**The Window Object**

**The Browser Object Model**

* The Browser Object Model (or BOM for short) is a collection of properties and methods that contain information about the browser and computer screen
* Every browser window, tab, popup, frame, and iframe has a window object.
* ES6 made parseInt() and isNaN() methods of the Number object

**Dialogs**

* The most use **alert( ), confirm( ) and prompt( ),** all major browsers support them as methods of the **window** object.

**Browser Information**

* The **Navigator** object contains information about the browser being used
* Its **userAgent** property will return information about the browser and operating system
* The **window.location** property is an object that contains information about the URL of the current page
* The **pathname** property returns a string of the path that follows the domain
* The toString() method returns a string containing the whole URL:

**Controlling Windows**

* A new window can be opened using the **window.open( )**method

**Screen Information**

* The **window.screen** object contains information about the screen the browser is displayed on
* The **availHeight** and **availWidth** can be used to find the height and width of the screen, excluding any operating system menus.
* The **colorDepth** property can be used to find the color bit depth of the user’s monitor, although there are few use cases for doing this other than collecting user statistics.

**The Document Object**

* The write() method simply writes a string of text to the page - document.write('Hello, world!');
* document.write('<h1>Hello, world!</h1>');

**Cookies**

* Cookies are small files that are saved locally on a user’s computer. They were invented by Netscape as a way of getting round HTTP being a stateless protocol.
* Cookies can be used to storing information that can then be retrieved between requests.
* To create a cookie, you assign it to JavaScript’s “cookie jar”, using the document.cookie property, like so: document.cookie = 'name=Superman';<< "name=Superman"

**Timing Functions**

**setTimeout( )**

* The **window** object provides some useful methods for scheduling the execution of a function, and for repeatedly executing functions at regular intervals.
* The **window.setTimeout( )** method accepts a callback to a function as its first parameter and a number of milliseconds as its second parameter

**<template>: The Content Template element**

* The <template> [HTML](https://developer.mozilla.org/en-US/docs/Web/HTML) element is a mechanism for holding [HTML](https://developer.mozilla.org/en-US/docs/Glossary/HTML) that is not to be rendered immediately when a page is loaded but may be instantiated subsequently during runtime using JavaScript.