Objects

* Object literals

An object literal is an object that is created directly in the language by wrapping all its properties and methods in curly braces {}. They allow objects to be created quickly without need for defining a class.

* Adding properties to objects

ES6 has added way to compute properties, such as

const hulk = { name: ‘Hulk’, [‘catch’ + ‘Phrase’]: ‘Hulk Smash!’ };

This is the same as the *hulk* object having the following properties:

{ name: ‘Hulk’, catchphrase: ‘Hulk Smash!’ }

Values of properties can also be JavaScript expressions, like

const bewitched = true;

const captainBritain = { name: ‘Captain Britain’, hero: bewitched ? false : true };

This makes the value of the *hero* property dependent upon the Boolean value of *bewitched*.

Symbol data-type objects can also be used as a computed property key.

Accessing properties are done by square brackets [].

variable[property];

>> propertyValue

* Object methods

Calling methods of objects is done the same as functions, using parentheses ().

superman.fly();

>> ‘Up, up, and away!’

Checking if they exist is done by using the *in* operator:

‘city’ in superman;

>> false

You can also use *hasOwnProperty*() to check if the object has its own, rather than if it inherited it. This will **only** return any properties that belong to that particular object, whereas *in* or “!== undefined” will return *true*, even if the property has been inherited from another object.

You can find all properties of an object using *for-in* loops. For example;

*for(const* key *in* superman) {

console.log(key + “: ” + superman[key]);

}

returns all the property names (keys) and their values to the console window.

Objects are **NEVER** like an ordered list like an array, set, or map, so do not rely on them as such, with properties in a certain order!

* JSON

JSON – **FUNCTIONS ARE NOT PERMITTED VALUES**

Use *stringify*() to format JavaScript Objects to a String object of JSON data. Methods defined within the object are ignored by the *stringify*() method.

* Math object
* Date object

**Use getFullYear(), not getYear().**

* RegExp object

RegExp, or Regular Expression, is a pattern to search string for matches to the pattern (common to use in “find and replace”).

Using carots ^ negates the expression sequence, like

/[^A-Z]/

represents anything *not* a capital letter.