SUNSTONE

JavaScript

LECTURE 1 NOTES

Introduction to JavaScript

What is JavaScript?

- JavaScript is a high-level programming language that is primarily used for web development. It allows developers to add interactive elements, dynamic content, and behavior to websites. JavaScript is a versatile language that can be used both on the client-side (in the web browser) and on the server-side (with the help of technologies like Node.js).

- Here is the difference between HTML, CSS, and JavaScript.

- Here are some key features and uses of JavaScript:

1) All popular web browsers support JavaScript because they provide built-in execution environments.

2) JavaScript follows the syntax and structure of the C programming language. Thus, it is a structured programming language.

3) JavaScript is a weakly typed language, where certain types are implicitly cast (depending on the operation).

4) JavaScript is an object-oriented programming language that uses prototypes rather than using classes for inheritance.

5) It is a light-weighted and interpreted language.

6) It is a case-sensitive language.

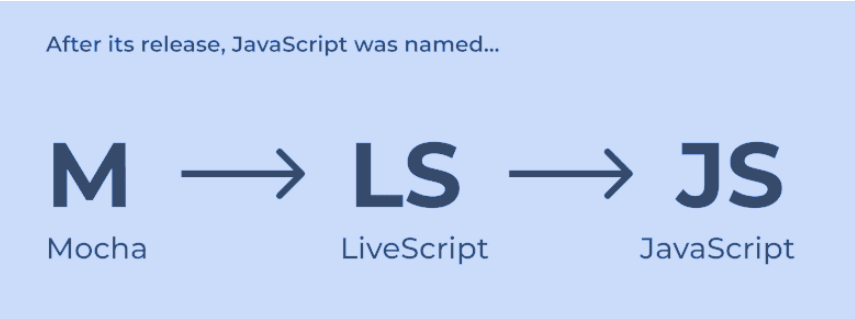
7) JavaScript is supportable in several operating systems including Windows, macOS, etc.

8) It provides good control to the users over the web browsers

Overall, JavaScript is a versatile and essential programming language for web development, enabling developers to create interactive and dynamic websites, build server-side applications, and develop mobile apps

History of JavaScript

- JavaScript was created by **Brendan Eich,** an American computer programmer in just 10 days while he was working at Netscape Communications Corporation (now known as Mozilla) in **1995**. It was originally named **'Mocha'** but was quickly renamed to **LiveScript** and finally to **JavaScript** due to marketing reasons and its potential association with the growing popularity of Java at that time.



- The motivation behind creating JavaScript was to add interactivity and dynamic functionality to web pages, which were predominantly static at that time. The language was designed to be lightweight and easy to learn, allowing web developers to write scripts that could be executed on the client-side, directly within web browsers.

- In **1996**, Netscape submitted JavaScript to the **European Computer Manufacturers Association (ECMA)** to standardize the language. The standardization process resulted in the creation of ECMAScript, which is the official name of the JavaScript language specification. The first edition of the ECMAScript standard, ECMAScript 1, was released in 1997.

- Over the years, JavaScript has undergone significant improvements and updates to meet the evolving needs of web development. Major versions of ECMAScript have been released, introducing new features and improving the language's capabilities. Some notable versions include ECMAScript 3 (1999), ECMAScript 5 (2009), ECMAScript 6 (ES6) or ECMAScript 2015 (2015), ECMAScript 2016, ECMAScript 2017, ECMAScript 2018, and ECMAScript 2019. Each version introduced new syntax, features, and improvements to the language.

- JavaScript's popularity soared as web development expanded, and it became a fundamental technology for building interactive websites and web applications. The introduction of **AJAX (Asynchronous JavaScript and XML)** in the early 2000s further propelled JavaScript's usage by enabling seamless communication between the browser and the server, facilitating dynamic updates without page reloads.

- In recent years, JavaScript has experienced significant advancements, with modern frameworks and libraries such as **React.js, AngularJS, Vue.js,** and **Node.js** gaining prominence. These tools have contributed to JavaScript's expansion into server-side development, desktop application development, and even mobile app development through technologies such as React Native and Ionic.

- JavaScript has evolved into a versatile language that is now widely used across different platforms and has a massive ecosystem of libraries, frameworks, and tools supporting its development. It continues to be a dominant force in web development, powering countless websites and applications.

Applications of JavaScript

- Client-side validation
- Dynamic drop-down menus
- Displaying date and time
- Displaying pop-up windows and dialog boxes (like an alert dialog box, confirm dialog box and prompt dialog box)
- Displaying clocks etc

JavaScript in Web Development

JavaScript is a widely used programming language for web development. It is primarily used for creating dynamic and interactive elements on websites. Here are some key points about JavaScript in web development:

- **Client-side Language:** JavaScript is primarily executed on the client-side, meaning it runs in the web browser of the user. It enables developers to add interactivity, manipulate the DOM (Document Object Model), and respond to user actions without the need for server-side communication.
- **Web Browsers:** JavaScript is supported by all modern web browsers, including Chrome, Firefox, Safari, and Edge. This widespread support makes it a reliable choice for building web applications.
- **DOM Manipulation:** JavaScript allows developers to access and manipulate the HTML document's elements, structure, and styles using the DOM API. With JavaScript, you can dynamically change content, modify styles, create and remove elements, handle events, and more.
- **Event Handling:** JavaScript provides a mechanism to handle user interactions such as button clicks, form submissions, mouse movements, and keyboard events. You can write event handlers to respond to these events and trigger appropriate actions or behaviors.
- **AJAX and Fetch:** JavaScript enables asynchronous communication with web servers using techniques like AJAX (Asynchronous JavaScript and XML) and the Fetch API. These features allow you to retrieve data from the server without refreshing the entire web page, resulting in a more interactive and responsive user experience.
- **Frameworks and Libraries:** JavaScript has a rich ecosystem of frameworks and libraries that simplify web development. Popular frameworks such as React, Angular, and Vue.js provide reusable components and efficient ways to manage the application state.
- **Form Validation:** JavaScript can be used to validate user input on forms before submitting them to the server. You can perform various validations such as checking for required fields, validating email addresses, enforcing password strength, and more.

- **Animations and Effects:** JavaScript, combined with CSS, enables the creation of animations and visual effects on web pages. You can animate elements, create transitions, and apply dynamic effects to enhance the user interface.
- **Browser APIs:** JavaScript provides access to a wide range of browser APIs that allow developers to leverage device features. These APIs include the Geolocation API, Web Storage API, Canvas API, Web Audio API, and more, enabling web applications to utilize various capabilities of the user's device.
- **Third-Party Integration:** JavaScript allows integration with third-party APIs and services, such as social media platforms, payment gateways, mapping services, and more. This integration enables developers to extend the functionality of their web applications by leveraging existing services.

JavaScript's versatility and widespread adoption make it a powerful tool for web development. It is constantly evolving, with new features and improvements being added regularly. Learning JavaScript opens up a world of possibilities for creating dynamic, interactive, and engaging web applications.

Basic Syntax and Structure of a JavaScript Program

The basic syntax and structure of a JavaScript program consist of a set of rules that define how the code should be written and organized.

The basic structure of JavaScript Program is:

```
//HTML Code
<HTML>
   <BODY>
     <SCRIPT>
     document.write(" WELCOME TO SUNSTONE ");
      //This is Visible in Console Window
     </SCRIPT>
   </BODY>
</HTML>
```

Output in Console Window:

WELCOME TO SUNSTONE

- ## Running JavaScript Code

To run JavaScript code, you can choose between a web browser environment or a Node.js environment. Here's a brief explanation of how to run JavaScript code in each of these environments:

1. **Running JavaScript in a Web Browser:**

To run JavaScript code in a web browser, you have a few options:

a. **Inline Script:** You can include JavaScript code directly within an HTML file using the <script> tag. Here's an example:

```
// HTML Code

<HTML>

<HEAD>

  <TITLE> JavaScript Page </TITLE>

</HEAD>

<BODY>

  <H1> JavaScript </H1> //This will be visible in
browser page

  <SCRIPT>

       // JavaScript code goes here

       console.log(' Hello, World! ');

  </SCRIPT>

</BODY>

</HTML>
```

You can open the HTML file in a web browser, and the JavaScript code will be executed. You can view the output or errors in the browser's developer console (accessible through the browser's developer tools).

b. **External Script:** Alternatively, you can write your JavaScript code in a separate .js file and include it in an HTML file using the <script> tag's src attribute. Here's an example:

```
//HTML Code

<HTML>

<HEAD>

<TITLE> JavaScript Page </TITLE>

 <SCRIPT SRC="script.js"> // External JavaScript

</SCRIPT>

</HEAD>

<BODY>

<H1> JavaScript </H1>  //This will be visible in
browser page

</BODY>

</HTML>
```

In this example, the JavaScript code is written in a file named script.js. When you open the HTML file in a browser, the browser will load and execute the JavaScript code from the external file.

2. **Running JavaScript in Node.js:**

Node.js is a runtime environment that allows you to run JavaScript code outside of a web browser. Here's how you can run JavaScript code in Node.js:

a. **Create a JavaScript File:** Create a new file with a .js extension, such as script.js, and write your JavaScript code in it.

node script.js

b. **Execute the File with Node.js:** Open a terminal or command prompt, navigate to the directory where your JavaScript file is located, and run the following command:

Replace script.js with the name of your JavaScript file. Node.js will execute the JavaScript code, and you'll see the output or errors in the terminal.

Whether you choose to run JavaScript code in a web browser or a Node.js environment depends on your specific use case. Web browsers are suitable for client-side web development, while Node.js is commonly used for server-side applications and command-line tools.