JavaScript Functions

Functions Overview

- JavaScript includes the ability to create functions
- Functions group logical blocks of code and promote reusability of code
- Functions are declared with the keyword function and exist between opening and closing braces:

```
function myFunction() {

}
```

Parameters

- Functions accept zero to many parameters.
- The parameters create local variables within the function scope
- Those local variables can be manipulated and used within the function for calculations and other processing.

Return

- Functions can also return a value.
- The return keyword is used to do so.
- When 'return' is encountered, all processing within the function stops immediately and control is given back to the program at the place where the function was called.

Function Example

Example Discussion

- The example used a basic function first, and just logged the parameter passed in.
- The second example accepted two arguments (parameters). The second example used isNaN to test whether the parameters were numbers. If they were, then the numbers were added.
- Note the use also of || for a logical OR within the if statement.

More on Returns

- You can return anything you want from a function and then use it elsewhere.
- But you should know what you're returning or check it when you get it back.
- The previous examples might easily have expected to use a number from addTwo but the "fail" condition returns a string.

Scoping Revisited

- Variables are accessible within their block and any inner blocks.
- Variables from inner blocks are not typically available outside of their block.
- Demo

Built-In Functions & Methods

- JavaScript, through the browser interface, exposes several built-in functions and methods.
- We'll encounter some of these in upcoming lectures.
- The window and document objects provide methods that we'll work with.
- For example, setTimeout() is a function that calls a function after a certain time. It's typically used for evil.

Function Practice

- Create a basic HTML5 page.
- Within the body, add the following JavaScript:

```
<script type="text/javascript">
function myFunction(myName) {
        console.log(myName);
        return false;
}
myFunction("Your Name Here");
</script>
```

Closures

- There is a way to get access to inner variables.
- The (horribly named) term is "Closures".
- Basically, set a variable equal to the function and within the function, have an inner function.
- Closures are covered in WDMD211.
- Use them to imitate private methods.

Anonymous Function Expressions

- The functions we've seen so far have been named:
- function myFunction() { // code here }
- Functions can also be anonymous when used as expressions.
- Inline and Immediately Invoked Function Expressions (IIFE) – See Demo
- IIFEs are used more heavily with jQuery and will be discussed further in WDMD211.

Summary

- Functions are used for repeated code and promote reusability.
- JavaScript functions are similar to functions in other languages.
- Send parameters in (optional)
- Do something
- Send a return out (optional)