**JavaScript (JS) Introduction**

Some Basic Facts

* JavaScript ≠ Java programming language
* HTML is used for defining the framework of a website, CSS is used to stylize the website, whereas Js is used to make the website responsive (it makes user interaction to a website possible)
* There are various frameworks of Js in use, like:
  + Website client side (Js, jQuery, React js ….)
  + Website server side (Node Js, Express Js)
  + Mobile Development (React Native, Pone Gap, iconic …)
  + Software Development (electronjs) [ primary example: VS code]

ECMA and Js

* **ECMA International** is an organization that creates standards / protocols for technologies
* **ECMA-262** is a standard provided by ECMA International. It contains specifications for a general-purpose scripting language.
* **ECMAScript** comes under ECMA-262. It provides rules, details & guidelines that a scripting language must observe to be considered ECMAScript compliant. It specifies how to create a scripting language.
* **JavaScript** is a general-purpose scripting language that confirms to the ECMAScript specifications.
* In short, JavaScript is a scripting language that implements ECMAScript specification as described in ECMA-262.
* ECMAScript has a lot of versions. Before, a new version usually brings in a lot of major changes so the time duration was large. After **ES6 (ECMAScript 6 / ECMAScript 2015)** which was a significant major update to previous scripts, ECMA started releasing updates to ECMAScript yearly.
* ECMAScript versions and their release year are:
  + ES1 1997
  + ES2 1998
  + ES3 1999
  + ES4 (**Abandoned,** due to bugs and instability)
  + ES5 2009
  + **ES6 2015** / **ECMAScript 2015 (ES2015) –** 6th edition [ Major change]
  + ES7 2016 / ECMAScript 2016 (ES2016) –7th edition
  + ES8 2017 / ECMAScript 2017 (ES2017) –8th edition
  + ES9 2018 / ECMAScript 2018 (ES2018) –9th edition
  + ES10 2019 / ECMAScript 2019 (ES2019) –10th edition
  + ES11 2020 / ECMAScript 2020 (ES2020) –11th edition
* Each ECMAScript version comes with new features, but they are not immediately added into function to all the web servers. The updates take time to roll in into being features in web servers after their stability is confirmed.

**Note:**

* **MDN** website is best for checking in documentation over HTML, CSS and Js. It also gives information on which features are present in which version of the browser.
* We can also use **Can I use** website to look into browser compatibility.

Where to write Js ?

* Js code can be written anywhere in head-tag / body-tag inside **script-tag** .
* Js can also be written externally and added to HTML file anywhere in head-tag / body-tag by using **src-attribute** of **script-tag.**
* Js can also be written in the **console area of any browser** to do debug or to do inspection of website.
* We can include as many scripts as we want in an HTML file and they will be executed as per their order in the file.

**Note:**

**Js is mainly attached at the bottom of the body tag**. Much likely because the web browser reads the HTML code with top-down approach, so if the Js code is attached at the top (head-tag) then it will load the Js and run it before reading the whole HTML code. It will throw an error as the Js code will not be able to find the element for which the code was written as the browser has not yet fully read the complete HTML file.

async & defer – attribute of script-tag

* **async** is the default way script-tag works.

Syntax : <script **async** src=”address of Js”> </script>

If async is specified in script-tag, the browser will load the Js file and run it as soon as it finds the script-tag.

* **defer** is a way to mimic the behaviour of including Js at the bottom of the body-tag without actually writing it there.

Syntax : <script **defer** src=”address of Js”> </script

If defer is specified in script-tag, the browser will read the whole HTML file first ignoring the position of the script-tag and then finally load and run the Js.