Advanced Javascript

**Object-oriented JS**

Has a state, for example in human form two legs, two eyes a head, two arms…  
and behaviour, for example in human form, can talk, walk, eat etc…

Terminology  
  
**Namespace** – A container that allows developers to bundle all functionality under a unique specific name

For example instead of having global script….

<script type=”text/javascript”>

var temp = 10;

console.log(temp);

</script>

Create a named function (namespace, in the example abc) to stop the content becoming accessible in the browser source

<script type=”text/javascript”>

function abc() {

var temp = 10;

console.log(temp);

}

abc();

</script>

**Property** – An Object characteristic, such as colour or in the example below eyes…

<script type=”text/javascript”>

// create properties/characteristics in the function

function bodyParts() {

this.eyes = 2;

};

//apply the properties to a new variable, in this case a variable called human1

var human1 = new bodyParts();

//print a specific property to the console

console.log(human1.eyes);

</script>

**Method** – An object capability, such as walk. It is a subroutine or function associated with a class. Method is a behaviour

<script type=”text/javascript”>

function bodyParts() {

this.eyes = 2;

//this is a method

this.getEyes = function () {

return this.eyes;

}

};

var human1 = new bodyParts();

console.log(human1.getEyes());

</script>

**Constructor** – Used for the construction of objects and a way of not repeating functions

Begin by assigning a name in the function brackets that links to the function copy below.

This enables you to change the variable

<script type=”text/javascript”>

function bodyParts(eyes) {

this.eyes = eyes;

//this is a method

this.getEyes = function () {

return this.eyes;

}

};

var human1 = new bodyParts(2);  
console.log(“Eyes of human 1 “, human1.getEyes());

var human2 = new bodyParts(2);  
console.log(“Eyes of human 2 “, human2.getEyes());

</script>