Javascript

Kameswari Chebrolu

https://d2v4zi8pl64nxt.cloudfront.net/javascript-seo/5948abfc0e2df5.02876591.gif

Examples are from : https://www.w3schools.com/html/default.asp

JavaScript

- A growing demand for more interactive and dynamic content → Client-Side Scripting
  - Originated in Netscape Communications, initially named Mocha and later LiveScript
  - Was originally designed to run on the client side (i.e. in the browser)
  - Netscape and Sun later entered a collaboration, and renamed it Javascript
    - Why? Java of Sun Microsystems very popular, renamed to leverage popularity of Java
    - Note: JavaScript and Java are different languages with different purposes!

- Lots of nice features:
  - Light-weight
  - Cross-Platform Compatibility (across browsers, OS)
  - Can interact with Document Object Model (DOM)
    - Helps manipulate elements in the HTML document
- Standardized in 1997; led to widespread adoption

- Evolved new features
  - Libraries like jQuery
    - Simplifies DOM manipulation, event handling, animation, and Ajax interactions
  - Frameworks such as Angular, React, and Vue.js
    - Pre-written reusable code libraries or sets of tools
    - Help simplify and streamline development of web applications

Example: Change HTML Content

Run

```
<!DOCTYPE html>
<html>
<body>

<h2>What Can JavaScript Do?</h2>

<p id="demo">JavaScript can change HTML content.</p>

<button type="button" onclick='document.getElementById("demo").innerHTML = "Hello
JavaScript!"'>Click Me!</button>

</body>
</html>
```

What Can JavaScript Do?

JavaScript can change HTML content.

Click Me!

Run

```
<!DOCTYPE html>
<html>
<body>

<h2>What Can JavaScript Do?</h2>

<p id="demo">JavaScript can change HTML content.</p>

<button type="button" onclick='document.getElementById("demo").innerHTML = "Hello
JavaScript!"'>Click Me!</button>

</body>
</html>
```

What Can JavaScript Do?

Hello JavaScript!

Click Me!

Example: Change Attribute Values

Run

Result Size: 908 x 784

```
<!DOCTYPE html>
<html>
<body>

<h2>What Can JavaScript Do?</h2>

<p>JavaScript can change HTML attribute values.</p>

<p>In this case JavaScript changes the value of the src (source) attribute of an image.</p>

<button onclick="document.getElementById('myImage').src='pic_bulbon.gif'">Turn on the
light</button>

<img id="myImage" src="pic_bulboff.gif" style="width:100px">

<button onclick="document.getElementById('myImage').src='pic_bulboff.gif'">Turn off the
light</button>

</body>
</html>
```

What Can JavaScript Do?

JavaScript can change HTML attribute values.

In this case JavaScript changes the value of the src (source) attribute of an image.

Turn on the light Turn off the light

Run

Result Size: 908 x 784

```
<!DOCTYPE html>
<html>
<body>

<h2>What Can JavaScript Do?</h2>

<p>JavaScript can change HTML attribute values.</p>

<p>In this case JavaScript changes the value of the src (source) attribute of an image.</p>

<button onclick="document.getElementById('myImage').src='pic_bulbon.gif'">Turn on the
light</button>

<img id="myImage" src="pic_bulboff.gif" style="width:100px">

<button onclick="document.getElementById('myImage').src='pic_bulboff.gif'">Turn off the
light</button>

</body>
</html>
```

What Can JavaScript Do?

JavaScript can change HTML attribute values.

In this case JavaScript changes the value of the src (source) attribute of an image.

Turn on the light Turn off the light

Run

Result Size: 908 x 784

```
<!DOCTYPE html>
<html>
<body>

<h2>What Can JavaScript Do?</h2>

<p>JavaScript can change HTML attribute values.</p>

<p>In this case JavaScript changes the value of the src (source) attribute of an image.</p>

<button onclick="document.getElementById('myImage').src='pic_bulbon.gif'">Turn on the
light</button>

<img id="myImage" src="pic_bulboff.gif" style="width:100px">

<button onclick="document.getElementById('myImage').src='pic_bulboff.gif'">Turn off the
light</button>

</body>
</html>
```

What Can JavaScript Do?

JavaScript can change HTML attribute values.

In this case JavaScript changes the value of the src (source) attribute of an image.

Turn on the light Turn off the light

Example: Change HTML Styles (CSS)

Run

```
<!DOCTYPE html>
<html>
<body>

<h2>What Can JavaScript Do?</h2>

<p id="demo">JavaScript can change the style of an HTML element.</p>

<button type="button" onclick="document.getElementById('demo').style.fontSize='35px'">Click
Me!</button>

</body>
</html>
```

What Can JavaScript Do?

JavaScript can change the style of an HTML element.

Click Me!

Run

Result Size: 908 x 784

Get your own website

```
<!DOCTYPE html>
<html>
<body>

<h2>What Can JavaScript Do?</h2>

<p id="demo">JavaScript can change the style of an HTML element.</p>

<button type="button" onclick="document.getElementById('demo').style.fontSize='35px'">Click
Me!</button>

</body>
</html>
```

What Can JavaScript Do?

JavaScript can change the style of an HTML element.

Click Me!

Example: Hide HTML Elements

Run

```
<!DOCTYPE html>
<html>
<body>

<h2>What Can JavaScript Do?</h2>

<p id="demo">JavaScript can hide HTML elements.</p>

<button type="button" onclick="document.getElementById('demo').style.display='none'">Click
Me!</button>

</body>
</html>
```

What Can JavaScript Do?

JavaScript can hide HTML elements.

Click Me!

Run

```
<!DOCTYPE html>
<html>
<body>

<h2>What Can JavaScript Do?</h2>

<p id="demo">JavaScript can hide HTML elements.</p>

<button type="button" onclick="document.getElementById('demo').style.display='none'">Click
Me!</button>

</body>
</html>
```

What Can JavaScript Do?

Click Me!

Other Capabilities

- Respond to user events such as clicks, keypresses, and form submissions.
- Validate and process user input
- Fetch data from servers using AJAX
- Store and retrieve data on the client-side using local storage
- Create animations and transitions to enhance the user interface
- Access various browser APIs for features like geolocation, notifications, and more

Javascript Basics

- JavaScript has C-style syntax
  - Usual: variables, data types, operators, control structures (such as loops and conditionals), functions, and objects
  - Will cover shortly!

- JavaScript code is inserted between <script> and </script> tags
  - Can place any number of scripts in a HTML document.
  - Scripts can be placed in <body>, or in <head> section or both
  - Scripts can also be placed in external files and included in HTML
    - JavaScript files have the file extension .js
    - Useful when same code is used in many different web pages
    - Separates HTML and code → make them easier to read and maintain
    - Cached JavaScript files can speed up page loads

- An external script can be referenced in 3 different ways:
  - With a full URL (a full web address)
    - <script src="https://www.w3schools.com/js/myScript.js"></script>
  - With a file path (like /js/)
    - <script src="/js/myScript.js"></script>
  - Without any path
    - <script src="myScript.js"></script>
    - located in the same folder as the current page
  - You don't include <script> tags inside external files!

Run

```html
<!DOCTYPE html>
<html>
<body>

<h2>Demo JavaScript in Body</h2>

<p id="demo">A Paragraph.</p>

<button type="button" onclick="myFunction()">Try it</button>

<script>
function myFunction() {
  document.getElementById("demo").innerHTML = "Paragraph changed.";
}
</script>

</body>
</html>
```

Demo JavaScript in Body

Paragraph changed.

Try it

Run

```html
<!DOCTYPE html>
<html>
<body>

<h2>External JavaScript</h2>

<p id="demo">A Paragraph.</p>

<button type="button" onclick="myFunction()">Click Me</button>

<p>This example uses a full web URL to link to "myScript.js".</p>
<p>(myFunction is stored in "myScript.js")</p>

<script src="https://www.w3schools.com/js/myScript.js"></script>

</body>
</html>
```

External JavaScript

A Paragraph.

Click Me

This example uses a full web URL to link to "myScript.js".

(myFunction is stored in "myScript.js")

Variables and Operators

- JavaScript Variables can be declared in 4 ways: Automatically; Using var; Using let; Using const
  - Var is not used any more
  - Good practice to declare variables, so don't do it automatically
  - Always use const if the value should not be changed
- Operators:
  - Very similar to C (+, -, *, /, ++, –, **, &&, ! etc)
  - Comparison also similar to C (<, >, >=, !=, == etc)
  - "==" equal to vs "===" equal value and equal type

Run

```html
<!DOCTYPE html>
<html>
<body>
<h1>JavaScript Variables</h1>

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

<script>
let x = 5;
let y = 6;
let z = x * y;

const price1 = 10000;
const price2 = 20000;
let total = price1 + price2;
total += 50000;

let person = "John" + " " + "Doe";

document.getElementById("demo").innerHTML =
person + " has an id of " + z + " with a bank balance of " + total;
</script>

</body>
</html>
```

No explicit declaration of type! Interpreted based on context!

Run

```html
<!DOCTYPE html>
<html>
<body>
<h1>JavaScript Variables</h1>

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

<script>
let x = 5;
let y = 6;
let z = x * y;

const price1 = 10000;
const price2 = 20000;
let total = price1 + price2;
total += 50000;

let person = "John" + " " + "Doe";

document.getElementById("demo").innerHTML =
person + " has an id of " + z + " with a bank balance of " + total;
</script>

</body>
</html>
```

JavaScript Variables

John Doe has an id of 30 with a bank balance of 800(

No explicit declaration of type! Interpreted based on context!

Data Types

- JavaScript has 8 Datatypes:
  - String, Number, Bigint, Boolean, Undefined, Null, Symbol, Object
  - Object data type can contain:
    - An object, array or date
- JavaScript has dynamic types
  - Type of a variable can change during execution of the program
  - Same variable can be used to hold different data types
  - When adding a number and a string, JavaScript will treat the number as a string
  - Evaluates expressions from left to right

Run

```
<h2>JavaScript Data Types</h2>

<p>JavaScript has dynamic types. </p>

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

<script>
let x;          // Now x is undefined
x = 5;          // Now x is a Number
x = "John";     // Now x is a String

// Numbers:
x = 16;
let y = 123e-5;   // decimal in exponential, 0.00123

// Strings: double or single quotes are fine
let color = "Yellow";
let lastName = 'Johnson';

// Booleans
x = true;
y = false;

// Object:
const person = {firstName:"John", lastName:"Doe"};

// Array Object
const cars = ["Saab", "Volvo", "BMW"];

// Date object:
const date = new Date("2023-03-25");

x = 16 + 4 + "Volvo";
y = "Volvo" + 16 + 4;

document.getElementById("demo").innerHTML = x + "<br>" + y + "<br>"+ cars + "<br>" + date;
</script>

</body>
</html>
```

Run

```html
<h2>JavaScript Data Types</h2>

<p>JavaScript has dynamic types. </p>

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

<script>
let x;          // Now x is undefined
x = 5;          // Now x is a Number
x = "John";     // Now x is a String

// Numbers:
x = 16;
let y = 123e-5;   // decimal in exponential, 0.00123

// Strings: double or single quotes are fine
let color = "Yellow";
let lastName = 'Johnson';

// Booleans
x = true;
y = false;

// Object:
const person = {firstName:"John", lastName:"Doe"};

// Array Object
const cars = ["Saab", "Volvo", "BMW"];

// Date object:
const date = new Date("2023-03-25");

x = 16 + 4 + "Volvo";
y = "Volvo" + 16 + 4;

document.getElementById("demo").innerHTML = x + "<br>" + y + "<br>"+ cars + "<br>" + date;
</script>

</body>
</html>
```

JavaScript Data Types

JavaScript has dynamic types.

20Volvo
Volvo164
Saab,Volvo,BMW
Sat Mar 25 2023 05:30:00 GMT+0530 (India Standard Time)

```html
<!DOCTYPE html>
<html>
<body>

<h1>JavaScript Numbers</h1>
<h2>Integer and BigInt</h2>

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

<script>
let x = 9999999999999999;
let y = BigInt("9999999999999999");
document.getElementById("demo").innerHTML = x + "<br>" + y;
</script>

</body>
</html>
```

JavaScript Numbers

Integer and BigInt

10000000000000000
9999999999999999

Arrays

- An array is a special variable, which can hold more than one value
- Real strength of JavaScript arrays are the built-in array properties and methods
  - length property of an array returns the length of an array
  - add a new element to an array using the push() method:
  - pop() method removes the last element from an array:
  - sort() method sorts an array alphabetically:
  - reverse() method reverses the elements in an array.

```
const cars = ["Saab",
"Volvo", "BMW"];

const cars = [
  "Saab",
  "Volvo",
  "BMW"
];

const cars = [];
cars[0]= "Saab";
cars[1]= "Volvo";
cars[2]= "BMW";
```

Run

Result Siz

```
<!DOCTYPE html>
<html>
<body>
<h1>JavaScript Arrays</h1>

<p id="demo1"></p>
<p id="demo2"></p>

<script>

const cars = ["Saab", "Volvo", "BMW"];

cars.push("Honda");

cars.push("Toyota");

cars.pop();

let size = cars.length;

// First sort the array
cars.sort();
document.getElementById("demo1").innerHTML = cars + " " + size;

// Then reverse it:
cars.reverse();
document.getElementById("demo2").innerHTML = cars;
</script>

</body>
</html>
```

Run ❯

Result Siz

```
<!DOCTYPE html>
<html>
<body>
<h1>JavaScript Arrays</h1>

<p id="demo1"></p>
<p id="demo2"></p>

<script>

const cars = ["Saab", "Volvo", "BMW"];

cars.push("Honda");

cars.push("Toyota");

cars.pop();

let size = cars.length;

// First sort the array
cars.sort();
document.getElementById("demo1").innerHTML = cars + " " + size;

// Then reverse it:
cars.reverse();
document.getElementById("demo2").innerHTML = cars;
</script>

</body>
</html>
```

JavaScript Arrays

BMW,Honda,Saab,Volvo 4

Volvo,Saab,Honda,BMW

Functions

- Defined with “function” keyword, followed by a name, followed by parentheses ()
- Code to be executed, by the function, is placed inside curly brackets {}

```html
<!DOCTYPE html>
<html>
<body>

<h1>JavaScript Functions</h1>

<p>Invoke (call) a function that converts from Fahrenheit to
Celsius:</p>
<p id="demo"></p>

<script>
function toCelsius(f) {
  return (5/9) * (f-32);
}

let value = toCelsius(77);
document.getElementById("demo").innerHTML = value;
</script>

</body>
</html>
```

JavaScript Functions

Invoke (call) a function that converts from Fahrenheit to Celsius:

25

Objects

Object	Properties	Methods
	car.name = Fiat	car.start()
	car.model = 500	car.drive()
	car.weight = 850kg	car.brake()
	car.color = white	car.stop()

All cars have the same **properties**, but the property **values** differ from car to car.

All cars have the same **methods**, but the methods are performed **at different times**.

- Objects are variables but can contain many values and also methods!
  - values are written as name:value pairs (called properties)
  - Can be accessed via objectName.propertyName or objectName["propertyName"]
  - Methods are actions that can be performed on objects
    - Are stored in properties as function definitions
    - “this” keyword refers to an object
      - which object depends on how this is being invoked (used or called)?
  - When a JavaScript variable is declared with the keyword "new", variable is created as an object

```html
<!DOCTYPE html>
<html>
<body>

<h2>JavaScript Objects</h2>

<p>There are two different ways to access an object property.</p>

<p>You can use person.property or person["property"].</p>

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

<script>
// Create an object:
const person = {
  firstName: "John",
  lastName : "Doe",
  id     :  5566
};

// Display some data from the object:
document.getElementById("demo").innerHTML =
person.firstName + " " + person["lastName"];
</script>

</body>
</html>
```

JavaScript Objects

There are two different ways to access an object property.

You can use person.property or person["property"].

John Doe

```html
<!DOCTYPE html>
<html>
<body>

<h2>JavaScript Objects</h2>
<p>Creating a JavaScript Object:</p>

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

<script>
const person = new Object();
person.firstName = "John";
person.lastName = "Doe";
person.age = 50;
person.eyeColor = "blue";

document.getElementById("demo").innerHTML =
person.firstName + " is " + person.age + " years old.";
</script>

</body>
</html>
```

JavaScript Objects

Creating a JavaScript Object:

John is 50 years old.

```html
<!DOCTYPE html>
<html>
<body>

<h2>JavaScript Objects</h2>
<p>An object method is a function definition, stored as a property
value.</p>

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

<script>
// Create an object:
const person = {
  firstName: "John",
  lastName: "Doe",
  id: 5566,
  fullName: function() {
    return this.firstName + " " + this.lastName;
  }
};

// Display data from the object:
document.getElementById("demo").innerHTML = person.fullName();
</script>

</body>
</html>
```

JavaScript Objects

An object method is a function definition, stored as a property valu

John Doe

Javascript Errors

- “try” statement allows you to define a block of code to be tested for errors while it is being executed.
- “catch” statement allows you to define a block of code to be executed, if an error occurs in the try block
  - JavaScript statements try and catch come in pairs
- “throw” statement allows you to create a custom error

Run

```
<!DOCTYPE html>
<html>
```

```
<!DOCTYPE html>
<html>
<body>

<h2>JavaScript const</h2>

<p>Declaring a constant array does NOT make the elements
unchangeable:</p>

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

<script>
// Create an Array:
const cars = ["Saab", "Volvo", "BMW"];

// Change an element:
cars[0] = "Toyota";

// Add an element:
cars.push("Audi");

// Display the Array:
document.getElementById("demo").innerHTML = cars;
</script>

</body>
</html>
```

JavaScript const

Declaring a constant array does NOT make the elements unchangeable:

Toyota,Volvo,BMW,Audi

```html
<h2>JavaScript const</h2>

<p>Declaring a constant array does NOT make the elements
unchangeable:</p>

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

<script>
// Create an Array:
const cars = ["Saab", "Volvo", "BMW"];

// Change an element:
cars[0] = "Toyota";

// Add an element:
cars.push("Audi");

// Display the Array:
document.getElementById("demo").innerHTML = cars;

try {
  cars = ["Toyota", "Volvo", "Audi"];
}
catch (err) {
  document.getElementById("demo").innerHTML = err;
}

</script>

</body>
</html>
```

JavaScript const

Declaring a constant array does NOT make the elements unchangeable:

TypeError: invalid assignment to const 'cars'

```html
<!DOCTYPE html>
<html>
<body>

<h2>JavaScript try catch</h2>

<p>Please input a number between 5 and 10:</p>

<input id="demo" type="text">
<button type="button" onclick="myFunction()">Test Input</button>
<p id="p01"></p>

<script>
function myFunction() {
  const message = document.getElementById("p01");
  message.innerHTML = "";
  let x = document.getElementById("demo").value;
  try {
    if(x.trim() == "")  throw "empty";
    if(isNaN(x)) throw "not a number";
    x = Number(x);
    if(x < 5)  throw "too low";
    if(x > 10)   throw "too high";
  }
  catch(err) {
    message.innerHTML = "Input is " + err;
  }
}
</script>

</body>
</html>
```

JavaScript try catch

Please input a number between 5 and 10:

3 Test Input

Input is too low

Classes

- Classes are not objects, they are just a template for objects!
- Use keyword class to create a class
  - Always add a method named constructor() (same exact name)
    - Executed automatically when a new object is created and is used to initialize object properties

```html
<!DOCTYPE html>
<html>
<body>
<h1>JavaScript Classes</h1>
<p>Creating two car objects from a car class:</p>

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

<script>
class Car {
  constructor(name, year) {
    this.name = name;
    this.year = year;
  }
}

const myCar1 = new Car("Ford", 2014);
const myCar2 = new Car("Audi", 2019);

document.getElementById("demo").innerHTML =
myCar1.name + " " + myCar2.name;
</script>

</body>
</html>
```

JavaScript Classes

Creating two car objects from a car class:

Ford Audi

```html
<!DOCTYPE html>
<html>
<body>
<h1>JavaScript Class Method</h1>
<p>Pass a parameter into the "age()" method.</p>

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

<script>
class Car {
  constructor(name, year) {
    this.name = name;
    this.year = year;
  }
  age(x) {
    return x - this.year;
  }
}

const date = new Date();
let year = date.getFullYear();

const myCar = new Car("Ford", 2014);
document.getElementById("demo").innerHTML=
"My car is " + myCar.age(year) + " years old.";
</script>

</body>
</html>
```

JavaScript Class Method

Pass a parameter into the "age()" method.

My car is 9 years old.

Conditionals

```html
<!DOCTYPE html>
<html>
<body>

<h2>JavaScript if .. else</h2>

<p>A time-based greeting:</p>

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

<script>
const time = new Date().getHours();
let greeting;
if (time < 10) {
  greeting = "Good morning";
} else if (time < 20) {
  greeting = "Good day";
} else {
  greeting = "Good evening";
}
document.getElementById("demo").innerHTML = greeting;
</script>

</body>
</html>
```

JavaScript if .. else

A time-based greeting:

Good day

Loops

```html
<!DOCTYPE html>
<html>
<body>

<h2>JavaScript For Loop</h2>

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

<script>
const cars = ["BMW", "Volvo", "Saab", "Ford"];
let i, len, text;
for (i = 0, len = cars.length, text = ""; i < len; i++) {
  text += cars[i] + "<br>";
}
document.getElementById("demo").innerHTML = text;
</script>

</body>
</html>
```

JavaScript For Loop

BMW
Volvo
Saab
Ford

```html
<!DOCTYPE html>
<html>
<body>

<h2>JavaScript For In Loop</h2>
<p>The for in statement loops through the properties of an object:</p>

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

<script>
const person = {fname:"John", lname:"Doe", age:25};

let txt = "";
for (let x in person) {
  txt += person[x] + " ";
}

document.getElementById("demo").innerHTML = txt;
</script>

</body>
</html>
```

JavaScript For In Loop

The for in statement loops through the properties of an object:

John Doe 25

For in loop

- for in statement loops through the properties of an Object
- Each iteration returns a key x
- key is used to access the value of the key
- value of the key is person[x]

```html
<!DOCTYPE html>
<html>
<body>

<h2>JavaScript For Of Loop</h2>
<p>The for of statement loops through the values of any iterable object:</p>

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

<script>
const cars = ["BMW", "Volvo", "Mini"];

let text = "";
for (let x of cars) {
  text += x + "<br>";
}

document.getElementById("demo").innerHTML = text;
</script>

</body>
</html>
```

JavaScript For Of Loop

The for of statement loops through the values of any iterable object:

BMW
Volvo
Mini

for of statement loops through the values of an iterable object

While

```html
<!DOCTYPE html>
<html>
<body>

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

<script>
const cars = ["BMW", "Volvo", "Saab", "Ford"];

let i = 0;
let text = "";
while (cars[i]) {
  text += cars[i] + "<br>";
  i++;
}

document.getElementById("demo").innerHTML = text;
</script>

</body>
</html>
```

BMW
Volvo
Saab
Ford

Display

- JavaScript can "display" data in different ways:
  - Writing into an HTML element, using innerHTML
  - Writing into the HTML output using document.write()
  - Writing into an alert box, using window.alert()
  - Writing into the browser console, using console.log()

Run

```html
<!DOCTYPE html>
<html>
<body>

<h2>My First Web Page</h2>
<p>My First Paragraph.</p>

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

<script>
document.getElementById("demo").innerHTML = 1 + 1;
</script>

<script>
document.write(2 + 3);
</script>

<br><br>

<button type="button" onclick="document.write(4 + 5)">Try it</button>

<script>
alert(6 + 7);
</script>


</body>
</html>
```

My First Web Page

My First Paragraph.

2

5

Try it

www.w3schools.com

13

OK

Using document.write() after an HTML document is loaded, will delete all existing HTML.
Should only be used for testing.
Try it (document.write) , will overwrite everything and show only number 9

Run ❯

Result Size: 908 x 534

Get your own website

```
<!DOCTYPE html>
<html>
<body>

<h2>Activate Debugging</h2>

<p>F12 on your keyboard will activate debugging.</p>
<p>Then select "Console" in the debugger menu.</p>
<p>Then click Run again.</p>

<script>
console.log(5 + 6);
</script>

</body>
</html>
```

Activate Debugging

F12 on your keyboard will activate debugging.

Then select "Console" in the debugger menu.

Then click Run again.

Inspector | Console | Debugger | Network | Style Editor | Performance | Memory | Storage | Accessibility | Application

Filter Output | Errors | Warnings | Logs | Info | Debug | CSS | XHR | Requests

Request to access cookie or storage on "https://googleads.g.doubleclick.net/xbbe/pixel?d=CNurWxDLw_UBGPv4s4ICMAE&v=APEucNVsIU0814lg-bJxhDDbNLPlN6XlTpVx-5mka2l0vtFhNXmUWEnPlnooi13FarxCDfk4vSQ4QgLUl_gdNwP4ufoCI7mB4MdmmbDtqR-5VlKQFETtuQw" was blocked because it came from a tracker and content blocking is enabled. [Learn More]

WEBGL_debug_renderer_info is deprecated in Firefox and will be removed. Please use RENDERER. … _-LR3WxpzwV0Gscftq1A_DlowstvxoTnWWhwY4Ow.js line 2 > eval line 3982 > eval line 1 > eval line 1 > eval:1:115

A resource is blocked by OpaqueResponseBlocking, please check browser console for details. gen_204

A resource is blocked by OpaqueResponseBlocking, please check browser console for details. sodar

11 tryit.asp:13:9

A resource is blocked by OpaqueResponseBlocking, please check browser console for details. interaction

Event Handling

- Handling events is a crucial aspect of web development
- JavaScript provides mechanisms for interacting with and responding to events
  - <element event='some JavaScript'>
  - <element event="some JavaScript">

Event Types

UI Events

- Click
- Double click
- Mouseover
- Mouseout
- Keydown, keyup
- Focus, blur

Form Events:

- Submit
- Change
- Input

Document/Window Events:

- Load
- Resize
- Scroll

an `onclick` attribute (with code), is added to a `<button>` element:

```html
<!DOCTYPE html>
<html>
<body>
<h1>JavaScript HTML Events</h1>
<h2>The onclick Attribute</h2>

<button onclick="this.innerHTML=Date()">The time is?</button>

</body>
</html>
```

JavaScript HTML Events

The onclick Attribute

The time is?


```html
<!DOCTYPE html>
<html>
<body>
<h1>JavaScript HTML Events</h1>
<h2>The onclick Attribute</h2>

<button onclick="this.innerHTML=Date()">The time is?</button>

</body>
</html>
```


JavaScript HTML Events

The onclick Attribute

Thu Dec 28 2023 16:39:48 GMT+0530 (India Standard Time)

```html
<!DOCTYPE html>
<html>
<body>


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

<button id="myButton">Click me</button>


<script>
    document.getElementById("myButton").addEventListener("click",
function() {
        alert("Button clicked!");
    });
</script>


</body>
</html>
```

Click me

www.w3schools.com

Button clicked!

OK

Demo

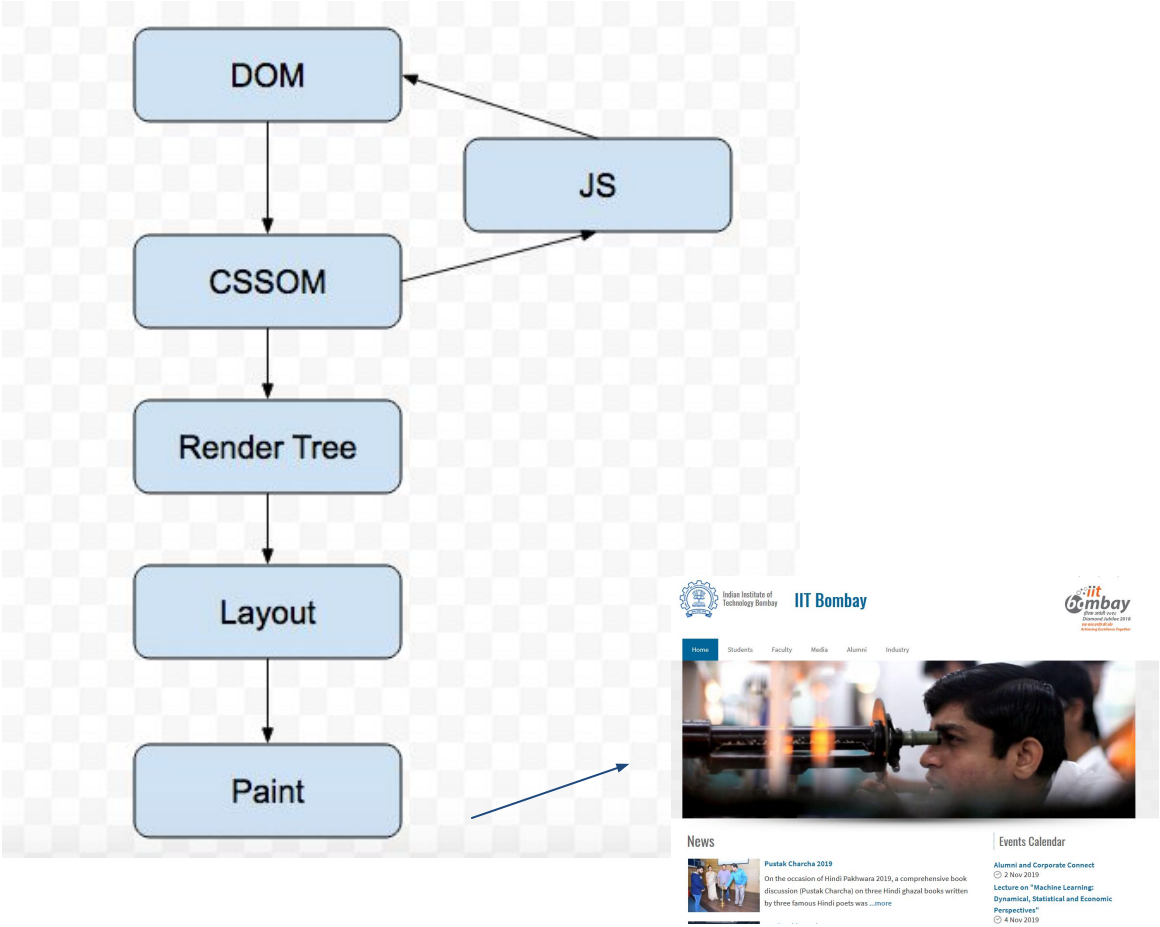
events.html

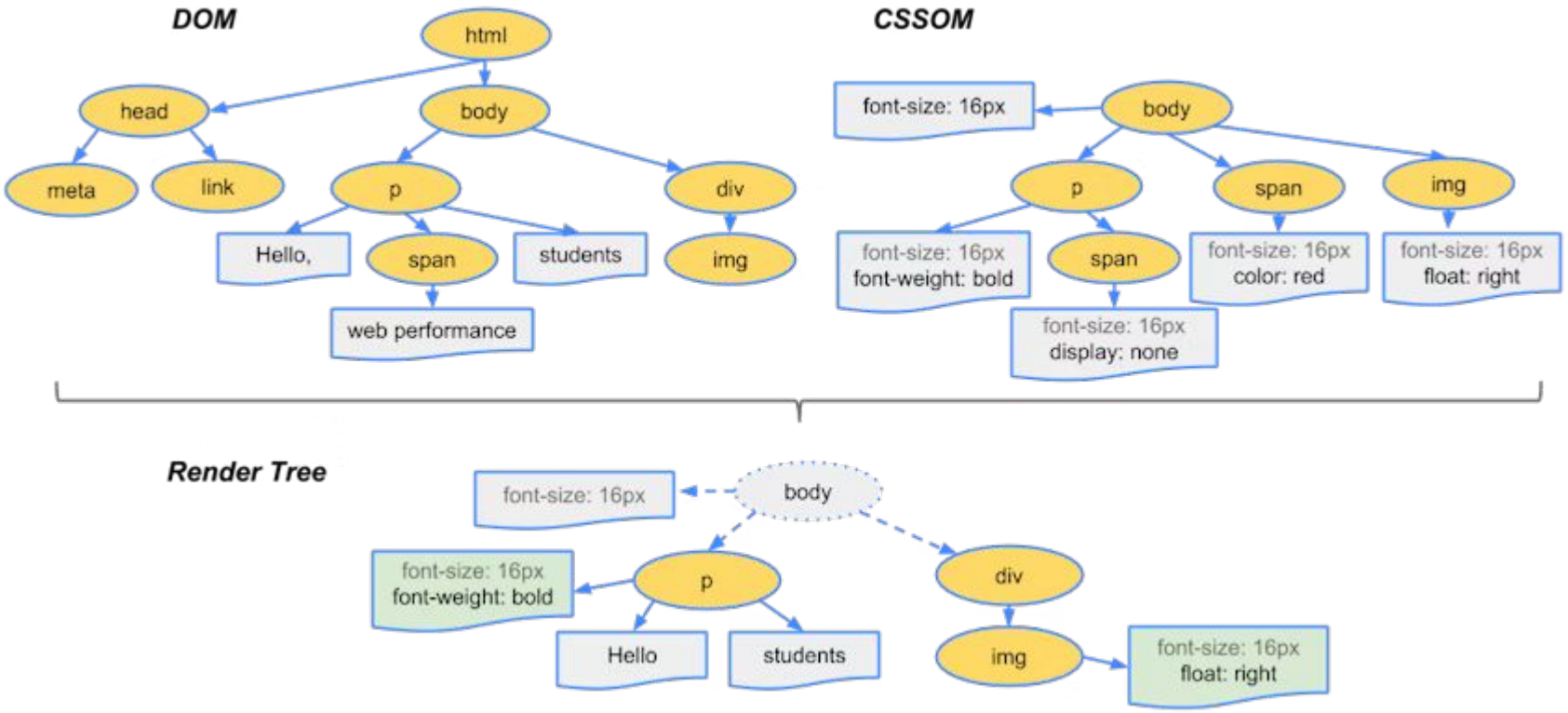
DOM

- JavaScript and Document Object Model (DOM) are closely intertwined
- Help building interactive and dynamic web pages

Document Object Model (DOM)

- An application programming interface (api) that extracts a tree structure out of HTML
  - Each node is an object representing a part of the document
  - Objects can be manipulated programmatically via JavaScript


DOM
JS
CSSOM
Render Tree
Layout
Paint
Indian Institute of Technology Bombay
IIT Bombay
Home
Students
Faculty
Media
Alumni
Industry
News
Pustak Charcha 2019
On the occasion of Hindi Pakhwara 2019, a comprehensive book discussion (Pustak Charcha) on three Hindi ghazal books written by three famous Hindi poets was ...more
Events Calendar
Alumni and Corporate Connect
2 Nov 2019
Lecture on "Machine Learning: Dynamical, Statistical and Economic Perspectives"
4 Nov 2019


DOM
html
head
body
meta
link
p
div
Hello,
span
students
img
web performance
CSSOM
font-size: 16px
body
p
span
img
font-size: 16px
font-weight: bold
span
font-size: 16px
color: red
font-size: 16px
float: right
font-size: 16px
display: none
Render Tree
font-size: 16px
body
font-size: 16px
font-weight: bold
p
div
Hello
students
img
font-size: 16px
float: right

DOM Manipulation

- JavaScript can change all the HTML elements in the page
- JavaScript can change all the HTML attributes in the page
- JavaScript can change all the CSS styles in the page
- JavaScript can remove existing HTML elements and attributes
- JavaScript can add new HTML elements and attributes
- JavaScript can react to all existing HTML events in the page
- JavaScript can create new HTML events in the page

- In DOM, all HTML elements are defined as objects
- Example: document.getElementById("demo").innerHTML = "Hello World!";
  - Document is the object; getElementById is a method of document object, while innerHTML is a property

Example: Change HTML Content

Run

```
<!DOCTYPE html>
<html>
<body>

<h2>What Can JavaScript Do?</h2>

<p id="demo">JavaScript can change HTML content.</p>

<button type="button" onclick='document.getElementById("demo").innerHTML = "Hello
JavaScript!"'>Click Me!</button>

</body>
</html>
```

What Can JavaScript Do?

JavaScript can change HTML content.

Click Me!

Run

```
<!DOCTYPE html>
<html>
<body>

<h2>What Can JavaScript Do?</h2>

<p id="demo">JavaScript can change HTML content.</p>

<button type="button" onclick='document.getElementById("demo").innerHTML = "Hello
JavaScript!"'>Click Me!</button>

</body>
</html>
```

What Can JavaScript Do?

Hello JavaScript!

Click Me!

Example: Change Attribute Values

Run

Result Size: 908 x 784

```
<!DOCTYPE html>
<html>
<body>

<h2>What Can JavaScript Do?</h2>

<p>JavaScript can change HTML attribute values.</p>

<p>In this case JavaScript changes the value of the src (source) attribute of an image.</p>

<button onclick="document.getElementById('myImage').src='pic_bulbon.gif'">Turn on the
light</button>

<img id="myImage" src="pic_bulboff.gif" style="width:100px">

<button onclick="document.getElementById('myImage').src='pic_bulboff.gif'">Turn off the
light</button>

</body>
</html>
```

What Can JavaScript Do?

JavaScript can change HTML attribute values.

In this case JavaScript changes the value of the src (source) attribute of an image.

Turn on the light Turn off the light

Run

Result Size: 908 x 784

```
<!DOCTYPE html>
<html>
<body>

<h2>What Can JavaScript Do?</h2>

<p>JavaScript can change HTML attribute values.</p>

<p>In this case JavaScript changes the value of the src (source) attribute of an image.</p>

<button onclick="document.getElementById('myImage').src='pic_bulbon.gif'">Turn on the
light</button>

<img id="myImage" src="pic_bulboff.gif" style="width:100px">

<button onclick="document.getElementById('myImage').src='pic_bulboff.gif'">Turn off the
light</button>

</body>
</html>
```

What Can JavaScript Do?

JavaScript can change HTML attribute values.

In this case JavaScript changes the value of the src (source) attribute of an image.

Turn on the light Turn off the light

Run

Result Size: 908 x 784

```
<!DOCTYPE html>
<html>
<body>

<h2>What Can JavaScript Do?</h2>

<p>JavaScript can change HTML attribute values.</p>

<p>In this case JavaScript changes the value of the src (source) attribute of an image.</p>

<button onclick="document.getElementById('myImage').src='pic_bulbon.gif'">Turn on the
light</button>

<img id="myImage" src="pic_bulboff.gif" style="width:100px">

<button onclick="document.getElementById('myImage').src='pic_bulboff.gif'">Turn off the
light</button>

</body>
</html>
```

What Can JavaScript Do?

JavaScript can change HTML attribute values.

In this case JavaScript changes the value of the src (source) attribute of an image.

Turn on the light Turn off the light

Example: Change HTML Styles (CSS)

Run

```
<!DOCTYPE html>
<html>
<body>

<h2>What Can JavaScript Do?</h2>

<p id="demo">JavaScript can change the style of an HTML element.</p>

<button type="button" onclick="document.getElementById('demo').style.fontSize='35px'">Click
Me!</button>

</body>
</html>
```

What Can JavaScript Do?

JavaScript can change the style of an HTML element.

Click Me!

Run

Result Size: 908 x 784

Get your own website

```
<!DOCTYPE html>
<html>
<body>

<h2>What Can JavaScript Do?</h2>

<p id="demo">JavaScript can change the style of an HTML element.</p>

<button type="button" onclick="document.getElementById('demo').style.fontSize='35px'">Click
Me!</button>

</body>
</html>
```

What Can JavaScript Do?

JavaScript can change the style of an HTML element.

Click Me!

Example: Hide HTML Elements

Run

```
<!DOCTYPE html>
<html>
<body>

<h2>What Can JavaScript Do?</h2>

<p id="demo">JavaScript can hide HTML elements.</p>

<button type="button" onclick="document.getElementById('demo').style.display='none'">Click
Me!</button>

</body>
</html>
```

What Can JavaScript Do?

JavaScript can hide HTML elements.

Click Me!

Run

```
<!DOCTYPE html>
<html>
<body>

<h2>What Can JavaScript Do?</h2>

<p id="demo">JavaScript can hide HTML elements.</p>

<button type="button" onclick="document.getElementById('demo').style.display='none'">Click
Me!</button>

</body>
</html>
```

What Can JavaScript Do?

Click Me!

Form Manipulation

```html
<!DOCTYPE html>
<html>
<body>
<h1>JavaScript HTML Events</h1>
<h2>The oninput Attribute</h2>

Enter your name: <input type="text" id="fname" oninput="upperCase()">
<p>When you write in the input field, a function is triggered to transform the
input to upper case.</p>

<script>
function upperCase() {
  const x = document.getElementById("fname");
  x.value = x.value.toUpperCase();
}
</script>

</body>
</html>
```

JavaScript HTML Events

The oninput Attribute

Enter your name: KAMESWARI

When you write in the input field, a function is triggered to transform the input to upper case.

Mouse Interactions

Run

```html
<!DOCTYPE html>
<html>
<body>
<h1>JavaScript HTML Events</h1>
<h2>The onmouseover Attribute</h2>

<div onmouseover="mOver(this)" onmouseout="mOut(this)"
style="background-color:#D94A38;width:120px;height:20px;padding:40px;">
Mouse Over Me</div>

<script>
function mOver(obj) {
  obj.innerHTML = "Thank You"
}

function mOut(obj) {
  obj.innerHTML = "Mouse Over Me"
}
</script>

</body>
</html>
```

Result

JavaScript HTML Events

The onmouseover Attribute

Thank You

New Element

```html
<!DOCTYPE html>
<html>
<body>

<h2>JavaScript HTML DOM</h2>
<p>Add a new HTML Element.</p>

<div id="div1">
<p id="p1">This is a paragraph.</p>
<p id="p2">This is another paragraph.</p>
</div>

<script>
const para = document.createElement("p");
const node = document.createTextNode("This is new.");
para.appendChild(node);
const element = document.getElementById("div1");
element.appendChild(para);
</script>

</body>
</html>
```

JavaScript HTML DOM

Add a new HTML Element.

This is a paragraph.

This is another paragraph.

This is new.

Explanation

- Create a new <p> element:
  - const para = document.createElement("p");
- Add text to the <p> element, create a text node first
  - const node = document.createTextNode("This is a new paragraph.");
- Append text node to the <p> element:
  - para.appendChild(node);
- Append new element to an existing element
  - Find element via
    - const element = document.getElementById("div1");
  - Appends via
    - element.appendChild(para);

Noscript

The HTML <noscript> tag defines an alternate content to be displayed to users that have disabled scripts in their browser or have a browser that doesn't support scripts:

Example:

<noscript>Sorry, your browser does not support JavaScript!</noscript>

References

- Javascript in depth: https://www.w3schools.com/js/default.asp