

Flow Control in JavaScript

BY JAROSLAV MOHAPL

Abstract

Examples of flow control in JavaScript and some basic language characteristics.

Key Phrases JavaScript basics, flow control in Java Script.

Key Words JavaScript, flow control.

Introduction

A loop is the most important and most frequently used component of a programming language, because it allows automate tasks one might be forced to do by hand. That is why this column describes briefly the structure and use of the basic JavaScript controls such as the while loop, the for loop, the switch, and the if-then-else structure.

Generated Code

Suppose we have a set of images named img1.jpg, . . . , img20.jpg and want them all displayed on one web page. Rather than mechanically writing one line of HTML code after another to place the pictures on the page, we can use the for loop.

```
var N=20;
for(var i=1; i<=N; i++){
    document.write("<img scr='img"+i+".jpg' \
                    alt='Image "+i+"' /><br/>");
}
```

The code is self-explanatory, as long as you remember that in JavaScript, string+number produces a string. Curiously enough, upon opening of the HTML document, the lines generated by the for loop do properly initialize the document's array of images, but they are not written in the body of the HTML document. The same task, though with a slightly more writing, can be achieved using the while loop.

```
var N=20;
var i=1;
while( i<=N){
    document.write("<img scr='img"+i+".jpg' \
                    alt='Image "+i+"' /><br/>");
    i=i+1;
}
```

Iterating a JavaScript hash

Another use of the for loop is iteration through a hash, as illustrated by the following snippet:

```
month={JAN:"January", FEB:"February", MAR:"March"};
for (i in month){
    document.write(i+" "+month[i]+"<br />");
}
```

Switching Between Pages

Most web sites have a file called `index.html`, which mediates access to other documents on the site. Rather than constructing a menu directly in the index file, it may be more advantageous to use the index file merely for redirection to other pages. This way, if we want by default show to the user the new front page, we just change the default setting in the switch, instead of re-naming our files. The code snippet implementing the switch relies on the query string available through the `window.location.search.substring`.

```
var query = window.location.search.substring(1);
switch(query){
  case "pictures":
    window.location="http://www.foo.com/mypics.html";
    break;
  case "news":
    window.location="http://www.foo.com/news.html";
    break;
  case "blog":
    window.location="http://www.foo.com/rambling.html";
    break;
  default:
    window.location="http://www.foo.com/sitemap.html";
}
```

For example, to call the page `mypics.html` mentioned in the switch we must type in the browser's URL box: `http://www.foo.com/index.html?pictures`. The case labels do not have to be strings. Numbers are admissible as well. In that case, the argument of the switch must be a number too, of course. The above snippet changes for numeric case-labels as follows. It is to keep

in mind that the query string is what it says it is, a string, and must be converted to a number before the query argument enters the switch. That is achieved by subtracting the zero from the query string value.

```
var query = window.location.search.substring(1)-0;
switch(query){
  case 1:
    window.location="http://www.foo.com/mypics.html";
    break;
  case 2:
    window.location="http://www.foo.com/news.html";
    break;
  case 3:
    window.location="http://www.foo.com/rambling.html";
    break;
  default:
    window.location="http://www.foo.com/sitemap.html";
}
```

Inspecting Data Types

The following code illustrates the use of the if-then-else statements for determining of a user-entered data type. The code prompts the user to enter a string. If the string converts to a floating number then it is recognized as a number. If its string value is true or false, it qualifies as a Boolean value. Otherwise it is considered as a string.

```
var x=prompt("Enter a string, a number or a boolean value","");
if (!isNaN(parseFloat(x)))
{
    document.write("You entered the number "+x);
}
else if (x=="true"||x=="false")
{
    document.write("You entered the boolean value "+x);
}
else
{
    document.write("You entered the string "+x);
}
```

References

David Barron: Scripting Languages. *Wiley* 2000.

The ECMAScript scripting language. *World Wide Web Consortium* 1999.