all'interno dei tag HTML** come valore di nuovi attributi che sono stati introdotti per gestire gli eventi generati dall'utente

```
<a href=""
   onMouseOver ="codice JavaScript;"
   onMouseOut ="codice JavaScript;"
   onClick ="codice JavaScript;">
</a>
```

JavaScript

- L'**interazione** con l'utente è guidata dagli **eventi**
- L'**utente genera un evento**
- Se nel file HTML esiste del codice JavaScript associato a quell'evento (**event handler**), questo viene eseguito

Core JavaScript

- Fornisce i **costrutti di base** di un linguaggio di programmazione:
  variabili, espressioni, istruzioni, …
- Per certi aspetti ricorda i linguaggi C e Java

JavaScript: struttura lessicale

- Le istruzioni devono essere separate tra loro dal ;
- **Commenti**

  es. // questo è un commento

  es. /* questo è un commento */

  es. <!--questo è un commento//-->

JavaScript: struttura lessicale

- Gli identificatori possono iniziare con _ **$** o con una **lettera** e non possono iniziare con un numero
- Non si possono usare le parole riservate
- JavaScript è case sensitive

JavaScript: tipi di dati

- **Numeri** (floating point, 8 byte)
  - ✓ **Infinity, -Infinity, NaN**
  - ✓ **Number.MAX_VALUE, Number.MIN_VALUE**
- **Stringhe**
  - ✓ **msg = "hello world"**
  - ✓ **msg.length**
- **Booleani**

JavaScript: tipi di dati

- **Array**
  - ✓ **a = new Array()**
  - ✓ **b = [10, "ciao", true]**
- **Oggetti**
  - ✓ JavaScript permette al programmatore di definire i propri oggetti ma, soprattutto, fornisce una gerarchia di **oggetti predefiniti** che permettono di manipolare i documenti HTML

JavaScript: tipi di dati

- Si possono creare oggetti inizialmente vuoti (senza proprietà) usando l'operatore **new** e il costruttore **Object()**

```
var auto = new Object();
```

- Si possono poi definire le proprietà di un oggetto semplicemente usando degli identificatori e assegnandovi dei valori
- Si usa la notazione con il simbolo .

JavaScript: tipi di dati

- **Funzioni**
  (sono viste come tipi di dati)

```
function nomefunzione (arg1, arg2, … )
 {
   body della funzione
   return espressione;
 }
```

JavaScript: variabili

- JavaScript è un linguaggio **debolmente tipato**
  - ✓ `i = 1;`
  - ✓ `i = "Hello world";`
  - ✓ `i = true;`

JavaScript: variabili

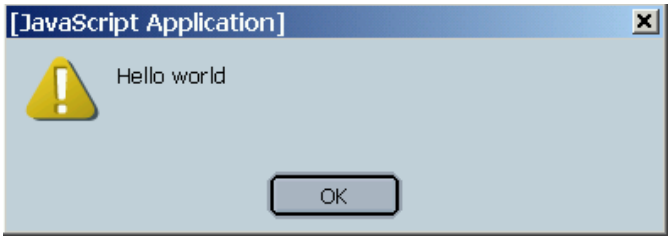
- **Non è obbligatorio** dichiarare le variabili
  - ✓ per dichiarare una variabile si usa la parola chiave **var**
  - ✓ se si introduce una variabile senza averla prima dichiarata lo spazio ad essa riservato viene allocato automaticamente

JavaScript: variabili

- Esistono variabili **globabli** e **locali**
- All'interno delle funzioni, per essere sicuri di usare **variabili locali**, si devono dichiarare esplicitamente con **var**
- Attenzione a dove si dichiarano le variabili! (vedi esempio 4)

JavaScript: input/output

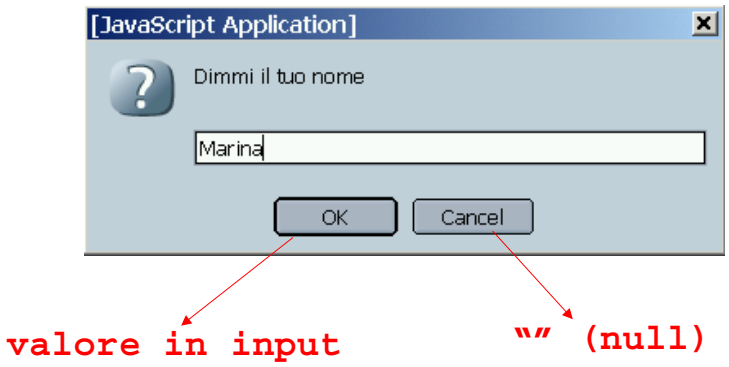
- **window.alert("Hello world")**



undefined

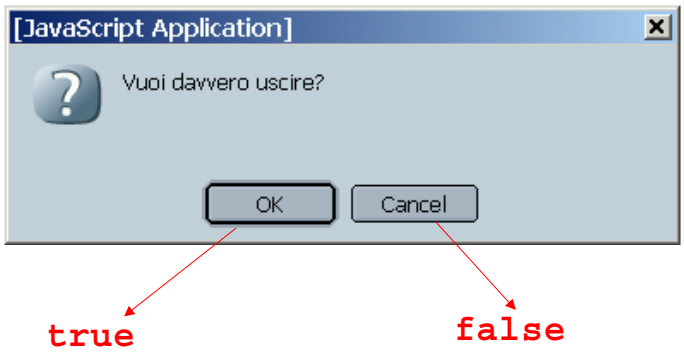
JavaScript: input/output

- window.prompt("Dimmi il tuo nome","")



JavaScript: input/output

- window.confirm("Vuoi davvero uscire?")



JavaScript: if

```
if (condizione)
   {
        ramo true
   }
 else
   {
        ramo false
   }
```


JavaScript: while e do/while

```
while (condizione)
   {
        istruzioni
   }
```

```
do
   {
        istruzioni
   }
while (condizione)
```

JavaScript: for e for/in

```
for (inizializza; test; incremento)
   {
       istruzioni
   }
```

```
for (variabile in oggetto)
   {
       istruzioni
   }
```


JavaScript: break e continue

break

causa l'uscita da un ciclo

continue

salta l'iterazione corrente in un ciclo e passa a quella successiva

JavaScript: switch

```
switch (n) {
  case value1
      { istruzioni; break;}
  case value2
      { istruzioni; break;}

        …
  default
      { istruzioni; break;}
```


Esercizi

- Esempio 1
- Esempio 2
- Esempio 3
- Esempio 4
- Esempio 5