About JavaScript:-

JavaScript was introduced as a full-fledged client-side language used for developing web applications in 1995. JavaScript is easy to learn, debug, and test. It is an event-based, platform-independent, and an interpreted language with all the procedural programming capabilities.

Web applications that were developed using server-side programming languages like Java and Dot Net involved travel of user requests all the way from the client(browser) to the server hosting the application. The multiple request-response cycles between client and server who were consuming both time and network bandwidth.

JavaScript got introduced as a Client-Side programming language with the capability of executing user requests on the Client-Side. This could help in reducing the number of request-response cycles between client and server and decrease the network bandwidth thus reducing the overall response time.

Later, in 1997 ECMAScript established a standard for the scripting languages that redefined the core features any scripting language should have and how to implement those features. From then, JavaScript evolved year after year with every new version of ECMAScript introducing new features. Developers prefer JavaScript to create dynamic, interactive, and scalable web applications as it helps developers in extending the functionalities of the web pages effectively.

This course will enable you to understand the features of ECMA and help to develop more robust web applications using JavaScript and handle user requests on the Client-Side.

Demonstration:-

When an application is loaded on the browser, there is a 'SignUp' link on the top right corner.

When this link is clicked, the 'SignUp' form is displayed. It contains three fields - 'Username', 'Email', and 'Password' and in some cases a 'Submit' button as well.

When data is entered in the fields and the button is clicked, then data entered in the fields will be validated and accordingly,next view page loaded. If data is invalid, an error message is displayed, if valid, the application navigates to homepage

How to handle the user click, validate the user data, and display the corresponding view?

To implement the requirement of handling user action like a click of a button or link and to respond to these requests by displaying the expected output, server-side languages like Java/JSP can be used as shown in the below diagram.

But server-side languages have certain limitations such as :-

* Multiple request-response cycles to handle multiple user requests
* More network bandwidth consumption
* Increased response time

If client-side scripting language JavaScript is used then, this can be done without consulting the server as can be seen in the below diagram.

The home page of MyMovie.com contains the SignUp link. The user performs click action on this link. The user action is handled on the client side itself with the help of the JavaScript code. This code arrives on the client along with the home page of the application.

When invoked on click of the link, this code executes on the client-side itself to validate the user-entered data and accordingly display the corresponding view.

Following are the advantages of this approach:

* No need for back and forth request-response cycles
* Less network bandwidth consumption
* In comparison to Java: JavaScript provides a 35% decrease in average response time and Pages being served 200ms faster.

**About ES6**

**About ES6:**

* **JavaScript was introduced as a client-side scripting language in 1995.**
* **ECMAScript established a standard for scripting languages in 1997.**
* **ES is a parent of many scripting languages like TypeScript, JScript, ActionScript, and JavaScript.**
* **JavaScript evolved year after year with every new version of ECMAScript introducing new features.**
* **ES6 also called ES2015.**

**ES6 introduces new transformed syntax to extend existing JavaScript constructs to meet the demands of complex applications written in JavaScript.**

**Below are some of the features that got introduced which extend the ability of JavaScript to ease the developer's web development**

* **Block scope variables using let and const keywords**
* **Template Literals**
* **Destructuring**
* **Arrow functions**
* **Enhanced for loop**
* **Default and Rest parameters**
* **Spread operator**
* **Classes and Objects**
* **Inheritance**
* **Native Promises**
* **Many other built-ins**

**The applications which have been implemented in ES6 uses the best practices, new web standards, and cutting-edge features, without using additional frameworks or libraries.**

**ES6 is also completely backward compatible. The features like Object Oriented support, New programming constructs, Modules, Templates, support for promises, etc. made ES6 faster.**

JavaScript is the programming language for web users to convert static web pages to dynamic web pages.

Web page designed using HTML and CSS is static.

JavaScript combined with HTML and CSS makes it dynamic.

**JavaScript was not originally named as JavaScript. It was created as a scripting language in 1995 over the span of 10 days with the name 'LiveScript'.**

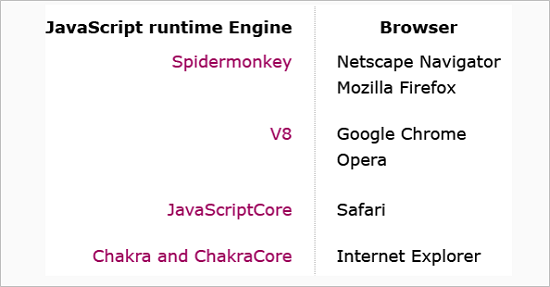
**The Scripting language is the one that controls the environment in which it runs.**

**But now JavaScript is a full-fledged programming language because of its huge capabilities for developing web applications. It contains core language features like control structures, operators, statements, objects, and functions.**

**JavaScript is an interpreted language. The browser interprets the JavaScript code embedded inside the web page, executes it, and displays the output. It is not compiled to any other form to be executed.**

**All the modern web browsers are with the JavaScript Engine, this engine interprets the JavaScript code. There is absolutely no need to include any file or import any package inside the browser for JavaScript interpretation.**

**Below are commonly used browser JavaScript engines.**

****

**Even though the latest version of JavaScript has advanced features still developer face challenges in executing the advanced code directly in the browser as:**

* **Latest syntax support is still low across browsers & servers (max is less than 70%)**
* **The features that are supported differ between browsers (with some overlap)**

**None of the IE browsers significantly support the latest features (the new Microsoft Edge browser does)**

**So, to overcome these drawbacks, the conversion of JavaScript code written using the latest syntax to browser understandable code takes place using the transpilers such as Babel, Traceur, TypeScript, etc.**

**Thus, after the code is transpiled,it will be cross-browser compatible.**

**JavaScript code can be embedded within the HTML page or can be written in an external file.**

**There are three ways of writting JavaScript depending on the platform :**

* **Inline Scripting**
* **Internal Scripting**
* **External Scripting**

**When JavaScript code are written within the HTML file itself, it is called internal scripting.**

**Internal scripting, is done with the help of HTML tag :  <script> </script>**

**This tag can be placed either in the head tag or body tag within the HTML file.**

**JavaScript code written inside <head> element is as shown below :**

1. **<html>**
2. **<head>**
3. **<script>**
4. ***//internal script***
5. **</script>**
6. **</head>**
7. **<body>**
8. **</body>**
9. **</html>**

**JavaScript code written inside <body> element is as shown below :**

1. **<html>**
2. **<head>**
3. **</head>**
4. **<body>**
5. **<script>**
6. ***//internal script***
7. **</script>**
8. **</body>**
9. **</html>**

****

**JavaScript code can be written in an external file also. The file containing JavaScript code is saved with the extension \*.js (e.g. fileName.js)**

**To include the external JavaScript file, the script tag is used with attribute 'src' as shown in the below-given code-snippet:**

1. **<html>**
2. **<head>**
3. ***<!-- \*.js file contain the JavaScript code -->***
4. **<script src="\*.js"></script>**
5. **</head>**
6. **<body>**
7. **</body>**
8. **</html>**

**Example:**

**Demo.js :-**

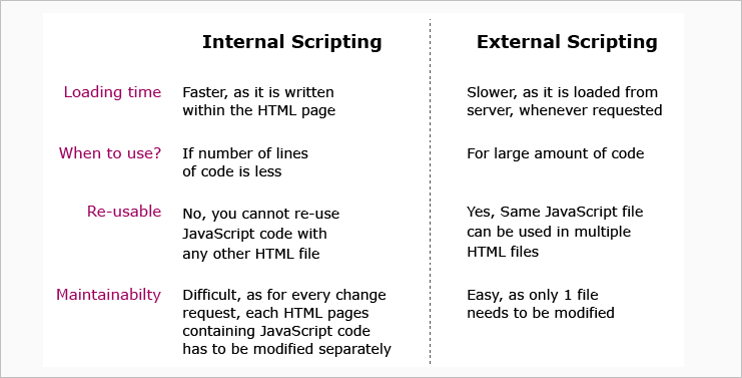
1. **let firstName="Rexha";**
2. **let lastName ="Bebe";**
3. **console.log(firstName+" "+lastName);**

**Demo.html :-**

1. **<html>**
2. **<head>**
3. **<script src="Demo.js"></script>**
4. **</head>**
5. **<body>**
6. **</body>**
7. **</html>**

**NOTE: In external file, JavaScript code is not written inside <script> </script> tag.**

**The below-mentioned points can help you choose between any two ways of writing the script based on some parameters.**

****

**Note: External scripting is used throughout this course. But due to the platform built-in feature,HTML code cannot be explicitly linked to external script using <script></script> tag. HTML code that are used to write is automatically linked to JavaScript code written on that page.**