Client-side programming with JavaScript

Laura Farinetti
Dipartimento di Automatica e Informatica
Politecnico di Torino
laura.farinetti@polito.it

Summary

- Introduction
- Language syntax
- Functions
- Objects
- Events
- The HTML Document Object Model (DOM)
- Examples

What and why JavaScript?

- JavaScript is a lightweight, interpreted programming language with object-oriented capabilities that allows you to build interactivity into otherwise static HTML pages
  - JavaScript made its first appearance in Netscape 2.0 in 1995 with the name "LiveScript"
  - Later standardized by ECMA (www.ecma.ch): ECMAScript
- JavaScript is one of the 3 languages all web developers must learn
  - HTML to define the content of web pages
  - CSS to specify the layout of web pages
  - JavaScript to program the behavior of web pages

What can JavaScript do?

- JavaScript can change HTML content
- JavaScript can change HTML attributes
- JavaScript can change HTML styles (CSS)
- JavaScript can validate data

- http://www.w3schools.com/js/js_intro.asp

JavaScripts

- A JavaScript consists of JavaScript statements placed within the <script>... </script> HTML tags in a web page
- The <script> tag containing JavaScript code can be placed anywhere in a web page
  - In the head or the body section

prova.html

```
<html>
<body>
<script language="javascript" type="text/javascript">
<!--
   document.write("Hello World!")
//-->
</script>
</body>
</html>
```

Where to embed JavaScript code?

- In the head section
  - Scripts to be executed when they are called, or when an event is triggered, go in the head section
  - When you place a script in the head section, you will ensure that the script is loaded before anyone uses it
- In the body section
  - Scripts to be executed when the page loads go in the body section
  - When you place a script in the body section it generates the content of the page

JavaScript functions and events

- Functions are usually defined in the head section
- Functions can be executed when an event occurs, e.g. when the user clicks a button

```
<html>
<head>
<script type="text/javascript">
<!--
function sayHello() {
   alert("Hello World")
}
//-->
</script>
</head>
<body>
<input type="button" onclick="sayHello()" value="Say Hello" />
</body>
</html>
```

Example

- JavaScript can change HTML content

```
<html>
<head>
<script>
function myFunction() {
    document.getElementById("demo").innerHTML =
       "... e vivo a Torino."; }
</script>
</head>

<body>
<h1>JavaScript</h1>
<p id="demo">Mi chiamo Andrea Rossi ...</p>
<button type="button" onclick="myFunction()">Prova</button>
</body>
</html>
```

External JavaScripts

- Scripts can be placed in external files too
  - Useful when the same code is used in many different web pages
  - Can be called in <head> or <body>
- JavaScript files: extension .js

```
<!DOCTYPE html>
<html>
<body>
   <script src="myScript.js"></script>
</body>
</html>
```

JavaScript display possibilities

- JavaScript can “display” data in different ways
  - Writing into an alert box: window.alert()
  - Writing into the HTML output: document.write()
  - Writing into an HTML element: innerHTML
  - Writing into the browser console: console.log()

  - http://www.w3schools.com/js/js_output.asp

JavaScript display possibilities

- Using document.write() after an HTML document is fully loaded deletes all existing HTML
  - document.write() is useful only for testing purposes

```
<!DOCTYPE html>
<html>
<body>

<h1>Esempio</h1>
<p>Quanto fa 5 + 6 ?</p>

<button type="button"
   onclick="document.write(5 + 6)">Prova</button>

</body>
</html>
```

JavaScript display possibilities

- To access an HTML element, JavaScript can use the document.getElementById(id) method
- The id attribute defines the HTML element
- The innerHTML property defines the HTML content

```
<!DOCTYPE html>
<html>
<body>

<h1>Esempio</h1>
<p>Quanto fa 5 + 6 ?</p>
<p id="demo"></p>
<button type="button"
   onclick="document.getElementById('demo').innerHTML
            = 5 + 6;">Prova</button>
</body>
</html>
```

JavaScript display possibilities

- Example, with the predefined function Date()

```
<!DOCTYPE html>
<html>
<body>
<h1>Esempio</h1>
<button type="button" onclick=
   "document.getElementById('demo').innerHTML = Date()">
   Premi qui per sapere data e ora</button>
<p id="demo"></p>
</body>
</html>
```

What can JavaScript do?

- Generate dialog boxes
- Redirect a page
- Open new browser windows (pop-ups)
- Intercept mouse events
  - Clicks on links, buttons, ...
  - Mouse-overs
- Read user input in forms
- Modify HTML pages
  - Add/remove content
  - Change images
  - Modify form controls

What to know…

- JS variables and expressions
- JS language constructs (if, while, ...)
- JS objects
  - The most important built-in objects
- Interaction with the user
  - mouse, keyboard
- Interaction with the browser
  - windows, pages
- Interaction with the page: the Document Object Model

JavaScript syntax

- Similar to C language (and Ruby too)
  - Choice, loops and other constructs are the same
  - Blocks are delimited by { }
  - Most operators are identical
  - Variables are different, however, …
- JavaScript is a case-sensitive language
- Semi-colons (at the end of a line) can be omitted
- Comments:

```
<script>
// This is a comment. It is similar to comments in C++
/*
 * This is a multiline comment in JavaScript
 * It is very similar to comments in C Programming
 */
</script>
```

JavaScript data types and variables

- Three primitive data types
  - Numbers (123, 120.50, …) – no distinction between integers are real numbers
  - Strings of text ("This text string“, …)
  - Booleans (true or false)
- A composite data type known as “object”
- In JavaScript all variables must be declared before their use
- Data types are converted as needed

```
<script>
  var money;
  var x;
  var y = 10;
  var z = "Hello!";
  var one, two, three;
  var d = new Date();  //object
</script>
```

Main Javascript operators

- Numeric operators
  - + - * / %
- Increment operators
  - ++ --
- Assignment operators
  - = += -= *= /= %=
- String operator
  - + (concatenation)
- Comparison operators
  - == (same value) === (same value and same type)
  - != > < >= <=
- Boolean and Logic operators
  - && (logical “and”) || (logical “or”) ! (logical “not”)

Choice statements

```
if (condition)
{
   ...code...
}
```

```
if (condition)
{
   ...code if true...
}
else
{
   ...code if false...
}
```

```
if (condition1)
{
   ...code if 1 true...
}
else if (condition2)
{
   ...code if 2 true...
}
else
{
   ...if both false...
}
```

Choice statements

```
switch(n)
{
  case 1:
    code block 1
    break

  case 2:
    code block 2
    break

  default:
    code to be executed if n is
    different from case 1 and 2
}
```

Loop statements

```
for ( var=startvalue; var<=endvalue; var=var+increment )
{
    code to be executed
}
```

```
while ( condition_is_true )
{
    code to be executed
}
```

```
do {
    code to be executed
} while ( condition_is_true )
```

Loop statements

```
while ( ... ) // or for
{
     code
     break
     code
}
```

```
while ( ... ) // or for
{
     code
     continue
     code
}
```

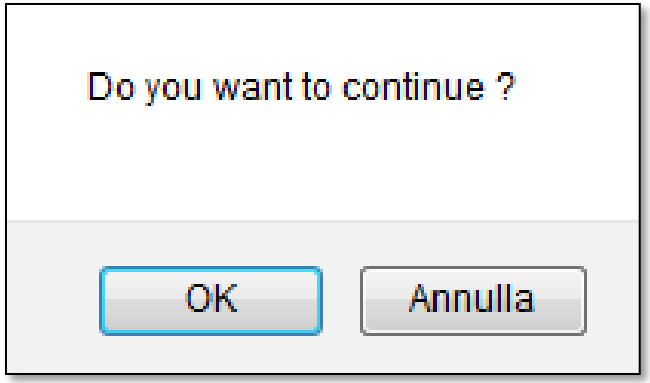
Basic interaction methods

- Alert dialog box
  - OK to confirm
- Mostly used to give a warning message to the users

```
<head>
<script type="text/javascript">
<!--
   alert("Warning Message");
//-->
</script>
</head>
```

Basic interaction methods

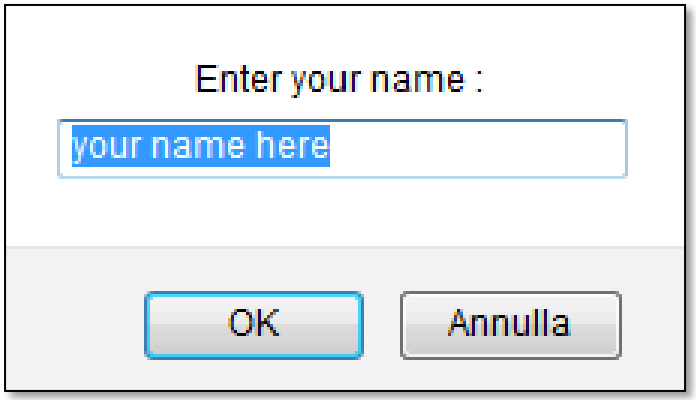
- Confirmation dialog box
  - OK, cancel
  - True if user clicks on OK
- Mostly used to take user's consent on any option




```
<script type="text/javascript">
 var retVal = confirm("Do you want to continue ?");
  if( retVal == true ){
    alert("User wants to continue!");
  }else{
    alert("User does not want to continue!");
  }
</script>
```

Basic interaction methods

- Prompt dialog box
  - Returns a string with the text written by the user
  - Returns null if user clicks on Cancel
- Used to get user input




```
<script type="text/javascript">
<!--
   var retVal = prompt("Enter your name : ",
     "your name here");
   alert("Hello " +  retVal );
//-->
</script>
```

Functions

- Function definition

```
function functionname(var1,var2,...,varX)
{
        some code
}
```

  - No parameters:

```
function functionname()
{
        some code
}
```

- A function may return a value to its caller by executing the return statement
  - return value ;
  - The value may be of any type (boolean, numeric, string, ...)

Example

```
<html>
<head>
<script type="text/javascript">
  function product(a,b)
  {
    return a*b;
  }
</script>
</head>

<body>
<script type="text/javascript">
  document.write(product(4,3)) ;
</script>
</body>
</html>
```

Objects in JavaScript

- An object is a complex data type characterized by
- A current value
  - Sometimes the internal value is “hidden”
- A set of properties
  - Various values that be read, associated in some way to the object value
  - Some values that may be written, that modify in some way the object value
- A set of methods
  - Operations (with parameters) that can be asked to the object

Example

```
<html>
<head>
<title>User-defined objects</title>
<script type="text/javascript">
  var book = new Object();   // Create the object
  book.subject = "Perl"; // Assign properties to the object
  book.author  = "Mohtashim";
</script>
</head>
<body>
<script type="text/javascript">
  document.write("Book name is: " + book.subject + "<br>");
  document.write("Book author is: " + book.author + "<br>");
</script>
</body>
</html>
```

JavaScript native objects

- JavaScript has several built-in objects
  - Accessible anywhere in a program
  - Work the same way in any browser running in any operating system
- List of native objects
  - JavaScript Number Object
  - JavaScript Boolean Object
  - JavaScript String Object
  - JavaScript Array Object
  - JavaScript Date Object
  - JavaScript Math Object
  - JavaScript RegExp Object

The String object

- Strings are used to store and manipulate sequences of characters
- The only property is
  - .length (the number of characters in the string)
- Many general methods
  - .charAt(), .concat(), .indexOf(), .localeCompare(), .match(), .replace(), .search(), .slice(), .split(), .substr(), .substring(), .toLowerCase(), .toUpperCase(), .toString(), .valueOf(), …
- Many methods specific for writing HTML

String methods for HTML formatting

- Methods that returns a copy of the string wrapped inside the appropriate HTML tag
  - Warning: not standard methods, may not work as expected in all browsers
- List of main methods
  - .big(), .small(), .italic(), .bold(), .fixed(), .sub(), .sup()
  - .fontcolor(c), .fontsize(s)
  - .anchor("name"), .link("url")

```
<script>
    var str = "Hello World!";
    document.write(str);
    document.write("<br />");
    str = str.fontcolor("red");
    document.write(str + "<br/>");
    str = str.fontsize(7);
    document.write(str);
</script>
```

http://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_str_style

Example

```
<!DOCTYPE html>
<html>
<body>
<p>Click the button to create an HTML link around a string.</p>

<button onclick="myFunction()">Try it</button>

<script>
function myFunction() {
    var txt = document.getElementById("demo").innerHTML;
    txt2 = txt.link("chap10.html");
       document.getElementById("demo").innerHTML = txt2;
}
</script>

<p id="demo">Chapter 10</p>

</body>
</html>
```

References

- JavaScript and HTML DOM Reference
  - http://www.w3schools.com/jsref/default.asp
- JavaScript tutorials
  - http://www.w3schools.com/js/
  - http://www.html.it/guide/guida-javascript-di-base/
  - http://www.codecademy.com/tracks/javascript
  - http://www.tutorialspoint.com/javascript/

Licenza d'uso

- Queste diapositive sono distribuite con licenza Creative Commons "Attribuzione - Non commerciale - Condividi allo stesso modo 2.5 Italia (CC BY-NC-SA 2.5)"
- Sei libero:
  - di riprodurre, distribuire, comunicare al pubblico, esporre in pubblico, rappresentare, eseguire e recitare quest'opera
  - di modificare quest'opera
- Alle seguenti condizioni:
  - **Attribuzione** — Devi attribuire la paternità dell'opera agli autori originali e in modo tale da non suggerire che essi avallino te o il modo in cui tu usi l'opera.
  - **Non commerciale** — Non puoi usare quest'opera per fini commerciali.
  - **Condividi allo stesso modo** — Se alteri o trasformi quest'opera, o se la usi per crearne un'altra, puoi distribuire l'opera risultante solo con una licenza identica o equivalente a questa.
- http://creativecommons.org/licenses/by-nc-sa/2.5/it/