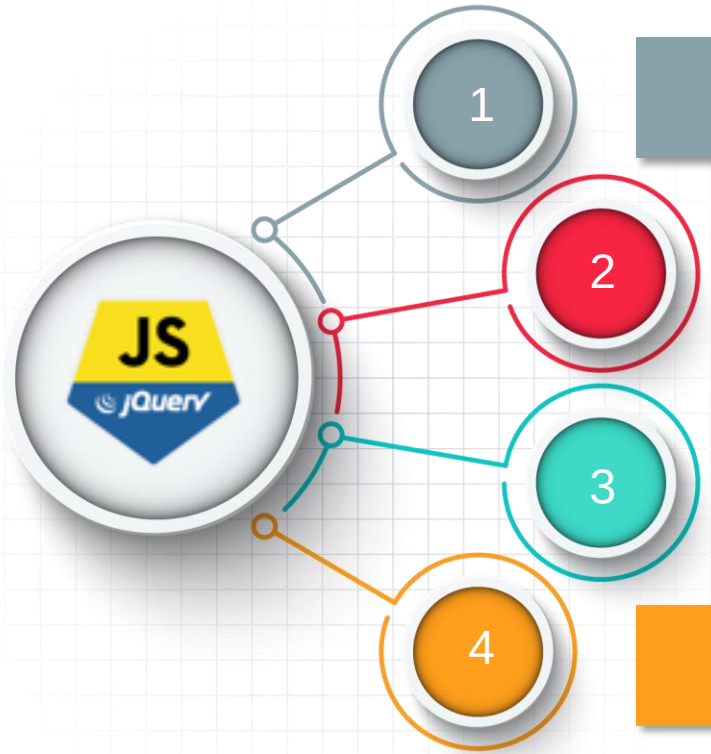
edureka!

JavaScript & JQuery

Course Outline



Introduction to JavaScript

Functions and Internal Memory Management

Event Handling and Objects

Libraries and Frameworks

edureka!

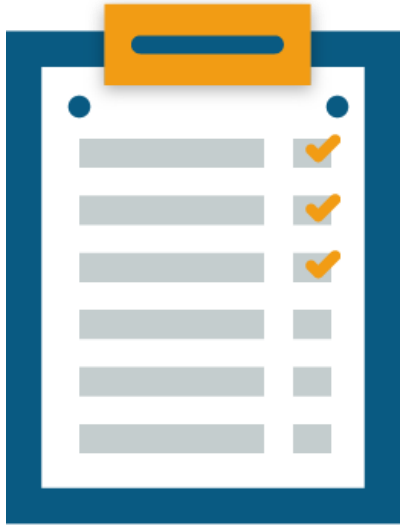
Module 3 –Objects and Event Handling

edureka!

Objectives

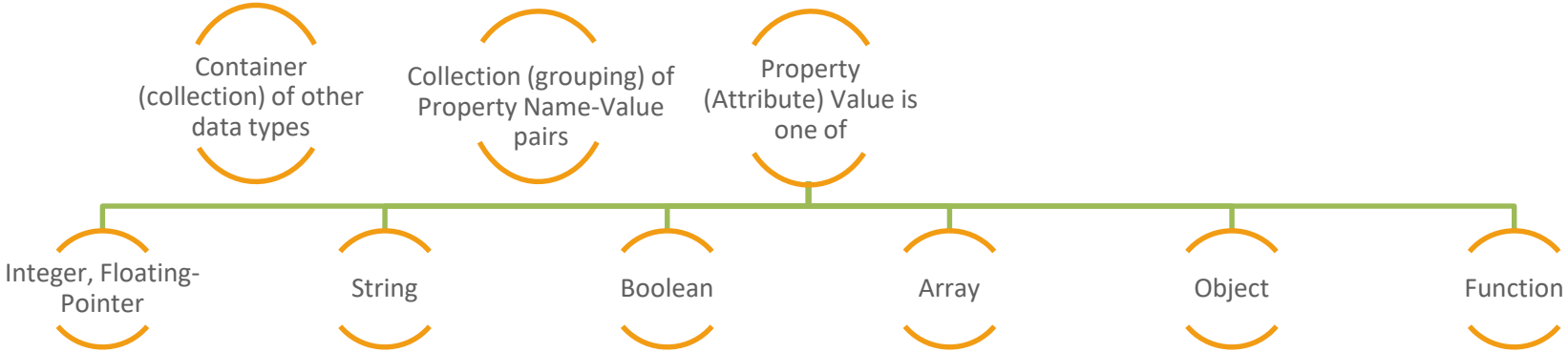
After completing this module, you should be able to:

- Create and use objects
- Access BOM methods and elements
- Access HTML Elements through DOM objects
- Understand how to change HTML Element style with DOM
- Attach Event Listeners to DOM elements
- Validate HTML forms before sending a request to the Server



edureka!

Javascript Objects



Object Creation

- Creating with new keyword

```
var person = new Object();
person.name = "saya";
person.age = 46;
```

- Creating { } with initial set of properties (attributes) with . (dot)

```
var person = {};
person.name = "saya";
person.age = 46;
```

Object Creation

- Creating initial set of properties (attributes) with [ ]

```
var person = {};
person['name'] = "saya";
person['age'] = 46;
```

- Creating with initial set of properties (attributes) with { }

```
var person = { name: "saya", age: 46 };
alert("Your name is: " + person.name + "\nYour age
is: " + person.age);
```

Object Creation

- Creating objects with Constructor function
  - Set values for the initial set of properties (attributes)
  - Constructor is executed when the object is created with **new** keyword

```
function Person () {
 this.name = "Sreenivasulu Saya";
 this.age = 46;
}
var person = new Person();
alert("Your name is: " + person.name + "\nYour age is: " +
person.age);
```

Object Creation

- Object Creation with Object.create()
- Creates (duplicates) an object with a prototype object
- No need to define a constructor function for the object
- Set values for the initial set of properties (attributes):

```
var Animal = {
  type: "Invertebrates",                              // Default value of
properties
  displayType : function(){                          // Method which will
display type of Animal
    alert("The type is:: " + this.type);
  }}
var animal1 = Object.create(Animal);
animal1.displayType();                              // Output: Invertebrates
var fish = Object.create(Animal);
fish.type = "Fishes";
fish.displayType();                                  // Output: Fishes
```

Object Property Deletion

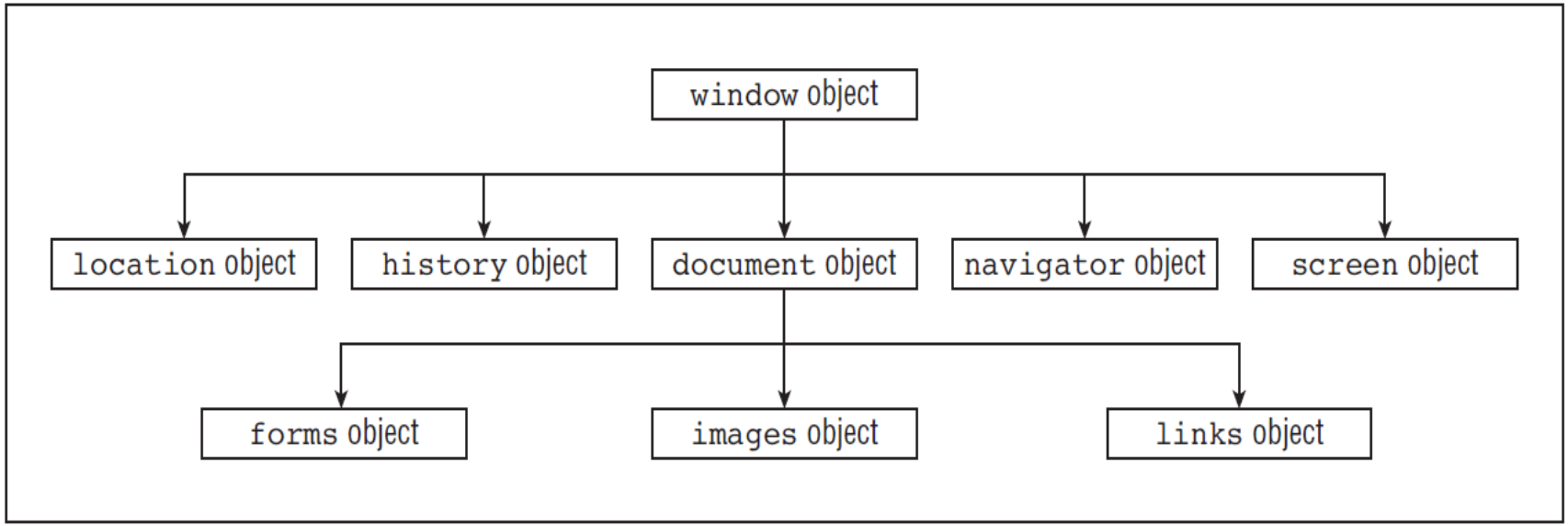
- Object Property (Attribute) Deletion
- Delete an Object's property with delete keyword

```
person = { name : "Saya", age: 40, weight: 70};
var props = "";
for( p in person)
    props += p + " ";
alert("Properties in Person object are: " + props);
delete person.age;   // delete person['age']; OR var pro= "age";
alert("Properties in Person object are after DELETION: " + props);
```

edureka!

Browser Object Model (BOM)

- Browser Objects are global objects
- Available throughout the JavaScript code
- BOM allows JavaScript code to interact with the browser properties
- The image below shows different BOM objects :



BOM Window Object

- Represents the browser's window
- Global variables are properties of the window object.
- Global functions are methods of the window object.
- Even the document object (of the HTML DOM) is a property of the window object
  - window.open() - open a new window
  - window.close() - close the current window
  - window.moveTo() -move the current window
  - window.resizeTo() -resize the current window
- Example of Window Object can be seen further in Screen, Navigator, History, Location and Document Objects, as they all are Window Objects

BOM – Screen Object

- The Screen object contains information of the users screen
- The window.screen object can be written without the Window prefix
- Some basic properties of the screen are:

screen.width	• Returns the width of the users screen
screen.height	• Returns the height of the users
screen.availWidth	• Returns the width of the users screen, excluding features like Windows Taskbar
screen.availHeight	• Returns the width of the users screen, excluding features like Windows Taskbar
screen.colorDepth	• Returns the number of bits used to display one color
screen.pixelDepth	• Returns the pixel depth of the screen.

BOM Screen Object – Example

- Using Screen Objects:

```
<script>
alert("Screen width: " + screen.width + "\n Screen height: " +
        screen.height + "\n Color Depth: " + screen.colorDepth +
        "\n ,Window Width: " + window.innerWidth + "\n Window Height:
        " +
        window.innerHeight);
</script>
```

- Output:

BOM – Navigator Object

- The Navigator object contains information of the visitors browser
- The window.navigator object can be written without the window prefix
- Some basic properties of the Navigator are:

navigator.appName

- Returns the application name on the browser

navigator.appCodeName

- Returns the application codename on the browser

navigator.platform

- Returns the platform(Operating System) name

navigator.cookieEnabled

- Returns true if cookies are enabled

navigator.product

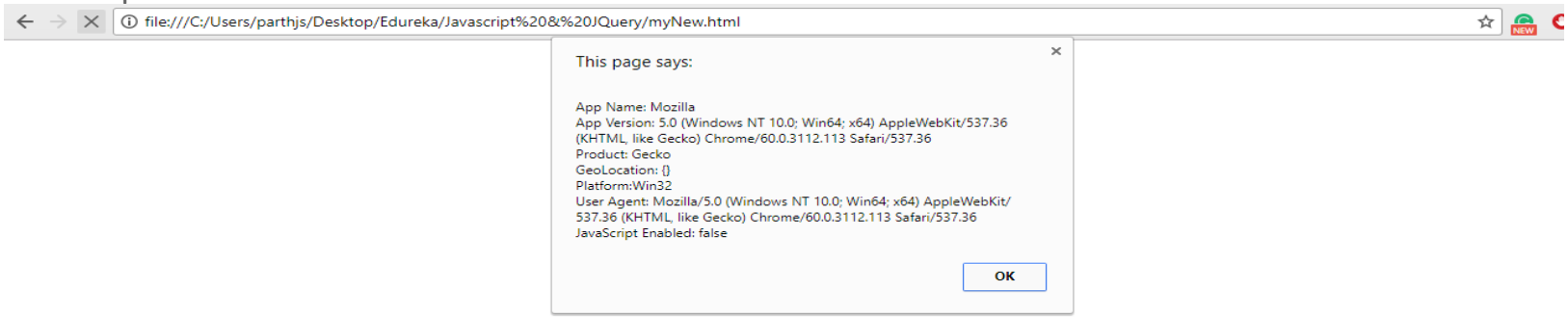
- Returns the product name of the browser engine

BOM Navigator Object – Example

- Using Navigator Objects:

```
alert( "\nApp Name: " + navigator.appCodeName +
       "\nApp Version: " + navigator.appVersion +
       "\nProduct: " + navigator.product +
       "\nGeoLocation: " + JSON.stringify(navigator.geolocation) +
       "\nPlatform:"  + navigator.platform +
       "\nUser Agent: " + navigator.userAgent +
       "\nJavaScript Enabled: " + navigator.javaEnabled());
```

- Output:



BOM Location Object

The window.location object can be used to:

- Get the current page address (URL)
- Redirect the browser to a new page.

window.location.href

Returns the href (URL) of the current page

window.location.hostname

Returns the domain name of the web host

window.location.pathname

Returns the path and filename of the current page

window.location.protocol

Returns the web protocol used (http: or https:)

window.location.assign

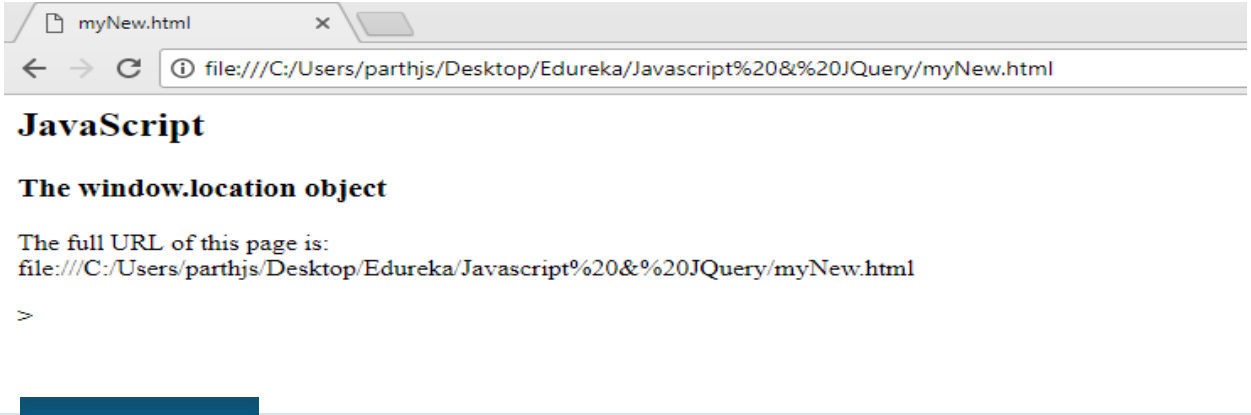
Loads a new document

BOM Location Object – Example

Using Location Object

```
<p id="demo"></p>

<script>
document.getElementById("demo").innerHTML =
"The full URL of this page is:<br>" + window.location.href;
</script>
```

Output:



BOM History Object

- The window.history object contains the browsers history
- Some methods of history object :
  - history.back() - same as clicking back in the browser
  - history.forward() - same as clicking forward in the brow

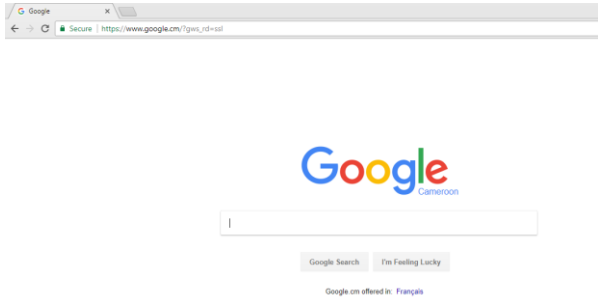
```
<p id="demo"></p>

<script>
document.getElementById("demo").innerHTML =
"The full URL of this page is:<br>" + window.location.href;
</script>
```

BOM History Object – Example

Using History Object

```
<input type="button" value="Forward" onclick="goForward()">
<script>
function goForward() {
    window.history.forward()
}
</script>
```

Output for the above piece of code will be the page that lies after the current web page in the browser history

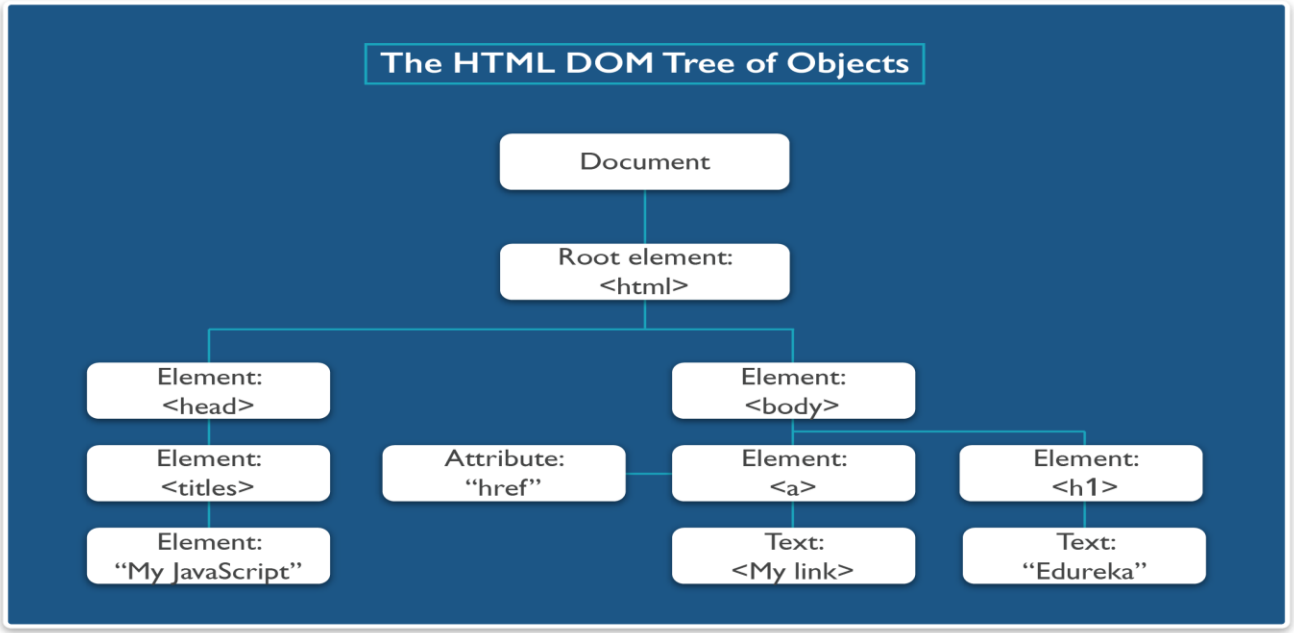


Document Object Model (DOM)

- The HTML DOM is a standard **object** model and **programming interface** for HTML.
- It defines:
  - The HTML elements as **objects**
  - The **properties** of all HTML elements
  - The **methods** to access all HTML elements
  - The **events** for all HTML elements
- **D**ocument **O**bject **M**odel of the page is created by the browser, when a webpage is loaded
- The HTML DOM model is constructed as a tree of Objects:

Document Object Model

- The HTML DOM model is constructed as a tree of Objects:



What can JavaScript do with DOM?

JavaScript + DOM = Dynamic HTML on Client-Side

Change all the HTML elements and attributes in the page

Change all the CSS styles in the page

Add and Remove existing HTML elements and attributes

JavaScript can react to all existing HTML events in the page

JavaScript can create new HTML events in the page

Document Object Model – Finding Elements

- The document object represents your web page
- To access any HTML element, you will start with accessing the document object and DOM methods to find them

getElementById() Method	getElementsByClassName() Method	getElementsByTagName() method
Gets the HTML Element with specified ID	Gets the HTML Elements with specified class name	Gets the HTML Elements with specified element tag

Document Object Model – Finding Elements

- getElementById() Method

```
document.getElementBy(“demo").innerHTML = "Welcome to Edureka!";
```

- getElementsByClassName() Method

```
document.getElementsByClassName("middlePara")[0].innerHTML = "Welcome to Edureka!";
document.getElementsByClassName("middlePara")[1].innerHTML = "This is HTML5, CSS3 and JavaScript Course!";
```

- getElementsByTagName() Method

```
document.getElementsByTagName("p")[0].innerHTML = "Welcome to Edureka!";
document.getElementsByTagName("p")[1].innerHTML = "This is HTML5, CSS3 and JavaScript Course!";
```

Document Object Model – Changing HTML Elements

- element.innerHTML Property
  - Accesses the content of an HTML Element

```
document.getElementById("welcomePara").innerHTML = "Hello World!";
```

- element.attribute Property
  - Changes the attribute of an HTML Element

```
document.getElementsByTagName("p")[0].className = "myPara";
```

- element.setAttribute Method
  - Sets the attribute of an HTML Element

```
document.getElementsByTagName("p")[0].setAttribute("class", "myPara");
```

Document Object Model – Changing HTML Elements

- element.innerHTML Property
  - Accesses the content of an HTML Element

```
document.getElementById("welcomePara").innerHTML = "Hello World!";
```

- element.attribute Property
  - Changes the attribute of an HTML Element

```
document.getElementsByTagName("p")[0].className = "myPara";
```

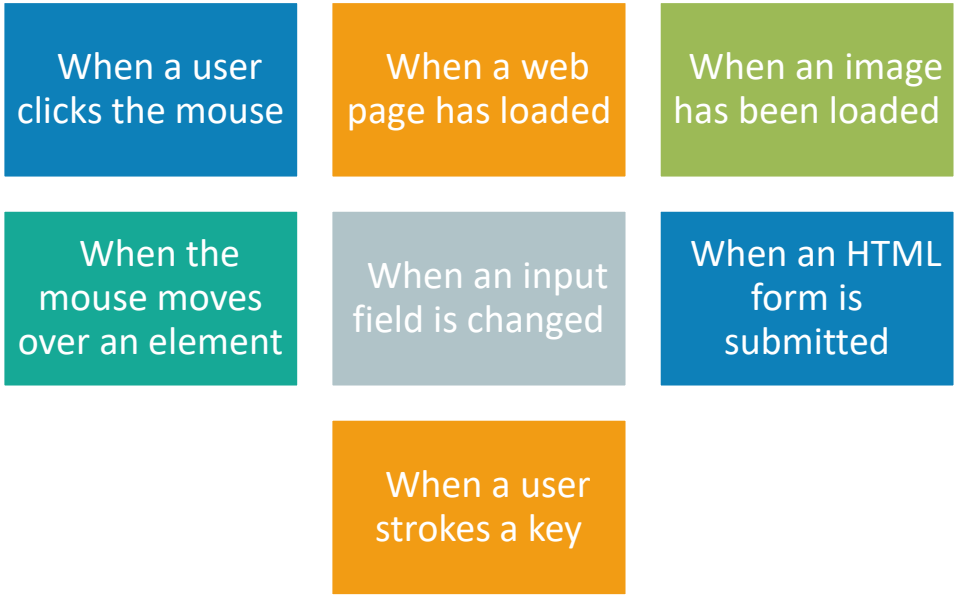
- element.setAttribute Method
  - Sets the attribute of an HTML Element

```
document.getElementsByTagName("p")[0].setAttribute("class", "myPara");
```

Document Object Model – Events

Events are “things that happen”

Example of events are:

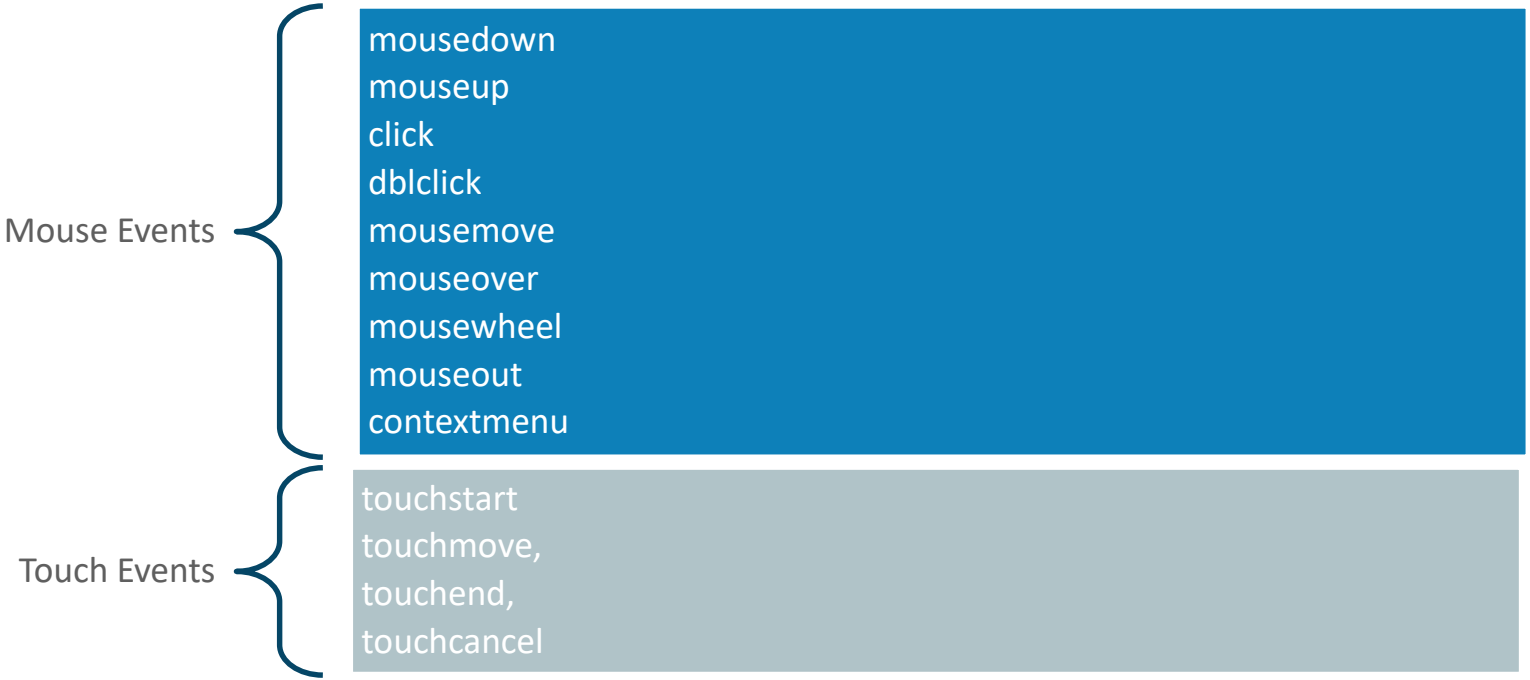


- JavaScript has the ability to react on these events

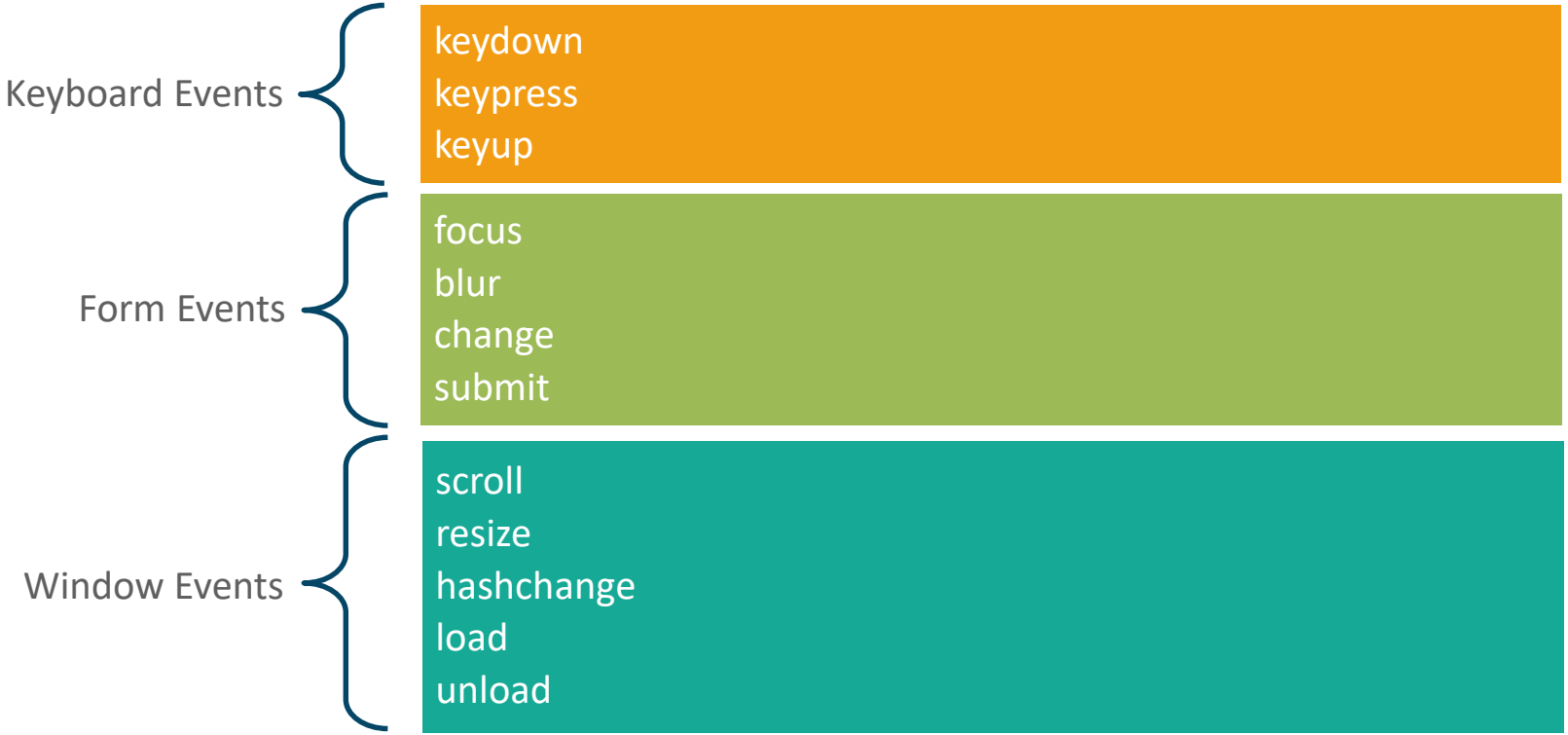
edureka!

Document Object Model – Events

- These Events can be accessed by adding them as Event Listeners to Elements



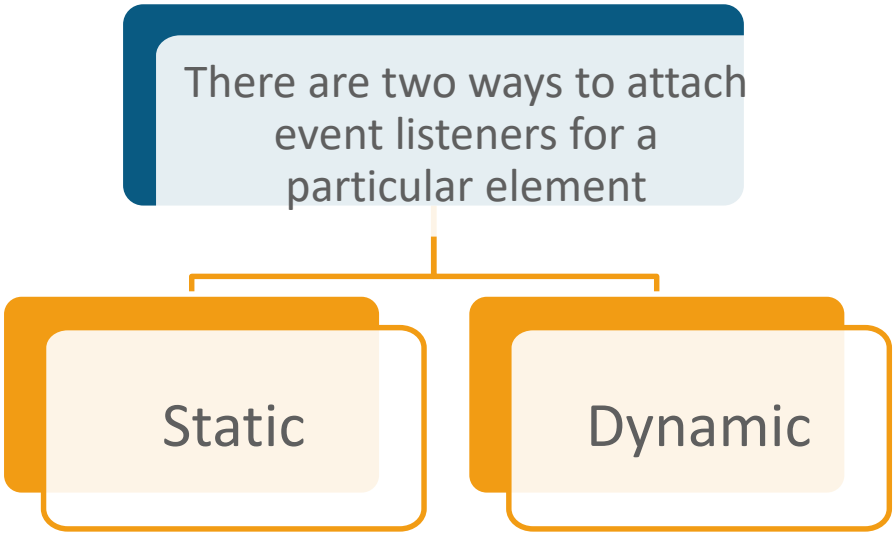
Document Object Model – Events



edureka!

Document Object Model – Event Listeners

- Event Listeners are: The type of event that is occurring
- We can attach any number of event listeners to an element



Document Object Model – Event Listeners

- Attach Event Listeners with addEventListener method, dynamically

- Syntax :

  element.addEventListener(event, function);

- Example : Event “click” will occur and call the function paraClicked(), which will change the CSS properties of the element

```
<script>
document.getElementById("myPara").addEventListener("click",
paraClicked);
function paraClicked()
{
 document.getElementById("myPara").setAttribute("style", "color:
cyan");
}
</script>
```

Document Object Model – Event Listeners

- Attach Event Handlers as HTML Element Attributes, Statically

- Syntax : <element event_listener1=“ function_name1(); ” event_listener2=“ function_name2(); ” >
- Example : Event “click” will occur, when we click on the element. It will call the function paraClicked(), which will change the CSS properties of the element

```
<p id="myPara" onclick="paraClicked();" style="color: green;">
<script>
 function paraClicked() {
document.getElementById("myPara").setAttribute("style", "color:
green"); }
</script>
```

edureka!

Form Validations

- HTML forms can be validated by JavaScript
- The main goal of Data Validation is to ensure correct user input
- Validation can be defined by many different methods, and deployed in many different ways :
  - **Server-side validation** is performed by a web server, after input has been sent to the server
  - **Client-side validation** is performed by a web browser, before input is sent to a web server
- Saves a lot of unnecessary calls to the server as all processing is handled by the web browser

Typical Validations

Check if the input field is empty
Check if the input field is a numeric value
Check if the input field is a valid email id

Form Validations – Example

Example for checking if the field in the form is empty

```
function validateForm() {
    var x = document.forms["myForm"][“text1"].value;
    if (x == "") {
        alert("Name must be filled out");
        return false;
    }
}
```

Automatic Form Validations

- HTML form validation can be performed automatically by the browser, using the **required** keyword
- Example: If the field with name “text1” is empty, the form will not be submitted

  The **required** attribute will prevent the form from being submitted

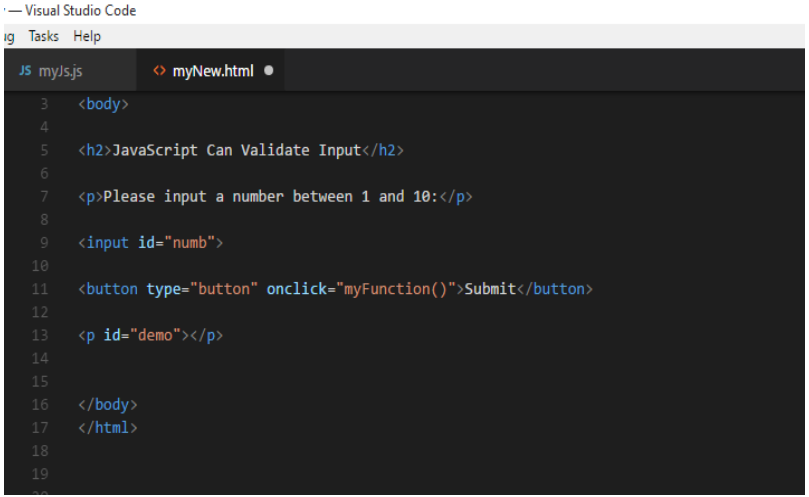
```
<form action=“”>
  <input type="text" name=“text1" required>
  <input type="submit" value="Submit">
</form>
```

Demo – Validate an HTML Page

edureka!

Validate an Input to Check if it is a Number

Step1: Type HTML code and take input from user in a Text Box



Step2: We will add onclick Event Listener in <button> which will return a Function myFunction()

```
11  <button type="button" onclick="myFunction()">Submit</button>
12
```

Step3: In myFunction(), we will find the input element, and extract out the text value in Variable x

```
function myFunction() {
    var x, text;

    // Get the value of the input field with id="numb"
    x = document.getElementById("numb").value;
```

edureka!

Validate an Input to Check if it is a Number

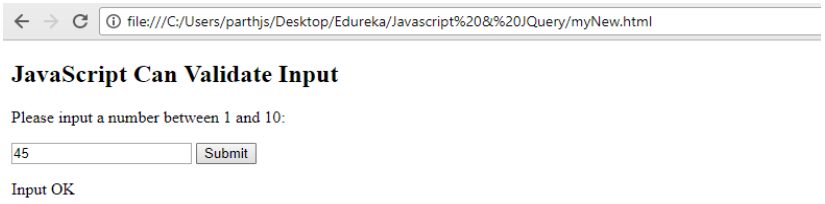
Step4: We will check if it is a numeric value by using an inbuilt function isNaN(), which will return True if the x is Not a Number

```
if (isNaN(x)) {
    text = "Input not valid";
} else {
    text = "Input OK";
}
```

Step5 : We will print the result in our paragraph having id ”demo”

```
}
document.getElementById("demo").innerHTML = text;
```

Step6: Check the output after entering a Numeric value. It will print ”Input Ok ”



Step7 : Check the output after entering any character in the input field, which is not a Numeric Value. It will print “Input not valid”

edureka!

Summary

In this module, we have learnt to:

- Create, Access and Delete Objects
- Access Browser properties and methods using BOM objects
- Change properties of HTML and CSS elements using DOM objects
- Attach Event Listeners, Statically and Dynamically
- Perform Validations on HTML Form elements

edureka!

Thank You

For more information please visit our website
www.edureka.co