# Task 1.1

# Explain how JavaScript is used as a web programming language.

Headings/bold

The eight datatypes https://www.w3schools.com/js/js\_datatypes.asp

JavaScript was originally developed in 1995 to allow interactivity on webpages. Originally called Mocha, or LiveScript, it was first implemented in Netscape Navigator 2. JavaScript began to be adopted by more web clients and proved more popular than competing standards such as Microsoft’s JScript. It became a de facto standard before being officially standardized as ECMAScript (as Oracle Systems hold a copyright on JavaScript) in 1997.

There have been multiple versions of ECMAScript over the years, and some web browsers have incomplete or quirky implementation of the standard. We just to have to code to specific browser versions due to the differences between them but this has greatly improved over the years requiring less and less hacks or workarounds.

In web browsers JavaScript interacts with both data and user interface to allow a wide range of interactions. The DOM allows scripts to both view and modify the view rendered in a web browser from simple effects (such as roll overs) to complex data modelling, modification and UI elements.

JavaScript does not interact directly with HTML or CSS (as these are just the languages used to describe the UI data initially loaded into the DOM) but can effectively achieve this by manipulating its model stored in the DOM. It can be used to perform CRUD operation on any DOM element, allows quite complex user interface modifications.

JavaScript is an event driven language, where the host of the JavaScript engine (eg web browser) raises event when something happens such as a UI click, a timer fires or a system event is raised (such as close the window). To react to this event, JavaScript must declare an event handler in the DOM. These are functions that are called by the JavaScript engine when it receives a notification for a matching event.

These interactions with the UI (through the DOM) allow JavaScript to perform a range of activities. Amongst the activities it can perform it is widely used for Form Validation and UI animation effects. Form Validation is the function of ensuring that any data inputted by the user is correct and suitable for use. Common examples include check a text input conforms to the format used by telephone numbers, dates or credit card numbers, or that data does not exceed minimal or maximal lengths. It can also be used to show/hide or create UI elements dynamically – a user option could change the language shown on a page without a reload, or change the visual theme on a page by requesting new data from a server and incorporating it into the DOM.

Animation effects are simply modification of the DOM, and can be simple changes to CSS (make buttons green to show ok, or disabled/grey when data is incomplete) or more complicated drawings generated on the fly.

JavaScript also allows asynchronous operations, giving the appearance of multi-threaded operations. Asynchronous operations allow the JavaScript engine to appear to process multiple operations simultaneously without blocking the processing of code while a long running process completes. An example of this is when that while a server request is being processed (potentially taking a long time) over JavaScript events can be processed, so a ‘submit’ button would not stop the processing over animations.

On the server-side execution engines such as Node allow JavaScript to be executed to perform not client operations. These could be used to generate data and return it (generate an image or a report perhaps), or to modify data stored in a file system or database depending on validated user requests. The security issues on server side execution are more arduous that web client interfaces but are far more important to get correct. [Little Bobby Tables](https://www.explainxkcd.com/wiki/index.php/327:_Exploits_of_a_Mom) is still true today.

More traditional web applications used a submit and redirect structure where after a browser request to a server has been processed, the server redirects the browser to show a different URL. So for example, an Edit page would save any changes and instruct the browser to show the List page. JavaScript allows a different paradigm called Single Page Applications (SPA), where after the server processes the clients request, the JavaScript running the browser simply updates the elements that have been changed without having to refresh the entire DOM from server files. An example or this is chat window, where changes from the server update the chat content dynamically and across multiple clients simultaneously and no clients need to reload the entire page, which would ruin the chat experience.