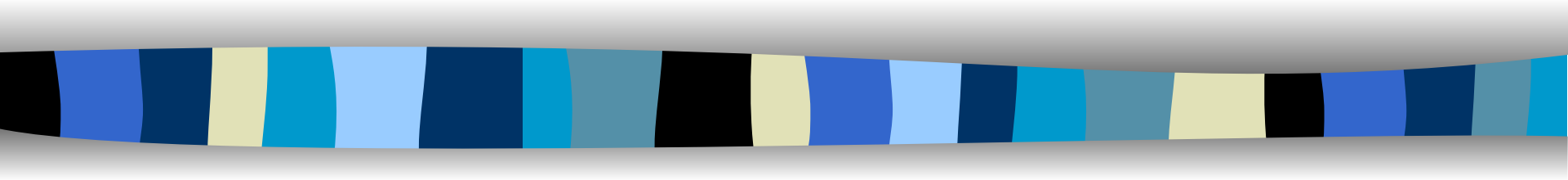


Internet Programming

By Dr. Ashfaq Shaikh Ph.D Computer Engg.
Assit. Professor MHSS Mumbai

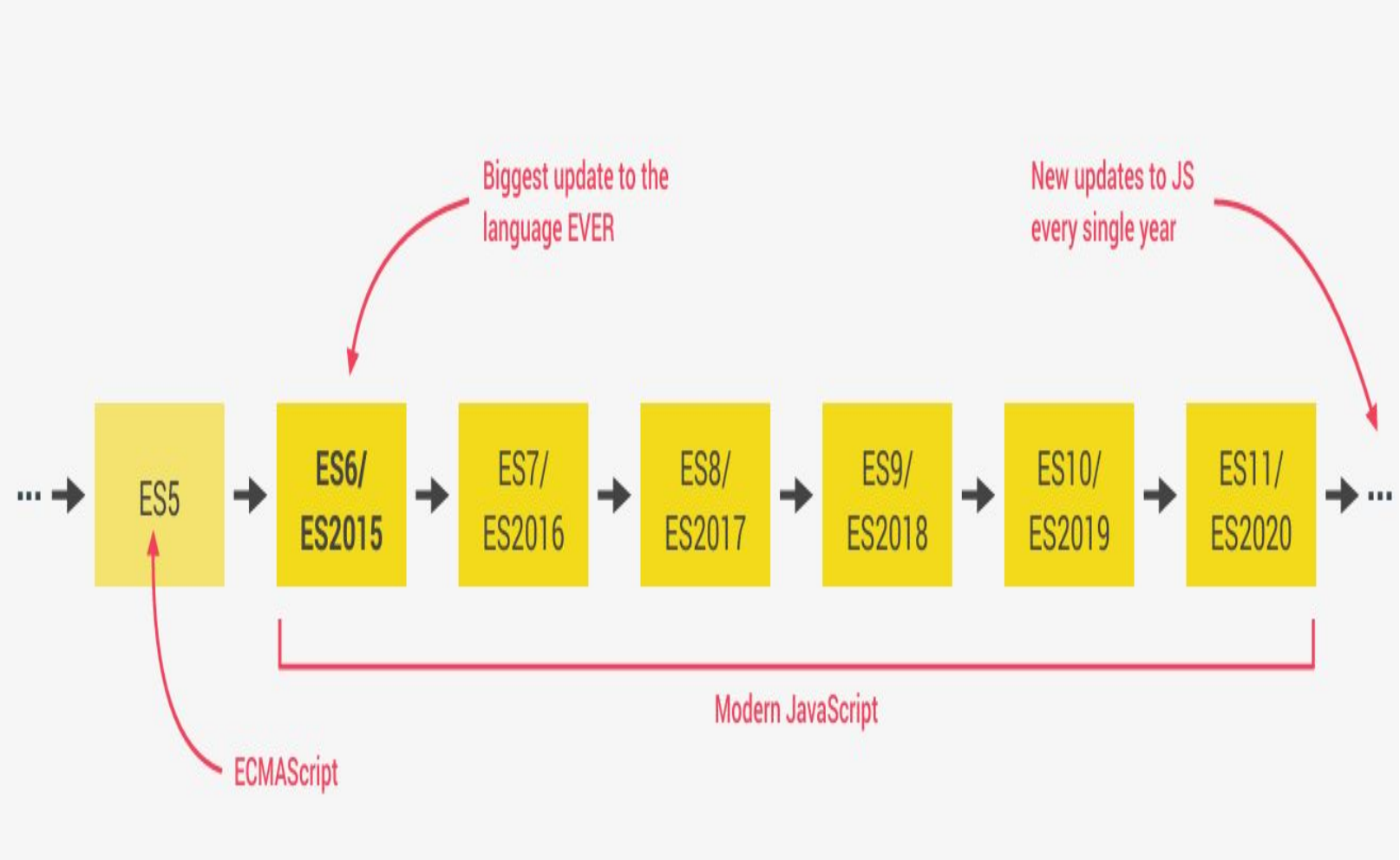
Introduction To Java Script

- Introduction to Java Script
- ES 6 As Modern JS
- JS programming
- Basics Syntax
- OOP in JS
- Arrow Function

What is Java Script

- **JavaScript** is a **programming language** used primarily to create **interactive and dynamic content on websites**.
- It runs in the browser and allows developers to build features like image sliders, form validation, animations, games, and dynamic page updates without reloading the page.
- **Event-driven**: Can respond to user actions like clicks, typing, etc.

History of JS



Three way to Write Java Script

- **1. Inline JavaScript (inside HTML tag)**
- JavaScript code is directly written inside an HTML tag's attribute, like onclick.
- <button onclick="alert('Hello!')">Click Me</button>
- 2. JavaScript is written inside a <script> tag within the same HTML file — usually in the <head> or at the bottom of <body>.

Internal Java Script

```
<!DOCTYPE html>
<html>
<head>
  <title>Internal JS</title>
  <script>
    function greet() {
      alert("Hello from internal JavaScript!");
    }
  </script>
</head>
<body>
  <button onclick="greet()">Click Me</button>
</body>
</html>
```

Ways to display O/P in JS

- **1. Using console.log() use *for debugging***
- Displays output in the **browser console**.
- console.log("This is a message in the console");
- Great for developers during development and debugging.

2. Using alert()

- **Using alert()** ***(popup dialog box***
- Shows a **popup alert box** with a message

- alert("This is an alert box!");

- Good for quick demos, but not user-friendly for real apps.

3. Using document.write()

- Writes directly to the **HTML document.**
- document.write("This will appear in the browser window");
- Avoid using it after the page has loaded — it can overwrite the entire HTML content.

4. **Using innerHTML** *(dynamic HTML content)*

- Updates content inside an HTML element.

- <p id="demo"></p
<script>
document.getElementById("demo").innerHTML =
"Output shown here!";
</script>

- Commonly used to dynamically update webpage content.

5. Using window.prompt()

- **5. Using window.prompt()** ***(to take input, but also shows output)***
- Displays a dialog box to take user input, but you can also use it to show default messages.
- let name = prompt("Enter your name:", "Guest");
alert("Hello, " + name);

How to Run the JS Code

- 1. Use NodeJs to run JS Code

a. Install nodejs
b. Save java script files as .js
c. Use command

d. >node filename.js

e. To Run JS code

How to run JS using HTML

```
<!DOCTYPE html><html> <body>
<script>
  // Take input from user
  let num1 = prompt("Enter first number:");
  let num2 = prompt("Enter second number:");
  // Convert string input to numbers
  num1 = parseFloat(num1);
  num2 = parseFloat(num2);
  // Calculate sum
  let sum = num1 + num2;
  // Show output
  alert("The sum is: " + sum);
</script></body></html>
```

Using Function

```
<!DOCTYPE html>
<html>
<body>
<h3>Add Two Numbers</h3>
<input type="number" id="num1" placeholder="Enter first
    number">
<input type="number" id="num2" placeholder="Enter second
    number">
<button onclick="addNumbers()">Add</button>

<p id="result"></p>
<script>
  function addNumbers() {
    let n1 = parseFloat(document.getElementById("num1").value);
    let n2 = parseFloat(document.getElementById("num2").value);
    let sum = n1 + n2;
    document.getElementById("result").innerText = "Sum: " + sum;
  }</script></body></html>
```

Example1

```
function addNumbers()
 {  const num1 =
    parseFloat(document.getElementById("num1").value
    );
const num2 =
    parseFloat(document.getElementById("num2").value
    );
if (!isNaN(num1) && !isNaN(num2)) {    const sum =
    num1 + num2;
    document.getElementById("result").innerText =
    "Sum: " + sum;  }
else
{    document.getElementById("result").innerText =
    "Please enter valid numbers.";  }}
```

Code for JS

```
<!DOCTYPE html>
<html>
<head>
<title>Addition of Two Numbers</title>
 <script src="add.js" defer>
</script>
</head>
<body>
<h2>
Add Two Numbers</h2>
<input type="number" id="num1" placeholder="Enter
   first number">
<input type="number" id="num2" placeholder="Enter
   second number">
<button onclick="addNumbers()">Add</button>  <p
   id="result"></p></body></html>
```

DOM Introduction

- **DOM** stands for **Document Object Model**.
- It is a **tree-like structure** that represents all elements of an HTML or XML document.
- Every element (like , <p>, <img>) becomes a **node** in the DOM tree.
- JavaScript can access and change any part of this tree.

DOM Manipulation

- **DOM Manipulation** refers to the process of **programmatically accessing and modifying the structure, content, and style** of a web page using JavaScript through the **Document Object Model (DOM)**.

- DOM Manipulation Allows **dynamic behavior** (change content without reloading the page).

- Used in **interactive features** like:

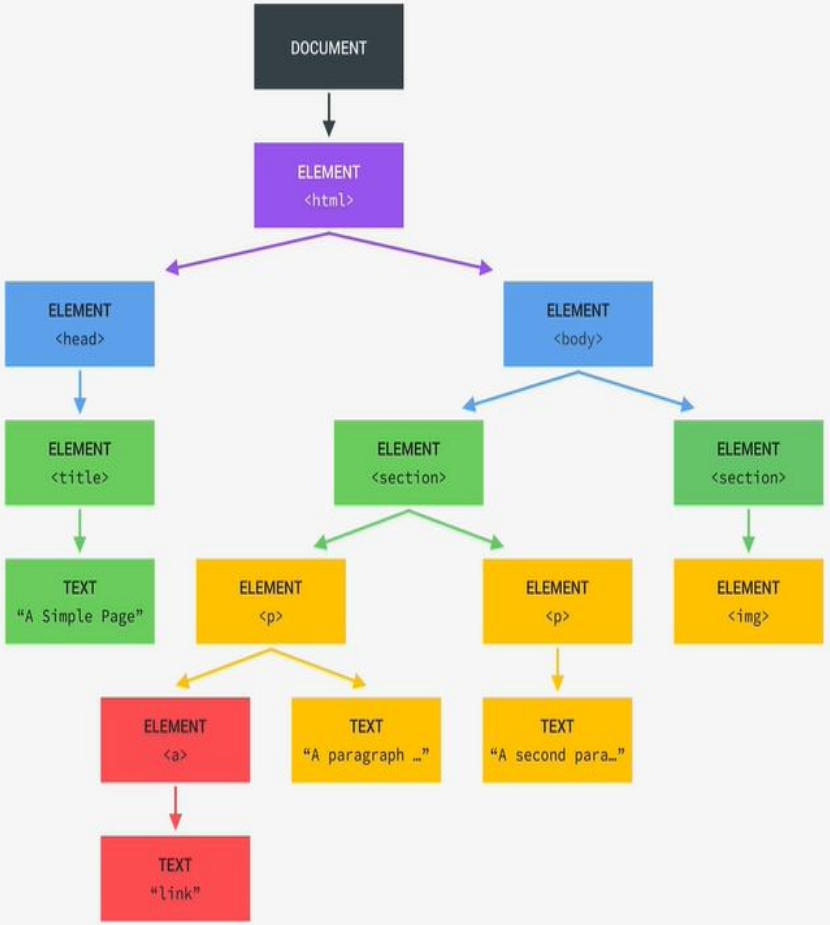
- Form validation
- Modals and popups
- Live search/filtering
- AJAX content updates

What for DOM Manipulation

Task	Example
Access elements	`document.getElementById("title")`
Change content	`element.innerHTML = "Hello"`
Modify styles	`element.style.color = "blue"`
Add or remove elements	`document.createElement("p"), element.remove()`
Handle user events	`button.addEventListener("click", function(){...})`

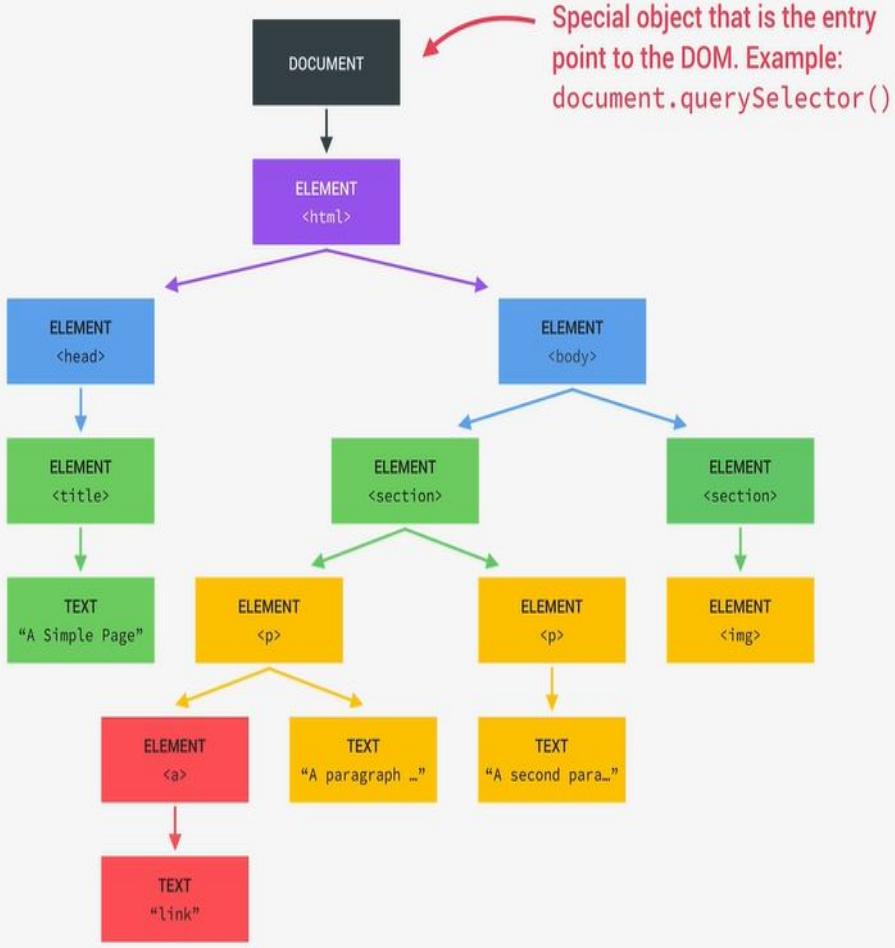
DOM

```
<html>
  <head>
    <title>A Simple Page</title>
  </head>
  <body>
    <section>
      <p>A paragraph with a <a>link</a></p>
      <p>A second paragraph</p>
    </section>
    <section>
      <img src="dom.png" alt="The DOM" />
    </section>
  </body>
</html>
```



DOM

```
<html>
  <head>
    <title>A Simple Page</title>
  </head>
  <body>
    <section>
      <p>A paragraph with a <a>link</a></p>
      <p>A second paragraph</p>
    </section>
    <section>
      <img src="dom.png" alt="The DOM" />
    </section>
  </body>
</html>
```



THANK YOU

Questions?