ROHINI COLLEGE OF ENGINEERING AND TECHNOLOGY

Kanyakumari Main Road, near Anjugramam, Palkulam, Anjugramam, Tamil Nadu 629401

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATION COURSE

ON

JAVA SCRIPT AND JQUERY

COURSE MATERIAL

JAVASCRIPT AND JQUERY

JAVA SCRIPT

JavaScript (js) is a light-weight object-oriented programming language which is used by several websites for scripting the webpages. It is an interpreted, full-fledged programming language that enables dynamic interactivity on websites when applied to an HTML document. It was introduced in the year 1995 for adding programs to the webpages in the Netscape Navigator browser. Since then, it has been adopted by all other graphical web browsers. With JavaScript, users can build modern web applications to interact directly without reloading the page every time. The traditional website uses js to provide several forms of interactivity and simplicity.

Although, JavaScript has no connectivity with Java programming language. The name was suggested and provided in the times when Java was gaining popularity in the market. In addition to web browsers, databases such as CouchDB and MongoDB uses JavaScript as their scripting and query language.

Features of JavaScript

There are following features of JavaScript:

1. All popular web browsers support JavaScript as 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.

Application of JavaScript

JavaScript is used to create interactive websites. It is mainly used for:

- 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 Example

```
<script>
document.write("Hello JavaScript by JavaScript");
</script>
```

A detailed explanation of first JavaScript example is given in next chapter.

JavaScript Tutorial

- JavaScript Introduction
- JavaScript Example
- External JavaScript

JavaScript Basics

- JavaScript Comment
- JavaScript Variable
- JavaScript Global Variable
- JavaScript Data Types
- JavaScript Operators

- JavaScript If Statement
- JavaScript Switch
- JavaScript Loop
- JavaScript Function

JavaScript Objects

- JavaScript Object
- JavaScript Array
- JavaScript String
- JavaScript Date
- JavaScript Math
- JavaScript Number
- JavaScript Boolean
- JavaScript DataView
- JavaScript Function
- JavaScript handler
- JavaScript JSON
- JavaScript Number
- JavaScript Reflect
- JavaScript RegExp
- JavaScript Symbol

JavaScript BOM

- Browser Objects
- 1) Window Object
- 2) History Object
- 3) Navigator Object

- 4) Screen Object

JavaScript DOM

- 5) Document Object
- getElementById
- getElementsByName
- getElementsByTagName
- JavaScript innerHTML property
- JavaScript innerText property

JavaScript Validation

- JavaScript form validation
- JavaScript email validation

JavaScript OOPs

- JavaScript Class
- JavaScript Object
- JavaScript Prototype
- JavaScript constructor Method
- JavaScript static Method
- JavaScript Encapsulation
- JavaScript Inheritance
- JavaScript Polymorphism
- JavaScript Abstraction

JavaScript Cookies

- JavaScript Cookies
- Cookie Attributes

- Array every() method
- Array fill() method
- Array filter() method
- Array find() method
- Array findIndex() method
- Array forEach() method
- Array includes() method
- Array indexOf() method
- Array join() method
- Array lastIndexOf() method
- Array map() method
- Array pop() method
- Array push() method
- Array reverse() method
- Array shift() method
- Array slice() method
- Array sort() method
- Array splice() method
- Array unshift() method

JavaScript DataView Object

- JavaScript DataView
- DataView getFloat32() method
- DataView getFloat64() method
- DataView getInt8() method
- DataView getInt16() method
- DataView getInt32() method

- DataView getUint8() method
- DataView getUint16() method
- DataView getUint32() method

JavaScript Function Object

- JavaScript Function
- Function apply() method
- Function bind() method
- Function call() method
- Function toString() method

JavaScript Date Object

- JavaScript Date
- date getDate() method
- date getDay() method
- date getFullYears() method
- date getHours() method
- date getMilliseconds() method
- date getMinutes() method
- date getMonth() method
- date getSeconds() method
- date getUTCDate() method
- date getUTCDay() method
- date getUTCFullYears() method
- date getUTCHours() method
- date getUTCMinutes() method
- date getUTCMonth() method

- date getUTCSeconds() method
- date setDate() method
- date setDay() method
- date setFullYears() method
- date setHours() method
- date setMilliseconds() method
- date setMinutes() method
- date setMonth() method
- date setSeconds() method
- date setUTCDate() method
- date setUTCDay() method
- date setUTCFullYears() method
- date setUTCHours() method
- date setUTCMilliseconds() method
- date setUTCMinutes() method
- date setUTCMonth() method
- date setUTCSeconds() method
- date toDateString() method
- date toISOString() method
- date toJSON() method
- date toString() method
- date toTimeString() method
- date toUTCString() method
- date valueOf() method

JavaScript handler Object

- JavaScript handler

- handler apply() method
- handler construct() method
- handler defineProperty() method
- handler deleteProperty() method
- handler get() method
- handler getOwnPropertyDescriptor() method
- handler getPrototypeOf() method
- handler has() method
- handler isExtensible() method
- handler ownKeys() method
- handler preventExtensions() method
- handler set() method
- handler setPrototypeOf() method

JavaScript JSON Object

- JavaScript JSON
- JSON.parse() method
- JSON.stringify() method

JavaScript Map Object

- JavaScript Map
- Map clear() method
- Map delete() method
- Map entries() method
- Map forEach() method
- Map get() method
- Map has() method

- Math tan() method
- Math tanh() method
- Math trunc() method

JavaScript Number Object

- JavaScript Number
- Number isFinite() method
- Number isInteger() method
- Number parseFloat() method
- Number parseInt() method
- Number toExponential() method
- Number toFixed() method
- Number toPrecision() method
- Number toString() method

JavaScript RegExp Object

- JavaScript RegExp
- RegExp.exec() method
- RegExp.test() method
- RegExp.toString() method

JavaScript Object

- JavaScript Object
- Object.assign() method
- Object.create() method
- Object.defineProperty() method
- Object.defineProperties() method

- Object.entries() method
- Object.freeze() method
- getOwnPropertyDescriptor() method
- getOwnPropertyDescriptors() method
- getOwnPropertyNames() method
- getOwnPropertySymbols() method
- Object.getPrototypeOf() method
- Object.is() method
- preventExtensions() method
- Object.seal() method
- Object.setPrototypeOf() method
- Object.values() method

JavaScript Reflect Object

- JavaScript Reflect
- Reflect.apply() method
- Reflect.construct() method
- Reflect.defineProperty() method
- Reflect.deleteProperty() method
- Reflect.get() method
- getOwnPropertyDescriptor() method
- Reflect.getPrototypeOf() method
- Reflect.has() method
- Reflect.isExtensible() method
- Reflect.ownKeys() method
- preventExtensions() method
- Reflect.set() method

- Reflect.setPrototypeOf() method

JavaScript Set Object

- JavaScript Set
- Set add() method
- Set clear() method
- Set delete() method
- Set entries() method
- Set forEach() method
- Set has() method
- Set values() method

JavaScript String Object

- String charAt() method
- String charAt() method
- String charCodeAt() method
- String concat() method
- String indexOf() method
- String lastIndexOf() method
- String search() method
- String match()
- String replace() method
- String substr() method
- String substring() method
- String slice() method
- String toLowerCase() method
- toLocaleLowerCase() method

- String toUpperCase() method
- toLocaleUpperCase() method
- String toString() method
- String valueOf() method

JavaScript Symbol Object

- JavaScript Symbol
- Symbol.for() method
- Symbol.keyFor() method
- Symbol.toString() method

Symbol Property

- Symbol.hasInstance Property
- isConcatSpreadable Property
- Symbol.match Property
- Symbol.prototype Property
- Symbol.replace Property
- Symbol.search Property
- Symbol.split Property
- Symbol.toStringTag Property
- Symbol.unscopables Property

JavaScript TypedArray Object

- JavaScript TypedArray
- TypedArray copyWithin() method
- TypedArray entries() method
- TypedArray every() method

- TypedArray fill() method
- TypedArray filter() method
- TypedArray find() method
- TypedArray findIndex() method
- TypedArray forEach() method
- TypedArray includes() method
- TypedArray indexof() method
- TypedArray join() method
- TypedArray Keys() method
- TypedArray lastIndexof() method
- TypedArray map() method
- TypedArray reduce() method
- TypedArray reduceRight() method
- TypedArray reverse() method
- TypedArray set() method
- TypedArray Slice() method
- TypedArray some() method
- TypedArray sort() method
- TypedArray subarray() method
- TypedArray values() method
- toLocaleString() method
- TypedArray toString() method

JavaScript WeakMap Object

- JavaScript WeakMap
- WeakMap delete() method
- WeakMap get() method

- WeakMap has() method
- WeakMap set() method

JavaScript WeakSet Object

- JavaScript WeakSet
- WeakSet add() method
- WeakSet delete() method
- WeakSet has() method

JavaScript Example

1. JavaScript Example
2. Within body tag
3. Within head tag

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.

Let’s create the first JavaScript example.

```
<script type="text/javascript">
document.write("JavaScript is a simple language for javatpoint learners");
</script>
```

The **script** tag specifies that we are using JavaScript.The **text/javascript** is the content type that provides information to the browser about the data.

The **document.write()** function is used to display dynamic content through JavaScript. We will learn about document object in detail later.

3 Places to put JavaScript code

1. Between the body tag of html

2. Between the head tag of html
3. In .js file (external javaScript)

1) JavaScript Example : code between the body tag

In the above example, we have displayed the dynamic content using JavaScript. Let's see the simple example of JavaScript that displays alert dialog box.

```
<script type="text/javascript">
 alert("Hello Javatpoint");
</script>
```

2) JavaScript Example : code between the head tag

Let's see the same example of displaying alert dialog box of JavaScript that is contained inside the head tag. In this example, we are creating a function msg(). To create function in JavaScript, you need to write function with function_name as given below. To call function, you need to work on event. Here we are using onclick event to call msg() function.

```
<html>
<head>
<script type="text/javascript">
function msg(){
 alert("Hello Javatpoint");
}
</script>
</head>
<body>
<p>Welcome to JavaScript</p>
<form>
    <input type="button" value="click" onclick="msg()"/>
    </form>
    </body>
```

```
</html>
```

External JavaScript file

We can create external JavaScript file and embed it in many html page.It provides **code re usability** because single JavaScript file can be used in several html pages.An external JavaScript file must be saved by .js extension. It is recommended to embed all JavaScript files into a single file. It increases the speed of the webpage.Let's create an external JavaScript file that prints Hello Javatpoint in a alert dialog box.

message.js

```
function msg(){
 alert("Hello Javatpoint");
}
```

Let's include the JavaScript file into html page. It calls the JavaScript function on button click.

index.html

```
<html>
<head>
<script type="text/javascript" src="message.js"></script>
</head>
<body>
<p>Welcome to JavaScript</p>
<form>
<input type="button" value="click" onclick="msg()"/>
</form>
</body>
</html>
```

Advantages of External JavaScript

There will be following benefits if a user creates an external javascript:

1. It helps in the reusability of code in more than one HTML file.
2. It allows easy code readability.
3. It is time-efficient as web browsers cache the external js files, which further reduces the page loading time.
4. It enables both web designers and coders to work with html and js files parallelly and separately, i.e., without facing any code conflictions.
5. The length of the code reduces as only we need to specify the location of the js file.

Disadvantages of External JavaScript

There are the following disadvantages of external files:

1. The stealer may download the coder's code using the url of the js file.
2. If two js files are dependent on one another, then a failure in one file may affect the execution of the other dependent file.
3. The web browser needs to make an additional http request to get the js code.
4. A tiny to a large change in the js code may cause unexpected results in all its dependent files.
5. We need to check each file that depends on the commonly created external javascript file.
6. If it is a few lines of code, then better to implement the internal javascript code.

JavaScript Comment

1. JavaScript comments
2. Advantage of javaScript comments
3. Single-line and Multi-line comments

The **JavaScript comments** are meaningful way to deliver message. It is used to add information about the code, warnings or suggestions so that end user can easily interpret the code.

The JavaScript comment is ignored by the JavaScript engine i.e. embedded in the browser.

Advantages of JavaScript comments

There are mainly two advantages of JavaScript comments.

1. **To make code easy to understand** It can be used to elaborate the code so that end user can easily understand the code.
2. **To avoid the unnecessary code** It can also be used to avoid the code being executed. Sometimes, we add the code to perform some action. But after sometime, there may be need to disable the code. In such case, it is better to use comments.

Types of JavaScript Comments.

There are two types of comments in JavaScript.

1. Single-line Comment
2. Multi-line Comment

JavaScript Single line Comment

It is represented by double forward slashes (//). It can be used before and after the statement.

Let’s see the example of single-line comment i.e. added before the statement.

```
<script>
// It is single line comment
document.write("hello javascript");
</script>
```

Let’s see the example of single-line comment i.e. added after the statement.

```
<script>
var a=10;
var b=20;
var c=a+b;//It adds values of a and b variable
```

```
document.write(c);//prints sum of 10 and 20
</script>
```

JavaScript Multi line Comment

It can be used to add single as well as multi line comments. So, it is more convenient.It is represented by forward slash with asterisk then asterisk with forward slash. For example:

```
/* your code here  */
```

It can be used before, after and middle of the statement.

```
<script>
/* It is multi line comment.
It will not be displayed */
document.write("example of javascript multiline comment");
</script>
```

JavaScript Variable

1. JavaScript variable
2. JavaScript Local variable
3. JavaScript Global variable

A **JavaScript variable** is simply a name of storage location. There are two types of variables in JavaScript : local variable and global variable.

There are some rules while declaring a JavaScript variable (also known as identifiers).

1. Name must start with a letter (a to z or A to Z), underscore(_), or dollar($) sign.
2. After first letter we can use digits (0 to 9), for example value1.
3. JavaScript variables are case sensitive, for example x and X are different variables.

Correct JavaScript variables

```
var x = 10;
var _value="sonoo";
```

Incorrect JavaScript variables

```
var  123=30;
var *aa=320;
```

Example of JavaScript variable

Let’s see a simple example of JavaScript variable.

```
<script>
var x = 10;
var y = 20;
var z=x+y;
document.write(z);
</script>
```

JavaScript local variable

A JavaScript local variable is declared inside block or function. It is accessible within the function or block only. For example:

```
<script>
function abc(){
var x=10;//local variable
}
</script>
```

Or,**<script>**

```
If(10<13){
var y=20;//JavaScript local variable
}
</script>
```

JavaScript global variable

A **JavaScript global variable** is accessible from any function. A variable i.e. declared outside the function or declared with window object is known as global variable. For example:

```
<script>
var data=200;//gloabal variable
function a(){
document.writeln(data);
}
function b(){
document.writeln(data);
}
a();//calling JavaScript function
b();
</script>
```

JavaScript Global Variable

A **JavaScript global variable** is declared outside the function or declared with window object. It can be accessed from any function.

Let’s see the simple example of global variable in JavaScript.

```
<script>
var value=50;//global variable
function a(){
alert(value);
}
function b(){
alert(value);
}
</script>
```

Declaring JavaScript global variable within function

To declare JavaScript global variables inside function, you need to use **window object**. For example:

window.value=90; Now it can be declared inside any function and can be accessed from any function. For example:

```
function m(){
window.value=100;//declaring global variable by window object
}
function n(){
alert(window.value);//accessing global variable from other function
}
```

Internals of global variable in JavaScript

When you declare a variable outside the function, it is added in the window object internally. You can access it through window object also. For example:

```
var value=50;
function a(){
alert(window.value);//accessing global variable
}
```

Javascript Data Types

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

1. Primitive data type
2. Non-primitive (reference) data type

JavaScript is a **dynamic type language**, means you don't need to specify type of the variable because it is dynamically used by JavaScript engine. You need to use **var** here to specify the data type. It can hold any type of values such as numbers, strings etc. For example:

```
var a=40;//holding number
var b="Rahul";//holding string
```

JavaScript primitive data types

There are five types of primitive data types in JavaScript. They are as follows:

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

JavaScript non-primitive data types

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

JavaScript Operators

JavaScript operators are symbols that are used to perform operations on operands. For example:

1. var sum=10+20;

Here, + is the arithmetic operator and = is the assignment operator.

There are following types of operators in JavaScript.

1. Arithmetic Operators
2. Comparison (Relational) Operators
3. Bitwise Operators
4. Logical Operators
5. Assignment Operators
6. Special Operators

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

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

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

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

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

JavaScript Special Operators

The following operators are known as JavaScript special operators.

Operator	Description
(?:)	Conditional Operator returns value based on the condition. It is like if-else.

,	Comma Operator allows multiple expressions to be evaluated as single statement.
delete	Delete Operator deletes a property from the object.
in	In Operator checks if object has the given property
instanceof	checks if the object is an instance of given type
new	creates an instance (object)
typeof	checks the type of object.
void	it discards the expression's return value.
yield	checks what is returned in a generator by the generator's iterator.

JavaScript If-else

The **JavaScript if-else statement** is used *to execute the code whether condition is true or false*. There are three forms of if statement in JavaScript.

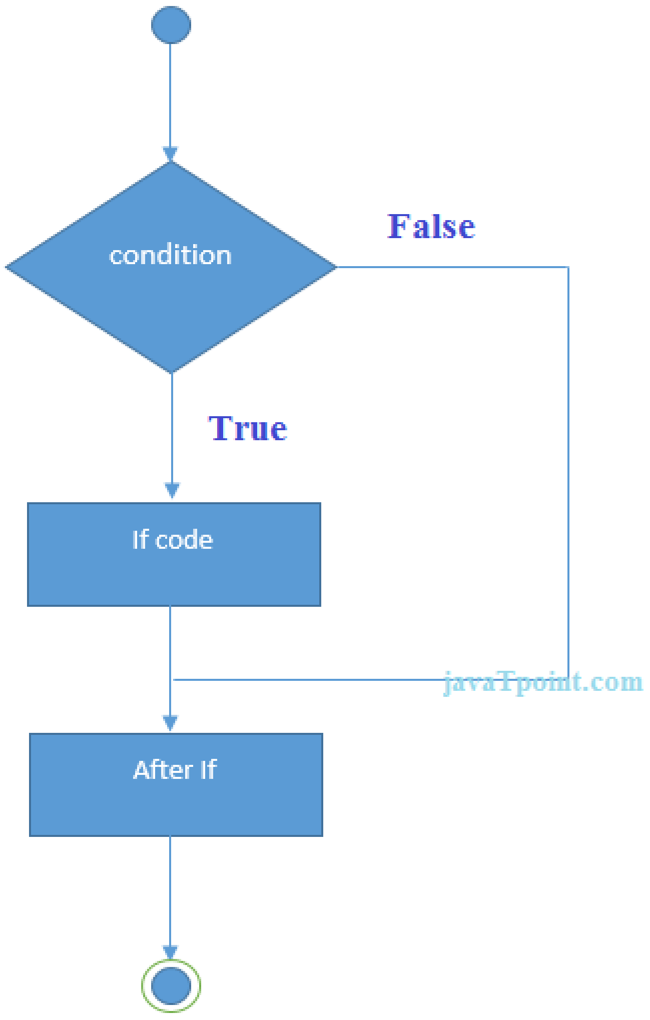
1. If Statement
2. If else statement
3. if else if statement

JavaScript If statement

It evaluates the content only if expression is true. The signature of JavaScript if statement is given below.

```
if(expression){
//content to be evaluated
}
```

Flowchart of JavaScript If statement



Let's see the simple example of if statement in javascript.

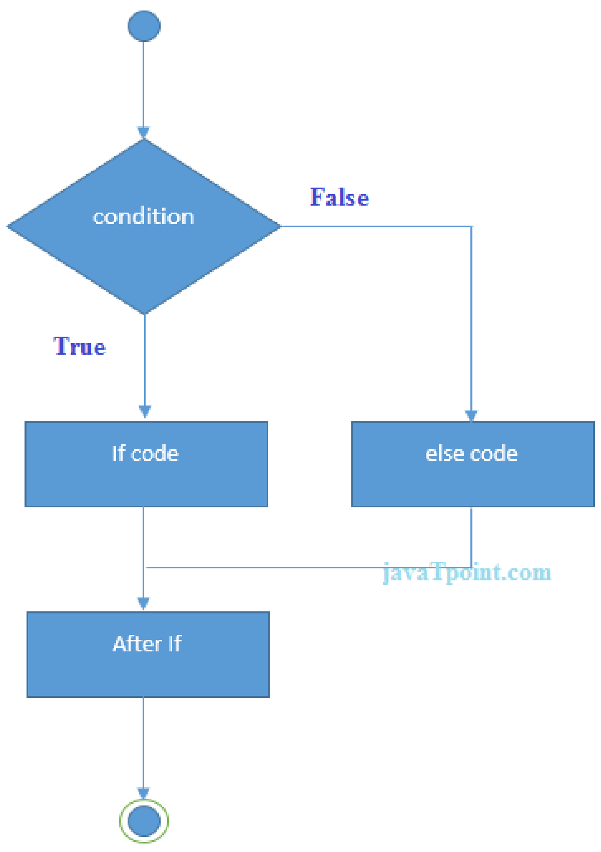
```
<script>
var a=20;
if(a>10){
document.write("value of a is greater than 10");
}
</script>
```

JavaScript If...else Statement

It evaluates the content whether condition is true of false. The syntax of JavaScript if-else statement is given below.

```
if(expression){
//content to be evaluated if condition is true
}
else{
//content to be evaluated if condition is false
}
```

Flowchart of JavaScript If...else statement



Let's see the example of if-else statement in JavaScript to find out the even or odd number.

<script>

```
var a=20;
if(a%2==0){
document.write("a is even number");
}
else{
document.write("a is odd number");
}
</script>
```

JavaScript If...else if statement

It evaluates the content only if expression is true from several expressions. The signature of JavaScript if else if statement is given below.

```
if(expression1){
//content to be evaluated if expression1 is true
}
else if(expression2){
//content to be evaluated if expression2 is true
}
else if(expression3){
//content to be evaluated if expression3 is true
}
else{
//content to be evaluated if no expression is true
}
```

Let’s see the simple example of if else if statement in javascript.

```
<script>
var a=20;
if(a==10){
document.write("a is equal to 10");
}
else if(a==15){
document.write("a is equal to 15");
}
else if(a==20){
document.write("a is equal to 20");
}
else{
document.write("a is not equal to 10, 15 or 20");
}
</script>
```

JavaScript Switch

The **JavaScript switch statement** is used *to execute one code from multiple expressions*. It is just like else if statement that we have learned in previous page. But it is convenient than *if..else..if* because it can be used with numbers, characters etc.

The signature of JavaScript switch statement is given below.

```
switch(expression){
case value1:
 code to be executed;
 break;
case value2:
 code to be executed;
 break;
......

default:
 code to be executed if above values are not matched;
}
```

Let’s see the simple example of switch statement in javascript.

```
<script>
var grade='B';
var result;
switch(grade){
case 'A':
result="A Grade";
break;
case 'B':
result="B Grade";
break;
case 'C':
result="C Grade";
break;
default:
result="No Grade";
}
document.write(result);
</script>
```

JavaScript Loops

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.There are four types of loops in JavaScript.

1. for loop
2. while loop
3. do-while loop
4. for-in loop

1) 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. The syntax of for loop is given below.

```
for (initialization; condition; increment)
{
   code to be executed
}
```

Let’s see the simple example of for loop in javascript.

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

2**) 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. The syntax of while loop is given below.

```
while (condition)
{
   code to be executed
}
```

Let’s see the simple example of while loop in javascript.

```
<script>
var i=11;
while (i<=15)
```

```
        {
        document.write(i + "<br/>");
        i++;
        }
</script>
```

3) 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. The syntax of do while loop is given below.

```
    do{
       code to be executed
    }while (condition);
```

Let’s see the simple example of do while loop in javascript.

```
    <script>
    var i=21;
    do{
    document.write(i + "<br/>");
    i++;
    }while (i<=25);
    </script>
```

JavaScript Example

1. JavaScript Example
2. Within body tag
3. Within head tag

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.

Let’s create the first JavaScript example.

```
<script type="text/javascript">
document.write("JavaScript is a simple language for javatpoint learners");
</script>
```

The **script** tag specifies that we are using JavaScript.

The **text/javascript** is the content type that provides information to the browser about the data.

The **document.write()** function is used to display dynamic content through JavaScript. We will learn about document object in detail later.

3 Places to put JavaScript code

1. Between the body tag of html
2. Between the head tag of html
3. In .js file (external javaScript)

1) JavaScript Example : code between the body tag

In the above example, we have displayed the dynamic content using JavaScript. Let’s see the simple example of JavaScript that displays alert dialog box.

```
<script type="text/javascript">
 alert("Hello Javatpoint");
</script>
```

2) JavaScript Example : code between the head tag

Let’s see the same example of displaying alert dialog box of JavaScript that is contained inside the head tag.In this example, we are creating a function msg(). To create function in JavaScript, you need to write function with function_name as given below.

To call function, you need to work on event. Here we are using onclick event to call msg() function.

```
<html>
<head>
<script type="text/javascript">
function msg(){
 alert("Hello Javatpoint");
}
          </script>
          </head>
          <body>
          <p>Welcome to JavaScript</p>
          <form>
          <input type="button" value="click" onclick="msg()"/>
          </form>
          </body>
          </html>
```

What is jQuery

- jQuery is a small and lightweight JavaScript library.
- jQuery is cross-platform.
- jQuery means "write less do more".
- jQuery simplifies AJAX call and DOM manipulation.

jQuery Example

In this tutorial, you will get a lot of jQuery examples to understand the topic well. Let's see a simple jQuery example.

File: firstjquery.html

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

```
 <title>First jQuery Example</title>
<script type="text/javascript" src="http://ajax.googleapis.com/ajax/libs/jquery/2.1.3/jquer
y.min.js">
 </script>
 <script type="text/javascript" language="javascript">
 $(document).ready(function() {
 $("p").css("background-color", "pink");
 });
 </script>
 </head>
<body>
<p>This is first paragraph.</p>
<p>This is second paragraph.</p>
<p>This is third paragraph.</p>
</body>
</html>
```

jQuery Features

Following are the important features of jQuery.

- HTML manipulation
- DOM manipulation
- DOM element selection
- CSS manipulation
- Effects and Animations
- Utilities
- AJAX
- HTML event methods
- JSON Parsing
- Extensibility through plug-ins

Why jQuery is required

Sometimes, a question can arise that what is the need of jQuery or what difference it makes on bringing jQuery instead of AJAX/ JavaScript? If jQuery is the replacement of AJAX and JavaScript? For all these questions, you can state the following answers.

- It is very fast and extensible.
- It facilitates the users to write UI related function codes in minimum possible lines.
- It improves the performance of an application.
- Browser's compatible web applications can be developed.
- It uses mostly new features of new browsers.

So, you can say that out of the lot of JavaScript frameworks, jQuery is the most popular and the most extendable. Many of the biggest companies on the web use jQuery. Some of these companies are:

- Microsoft
- Google
- IBM
- Netflix

jQuery Example

jQuery is developed by Google. To create the first jQuery example, you need to use JavaScript file for jQuery. You can download the jQuery file from jquery.com or use the absolute URL of jQuery file. In this jQuery example, we are using the absolute URL of jQuery file. The jQuery example is written inside the script tag.

Let's see a simple example of jQuery.

File: firstjquery.html

```
<!DOCTYPE html>
```

```
<html>
<head>
 <title>First jQuery Example</title>
 <script type="text/javascript" src="http://ajax.googleapis.com/ajax/libs/jquery/2.1.3/j
query.min.js">
 </script>
 <script type="text/javascript" language="javascript">
 $(document).ready(function() {
 $("p").css("background-color", "cyan");
 });
 </script>
 </head>
<body>
<p>The first paragraph is selected.</p>
<p>The second paragraph is selected.</p>
<p>The third paragraph is selected.</p>
</body>
</html>
```

$(document).ready() and $()

The code inserted between $(document).ready() is executed only once when page is ready for JavaScript code to execute.

In place of $(document).ready(), you can use shorthand notation $() only.

```
$(document).ready(function() {
$("p").css("color", "red");
});
```

The above code is equivalent to this code.

```
$(function() {
```

```
$("p").css("color", "red");
});
```

Let's see the full example of jQuery using shorthand notation $().

File: shortjquery.html

```
<!DOCTYPE html>
<html>
<head>
 <title>Second jQuery Example</title>
 <script type="text/javascript" src="http://ajax.googleapis.com/ajax/libs/jquery/2.1.3/jqu
ery.min.js">
 </script>
 <script type="text/javascript" language="javascript">
 $(function() {
 $("p").css("color", "red");
 });
 </script>
 </head>
<body>
<p>The first paragraph is selected.</p>
<p>The second paragraph is selected.</p>
<p>The third paragraph is selected.</p>
</body>
</html>
```

jQuery Selectors

jQuery Selectors are used to select and manipulate HTML elements. They are very important part of jQuery library. With jQuery selectors, you can find or select HTML elements based on their id, classes, attributes, types and much more from a DOM.

In simple words, you can say that selectors are used to select one or more HTML elements using jQuery and once the element is selected then you can perform various operation on that. All jQuery selectors start with a dollor sign and parenthesis e.g. $(). It is known as the factory function.

The $() factory function

Every jQuery selector start with thiis sign $(). This sign is known as the factory function. It uses the three basic building blocks while selecting an element in a given document.

S.No.	Selector	Description
1)	Tag Name:	It represents a tag name available in the DOM. For example: $('p') selects all paragraphs'p'in the document.
2)	Tag ID:	It represents a tag available with a specific ID in the DOM. For example: $('#real-id') selects a specific element in the document that has an ID of real-id.
3)	Tag Class:	It represents a tag available with a specific class in the DOM. For example: $('real-class') selects all elements in the document that have a class of real-class.

Let's take a simple example to see the use of Tag selector. This would select all the elements with a tag name and the background color is set to "pink".

File: firstjquery.html

```
<!DOCTYPE html>
<html>
<head>
 <title>First jQuery Example</title>
```

```
        <script type="text/javascript" src="http://ajax.googleapis.com/ajax/libs/jquery/2.1.3/jquer
y.min.js">
        </script>
        <script type="text/javascript" language="javascript">
        $(document).ready(function() {
        $("p").css("background-color", "pink");
        });
        </script>
        </head>
        <body>
        <p>This is first paragraph.</p>
        <p>This is second paragraph.</p>
        <p>This is third paragraph.</p>
        </body>
        </html>
```

How to use Selectors

The jQuery selectors can be used single or with the combination of other selectors. They are required at every steps while using jQuery. They are used to select the exact element that you want from your HTML document.

S.No.	Selector	Description
1)	Name:	It selects all elements that match with the given element name.
2)	#ID:	It selects a single element that matches with the given id.
3)	.Class:	It selects all elements that matches with the given class.
4)	Universal(*)	It selects all elements available in a DOM.

5)	Multiple Elements A,B,C	It selects the combined results of all the specified selectors A,B and C.

Different jQuery Selectors

Selector	Example	Description
*	$("*")	It is used to select all elements.
#id	$("#firstname")	It will select the element with id="firstname"
.class	$(".primary")	It will select all elements with class="primary"
class,.class	$(".primary,.secondary")	It will select all elements with the class "primary" or "secondary"
element	$("p")	It will select all p elements.
el1,el2,el3	$("h1,div,p")	It will select all h1, div, and p elements.
:first	$("p:first")	This will select the first p element
:last	$("p:last")	This will select he last p element
:even	$("tr:even")	This will select all even tr elements
:odd	$("tr:odd")	This will select all odd tr elements
:first-child	$("p:first-child")	It will select all p elements that are the first child of their parent
:first-of-type	$("p:first-of-type")	It will select all p elements that are the first p element of their parent

:last-child	$("p:last-child")	It will select all p elements that are the last child of their parent
:last-of-type	$("p:last-of-type")	It will select all p elements that are the last p element of their parent
:nth-child(n)	$("p:nth-child(2)")	This will select all p elements that are the 2nd child of their parent
:nth-last-child(n)	$("p:nth-last-child(2)")	This will select all p elements that are the 2nd child of their parent, counting from the last child
:nth-of-type(n)	$("p:nth-of-type(2)")	It will select all p elements that are the 2nd p element of their parent
:nth-last-of-type(n)	$("p:nth-last-of-type(2)")	This will select all p elements that are the 2nd p element of their parent, counting from the last child
:only-child	$("p:only-child")	It will select all p elements that are the only child of their parent
:only-of-type	$("p:only-of-type")	It will select all p elements that are the only child, of its type, of their parent
parent > child	$("div > p")	It will select all p elements that are a direct child of a div element
parent descendant	$("div p")	It will select all p elements that are descendants of a div element
element + next	$("div + p")	It selects the p element that are next to each div elements

element ~ siblings	$("div ~ p")	It selects all p elements that are siblings of a div element
:eq(index)	$("ul li:eq(3)")	It will select the fourth element in a list (index starts at 0)
:gt(no)	$("ul li:gt(3)")	Select the list elements with an index greater than 3
:lt(no)	$("ul li:lt(3)")	Select the list elements with an index less than 3
:not(selector)	$("input:not(:empty)")	Select all input elements that are not empty
:header	$(":header")	Select all header elements h1, h2 ...
:animated	$(":animated")	Select all animated elements
:focus	$(":focus")	Select the element that currently has focus
:contains(text)	$(":contains('Hello')")	Select all elements which contains the text "Hello"
:has(selector)	$("div:has(p)")	Select all div elements that have a p element
:empty	$(":empty")	Select all elements that are empty
:parent	$(":parent")	Select all elements that are a parent of another element
:hidden	$("p:hidden")	Select all hidden p elements
:visible	$("table:visible")	Select all visible tables
:root	$(":root")	It will select the document's root element
:lang(language)	$("p:lang(de)")	Select all p elements with a lang attribute value starting with "de"
[attribute]	$("[href]")	Select all elements with a href attribute

[attribute=value]	$("[href='default.htm']")	Select all elements with a href attribute value equal to "default.htm"
[attribute!=value]	$("[href!='default.htm']")	It will select all elements with a href attribute value not equal to "default.htm"
[attribute$=value]	$("[href$='.jpg']")	It will select all elements with a href attribute value ending with ".jpg"
[attribute|=value]	$("[title|='Tomorrow']")	Select all elements with a title attribute value equal to 'Tomorrow', or starting with 'Tomorrow' followed by a hyphen
[attribute^=value]	$("[title^='Tom']")	Select all elements with a title attribute value starting with "Tom"
[attribute~=value]	$("[title~='hello']")	Select all elements with a title attribute value containing the specific word "hello"
[attribute*=value]	$("[title*='hello']")	Select all elements with a title attribute value containing the word "hello"
:input	$(":input")	It will select all input elements
:text	$(":text")	It will select all input elements with type="text"
:password	$(":password")	It will select all input elements with type="password"
:radio	$(":radio")	It will select all input elements with type="radio"
:checkbox	$(":checkbox")	Itwill select all input elements with type="checkbox"
:submit	$(":submit")	It will select all input elements with type="submit"
:reset	$(":reset")	It will select all input elements with type="reset"

:button	$(":button")	It will select all input elements with type="button"
:image	$(":image")	It will select all input elements with type="image"
:file	$(":file")	It will select all input elements with type="file"
:enabled	$(":enabled")	Select all enabled input elements
:disabled	$(":disabled")	It will select all disabled input elements
:selected	$(":selected")	It will select all selected input elements
:checked	$(":checked")	It will select all checked input elements

jQuery hide()

The jQuery hide() method is used to hide the selected elements.

Syntax:

```
$(selector).hide();
$(selector).hide(speed, callback);
$(selector).hide(speed, easing, callback);
```

speed: It is an optional parameter. It specifies the speed of the delay. Its possible vales are slow, fast and milliseconds.

easing: It specifies the easing function to be used for transition.

callback: It is also an optional parameter. It specifies the function to be called after completion of hide() effect.

Let's take an example to see the jQuery hide effect.

```
<!DOCTYPE html>
```

```
<html>
<head>
<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"></script>
<script>
$(document).ready(function(){
   $("#hide").click(function(){
       $("p").hide();
   });
});
</script>
</head>
<body>
<p>
<b>This is a little poem: </b><br/>
Twinkle, twinkle, little star<br/>
How I wonder what you are<br/>
Up above the world so high<br/>
Like a diamond in the sky<br/>
Twinkle, twinkle little star<br/>
How I wonder what you are
</p>
<button id="hide">Hide</button>
</body>
</html>
```

jQuery show()

The jQuery show() method is used to show the selected elements.

Syntax:

```
$(selector).show();
$(selector).show(speed, callback);
```

```
$(selector).show(speed, easing, callback);
```

speed: It is an optional parameter. It specifies the speed of the delay. Its possible vales are slow, fast and milliseconds.

easing: It specifies the easing function to be used for transition.

callback: It is also an optional parameter. It specifies the function to be called after completion of show() effect.

Let's take an example to see the jQuery show effect.

```
<!DOCTYPE html>
<html>
<head>
<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"></script>
<script>
$(document).ready(function(){
    $("#hide").click(function(){
    $("p").hide();
  });
  $("#show").click(function(){
    $("p").show();
  });
});
</script>
</head>
<body>
<p>
<b>This is a little poem: </b><br/>
Twinkle, twinkle, little star<br/>
How I wonder what you are<br/>
Up above the world so high<br/>
```

```
Like a diamond in the sky<br/>
Twinkle, twinkle little star<br/>
How I wonder what you are
</p>
<button id="hide">Hide</button>
<button id="show">Show</button>
</body>
</html>
```

jQuery show() effect with speed parameter

Let's see the example of jQuery show effect with 1500 milliseconds speed.

```
 $(document).ready(function(){
        $("#hide").click(function(){
        $("p").hide(1000);
     });
     $("#show").click(function(){
        $("p").show(1500);
     });
 });
```

jQuery toggle()

The jQuery toggle() is a special type of method which is used to toggle between the hide() and show() method. It shows the hidden elements and hides the shown element.

Syntax:

```
$(selector).toggle();
$(selector).toggle(speed, callback);
$(selector).toggle(speed, easing, callback);
$(selector).toggle(display);
```

speed: It is an optional parameter. It specifies the speed of the delay. Its possible vales are slow, fast and milliseconds.

easing: It specifies the easing function to be used for transition.

callback: It is also an optional parameter. It specifies the function to be called after completion of toggle() effect.

display: If true, it displays element. If false, it hides the element.

Let's take an example to see the jQuery toggle effect.

```
<!DOCTYPE html>
<html>
<head>
<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"></script>

<script>
$(document).ready(function(){
   $("button").click(function(){
      $("div.d1").toggle();
   });
});
</script>
</head>
<body>
<button>Toggle</button>
<div class="d1" style="border:1px solid black;padding:10px;width:250px">
<p><b>This is a little poem: </b><br/>
Twinkle, twinkle, little star<br/>
How I wonder what you are<br/>
Up above the world so high<br/>
Like a diamond in the sky<br/>
```

```
        Twinkle, twinkle little star<br/>
        How I wonder what you are</p>
        </div>
        </body>
        </html>
```

jQuery toggle() effect with speed parameter

Let's see the example of jQuery toggle effect with 1500 milliseconds speed.

```
$(document).ready(function(){
    $("button").click(function(){
      $("div.d1").toggle(1500);
   });
});
```

jQuery fadeIn()

jQuery fadeIn() method is used to fade in the element.

Syntax:

```
$(selector).fadein();
$(selector).fadeIn(speed,callback);
$(selector).fadeIn(speed, easing, callback);
```

speed: It is an optional parameter. It specifies the speed of the delay. Its possible vales are slow, fast and milliseconds. **easing**: It specifies the easing function to be used for transition. **callback**: It is also an optional parameter. It specifies the function to be called after completion of fadein() effect.

Let's take an example to demonstrate jQuery fadeIn() effect.

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

```
<head>
<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"></script>
<script>
$(document).ready(function(){
    $("button").click(function(){
        $("#div1").fadeIn();
        $("#div2").fadeIn("slow");
        $("#div3").fadeIn(3000);
    });
});
</script>
</head>
<body>
<p>See the fadeIn() method example with different parameters.</p>
<button>Click to fade in boxes</button><br><br>
<div id="div1" style="width:80px;height:80px;display:none;background-
color:red;"></div><br>
<div id="div2" style="width:80px;height:80px;display:none;background-
color:green;"></div><br>
<div id="div3" style="width:80px;height:80px;display:none;background-color:blue;"></div>
</body>
</html>
```

jQuery fadeOut()

The jQuery fadeOut() method is used to fade out the element.

Syntax:

```
$(selector).fadeOut();
$(selector).fadeOut(speed,callback);
$(selector).fadeOut(speed, easing, callback);
```

speed: It is an optional parameter. It specifies the speed of the delay. Its possible vales are slow, fast and milliseconds.

easing: It specifies the easing function to be used for transition.

callback: It is also an optional parameter. It specifies the function to be called after completion of fadeOut() effect.

Let's take an example to demonstrate jQuery fadeOut() effect.

```
<!DOCTYPE html>
<html>
<head>
<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"></script
>
<script>
$(document).ready(function(){
    $("button").click(function(){
        $("#div1").fadeOut();
        $("#div2").fadeOut("slow");
        $("#div3").fadeOut(3000);
    });
});
</script>
</head>
<body>
<p>See the fadeOut() method example with different parameters.</p>
<button>Click to fade out boxes</button><br><br>
<div id="div1" style="width:80px;height:80px;background-color:red;"></div><br>
<div id="div2" style="width:80px;height:80px;background-color:green;"></div><br>
<div id="div3" style="width:80px;height:80px;background-color:blue;"></div>
</body>
```

```
</html>
```

jQuery fadeToggle()

jQuery fadeToggle() method is used to toggle between the fadeIn() and fadeOut() methods. If the elements are faded in, it will make them faded out and if they are faded out it will make them faded in. **Syntax**:

1. $(selector).fadeToggle();
2. $(selector).fadeToggle(speed,callback);
3. $(selector).fadeToggle(speed, easing, callback);

speed: It is an optional parameter. It specifies the speed of the delay. Its possible vales are slow, fast and milliseconds.

easing: It specifies the easing function to be used for transition.

callback: It is also an optional parameter. It specifies the function to be called after completion of fadeToggle() effect.

Let's take an example to demonstrate jQuery fadeToggle() effect.

```
<!DOCTYPE html>
<html>
<head>
<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"></script>
<script>
$(document).ready(function(){
   $("button").click(function(){
      $("#div1").fadeToggle();
      $("#div2").fadeToggle("slow");
      $("#div3").fadeToggle(3000);
   });
});
```

```
</script>
</head>
<body>
<p>See the fadeToggle() method example with different parameters.</p>
<button>Click to fade Toggle boxes</button><br><br>
<div id="div1" style="width:80px;height:80px;background-color:red;"></div><br>
<div id="div2" style="width:80px;height:80px;background-color:green;"></div><br>
<div id="div3" style="width:80px;height:80px;background-color:blue;"></div>
</body>
</html>
```

jQuery fadeTo()

jQuery fadeTo() method is used to fading to a given opacity.

Syntax:

```
$(selector).fadeTo(speed, opacity);
$(selector).fadeTo(speed, opacity, callback);
$(selector).fadeTo(speed, opacity, easing, callback);
```

speed: It specifies the speed of the delay. Its possible vales are slow, fast and milliseconds. **opacity**:It specifies the opacity. The opacity value ranges between 0 and 1. **easing**: It specifies the easing function to be used for transition. **callback**: It is also an optional parameter. It specifies the function to be called after completion of fadeToggle() effect.

Let's take an example to demonstrate jQuery fadeTo() effect.

```
<!DOCTYPE html>
<html>
<head>
<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"></script>
<script>
$(document).ready(function(){
    $("button").click(function(){
        $("#div1").fadeTo("slow", 0.3);
        $("#div2").fadeTo("slow", 0.4);
        $("#div3").fadeTo("slow", 0.5);
    });
});
</script>
</head>
<body>
<p>See the fadeTo() method example with different parameters.</p>
<button>Click to fade boxes</button><br><br>
<div id="div1" style="width:80px;height:80px;background-color:red;"></div><br>
<div id="div2" style="width:80px;height:80px;background-color:green;"></div><br>
<div id="div3" style="width:80px;height:80px;background-color:blue;"></div>
</body>
</html>
```

jQuery fadeTo()

jQuery fadeTo() method is used to fading to a given opacity.

Syntax:

```
$(selector).fadeTo(speed, opacity);
$(selector).fadeTo(speed, opacity, callback);
$(selector).fadeTo(speed, opacity, easing, callback);
```

speed: It specifies the speed of the delay. Its possible vales are slow, fast and milliseconds. **opacity**:It specifies the opacity. The opacity value ranges between 0 and 1. **easing**: It specifies the easing function to be used for transition. **callback**: It is also an optional parameter. It specifies the function to be called after completion of fadeToggle() effect.

Let's take an example to demonstrate jQuery fadeTo() effect.

```
<!DOCTYPE html>
<html>
<head>
<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"><
/script>
<script>
$(document).ready(function(){
    $("button").click(function(){
        $("#div1").fadeTo("slow", 0.3);
        $("#div2").fadeTo("slow", 0.4);
        $("#div3").fadeTo("slow", 0.5);
    });
});
</script>
</head>
<body>
<p>See the fadeTo() method example with different parameters.</p>
<button>Click to fade boxes</button><br><br>
<div id="div1" style="width:80px;height:80px;background-
color:red;"></div><br>
<div id="div2" style="width:80px;height:80px;background-
color:green;"></div><br>
<div id="div3" style="width:80px;height:80px;background-
color:blue;"></div>
</body>
</html>
```

jQuery slideDown()

jQuery slideDown() method is used to slide down an element.

Syntax:

```
$(selector).slideDown(speed);
$(selector).slideDown(speed, callback);
$(selector).slideDown(speed, easing, callback);
```

speed: It specifies the speed of the delay. Its possible vales are slow, fast and milliseconds.

easing: It specifies the easing function to be used for transition.

callback: It is also an optional parameter. It specifies the function to be called after completion of slideDown() effect.

Let's take an example to demonstrate jQuery slideDown() effect.

```
<!DOCTYPE html>
<html>
<head>
<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"></script>
<script>
$(document).ready(function(){
   $("#flip").click(function(){
      $("#panel").slideDown("slow");
   });
});
</script>
 <style>
#panel, #flip {
   padding: 5px;
   text-align: center;
   background-color: #00FFFF;
   border: solid 1px #c3c3c3;
}
#panel {
   padding: 50px;
```

```
    display: none;
}
</style>
</head>
<body>
<div id="flip">Click to slide down panel</div>
<div id="panel">Hello javatpoint.com!
It is the best tutorial website to learn jQuery and other languages.</div>
</body>
</html>
```

jQuery slideUp()

jQuery slideDown() method is used to slide up an element.

Syntax:

```
$(selector).slideUp(speed);
$(selector).slideUp(speed, callback);
$(selector).slideUp(speed, easing, callback);
```

speed: It specifies the speed of the delay. Its possible vales are slow, fast and milliseconds.

easing: It specifies the easing function to be used for transition.

callback: It is also an optional parameter. It specifies the function to be called after completion of slideUp() effect.

Let's take an example to demonstrate jQuery slideUp() effect.

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

```
<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"></script>

<script>
$(document).ready(function(){
  $("#flip").click(function(){
      $("#panel").slideUp("slow");
  });
});
</script>
 <style>
#panel, #flip {
    padding: 5px;
    text-align: center;
    background-color: #00FFFF;
    border: solid 1px #c3c3c3;
}
#panel {
    padding: 50px;
}
</style>
</head>
<body>
<div id="flip">Click to slide up panel</div>
<div id="panel">Hello javatpoint.com!
It is the best tutorial website to learn jQuery and other languages.</div>
</body>
</html>
```

jQuery animate()

The jQuery animate() method provides you a way to create custom animations.

Syntax:

1. $(selector).animate({params}, speed, callback);

Here, **params** parameter defines the CSS properties to be animated.

The **speed** parameter is optional and specifies the duration of the effect. It can be set as "slow" , "fast" or milliseconds.

The **callback** parameter is also optional and it is a function which is executed after the animation completes.

Let's take a simple example to see the animation effect.

```
<!DOCTYPE html>
<html>
<head>
<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"></script>
<script>
$(document).ready(function(){
    $("button").click(function(){
        $("div").animate({left: '450px'});
    });
});
</script>
</head>
<body>
<button>Start Animation</button>
<p>A simple animation example:</p>
<div style="background:#98bf21;height:100px;width:100px;position:absolute;"></div>
```

```
</body>
</html>
```

Output:

Start Animation

A simple animation example:

Note: The default position of all HTML elements is static. If you want to manipulate their position, set the CSS position property to the element to relative, fixed or absolute.

jQuery animate() method using multiple properties

You can use multiple properties to animate at the same time.

```
<!DOCTYPE html>
<html>
<head>
<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js
"></script>
<script>
$(document).ready(function(){
    $("button").click(function(){
        $("div").animate({
            left: '250px',
            opacity: '0.5',
            height: '150px',
            width: '150px'
        });
    });
});
</script>
```

```
                    </head>
                    <body>
                    <button>Start Animation</button>
                    <div style="background:#125f21;height:100px;width:100px;position:absolut
e;"></div>
                    </body>
                    </html>
```

Start Animation

jQuery animate() method using relative values

You can also define relative values (it is relative to the element's current value) by putting += or -= in front of the value.

```
<!DOCTYPE html>
<html>
<head>
<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"></script>
<script>
$(document).ready(function(){
  $("button").click(function(){
    $("div").animate({
      left: '250px',
      height: '+=150px',
      width: '+=150px'
    });
  });
});
</script>
</head>
<body>
<button>Start Animation</button>
```

```
<div style="background:#98bf21;height:100px;width:100px;position:absolute;"></div>
</body>
</html>
```

Output:

Start Animation

jQuery animate() method using predefined value

You can also specify a property's animation value as "show" , "hide" , or "toggle".

In this example, we are using "toggle" value for height, it means it will show/hide the selected element.

```
<!DOCTYPE html>
<html>
<head>
<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"></script>
<script>
$(document).ready(function(){
    $("button").click(function(){
        $("div").animate({
            height: 'toggle'
        });
    });
});
</script>
</head>
<body>
<button>Start Animation</button>
<div style="background:#98bf21;height:100px;width:100px;position:absolute;"></div>
</body>
```

```
</html>
```

Output:

jQuery Color animation

You can also animate the properties of elements between colors.

```
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>jQuery UI Effects - Animate demo</title>
  <link rel="stylesheet" href="http://code.jquery.com/ui/1.11.4/themes/smoothness/jquery-ui.css">
  <script src="http://code.jquery.com/jquery-1.10.2.js"></script>
  <script src="http://code.jquery.com/ui/1.11.4/jquery-ui.js"></script>
  <style>
    .toggler { width: 500px; height: 200px; position: relative; }
    #button { padding: .5em 1em; text-decoration: none; }
    #effect { width: 240px; height: 135px; padding: 0.4em; position: relative; background: #fff; }
    #effect h3 { margin: 0; padding: 0.4em; text-align: center; }
  </style>
  <script>
  $(function() {
    var state = true;
    $( "#button" ).click(function() {
      if ( state ) {
        $( "#effect" ).animate({
          backgroundColor: "#aa0000",
          color: "#fff",
          width: 500
        }, 1000 );
      } else {
        $( "#effect" ).animate({
          backgroundColor: "#fff",
          color: "#000",
          width: 240
        }, 1000 );
      }
      state = !state;
    });
  });
  </script>
</head>
<body>
<div class="toggler">
  <div id="effect" class="ui-widget-content ui-corner-all">
    <h3 class="ui-widget-header ui-corner-all">Animate</h3>
    <p>Javatpoint.com is the best tutorial website to learn Java and other programming languages.</p
>
  </div>
</div>
 <button id="button" class="ui-state-default ui-corner-all">Toggle Effect</button>
</body>
</html>
```

jQuery html()

jQuery html() method is used to change the entire content of the selected elements. It replaces the selected element content with new contents. Note: It is a very useful function but works in a limited area because of its API documentation. The API documentation of the jQuery html function consists of three method signatures.

The first method signature has no argument, so it just returns the HTML within that element. The remaining two signatures take a single argument: i.e. a string or a function that returns a string.

Syntax:

1. $(selector).html()

It is used to return content.

1. $(selector).html(content)

It is used to set content.

1. $(selector).html(function (index, currentcontent))

It is used to set content by calling function.

The jQuery html() method is used either for set the content or return the content of the selected elements.

- **To set content**: When you use this method to set content, it overwrites the content of the all matched elements.
- **To return content**: When you use this method to return content, it returns the content of the first matched element.

The text() method is used to set or return only the text content of the selected elements.

Parameters of jQuery html() method

Parameter	**Description**
Content	It is an essential parameter. It is used to specify the new content for the selecte elements. It can also contain HTML tags.
Function (index, currentcontent)	It is an optional parameter. It specifies a function that returns the new content for th selected elements.

	○ **Index**: It shows the index position of the element in the set. ○ **Currentcontent**: It shows the current HTML content of the selected elemen

Example of jQuery html() method

Let's take an example to demonstrate jQuery html() method. It is changing the content of all p elements.

```
    <!DOCTYPE html>
    <html>
    <head>
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>
    <script>
    $(document).ready(function(){
      $("button").click(function(){
        $("p").html("Hello <b>Javatpoint.com</b>");
      });
    });
    </script>
    </head>
    <body>
    <button>Click here to change the content of all p elements</button>
    <p>This is a paragraph.</p>
    <p>This is another paragraph.</p>
    </body>
</html>
```

Click here to change the content of all p elements

This is a paragraph.

This is another paragraph.

jQuery html() example 2

Let's see another example of jQuery html() method that returns HTML content. It returns the content of first paragraph only.

```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>
<script>
$(document).ready(function(){
    $("button").click(function(){
        alert($("p").html());
    });
});
</script>
</head>
<body>

<button>Return the content of p element</button>

<p>This is first <b>paragraph</b>.</p>
<p>This is another <b>paragraph</b>.</p>
</body>
</html>
```

Return the content of p element

This is first **paragraph**.

This is another **paragraph**.

jQuery html() example 3

Let's see another example of jQuery html() method that converts HTML to text.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>html demo</title>
  <style>
  p {
    margin: 8px;
    font-size: 20px;
    color: blue;
    cursor: pointer;
  }
  b {
    text-decoration: underline;
  }
  </style>
  <script src="https://code.jquery.com/jquery-1.10.2.js"></script>
</head>
<body>
<p>
  <b>Click</b> here to change the <span id="tag">html</span> to text
</p>
<script>
$( "p" ).click(function() {
  var htmlString = $( this ).html();
  $( this ).text( htmlString );
});
</script>
```

</body>

</html>

jQuery val()

There are two usage of jQuery val() method.

- It is used to get current value of the first element in the set of matched elements.
- It is used to set the value of every matched element.

Syntax:

1. $(selector).val()

It is used to get value.

1. $(selector).val(value)

It is used to set value.

1. $(selector).val(function(index,currentvalue))

It is used to set value using function.

Parameters of jQuery val() method

Parameter	**Description**
Value	It is a mandatory parameter. It is used specify the value of the attribute.
Function (index, currentvalue)	It is an optional parameter. It is used to specify a function that returns the value to set.

jQuery val() example

The val() method is primarily used to get the values of form elements. This method doesn't accept any arguments. This method returns a NULL when no option is selected and it returns an array containing the value of each selected options in the case of one or more selection.

Let's see the example of val() method.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>val demo</title>
  <style>
  p {
    color: red;
    margin: 4px;
  }
  b {
    color: blue;
  }
  </style>
  <script src="https://code.jquery.com/jquery-1.10.2.js"></script>
</head>
<body>
<p></p>
<select id="single">
  <option>Single</option>
  <option>Double</option>
 <option>Triple</option>
</select>
<script>
function displayVals() {
  var singleValues = $( "#single" ).val();
  $( "p" ).html( "<b>Value:</b> " + singleValues);
 }
 $( "select" ).change( displayVals );
 displayVals();
 </script>
 </body>
 </html>
```

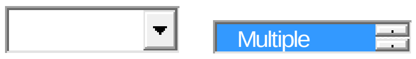
Value: Single

Let's see example of jQuery val() method with single and multiple select boxes.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>val demo</title>
  <style>
  p {
    color: red;
    margin: 4px;
  }
  b {
    color: blue;
  }
  </style>
  <script src="https://code.jquery.com/jquery-1.10.2.js"></script>
</head>
<body>
<p></p>
<select id="single">
```

```
  <option>Single</option>
  <option>Single2</option>
  <option>Single3</option>
</select>
<select id="multiple" multiple="multiple">
  <option selected="selected">Multiple</option>
  <option>Multiple2</option>
  <option>Multiple3</option>
</select>
<script>
function displayVals() {
  var singleValues = $( "#single" ).val();
  var multipleValues = $( "#multiple" ).val() || [];
  $( "p" ).html( "<b>Single:</b> " + singleValues +
    " <b>Multiple:</b> " + multipleValues.join( ", " ) );
}
$( "select" ).change( displayVals );
displayVals();
</script>
</body>
</html>
```

Single: Single **Multiple:** Multiple



jQuery val(value) example

This method is used to set a string of text, a number, an array of strings corresponding to the value of each matched element. This method facilitates you to set the value by passing in the function.

Let's see the example of val(value) method.

```
<!DOCTYPE html>
```

```
<html>
<head>
<script src="https://code.jquery.com/jquery-1.10.2.js"></script>
<script>
$(document).ready(function(){
    $("button").click(function(){
        $("input:text").val("JavaTpoint");
    });
});
</script>
</head>
<body>
<p>Name: <input type="text" name="user"></p>
<button>Set the value of the input field</button>
</body>
</html>
```

jQuery css()

The jQuery CSS() method is used to get (return)or set style properties or values for selected elements. It facilitates you to get one or more style properties. jQuery CSS() method provides two ways:

1) Return a CSS property

It is used to get the value of a specified CSS property.

Syntax:

```
css("propertyname");
```

Let's take an example to demonstrate this property.

```
<!DOCTYPE html>
```

```
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>
<script>
$(document).ready(function(){
    $("button").click(function(){
        alert("Background color = " + $("p").css("background-color"));
    });
});
</script>
</head>
<body>
<h2>This is a heading</h2>
<p style="background-color:#ff0000">The background-color of this paragraph is red.</p>
<p style="background-color:#00ff00">The background-
color of this paragraph is green.</p>
<p style="background-color:#0000ff">The background-color of this paragraph is blue.</p>
<button>Click here to get the background-color of first matched element</button>
</body>
</html>
```

2) Set a CSS property

This property is used to set a specific value for all matched element.

Syntax:

```
css("propertyname","value");
```

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

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>
<script>
$(document).ready(function(){
    $("button").click(function(){
        $("p").css("background-color", "violet");
    });
});
</script>
</head>
<body>
<p style="background-color:#ff0000">The background-color of this paragraph is red.</</p>
<p style="background-color:#00ff00">The background-color of this paragraph is green.</</p>
<p style="background-color:#0000ff">The background-color of this paragraph is blue.</</p>
<p>This paragraph has no background-color. </p>
<button>Click here to set a specific background-color of all matched element</button>
</body>
</html>
```

3) Set multiple CSS properties

It is just an extension of Set CSS property. It facilitates you to add multiple property values together.

Syntax:

```
1. css({"propertyname":"value","propertyname":"value",...});
```

Let's take an example to demonstrate this property. In this example we add two properties background-color and font-size for all element.

```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>
<script>
$(document).ready(function(){
   $("button").click(function(){
      $("p").css({"background-color": "yellow", "font-size": "200%"});
   });
});
</script>
</head>
<body>
<h2>This is a heading</h2>
<p style="background-color:#ff0000">The background-color of this paragraph is red.</p>
<p style="background-color:#00ff00">The background-color of this paragraph is green.</p>
<p style="background-color:#0000ff">The background-color of this paragraph is blue.</p>
<p>This paragraph has no background-color.</p>
<button>Click here to set multiple styles for all selected elements.</button>
</body>
</html>
```

jQuery before()

The jQuery before() method is used to insert the specified content before the selected elements. It adds the content specified by the parameter, before each element in the set of matched elements.

Syntax:

1. $(selector).before(content, function(index))

Parameters of jQuery before() method

Parameter	Description
Content	It is a mandatory parameter. It specifies the content to insert. Its possible values are: ○ HTML elements ○ jQuery objects ○ DOM elements
Function (index)	It specifies a function that returns the content which is used to insert. ○ **Index:** It provides the index position of the element in the set.

Example of jQuery before() method

Let's take an example to demonstrate the jQuery before() method.

```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></scrip
t>
<script>
$(document).ready(function(){
    $("button").click(function(){
        $("p").before("<p><b>Hello javatpoint.com</b></p>");
    });
});
</script>
```

```
</head>
<body>
<button>Insert content before each p element</button>
<p>This is a tutorial website.</p>
<p>This is a training institute.</p>
</body>
</html>
```

jQuery prepend()

The jQuery prepend() method is used to insert the specified content at the beginning (as a first child) of the selected elements. It is just the opposite of the jQuery append() method.

If you want to insert the content at the end of the selected elements, you should use the append method.

Syntax:

1. $(selector).prepend(content,function(index,html))

Parameters of jQuery prepend() method

Parameter	**Description**
Content	It is a mandatory parameter. It specifies the content which you want to insert. Its possib values are: ○ HTML elements ○ jQuery objects ○ DOM elements
Function (index, html)	It is an optional parameter. It specifies a function that returns the content which is inserte

	○ **Index:**It is used to provide the index position of the element in the set. ○ **Html:** : It provides the current HTML of the selected element.

Example of jQuery prepend() method

Let's take an example to demonstrate the jQuery prepend() method.

```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script
>
<script>
$(document).ready(function(){
   $("#btn1").click(function(){
      $("p").prepend("<b>Prepended text</b>. ");
   });
});
</script>
</head>
<body>
<p>This is the first paragraph.</p>
<p>This is the second paragraph.</p>
<button id="btn1">Prepend text</button>
</body>
</html>
```

This is the first paragraph.

This is the second paragraph.

Prepend text

jQuery prepend() example 2

```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>
<script>
$(document).ready(function(){
    $("#btn1").click(function(){
        $("p").prepend("<b>Prepended text</b>. ");
    });
    $("#btn2").click(function(){
        $("ol").prepend("<li>Prepended item</li>");
    });
});
</script>
</head>
<body>
<p>This is the first paragraph.</p>
<p>This is the second paragraph.</p>
<ol>
  <li>Item no.1</li>
  <li>Item no.2</li>
  <li>Item no.3</li>
</ol>
<button id="btn1">Prepend text</button>
<button id="btn2">Prepend list item</button>
</body>
</html>
```