JAVASCRIPT TOPIC 9

InfoTech 37
Jomar B. Colao
College Instructor

Intro

A JavaScript function is a block of code designed to perform a particular task.

A JavaScript function is executed when "something" invokes it (calls it).

Example

```
function myFunction(p1, p2) {
    return p1 * p2;          // the function returns the product of p1 and p2
}
```

Demo 1

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

  <p>This example calls a function which performs a calculation, and returns the result:</p>

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

  <script>
    function myFunction(a, b) {
      return a * b;
    }
    document.getElementById("demo").innerHTML = myFunction(4, 3);
  </script>

  </body>
</html>
```

JavaScript Function Syntax

A JavaScript function is defined with the function keyword, followed by a name, followed by parentheses ().

Function names can contain letters, digits, underscores, and dollar signs (same rules as variables).

The parentheses may include parameter names separated by commas: (parameter1, parameter2, ...)

The code to be executed, by the function, is placed inside curly brackets: {}

```
functionName(parameter1, parameter2, parameter3) {
    code to be executed
}
```

JavaScript Function Syntax

Function parameters are the names listed in the function definition.

Function arguments are the real values received by the function when it is invoked.

Inside the function, the arguments are used as local variables.

Note A Function is much the same as a Procedure or a Subroutine, in other programming languages.

Function Invocation

- The code inside the function will execute when "something" invokes (calls) the function:
- When an event occurs (when a user clicks a button)
- When it is invoked (called) from JavaScript code
- Automatically (self invoked)
- You will learn a lot more about function invocation later in this tutorial.

Function Return

- When JavaScript reaches a return statement, the function will stop executing.
- If the function was invoked from a statement, JavaScript will "return" to execute the code after the invoking statement.
- Functions often compute a return value. The return value is "returned" back to the "caller":

Function Return

Example

Calculate the product of two numbers, and return the result:

```
var x = myFunction(4, 3);    // Function is called, return value will
end up in x

function myFunction(a, b) {
      return a * b;          // Function returns the product of a and b
}
```

The result in x will be:

12

Demo 2

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

  <p>This example calls a function which performs a calculation, and returns the result:</p>

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

  <script>
    function myFunction(a, b) {
      return a * b;
    }
    document.getElementById("demo").innerHTML = myFunction(4, 3);
  </script>

  </body>
</html>
```

Why Functions?

- You can reuse code: Define the code once, and use it many times.
- You can use the same code many times with different arguments, to produce different results.

Example

Convert Fahrenheit to Celsius:

```
function toCelsius(fahrenheit) {
    return (5/9) * (fahrenheit-32);
}
```

Demo 3

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

  <p>This example calls a function to convert from Fahrenheit to Celcius:</p>
  <p id="demo"></p>

  <script>
    function toCelcius(f) {
      return (5/9) * (f-32);
    }
    document.getElementById("demo").innerHTML = toCelcius(32);
  </script>

  </body>
</html>
```

JavaScript Functions are Objects

- In JavaScript, functions are objects.
- JavaScript functions have properties and methods.
- You can add your own properties and methods to functions.

JavaScript Functions are Variables Too

In JavaScript, functions can be used as variables:

Example

Instead of:

```
temp = toCelsius(32);
text = "The temperature is " + temp + " Centigrade";
```

You can use:

```
text = "The temperature is " + toCelsius(32) + " Centigrade";
```

Demo 4

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

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

  <script>
    document.getElementById("demo").innerHTML =
    "The temperature is " + toCelsius(32) + " Centigrade";

    function toCelsius(fahrenheit) {
      return (5/9) * (fahrenheit-32);
    }
  </script>

  </body>
</html>
```