Programmazione client-side: JavaScript

JavaScript: Introduzione

- JavaScript è un linguaggio di programmazione/scripting molto flessibile, utilizzato in vari settori applicativi (proposto da Netscape)
- Nelle applicazioni web, è usato anche per programmazione server-side (p. es. JScript in ASP)
- Una versione "estesa" di JavaScript costituisce il meccanismo di scripting standard all'interno dei web browser.
- Standardizzazione ECMA-262: "ECMAScript"

Caratteristiche

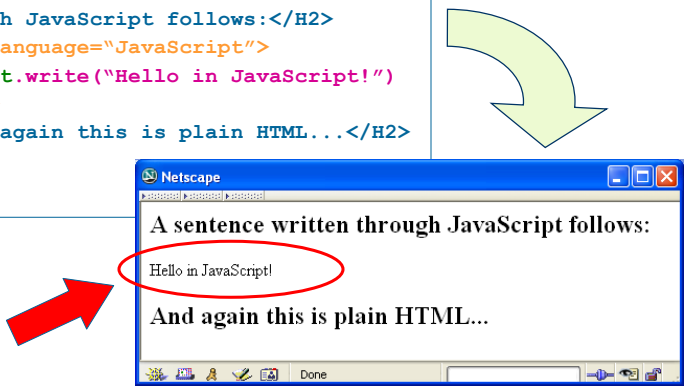
- JavaScript non ha niente a che fare con Java, tranne la somiglianza sintattica
- Linguaggio interpretato: prende spunto da Perl
- Debolmente tipizzato: tipicamente non si associa un tipo alle variabili
- Articolato e complesso, anche se spesso usato per applicazioni molto semplici.
- *Client-side JavaScript* = "core" JavaScript + DOM (Document Object Model)

Precisazioni su client-side JS

- I moderni browser contengono direttamente (no plug-in) l'interprete JavaScript, per poter eseguire programmi JavaScript
- I programmi JavaScript sono generalmente inseriti all'interno di documenti HTML, per mezzo dell'apposito tag `<script>`

Un primo esempio

```
<html>
  <head></head>
  <body>
    <H2> A sentence written
       through JavaScript follows:</H2>
    <script language="JavaScript">
      document.write("Hello in JavaScript!")
    </script>
    <H2> And again this is plain HTML...</H2>
  </body>
</html>
```



Primo esempio perfezionato

- Per evitare problemi con browser che non supportano JavaScript, il codice viene normalmente inserito dentro un commento HTML

```
...
  <body>
    <H2> A sentence written
       through JavaScript follows:</H2>
    <script language="JavaScript">
    <!- "hidden" JS code
      document.write("Hello in JavaScript!")
    // end of "hidden" JS code -->
    </script>
    <H2> And again this is plain HTML...</H2>
  </body>
</html>
```

Inizio commento

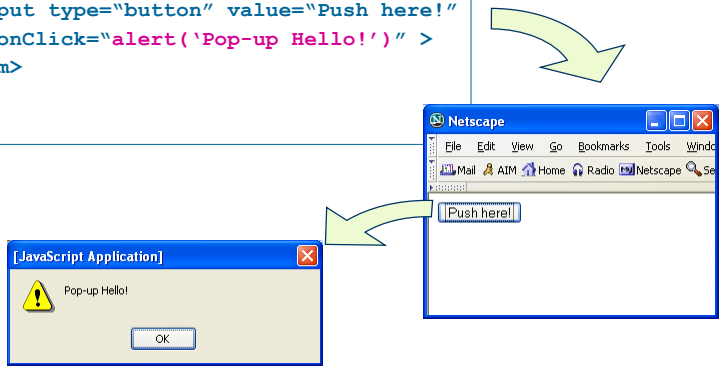
Fine commento

Gestione JS degli eventi grafici sul browser

- L'interazione dell'utente con l'ambiente grafico del browser determina l'occorrenza di "eventi"
- Esempi di eventi:
  - onClick, che si verifica quando un bottone viene premuto
  - mouseOver, quando si passa con il cursore del mouse sopra un componente grafico.
- Questi eventi possono essere riconosciuti e gestiti tramite codice e funzioni JavaScript: per far questo si usano i cosiddetti *event handlers*.

Eventi: esempio

```
</html>
  <head></head>
  <body>
    <form>
      <input type="button" value="Push here!"
         onClick="alert('Pop-up Hello!')" >
    </form>
  </body>
</html>
```

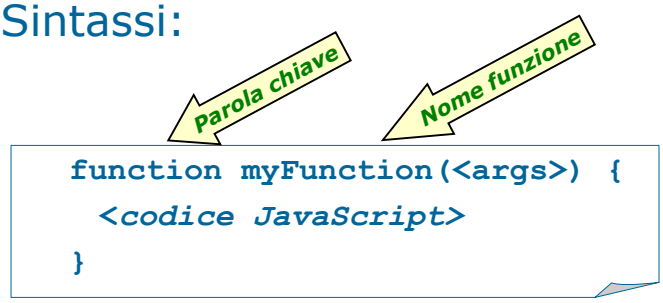


Alcuni eventi JavaScript

- onClick
- onDblClick
- onMouseDown
- onMouseUp
- onMouseOver
- onMouseOut
- onMouseMove
- onKeyPress
- onKeyUp
- onKeyDown
- onChange
  (su text fields)
- onLoad
- Altri…
  - Alcuni eventi dipendono dallo specifico browser

Funzioni JavaScript

- JavaScript fornisce il meccanismo delle "funzioni" per rendere più flessibile la programmazione
- Sintassi:

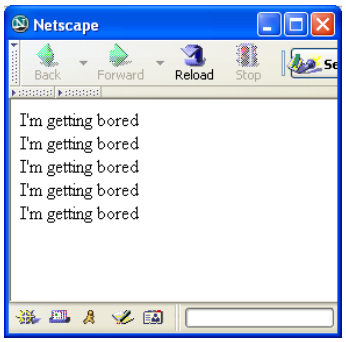


Esempio: uso di funzioni

```
...
  <body>
  <script language="JavaScri
    <!--
    function f1(msg, times) {
      for(i=1; i<=times; i++)
      document.write(msg);
    }
    f1("I'm getting bored"+"<br>", 5);
    //-->
  </script>
  </body>
</html>
```

DEFINIZIONE

CHIAMATA



Variabili JavaScript

- Una variabile è un nome associato a un dato
- Può contenere valori *di qualsiasi tipo*
- Prima di poterla usere,
  occorre dichiararla con la parola chiave **`var`**
  (ed eventualmente inizializzarla): se non lo si fa esplicitamente, JS lo fa implicitamente

```
...
  var i;
  var x = 5;

  x = "Che bella giornata!"
  var z = 0;
...
```

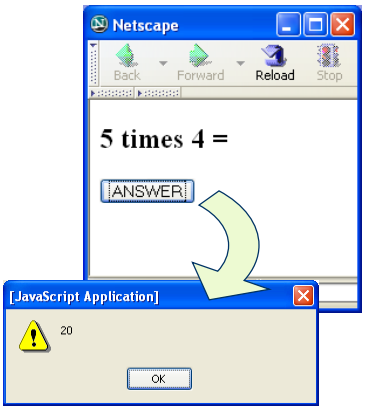
Funzioni ed event handler

- Si può specificare di richiamare una funzione al verificarsi di un evento

```
...
<body>
  <script language="JavaScript">
    <!--
    function f2(x, y) {
      alert (x*y);
    } //-->
  </script>
  <H2>5 times 4 =</H2>
  <form>
  <input type="button" value="ANSWER"
    onClick="f2(5,4)">
  </form>
...
```

DEFINIZIONE

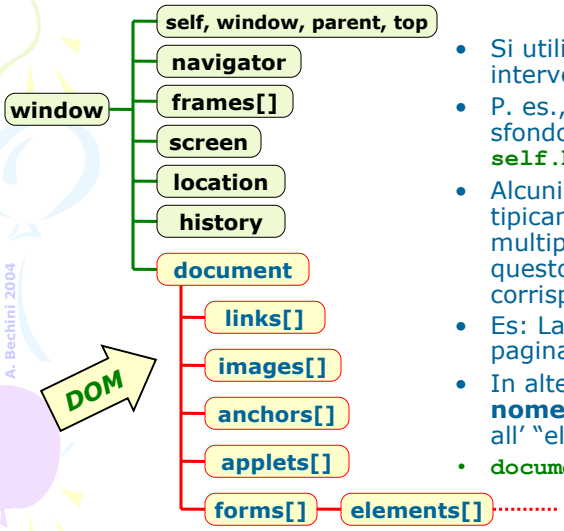
CHIAMATA



Gli oggetti JavaScript

- Al caricamento di una pagina web, il browser crea un insieme di oggetti che la definiscono e permettono di maneggiarla.
- Questi oggetti comprendono:
  - La finestra padre della pagina
  - L'URL della pagina corrente
  - La storia della sessione utente
  - Le proprietà del documento, p.es. il colore dello sfondo, il titolo, e i form HTML di input.
- Nel browser, questi oggetti sono organizzati in una gerarchia.

La gerarchia degli oggetti JavaScript

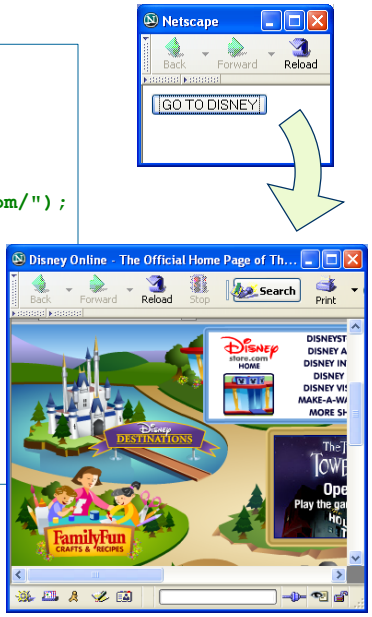


- Si utilizzano questi oggetti per intervenire sulla pagina web
- P. es., per intervenire sullo sfondo, ci si riferisce a `self.bgcolor`
- Alcuni elementi della pagina tipicamente compaiono in istanze multiple (p.es. link e form): in questo caso a un oggetto corrisponde in realtà un array
- Es: La seconda immagine della pagina sarà `document.images[1]`
- In alternativa, si può usare il **nome** esplicitamente assegnato all' "elemento":
- `document.myPrettyFace`

Aprire nuove finestre

- JavaScript permette di aprire nuove finestre del browser
- Tali finestre possono essere usate per visualizzare nuovi documenti (p.es. Pagine web o immagini)
- I documenti da visualizzare possono anche essere generati al volo da JavaScript
- L'apertura può essere fatta come specificato nel seguente esempio

```
...
<body>
 <script language="JavaScript">
  <!--
  function oW() {
   w = window.open("http://disney.go.com/");
  }
  //-->
 </script>
 <form>
 <input type="button"
    value="GO TO DISNEY"
    onClick="oW()">
 </form>
...
```



Proprietà di una finestra

- Si possono specificare le varie caratteristiche di una finestra:
- Dimensioni, presenza di toolbar, menubar, statusbar, possibilità di ridimensionare, ecc.

```
function oW() {
  w = window.open("doc.html",
    "WindowName",
    "width=400,height=300,status=yes,
     menubar=no,resizable=no"
    );
}
```

Operare sui form

- Una tipica operazione che si può svolgere sul lato client per migliorare le prestazioni di un'applicazione web è *l'elaborazione preventiva* dei dati presenti nei form HTML.
- Uno degli impieghi più tipici di JavaScript è la *validazione dei dati nei form*.
- Questa operazione consiste nel controllare che i vari campi di un form siano stati riempiti con dati del tipo corretto, prima di inviare la request al server.

Validazione dei form (I)

```
...
<body>
 <script language="JAVASCRIPT">
  <!--
  function checkIt() {
    if(document.forms[0].minnie.value == "")
       alert ("Your name is needed!")
    else
       alert ("Well done, "+document.mickey.elements[0].value);
    }
  //-->
 </script>
 <form name="mickey">
 Your name:
 <input name="minnie" type="text" >
 <input type="button" value="CORRECT?" onClick="checkIt()">
  </form>
...
```

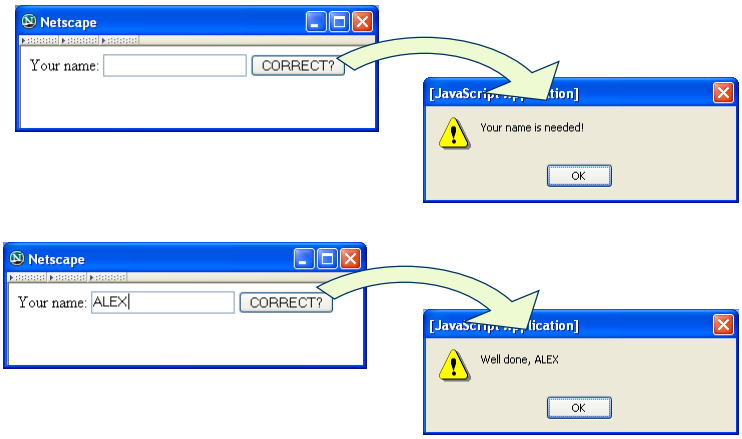
Funzione di validazione

Rif. con nome

Rif. oggetto DOM

Chiamata handler

Validazione dei form (II)



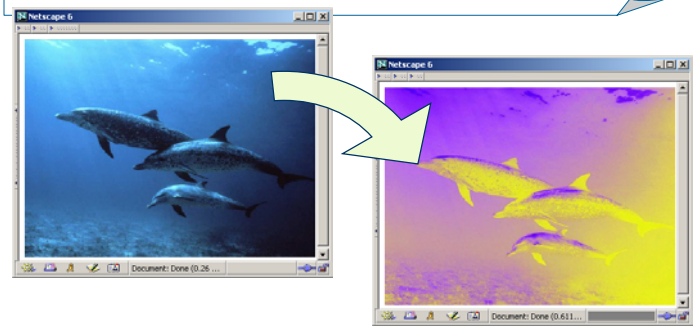
Validazione dei form (III)

- Nell'esempio precedente si è controllato semplicemente che una qualsiasi stringa fosse stata inserita.
- Per ingannare questo controllo, basterebbe inserire degli spazi in quel campo.
- In pratica, dobbiamo usare funzioni più complicate. Per esempio, per vedere se è stato inserito il carattere '@' in un campo per l'indirizzo e-mail:

```
document.forms[1].elements[3].
  value.indexOf('@', 0) == -1
```

JavaScript & immagini

```
<html>
 <head> </head>
 <body>
   <img src="d1.jpg" name ="delph"
     onMouseOver="document.delph.src='d2.jpg'">
 </body>
</html>
```



I "rollover"

- Si tratta di semplici animazioni che hanno luogo quando si posiziona il cursore del mouse sopra un elemento della pagina

```
...
  <img src="smallflower.jpg"
   onMouseOver="a=window.open('BIGflower.jpg')"
   onMouseOut="a.close()">
  </img>
...
```