Javascript

WEEK 9

Scoping and anonymous functions

JavaScript has function level scoping. All variables and functions defined within the anonymous function aren’t available to the code outside of it, effectively using closure to seal itself from the outside world.

```
(function(){

  console.log('Hello World!');

})
```

Self-executing anonymous function

```
(function(){
  console.log('Hello World!');
})();
```

Important Note

```
(function(){
  var hi = 'Hello world';
  function hiWorld(){
      return hi;
  }
  window.hiWorld = hiWorld; //Assign hiWorld to the global variable
})();
console.log(hiWorld()); //…and now this works.
//It's important to note that this won't work:
console.log(hi);
```

Now for something completely different!

Client-side Javascript

The Window object is the main entry point to all client-side JavaScript features and APIs. It represents a web browser window or frame, and you can refer to it with the identifier window. The Window object defines properties like location, which refers to a Location object that specifies the URL currently displayed in the window and allows a script to load a new URL into the window:

```
// Set the location property to navigate to a new web page
window.location = "http://www.javascript.com/";
```

window.

Notice that you do not need to explicitly use the window property. In client-side JavaScript, the Window object is also the global object. This means that the Window object is at the top of the scope chain and that its properties and methods are effectively global variables and global functions. The Window object has a property named window that always refers to itself. You can use this property if you need to refer to the window object itself, but it is not usually necessary to use window if you just want to refer to access properties of the global window object.

A few window functions

location

history
-back()
-forward()
-go(arg)

alert()
prompt()
confirm()