1. Javascript

Overview

- JavaScript – The language
  - Introduction
  - The Document Object Model
  - Basic syntax
    - Variables
    - Types
    - Statements

JavaScript

- JavaScript was originally Developed as a scripting language for Netscape web browser in 1995.
- Despite it’s name, it is not related to Java except for some similar syntax and function names.
- JavaScript has the following features:
  - Interpreted – not compiled
  - Object-oriented – but has no classes
  - Functional
  - Supports most basic data types such as booleans, ints, floats, and Strings.

Where JavaScript is used

- Browser
  - JavaScript is most commonly found in the browser
  - Most browsers now support JavaScript including mobile browsers
- Server
  - The Node.js server environment allows JavaScript to be used to write a server code
- Applications
  - Some applications support customisation through JavaScript scripts

Document Object Model

- The way JavaScript interacts with a web page is through the Document Object Model, abbreviated DOM.
- The DOM is a hierarchy of objects that represent a web page's:
  - Tags and attributes (including styles)
  - Structure
  - Events
- Through the DOM JavaScript can performed the following:
  - Access information about a web page:
    - Find a specific tag by tag name, class name, or id.
    - Access the attributes and contents of a tag.
  - Change the contents of a web page:
    - Modify tag attributes
    - Add event handlers
    - Create new tags and insert into DOM

DOM Hierarchy

- The DOM consists of the following hierarchy of JavaScript variables:
  - Window:
    - Document:
      - Head
      - Body
- JavaScript runs within a page and is automatically in the context of window, so window only needs to be used if directly accessing aspects of window.

Accessing a DOM element

- The document variable includes functions to access specific HTML tags.
- HTML tags can be identified through tag name, id, or name attribute.
- When accessing by tag or name a list of matching elements is returned.
- When accessing by id only one element is returned.

Accessing a DOM element

- **document.getElementById()** is used to access an element via its id.
- Your page that you built has a id called “**nav**”.
- We can access it with:
  - **document.getElementById(“nav”);**

- **document.getElementsByTagName()** returns every element that matches the tag name.
- For example, passing in “div” would return every div.
- **document.getElementsByTagName(“div”)** returns an array where each entry in the array is an element that matches the tag name.
- If you would like the first div you would write:
  - **document.getElementsByTagName(“div”)[0];**

JavaScript Variables

In JavaScript, variables have no declared type.
Instead of a keyword type like int, **var** is used to declare a variable, for example:

Java: int x = 0;
JavaScript: var x = 0;

In fact, Javascript, variables don’t have to be declared.

x = 0;
x = x + 1;

However not declaring a variable is **NOT** recommended.

Types of Variables and Comments

JavaScript has:

- Booleans
- Strings (either in double quotes or single quotes
- Integers
- Floating Point numbers

Comments are the same as most programming languages.
Single line comments are made with two slashes

- //This is a single line comment

Multi-line comments are everything between /* */

- /*This is a multi-line comment */

If statements

An **if statement** is a programming conditional statement that, if proved true, performs a function or displays information. Below is a general example of an if statement, specific to the JavaScript language.

```
var x = 1;

if (x < 10) {
    console.log("Hello John”);
}
```

In Node.js it is regularly used to check for database content and also config files when setting up a server!

For loops

A **loop** is a sequence of instructions that is continually repeated until a certain condition is reached. Typically, a certain process is done, such as getting an item of data and changing it, and then some condition is checked such as whether a counter has reached a prescribed number. Below is some example code of how to use a for loop in JavaScript:

```
for (var i = 0; i < 10; i++) {
    // Loop 10 times
}
```

Node.js is very fast, and is used a lot for multiplayer games on the web. Game loops are used a lot, it is also used to go through a set sequence of data incorporating the if statements.

Functions

Functions are "self contained" modules of code that accomplish a specific task. **Functions** usually "take in" data, process it, and "return" a result. Once a function is written, it can be used over and over and over again. **Functions** can be "called" from the inside of other functions.
It helps maintain code from not writing the same lines of code over and over again.

```
function showFullname(firstname, lastname) {
    var fullname = firstname + " " + lastname;
    console.log(fullname);
}
```

To call the function:

```
showFullname();
```

Strings

Strings are very easy to manipulate in javascript. for example changing a character in your string can be done by one line of code.

str = ”Hello@World!"
str.replace(”@”, ” “);

This code will find your @ mark in the string and change it to a space for you!

Now also if you use:

str = str.slice(0,-1);

This will cut the last character off your string and return:

“Hello World” as the final result.

Strings

Strings can get a bit difficult when you start learning how to use Regular expressions.

We will only use one to show what can be done.

```
var str = “mypassword”;
str.replace(/./g, “*”);
```

The section /./ will look for every character and “g” is to find in global space which will be the whole string in this case.

So every character will be changed to * with this line of code.

Use some interesting regular expressions and string manipulation becomes a lot more fun!

Difference between Browser and Server

JavaScript is used in both the Browser and the Node.js Server.

Most of JavaScript language can be used in both sides, however the server does not have the DOM elements.
So this means document.getElementById() and other document or window calls cannot be made on the server side coding.

There are more implementation methods for Node.js and in further lectures we will be learning them!