

Introduzione a Java Script

Sommario

- Introduzione
- Caratteristiche generali
- Oggetti e JavaScript
- Esempio: Stampa di una linea di testo in una pagina web
- Input
- Riepilogo concetti relativi alla memoria
- Aritmetica
- Operatori di relazione
- Risorse web

Obiettivi

- Scrivere semplici programmi JavaScript
- Gestire I/O
- Capire i concetti basilari relativi alla gestione della memoria
- Uso degli operatori aritmetici e logici

Introduzione (1)

- JavaScript è un linguaggio di scripting
 - Favorisce la funzionalità e la presentazione della pagina
 - Agisce sul lato client per rendere dinamiche e interattive le pagine
 - Costituisce la base per script complessi sul lato server
 - Permette lo sviluppo e il controllo di programmi

Introduzione (2)

- JavaScript **NON** deve essere confuso con il linguaggio Java
- I due linguaggi differiscono principalmente per
 - Object-orientation
 - JavaScript non supporta il paradigma o-o
 - In Java gli oggetti sono statici; in JavaScript sono dinamici
 - Tipizzazione
 - Java è fortemente tipizzato, JavaScript lo è debolmente

Introduzione (3)

- I due linguaggi sono molto simili riguardo la sintassi
 - delle espressioni
 - degli assegnamenti
 - delle strutture di controllo

Caratteristiche generali (1)

- Gli script in JavaScript svolgono attività computazionali in modalità Event-driven
 - Le azioni sono svolte come effetto del verificarsi di eventi, ad esempio risposte ad azioni dell'utente
- L'uso comune di script in JavaScript è quello di controllare le azioni dell'utente sul lato client e, se legittime e corrette, comunicarle al server per l'appropriata esecuzione

Caratteristiche generali (2)

- Se un documento XHTML **NON** contiene script
 - allora il browser elabora il documento linea per linea e ne presenta il contenuto
- Se invece il doc contiene script
 - il browser chiama l'interprete JavaScript per eseguirlo
 - finita l'elaborazione il browser torna al documento XHTML

Caratteristiche generali (3)

- Gli script possono comparire sia nella head section che nella body section di un doc XHTML
 - Nella head section ci sono gli script che producono un effetto solo quando esplicitamente chiamati o per effetto di una azione utente, ad es. click su un bottone
 - Nella body section ci sono gli script che vengono interpretati una e una sola volta durante l'elaborazione del doc

Oggetti e JavaScript (1)

- JavaScript è un linguaggio object-based
 - Non ci sono classi, gli object hanno scopo sia di oggetti che di modello di oggetti
 - Non si ha ereditarietà nelle modalità tradizionali dei linguaggi o-o, ma è simulata attraverso altri meccanismi (prototype object)
 - Non si ha polimorfismo

Oggetti e JavaScript (2)

- Gli oggetti di JavaScript sono collezioni di proprietà
 - Proprietà dei dati: possono essere valori primitivi o riferimenti ad altri oggetti
 - Proprietà dei metodi
- Tutti gli oggetti sono acceduti attraverso variabili

Esempio: Stampa di un testo in una pagina web

- Inline scripting
 - È scritto nel <body> del documento
 - Il tag <script>
 - Indica che il testo è parte di uno script
 - L'attributo type
 - specifica il tipo di file e il linguaggio di scripting utilizzato
 - Il metodo writeln
 - scrive una riga nel documento
 - Il carattere di Escape (\)
 - indica che un carattere "speciale" è usato nella stringa
 - Il metodo alert
 - Attiva una dialog box

```

1 <?xml version = "1.0"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4
5 <!-- Fig. 7.1: welcome.html -->
6 <!-- Displaying a line of text -->
7
8 <html xmlns = "http://www.w3.org/1999/xhtml">
9   <head>
10    <title>A First Program In JavaScript</title>
11
12    <script type = "text/javascript">
13      <!--
14      document.writeln(
15        "<h1>Welcome to JavaScript Programming!</h1>" );
16      // -->
17    </script>
18
19  </head><body></body>
20 </html >

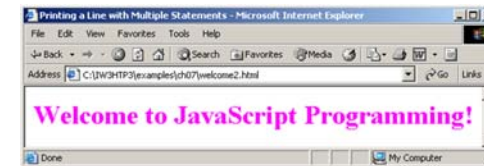
```



```

1 <?xml version = "1.0"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4
5 <!-- Fig. 7.2: welcome2.html -->
6 <!-- Printing a line with multiple statements -->
7
8 <html xmlns = "http://www.w3.org/1999/xhtml">
9   <head>
10    <title>Printing a Line with Multiple Statements</title>
11
12    <script type = "text/javascript">
13      <!--
14      document.write("<h1 style = 'color: magenta'>" );
15      document.write("Welcome to JavaScript " +
16        "Programming!</h1>" );
17      // -->
18    </script>
19
20  </head><body></body>
21 </html >

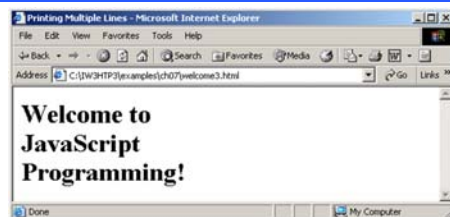
```



```

1 <?xml version = "1.0"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4
5 <!-- Fig. 7.3: welcome3.html -->
6 <!-- Printing Multiple Lines -->
7
8 <html xmlns = "http://www.w3.org/1999/xhtml">
9   <head><title>Printing Multiple Lines</title>
10
11   <script type = "text/javascript">
12     <!--
13     document.writeln("<h1>Welcome to<br />JavaScript" +
14       "<br />Programming!</h1>" );
15     // -->
16   </script>
17
18  </head><body></body>
19 </html >

```



```

1 <?xml version = "1.0"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4
5 <!-- Fig. 7.4: welcome4.html -->
6 <!-- Printing multiple lines in a dialog box -->
7
8 <html xmlns = "http://www.w3.org/1999/xhtml">
9   <head><title>Printing Multiple Lines In a Dialog Box</title>
10
11   <script type = "text/javascript">
12     <!--
13     window.alert("Welcome to\nJavaScript\nProgramming!");
14     // -->
15   </script>
16
17  </head>
18
19  <body>
20    <p>Click Refresh (or Reload) to run this script again.</p>
21  </body>
22 </html >

```



Esempio: Stampa di un testo in una pagina web

| Escape sequence | Description |
|-----------------|---|
| <code>\n</code> | Newline. Position the screen cursor at the beginning of the next line. |
| <code>\t</code> | Horizontal tab. Move the screen cursor to the next tab stop. |
| <code>\r</code> | Carriage return. Position the screen cursor to the beginning of the current line; do not advance to the next line. Any characters output after the carriage return overwrite the characters previously output on that line. |
| <code>\\</code> | Backslash. Used to represent a backslash character in a string. |
| <code>\"</code> | Double quote. Used to represent a double quote character in a string contained in double quotes. For example, <code>window.alert("\"in quotes\"");</code> displays "in quotes" in an alert dialog. |
| <code>'</code> | Single quote. Used to represent a single quote character in a string. For example, <code>window.alert('\'in quotes\'');</code> displays 'in quotes' in an alert dialog. |

Fig. 7.5 Some common escape sequences.

Variabili (1)

- Uno script può adattare il contenuto di una pagina sulla base di valori di input o di altre variabili
- Il concetto di variabile è lo stesso degli usuali linguaggi di programmazione
- Nei JavaScript **non** è necessario dichiarare le variabili, ma **è sempre regola di buona programmazione farlo**

Variabili (2)

- Gli identificatori sono qualunque stringa di
 - Alfanumerici
 - Underscore (`_`)
 - Dollaro (`$`)
- Il primo simbolo **non** può essere una cifra
- Non sono identificatori validi le keyword di JavaScript
- JavaScript è case sensitive

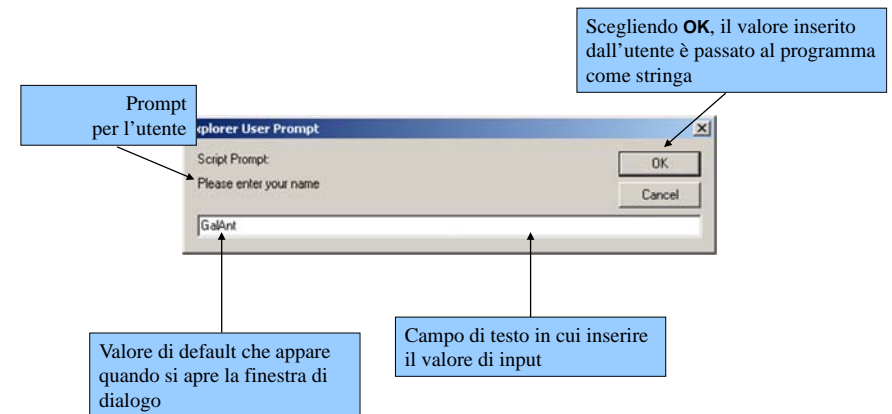
Variabili (3)

- Il metodo che permette di fornire valori di input è **prompt**
- Riceve due argomenti:
 - Il primo (obbligatorio) indica la stringa da stampare
 - Il secondo (opzionale) indica la stringa presentata per default

```
1 <?xml version = "1.0"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
3   "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
4
5 <!-- Fig. 7.6: welcome5.html -->
6 <!-- Using Prompt Boxes -->
7
8 <html xmlns = "http://www.w3.org/1999/xhtml">
9   <head>
10    <title>Using Prompt and Alert Boxes</title>
11
12    <script type = "text/javascript">
13      <!--
14      var name; // string entered by the user
15
16      // read the name from the prompt box as a string
17      name = window.prompt( "Please enter your name", "GalAnt" );
18
19      document.writeln( "<h1>Hello, " + name +
20        ", welcome to JavaScript programming!</h1>" );
21      // -->
22    </script>
```

```
23
24 </head>
25
26 <body>
27 <p>Click Refresh (or Reload) to run this script again.</p>
28 </body>
29 </html >
```

23

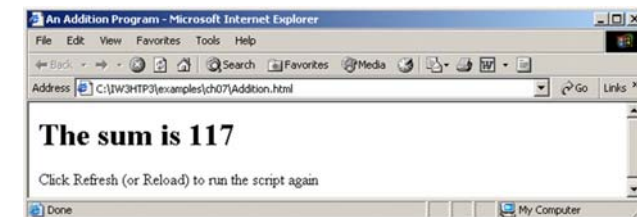
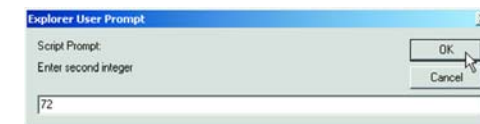
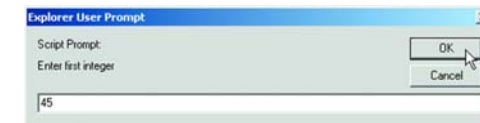


Somma di interi

- Si richiedono due numeri interi all'utente e si calcola la loro somma
- NaN (not a number)
- parseInt
 - Converte l'argomento da stringa di caratteri a intero

```
1 <?xml version = "1.0"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4
5 <!-- Fig. 7.8: Addition.html -->
6 <!-- Addition Program -->
7
8 <html xmlns = "http://www.w3.org/1999/xhtml">
9   <head>
10    <title>An Addition Program</title>
11
12    <script type = "text/javascript">
13      <!--
14        var firstNumber, // first string entered by user
15            secondNumber, // second string entered by user
16            number1, // first number to add
17            number2, // second number to add
18            sum; // sum of number1 and number2
19
20        // read in first number from user as a string
21        firstNumber =
22            window.prompt( "Enter first Integer", "0" );
23
```

```
24 // read in second number from user as a string
25 secondNumber =
26     window.prompt( "Enter second Integer", "0" );
27
28 // convert numbers from strings to integers
29 number1 = parseInt( firstNumber );
30 number2 = parseInt( secondNumber );
31
32 // add the numbers
33 sum = number1 + number2;
34
35 // display the results
36 document.writeln( "<h1>The sum is " + sum + "</h1>" );
37 // -->
38 </script>
39
40 </head>
41 <body>
42 <p>Click Refresh (or Reload) to run the script again</p>
43 </body>
44 </html >
```



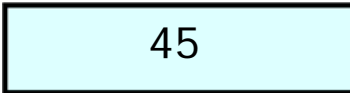
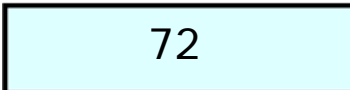
Riepilogo Concetti Memoria (1)

- I nomi di variabili corrispondono a locazioni di memoria
- Ogni variabile ha:
 - Un nome
 - Un tipo
 - Un valore
- La lettura di valori è un'operazione **non distruttiva** del valore
- La scrittura di valori è un'operazione **distruttiva** del valore



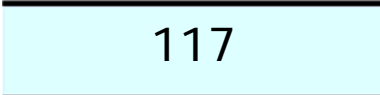
Riepilogo Concetti Memoria (2)

number1 

Riepilogo Concetti Memoria (3)

number1 
number2 

Riepilogo Concetti Memoria (4)

number1 
number2 
sum 

Aritmetica (1)

- Parecchi script svolgono operazioni aritmetiche

Aritmetica (2)

| JavaScript operation | Arithmetic operator | Algebraic expression | JavaScript expression |
|----------------------|---------------------|--------------------------------|-----------------------|
| Addition | + | $f+7$ | $f + 7$ |
| Subtraction | - | $p-c$ | $p - c$ |
| Multiplication | * | bm | $b * m$ |
| Division | / | x/y or $\frac{a}{y}$ or xy | x / y |
| Remainder | % | $r \text{ mod } s$ | $r \% s$ |

Fig. 7.12 Arithmetic operators.

| Operator(s) | Operation(s) | Order of evaluation (precedence) |
|-------------|---------------------------------------|--|
| *, / or % | Multiplication Division Modulus | Evaluated second. If there are several such operations, they are evaluated from left to right. |
| + or - | Addition Subtraction | Evaluated last. If there are several such operations, they are evaluated from left to right. |

Fig. 7.13 Precedence of arithmetic operators.

Aritmetica (3)

Step 1. $y = 2 * 5 * 5 + 3 * 5 + 7;$

$2 * 5$ is 10 (Leftmost multiplication)

Step 2. $y = 10 * 5 + 3 * 5 + 7;$

$10 * 5$ is 50 (Leftmost multiplication)

Step 3. $y = 50 + 3 * 5 + 7;$

$3 * 5$ is 15 (Multiplication before addition)

Step 4. $y = 50 + 15 + 7;$

$50 + 15$ is 65 (Leftmost addition)

Step 5. $y = 65 + 7;$

$65 + 7$ is 72 (Last addition)

Step 6. $y = 72;$ (Last operation—placed into Y)

Fig. 7.14 Order in which a second-degree polynomial is evaluated.

Operatori di relazione (1)

- Permettono di prendere decisioni sulla base dei valori di verità di una condizione
- Operatori di
 - Eguaglianza
 - Relazione

Operatori di relazione (2)

| Standard algebraic equality operator or relational operator | JavaScript equality or relational operator | Sample JavaScript condition | Meaning of JavaScript condition |
|---|--|-----------------------------|---------------------------------|
| <i>Equality operators</i> | | | |
| = | == | X == Y | X is equal to Y |
| ? | != | X != Y | X is not equal to Y |
| <i>Relational operators</i> | | | |
| > | > | X > Y | X is greater than Y |
| < | < | X < Y | X is less than Y |
| ≥ | >= | X >= Y | X is greater than or equal to Y |
| ≤ | <= | X <= Y | X is less than or equal to Y |

Fig. 7.15 Equality and relational operators.

```

1 <?xml version = "1.0"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
3 "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
4
5 <!-- Fig. 7.16: welcome6.html -->
6 <!-- Using Relational Operators -->
7
8 <html xmlns = "http://www.w3.org/1999/xhtml">
9 <head>
10 <title>Using Relational Operators</title>
11
12 <script type = "text/javascript">
13 <!--
14 var name, // string entered by the user
15     now = new Date(), // current date and time
16     hour = now.getHours(); // current hour (0-23)
17
18 // read the name from the prompt box as a string
19 name = window.prompt( "Please enter your name", "GalAnt" );
20
21 // determine whether it is morning
22 if ( hour < 12 )
23     document.write( "<h1>Good Morning, " );
24

```

```

25 // determine whether the time is PM
26 if ( hour >= 12 )
27 {
28     // convert to a 12 hour clock
29     hour = hour - 12;
30
31     // determine whether it is before 6 PM
32     if ( hour < 6 )
33         document.write( "<h1>Good Afternoon, " );
34
35     // determine whether it is after 6 PM
36     if ( hour >= 6 )
37         document.write( "<h1>Good Evening, " );
38 }
39
40 document.writeln( name +
41     ", welcome to JavaScript programming!</h1>" );
42 // -->
43 </script>
44
45 </head>
46

```

```

47 <body>
48 <p>Click Refresh (or Reload) to run this script again.</p>
49 </body>
50 </html>

```



Precedenze degli Operatori

| Operators | Associativity | Type |
|-----------|---------------|----------------|
| * / % | left to right | multiplicative |
| + - | left to right | additive |
| < <= > >= | left to right | relational |
| == != | left to right | equality |
| = | right to left | assignment |

Fig. 7.17 Precedence and associativity of the operators discussed so far.