Learn with Rabbil

JavaScript Fundamental

JavaScript (js) is a light-weight programming language which is used for web

Learn with Rabbil

Tools You Need

- Your Favorite Code Editor
- And Your Favorite Browser

Learn with Rabbil

MY FIRST JS

Javascript example is easy to code

JavaScript provides 3 places to put the JavaScript code: within body tag, within head tag and external JavaScript file

document.write() function is used to display dynamic content through JavaScript.

index.html

```
10   document.write("Hello JS");
```

Learn with Rabbil

JS COMMENT

The JavaScript comments are meaningful way to deliver message

It is used to add information about the code, warnings or suggestions

The JavaScript comment is ignored by the JavaScript engine

index.html

```
10  // It is single line comment
11  document.write("Hello JS");
12  /* It is multi line comment.
13  It will not be displayed */
```

Learn with Rabbil

JS VARIABLES

A JavaScript variable is simply a name of storage location

Name must start with a letter (a to z or A to Z), underscore(_), or dollar($) sign

After first letter we can use digits (0 to 9), for example value1.

JavaScript variables are case sensitive, for example x and X are different variables.

p1.html

```
8       var x = 10;
9       var y = 20;
10      var z=x+y;
11      document.write(z);
```

Learn with Rabbil

JAVASCRIPT DATA TYPES

JavaScript provides different data types to hold different types of values. There are two types of data types in JavaScript.

- Primitive data type
- Non-primitive (reference) data type

Learn with Rabbil

PRIMITIVE DATA TYPE

There are five types of primitive data types in JavaScript.

Data Type	Description
String	represents sequence of characters e.g. "hello"
Number	represents numeric values e.g. 100
Boolean	represents boolean value either false or true
Undefined	represents undefined value
Null	represents null i.e. no value at all

Learn with Rabbil

NON-PRIMITIVE DATA TYPE

The non-primitive data types are as follows

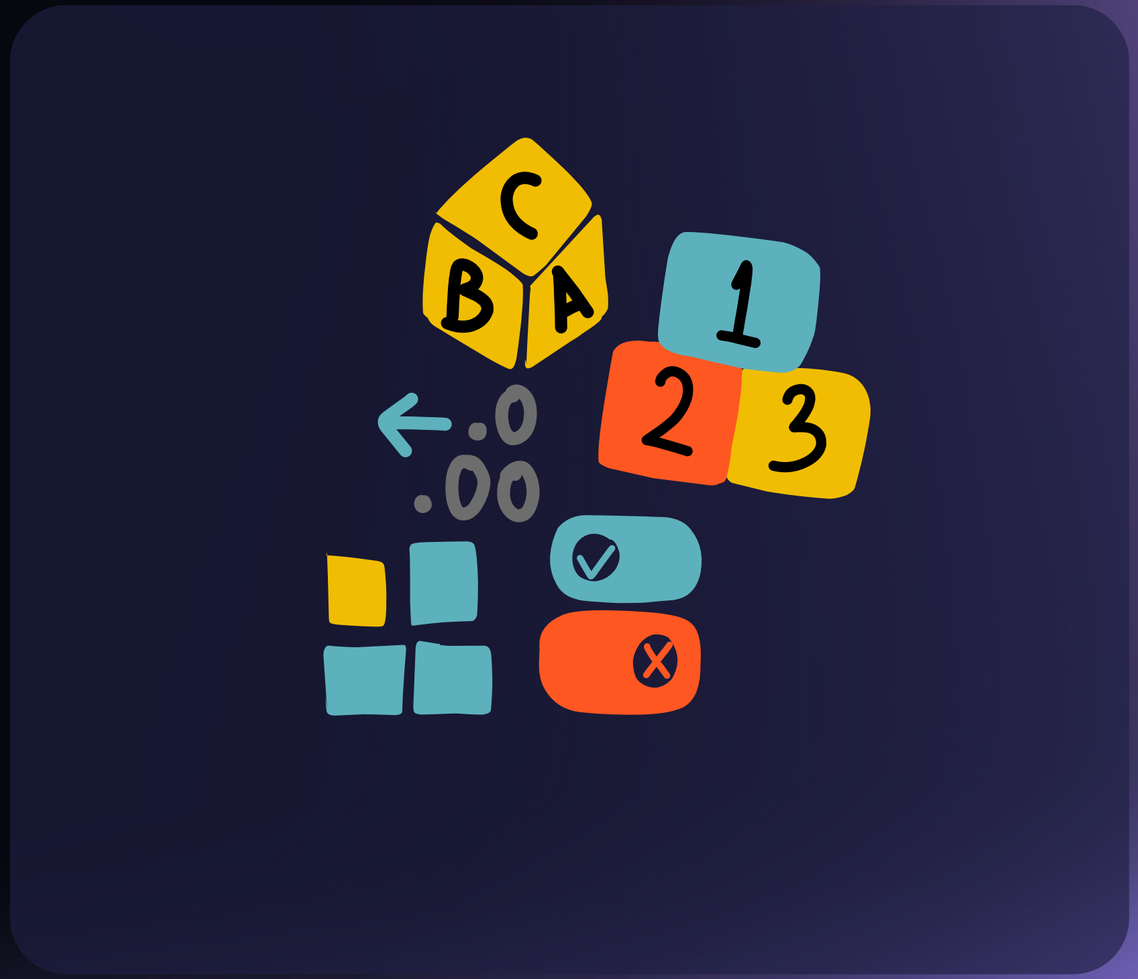
Data Type	Description
Object	represents instance through which we can access members
Array	represents group of similar values
RegExp	represents regular expression

Learn with Rabbil

JAVASCRIPT OPERATORS

JavaScript operators are symbols that are used to perform operations on operands

- Arithmetic Operators
- Comparison (Relational) Operators
- Bitwise Operators
- Logical Operators
- Assignment Operators
- Special Operators



Learn with Rabbil

JAVASCRIPT ARITHMETIC OPERATORS

Arithmetic operators are used to perform arithmetic operations on the operands. The following operators are known as JavaScript arithmetic operators.

Operator	Description	Example
+	Addition	10+20 = 30
-	Subtraction	20-10 = 10
*	Multiplication	10*20 = 200
/	Division	20/10 = 2
%	Modulus (Remainder)	20%10 = 0
++	Increment	var a=10; a++; Now a = 11
--	Decrement	var a=10; a--; Now a = 9

Learn with Rabbil

JAVASCRIPT COMPARISON OPERATORS

The JavaScript comparison operator compares the two operands. The comparison operators are as follows:

Operator	Description	Example
==	Is equal to	10==20 = false
===	Identical (equal and of same type)	10==20 = false
!=	Not equal to	10!=20 = true
!==	Not Identical	20!==20 = false
>	Greater than	20>10 = true
>=	Greater than or equal to	20>=10 = true
<	Less than	20<10 = false
<=	Less than or equal to	20<=10 = false

Learn with Rabbil

JAVASCRIPT BITWISE OPERATORS

The bitwise operators perform bitwise operations on operands. The bitwise operators are as follows:

Operator	Description	Example
&	Bitwise AND	(10==20 & 20==33) = false
|	Bitwise OR	(10==20 | 20==33) = false
^	Bitwise XOR	(10==20 ^ 20==33) = false
~	Bitwise NOT	(~10) = -10
<<	Bitwise Left Shift	(10<<2) = 40
>>	Bitwise Right Shift	(10>>2) = 2
>>>	Bitwise Right Shift with Zero	(10>>>2) = 2

Learn with Rabbil

JAVASCRIPT LOGICAL OPERATORS

The following operators are known as JavaScript logical operators

Operator	Description	Example
&&	Logical AND	(10==20 && 20==33) = false
||	Logical OR	(10==20 || 20==33) = false
!	Logical Not	!(10==20) = true

Learn with Rabbil

JAVASCRIPT ASSIGNMENT OPERATORS

The following operators are known as JavaScript assignment operators

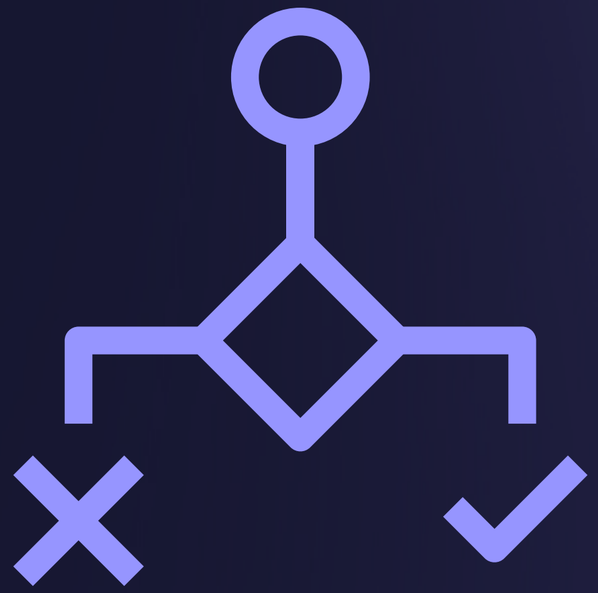
Operator	Description	Example
=	Assign	10+10 = 20
+=	Add and assign	var a=10; a+=20; Now a = 30
-=	Subtract and assign	var a=20; a-=10; Now a = 10
*=	Multiply and assign	var a=10; a*=20; Now a = 200
/=	Divide and assign	var a=10; a/=2; Now a = 5
%=	Modulus and assign	var a=10; a%=2; Now a = 0

Learn with Rabbil

JAVASCRIPT IF-ELSE

The JavaScript if-else statement is used to execute the code whether condition is true or false.

1. If Statement
2. If else statement
3. if else if statement



Learn with Rabbil

JAVASCRIPT IF STATEMENT

It evaluates the content only if expression is true.

index.html

```
10  var age=20;
11  if(age>18){
12  document.write("You are adult");
13  }
```

Learn with Rabbil

IF...ELSE STATEMENT

It evaluates the content whether condition is true of false.

index.html

```
10  var age=20;
11  if(age>18){
12      document.write("You are adult");
13  }
14  else{
15      document.write("You are child");
16  }
```

Learn with Rabbil

IF...ELSE IF STATEMENT

It evaluates the content only if expression is true from several expressions.

index.html

```
10  var a=20;
11  if(a==10){
12      document.write("a is equal to 10");
13  }
14  else if(a==15){
15      document.write("a is equal to 15");
16  }
17  else if(a==20){
18      document.write("a is equal to 20");
19  }
20  else{
21      document.write("a is not equal to 10, 15 or 20");
22  }
```

Learn with Rabbil

JAVASCRIPT SWITCH

The JavaScript switch statement is used to execute one code from multiple expressions. It is just like else if statement

index.html

```
12  switch(grade){
13      case 'A':
14      result="A Grade";
15      break;
16      case 'B':
17      result="B Grade";
18      break;
19      case 'C':
20      result="C Grade";
21      break;
22      default:
23      result="No Grade";
24  }
```

Learn with Rabbil

JS LOOP

The JavaScript loops are used to iterate the piece of code using for, while, do while or for-in loops. It makes the code compact. It is mostly used in array.

- FOR LOOP
- FOR IN LOOP
- WHILE LOOP
- DO WHILE LOOP

Learn with Rabbil

JAVASCRIPT FOR LOOP

The JavaScript for loop iterates the elements for the fixed number of times. It should be used if number of iteration is known.

index.html

```
10  for (i=1; i<=5; i++){
11      document.write(i + "<br/>")
12  }
```

Learn with Rabbil

JAVASCRIPT WHILE LOOP

The JavaScript while loop iterates the elements for the infinite number of times. It should be used if number of iteration is not known.

index.html

```
10  var i=11;
11  while (i<=15) {
12      document.write(i + "<br/>");
13      i++;
14  }
```

Learn with Rabbil

JAVASCRIPT DO WHILE LOOP

The JavaScript do while loop iterates the elements for the infinite number of times like while loop. But, code is executed at least once whether condition is true or false.

index.html

```
10  var i=21;
11  do{
12      document.write(i + "<br/>");
13      i++;
14  }
15  while (i<=25);
```

Learn with Rabbil

JavaScript Functions

01 Code Reusability

Functions are defined only once and can be invoked many times, like in other programming languages.

02 Less Code

It saves a lot of code because you don't need to write the logic many times. By the use of function, you can write the logic only once and reuse it.

03 Easy to understand

Functions separate the programming logic. So it is easier to understand the flow of the application because every logic is divided in the form of functions.

Learn with Rabbil

JAVASCRIPT FUNCTIONS

JavaScript functions are used to perform operations. We can call JavaScript function many times to reuse the code.

index.html

```
10  function msg(){
11      alert("hello! this is message");
12  }
```

Learn with Rabbil

FUNCTION ARGUMENTS

We can call function by passing arguments. Let’s see the example of function that has one argument.

index.html

```
10   function msg(text){
11       alert(text);
12   }
```

Learn with Rabbil

FUNCTION RETURN

We can call function that returns a value and use it in our program.

index.html

```
10  function addTwoNum(){
11      var x=10;
12      var y=20;
13      var z=x+y;
14      return z
15  }
```

Learn with Rabbil

JAVASCRIPT OBJECTS

A javaScript object is an entity having state and behavior (properties and method)

index.html

```
11  var Person={
12      Name:"Rabbil Hasan",
13      Age:30,
14      City:"Dhaka",
15      Country:"Bangladesh"
16  }
```

Learn with Rabbil

JAVASCRIPT ARRAY

JavaScript array is an object that represents a collection of similar type of elements

index.html

```
11  var city=["Dhaka","Rajshahi","Rangpur"];
12
13  for (i=0;i<city.length;i++){
14
15  document.write(city[i] + "<br/>");
16
17  }
```

Learn with Rabbil

JAVASCRIPT ARRAY CONCAT()

The JavaScript array concat() method combines two or more arrays and returns a new string.

index.html

```
11  var arr1=["A","B","C"];
12  var arr2=["D","E","F"];
13  var result=arr1.concat(arr2);
14  document.writeln(result);
```

Learn with Rabbil

JAVASCRIPT ARRAY FROM()

The from() method creates a new array that holds the shallow copy from an array or iterable object.

When applied to a string, each word gets converted to an array element in the new array

index.html

```
11  var arr=Array.from("RABBIL HASAN");
12  document.write(arr);
13
```

Learn with Rabbil

JAVASCRIPT ARRAY FILTER()

The JavaScript array filter() method filter and extract the element of an array that satisfying the provided condition.

index.html

```
11  var marks=[50,40,45,37,20];
12
13  function check(value)
14  {
15    return value>30;
16  }
17
18  document.writeln(marks.filter(check));
```

Learn with Rabbil

JAVASCRIPT ARRAY FIND()

The JavaScript array find() method returns the first element of the given array that satisfies the provided function condition.

index.html

```
11  var arr=[5,22,19,25,34];
12  var result=arr.find(x=>x>20);
13  document.writeln(result)
```

Learn with Rabbil

JAVASCRIPT ARRAY FINDINDEX()

The JavaScript array findIndex() method returns the index of first element of the given array that satisfies the provided function condition. It returns -1, if no element satisfies the condition.

index.html

```
11  var arr=[5,22,19,25,34];
12  var result=arr.findIndex(x=>x>20);
13  document.writeln(result
```

Learn with Rabbil

JAVASCRIPT ARRAY FOREACH()

The JavaScript array forEach() method is used to invoke the specified function once for each array element.

index.html

```
11  var arr=[5,22,19,25,34];
12  arr.forEach(function(fetch) {
13    document.writeln(fetch);
14  });
15
```

Learn with Rabbil

JAVASCRIPT ARRAY INCLUDES()

The JavaScript array includes() method checks whether the given array contains the specified element. It returns true if an array contains the element, otherwise false.

index.html

```
11  var arr=[5,22,19,25,34];
12  var result=arr.includes(22);
13  document.writeln(result);
```

Learn with Rabbil

JAVASCRIPT ARRAY INDEXOF()

The JavaScript array indexOf() method is used to search the position of a particular element in a given array. This method is case-sensitive.

index.html

```
11   var arr=[5,22,19,25,34];
12   var result= arr.indexOf(19);
13   document.writeln(result);
```

Learn with Rabbil

JAVASCRIPT ARRAY POP()

The JavaScript array pop() method removes the last element from the given array and return that element.

index.html

```
11  var arr=[5,22,19,25,34];
12  arr.pop();
13  document.writeln(arr);
```

Learn with Rabbil

JAVASCRIPT ARRAY PUSH()

The JavaScript array push() method adds one or more elements to the end of the given array.

index.html

```
11   var arr=[5,22,19,25,34];
12   arr.push(50);
13   document.writeln(arr);
```

Learn with Rabbil

JAVASCRIPT ARRAY REVERSE()

The JavaScript array reverse() method changes the sequence of elements of the given array and returns the reverse sequence.

index.html

```
11  var arr=[5,22,19,25,34];
12  array.reverse()
13  document.writeln(arr);
14
```

Learn with Rabbil

JAVASCRIPT ARRAY SLICE()

The JavaScript array slice() method extracts the part of the given array and returns it. This method doesn't change the original array

index.html

```
11  var arr=[5,22,19,25,34];
12  var result=arr.slice(1,3);
13  document.writeln(result);
```

Learn with Rabbil

JAVASCRIPT ARRAY SORT()

The JavaScript array sort() method is used to arrange the array elements in some order. By default, sort() method follows the ascending order.

index.html

```
11   var arr=[5,22,19,25,34];
12   var result=arr.sort();
13   document.writeln(result);
```

Learn with Rabbil

JAVASCRIPT ARRAY SPLICE()

The JavaScript array splice() method is used to add/remove the elements to/from the existing array. It returns the removed elements from an array.

The splice() method also modifies the original array.

index.html

```
11  var arr=["A","B","C","D","E"];
12  var result=arr.splice(2,0,"B")
13  document.writeln(arr);
```

JAVASCRIPT STRING METHODS

The JavaScript string is an object that represents a sequence of characters

charAt()
It provides the char value present at the specified index.

concat()
It provides a combination of two or more strings.

indexOf()
It provides the position of a char value present in the given string.

lastIndexOf()
It provides the position of a char value present in the given string by searching a character from the last position.

replace()
It replaces a given string with the specified replacement.

substr()
It is used to fetch the part of the given string on the basis of the specified starting position and length.

JAVASCRIPT STRING METHODS

The JavaScript string is an object that represents a sequence of characters

substring()

It is used to fetch the part of the given string on the basis of the specified index.

slice()

It is used to fetch the part of the given string. It allows us to assign positive as well negative index.

toLowerCase()

It converts the given string into lowercase letter.

toUpperCase()

It converts the given string into uppercase letter.

trim()

It trims the white space from the left and right side of the string.

JAVASCRIPT DATE OBJECT

The JavaScript date object can be used to get year, month and day.

getDate()
It returns the integer value between 1 and 31 that represents the day for the specified date on the basis of local time.

getDay()
It returns the integer value between 0 and 6 that represents the day of the week on the basis of local time.

getFullYears()
It returns the integer value that represents the year on the basis of local time.

getHours()
It returns the integer value between 0 and 23 that represents the hours on the basis of local time.

getMilliseconds()
It returns the integer value between 0 and 999 that represents the milliseconds on the basis of local time.

JAVASCRIPT DATE OBJECT

The JavaScript date object can be used to get year, month and day.

getMinutes()

It returns the integer value between 0 and 59 that represents the minutes on the basis of local time.

getMonth()

It returns the integer value between 0 and 11 that represents the month on the basis of local time.

getSeconds()

It returns the integer value between 0 and 60 that represents the seconds on the basis of local time.

JAVASCRIPT DATE OBJECT

Learn with Rabbil

The JavaScript date object can be used to get year, month and day.

getUTCDate()

It returns the integer value between 1 and 31 that represents the day for the specified date on the basis of universal time.

getUTCDay()

It returns the integer value between 0 and 6 that represents the day of the week on the basis of universal time.

getUTCFullYears()

It returns the integer value that represents the year on the basis of universal time.

getUTCHours()

It returns the integer value between 0 and 23 that represents the hours on the basis of universal time.

getUTCMinutes()

It returns the integer value between 0 and 59 that represents the minutes on the basis of universal time.

getUTCMonth()

It returns the integer value between 0 and 11 that represents the month on the basis of universal time.

JAVASCRIPT MATH

Learn with Rabbil

The JavaScript math object provides several constants and methods to perform mathematical operation.

abs()

It returns the absolute value of the given number.

ceil()

It returns a smallest integer value, greater than or equal to the given number.

floor()

It returns largest integer value, lower than or equal to the given number.

max()

It returns maximum value of the given numbers.

min()

It returns minimum value of the given numbers.

random()

It returns random number between 0 (inclusive) and 1 (exclusive).

round()

It returns closest integer value of the given number.

JAVASCRIPT NUMBER OBJECT

Learn with Rabbil

The JavaScript number object enables you to represent a numeric value. It may be integer or floating-point.

isFinite()

It determines whether the given value is a finite number.

isInteger()

It determines whether the given value is an integer.

parseFloat()

It converts the given string into a floating point number.

parseInt()

It converts the given string into an integer number.

toFixed()

It returns the string that represents a number with exact digits after a decimal point.

toString()

It returns the given number in the form of string.

WINDOW OBJECT

Learn with Rabbil

The window object represents a window in browser. An object of window is created automatically by the browser.

alert()

displays the alert box containing message with ok button.

confirm()

displays the confirm dialog box containing message with ok and cancel button.

prompt()

displays a dialog box to get input from the user.

open()

opens the new window.

close()

closes the current window.

setTimeout()

performs action after specified time like calling function, evaluating expressions etc.

Learn with Rabbil

JAVASCRIPT NAVIGATOR OBJECT

The JavaScript navigator object is used for browser detection. It can be used to get browser information such as appName, appCodeName, userAgent etc.

index.html

```
11  document.writeln("<br/>navigator.appCodeName: "+navigator.appCodeName);
12  document.writeln("<br/>navigator.appName: "+navigator.appName);
13  document.writeln("<br/>navigator.appVersion: "+navigator.appVersion);
14  document.writeln("<br/>navigator.cookieEnabled: "+navigator.cookieEnabled);
15  document.writeln("<br/>navigator.language: "+navigator.language);
16  document.writeln("<br/>navigator.userAgent: "+navigator.userAgent);
17  document.writeln("<br/>navigator.platform: "+navigator.platform);
18  document.writeln("<br/>navigator.onLine: "+navigator.onLine);
```

Learn with Rabbil

COMMON JAVASCRIPT EVENTS

HTML events are "things" that happen to HTML elements.

onclick()

The user clicks an HTML element

onchange()

An HTML element has been changed

onmouseover()

The user moves the mouse over an HTML element

onmouseout()

The user moves the mouse away from an HTML element

onkeydown()

The user pushes a keyboard key

onload()

The browser has finished loading the page

Learn with Rabbil

DOCUMENT OBJECT MODEL-DOM

When a web page is loaded, the browser creates a Document Object Model of the page. With the HTML DOM, JavaScript can access and change all the elements of an HTML document.

- 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

FINDING HTML ELEMENTS -DOM

index.html

```
4       <p id="MyId"></p>
5       <script>
6       document.getElementById("MyId").innerHTML = "Learn Confidently";
7       </script>
```

index.html

```
4       <p class="MyClass"></p>
5       <script>
6       document.getElementsByClassName("MyClass")[0].innerHTML = "Learn Confidently";
7       </script>
```

FINDING HTML ELEMENTS -DOM

index.html

```
4       <p></p>
5       <script>
6       document.getElementsByTagName("p")[0].innerHTML = "Learn Confidently";
7       </script>
```

index.html

```
4       <p name="MyName"></p>
5       <script>
6       document.getElementsByName("MyName")[0].innerHTML = "Learn Confidently";
7       </script>
```

Learn with Rabbil

HTML DOM DOCUMENT

- Display all name value pairs of cookies in a document
- Display the domain name of the server that loaded the document
- Display the date and time the document was last modified
- Display the title of a document
- Display the full URL of a document
- Replace the content of a document
- Open a new window, and add some content
- Display the number of elements with a specific tag name

HTML DOM DOCUMENT

index.html

```
4    <p id="demo">Click the button to display the cookies associated with this document.</p>
5    <button onclick="myFunction()">Try it</button>
6
7     <script>
8        function myFunction() {
9            document.getElementById("demo").innerHTML =
10           "Cookies associated with this document: " + document.cookie;
11        }
12   </script>
```

Learn with Rabbil

HTML DOM DOCUMENT

index.html

```
4    <button onclick="myFunction()">Try it</button>
5    <p id="demo"></p>
6    <script>
7        function myFunction() {
8         document.getElementById("demo").innerHTML = document.domain;
9        }
10   </script>
```

HTML DOM DOCUMENT

index.html

```
4   <p>This document was last modified <span id="demo"></span>.</p>
5   <script>
6       document.getElementById("demo").innerHTML = document.lastModified;
7   </script>
```

HTML DOM DOCUMENT

index.html

```
1   <html>
2     <head>
3     <title>Learn Confidently</title>
4     </head>
5   <body>
6
7   <p id="demo"></p>
8   <script>
9   document.getElementById("demo").innerHTML =
10  "The title of this document is: " + document.title;
11  </script>
12
13  </body>
14  </html>
```

HTML DOM DOCUMENT

index.html

```
7    <p>The full URL of this document is: <br><span id="demo"></span>.</p>
8    <script>
9        document.getElementById("demo").innerHTML = document.URL
10   </script>
```

HTML DOM DOCUMENT

index.html

```
7    <p id="demo">Click the button to replace this document with new content.</p>
8    <button onclick="myFunction()">Try it</button>
9    <script>
10       function myFunction() {
11         document.open("text/html","replace");
12         document.write("<h2>Learning about the HTML DOM is fun!</h2>");
13         document.close();
14       }
15   </script>
```

HTML DOM DOCUMENT

index.html

```
7    <p>Click the button to open a new window and add some content.</p>
8    <button onclick="myFunction()">Try it</button>
9
10   <script>
11       function myFunction() {
12         var w = window.open();
13         w.document.open();
14         w.document.write("<h2>Hello World!</h2>");
15         w.document.close();
16       }
17   </script>
```

HTML DOM DOCUMENT

index.html

```
7    <p></p>
8    <p></p>
9    <p></p>
10   <p id="demo"></p>
11   <input type="button" onclick="getElements()" value="How many elements tag p?">
12   <script>
13       function getElements() {
14         var x = document.getElementsByTagName("p");
15         document.getElementById("demo").innerHTML = x.length;
16       }
17   </script>
```

HTML DOM FINDING FORM INPUT VALUE

index.html

```
7    <input id="num1"/><br>
8    <input id="num2"/><br>
9    <button onclick="AddTwo()">Add</button>
10   <script>
11       function AddTwo() {
12         var x = document.getElementById("num1").value;
13         var y = document.getElementById("num2").value;
14         var z=parseFloat(x)+parseFloat(y);
15         alert(z);
16       }
17   </script>
```

DOM HTML CSS MANUPULATION

index.html

```
7     <h1 id="MyId">Learn Confidently</h1>
8     <button onclick="CSSClassManipulation()">Manipulate</button>
9     <script>
10        function CSSClassManipulation() {
11          var x = document.getElementById("MyId")
12          x.classList.add('text-primary')
13        }
14    </script>
```

DOM HTML CSS MANUPULATION

index.html

```
7    <h1 class="text-primary" id="MyId">Learn Confidently</h1>
8    <button onclick="CSSClassManipulation()">Manipulate</button>
9    <script>
10       function CSSClassManipulation() {
11         var x = document.getElementById("MyId")
12         x.classList.remove('text-primary')
13       }
14   </script>
```

DOM CREATE ELEMENT APPEND ELEMENT

Learn with Rabbil

index.html

```
7    <ol id="MyList"></ol>
8    <input id="Item"/>
9    <button onclick="AppendElement()">Append</button>
10   <script>
11       function AppendElement() {
12         var Item = document.getElementById("Item").value;
13         var MyList = document.getElementById("MyList");
14         let li = document.createElement("li");
15         li.innerHTML=Item;
16         MyList.appendChild(li)
17       }
18   </script>
```

DOM CHANGING ATTRIBUTE VALUE

Learn with Rabbil

index.html

```
7   <img id="image" src="https://cdn.rabbil.com/photos/images/2022/11/04/rabbilVai.png" width="160" height="120">
8   <button onclick="ChangeSrc()">Change Image Src</button>
9   <script>
10      function ChangeSrc() {
11        var image = document.getElementById("image")
12        image.src="https://cdn.rabbil.com/photos/images/2022/11/04/whyChoose.png"
13
14      }
15  </script>
```

DOM QUERY SELECTOR

Learn with Rabbil

index.html

```
7    <h1>H1</h1>
8    <h2 id="MyId">H2</h2>
9    <h3 class="MyClass">H3</h3>
10   <h4 name="MyName">H4</h4>
11   <input placeholder="..."/>
12   <button onclick="Change()">Change</button>
13
14   <script>
15       function Change() {
16         document.querySelector("h1").innerHTML='Hello H1'
17         document.querySelector("#MyId").innerHTML='Hello H2'
18         document.querySelector('.MyClass').innerHTML='Hello H3'
19         document.querySelector('h4[name="MyName"]').innerHTML='Hello H4'
20         document.querySelector('input').placeholder='New Placeholder'
21       }
22   </script>
```

AJAX

AJAX is a developer's dream, because you can:

- Update a web page without reloading the page
- Request data from a server - after the page has loaded
- Receive data from a server - after the page has loaded
- Send data to a server - in the background

Learn with Rabbil

AJAX PACKAGES

- The XMLHttpRequest object build in with javascript
- Fetch API supported by modern browser's
- Axios package
- JQuery Ajax
- Superagent

AJAX FETCH API GET REQUEST

index.html

```
8    <button onclick="FetchGetData()">Fetch Get Data</button>
9
10   <script>
11       function FetchGetData() {
12
13       var url="https://crud.teamrabbil.com/api/v1/ReadProduct"
14       var requestOptions = {method: 'GET'};
15
16       fetch(url, requestOptions)
17       .then(response => response.json())
18       .then(result => console.log(result))
19       .catch(error => console.log('error', error));
20
21       }
22   </script>
```

AJAX FETCH API POST REQUEST

index.html

```
8   <button onclick="FetchPostData()">Fetch Post Data</button>
9
10  <script>
11      function FetchPostData() {
12
13      var url="https://crud.teamrabbil.com/api/v1/CreateProduct"
14
15      var data={ Img:"A",ProductCode:"B",ProductName:"C",Qty:"D",TotalPrice:"E",UnitPrice:"F"}
16      var requestOptions = {
17          method: 'POST',
18          headers: {'Accept': 'application/json','Content-Type': 'application/json'},
19          body: JSON.stringify(data)
20      };
21
22      fetch(url, requestOptions)
23      .then(response => response.json())
24      .then(result => console.log(result))
25      .catch(error => console.log('error', error));
26      }
27  </script>
```