WP-UNIT-2

1. Discuss about JavaScript Advantages.

JavaScript is a lightweight, interpreted **programming** language. It is designed for creating network-centric applications. It is complimentary to and integrated with Java. **JavaScript** is very easy to implement because it is integrated with HTML. It is open and cross-platform..It is a client-Side scripting language.

Advantages of Javascript:

- Javascript is the most popular **programming language** in the world and that makes it a programmer's great choice. Once you learnt Javascript, it helps you developing great front-end as well as back-end softwares using different Javascript based frameworks like jQuery, Node.JS etc.
- Javascript is everywhere, it comes installed on every modern web browser and so to learn Javascript you really do not need any special environment setup. For example Chrome, Mozilla Firefox , Safari and every browser you know as of today, supports Javascript.
- Javascript helps you create really beautiful and crazy fast websites. You can develop your website with a console like look and feel and give your users the best Graphical User Experience.
- JavaScript usage has now extended to mobile app development, desktop app development, and game development. This opens many opportunities for you as Javascript Programmer.
- Due to high demand, there is tons of job growth and high pay for those who know JavaScript. You can navigate over to different job sites to see what having JavaScript skills looks like in the job market.
- Great thing about Javascript is that you will find tons of frameworks and Libraries already developed which can be used directly in your software development to reduce your time to market.

Sample Code:-

```
<html>
  <body>
    <script language = "javascript" type = "text/javascript">
        document.write("Hello World!")
    </script>
  </body>
</html>
```

There are many useful **Javascript frameworks** and libraries available:

- Angular,React,jQuery,Vue.js,Ext.js,Ember.js,Meteor,Mithril,Node.js

Limitations of JavaScript

We cannot treat JavaScript as a full-fledged programming language. It lacks the following important features −

- Client-side JavaScript does not allow the reading or writing of files. This has been kept for security reason.
- JavaScript cannot be used for networking applications because there is no such support available.
- JavaScript doesn't have any multi-threading or multiprocessor capabilities.

Once again, JavaScript is a lightweight, interpreted programming language that allows you to build interactivity into otherwise static HTML pages.

JavaScript Development Tools

- **Microsoft FrontPage** − Microsoft has developed a popular HTML editor called FrontPage. FrontPage also provides web developers with a number of JavaScript tools to assist in the creation of interactive websites.

- **Macromedia Dreamweaver MX** − Macromedia Dreamweaver MX is a very popular HTML and JavaScript editor in the professional web development crowd. It provides several handy prebuilt JavaScript components, integrates well with databases, and conforms to new standards such as XHTML and XML.
- **Macromedia HomeSite 5** − HomeSite 5 is a well-liked HTML and JavaScript editor from Macromedia that can be used to manage personal websites effectively.

2. Javascript Coding Syntax

JavaScript can be implemented using JavaScript statements that are placed within the **<script>... </script>** HTML tags in a web page.

You can place the **<script>** tags, containing your JavaScript, anywhere within your web page, but it is normally recommended that you should keep it within the **<head>** tags.

The <script> tag alerts the browser program to start interpreting all the text between these tags as a script. A simple syntax of your JavaScript will appear as follows.

```
<script ...>
  JavaScript code
</script>
```

The script tag takes two important attributes −

- **Language** − This attribute specifies what scripting language you are using. Typically, its value will be javascript. Although recent versions of HTML (and XHTML, its successor) have phased out the use of this attribute.
- **Type** − This attribute is what is now recommended to indicate the scripting language in use and its value should be set to "text/javascript".

So your JavaScript segment will look like −

```
<script language = "javascript" type = "text/javascript">
   JavaScript code
</script>
```

This function can be used to write text, HTML, or both. Take a look at the following code.

Live Demo

```
<html>
   <body>
      <script language = "javascript" type = "text/javascript">
         <!--
            document.write("Hello World!")
         //-->
      </script>
   </body>
</html>
```

This code will produce the following result −

```
Hello World!
```

Semicolons are Optional

Simple statements in JavaScript are generally followed by a semicolon character, just as they are in C, C++, and Java. JavaScript, however, allows you to omit this semicolon if each of your statements are placed on a separate line. For example, the following code could be written without semicolons.

```
<script language = "javascript" type = "text/javascript">
  <!--
    var1 = 10
    var2 = 20
  //-->
</script>
```

But when formatted in a single line as follows, you must use semicolons −

```
<script language = "javascript" type = "text/javascript">
  <!--
    var1 = 10; var2 = 20;
  //-->
</script>
```

Note − It is a good programming practice to use semicolons.

- JavaScript is a case-sensitive language. This means that the language keywords, variables, function names, and any other identifiers must always be typed with a consistent capitalization of letters.

JavaScript supports both C-style and C++-style comments, Thus −

- Any text between a // and the end of a line is treated as a comment and is ignored by JavaScript.
- Any text between the characters /* and */ is treated as a comment. This may span multiple lines

3. Write about JavaScript Datatypes and variables.

One of the most fundamental characteristics of a programming language is the set of data types it supports. These are the type of values that can be represented and manipulated in a programming language.

JavaScript allows you to work with three primitive data types −

- **Numbers,** eg. 123, 120.50 etc.
- **Strings** of text e.g. "This text string" etc.
- **Boolean** e.g. true or false.

JavaScript also defines two trivial data types, **null** and **undefined,** each of which defines only a single value. In addition to these primitive data types, JavaScript supports a composite data type known as **object**. We will cover objects in detail in a separate chapter.

JavaScript Variables

Like many other programming languages, JavaScript has variables. Variables can be thought of as named containers. You can place data into these containers and then refer to the data simply by naming the container.

Before you use a variable in a JavaScript program, you must declare it. Variables are declared with the var keyword as follows.

Ex:-

```
<script type = "text/javascript">
  <!--
    var money;
    var name;
  //-->
</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.

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

```
alert(value);
}
function b(){
alert(value);
}
</script>
```

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

4.Write about Operators in JavaScript

JavaScript Arithmetic Operators

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

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

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.

4. Write about JavaScript Decision making Statements

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.

```
1. if(expression){
2. //content to be evaluated
3. }
```

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
}
```

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
}
```

Example:-

```
<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;
}
```

Example

```
<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>
```

6.Discuss about 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

}
```

1. **Ex:-**
2. **<script>**
3. for (i=1; i**<**=5; i++)
4. {

5. document.write(i + "**
**")
6. }
7. **</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. 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.

Ex:-

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

for...in loop is used to loop through an object's properties. As we have not discussed Objects yet, you may not feel comfortable with this loop. But once you understand how objects behave in JavaScript, you will find this loop very useful.

Syntax

The syntax of 'for..in' loop is –

```
for (variablename in object) {
  statement or block to execute
}
```

In each iteration, one property from object is assigned to variablename and this loop continues till all the properties of the object are exhausted.

Example

8. Discuss about creation of arrays in JavaScript

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

There are 3 ways to construct array in JavaScript

- By array literal

- By creating instance of Array directly (using new keyword)
- By using an Array constructor (using new keyword)

By Array Literal Syntax:- var arrayname=[value1,value2.....valueN];

```
<script>
var emp=["Sonoo","Vimal","Ratan"];
for (i=0;i<emp.length;i++){
document.write(emp[i] + "<br/>");
}
</script>
```

By Using new Keyword

Syntax :- var arrayname=new Array();

```
<script>
var i;
var emp = new Array();
emp[0] = "Arun";
emp[1] = "Varun";
emp[2] = "John";

for (i=0;i<emp.length;i++){
document.write(emp[i] + "<br>");
}
</script>
```

JavaScript array constructor (new keyword)

to create instance of array by passing arguments in constructor so that we don't have to provide value explicitly.

The example of creating object by array constructor is given below.

```
<script>
var emp=new Array("Jai","Vijay","Smith");
for (i=0;i<emp.length;i++){
document.write(emp[i] + "<br>");
}
</script>
```

JavaScript Array Methods

Let's see the list of JavaScript array methods with their description.

Methods	Description
concat()	It returns a new array object that contains two or more merged arrays.
every()	It determines whether all the elements of an array are satisfying the provided function conditions.
fill()	It fills elements into an array with static values.
from()	It creates a new array carrying the exact copy of another array element.
filter()	It returns the new array containing the elements that pass the provided function conditions.
find()	It returns the value of the first element in the given array that satisfies the specified condition.
findIndex()	It returns the index value of the first element in the given array that satisfies the specified condition.

forEach()	It invokes the provided function once for each element of an array.
indexOf()	It searches the specified element in the given array and returns the index of the first match.
pop()	It removes and returns the last element of an array.
push()	It adds one or more elements to the end of an array.
reverse()	It reverses the elements of given array.
shift()	It removes and returns the first element of an array.
slice()	It returns a new array containing the copy of the part of the given array.
sort()	It returns the element of the given array in a sorted order.

Ex:-

```
var arr=[2,4,1,8,5];
var result=arr.sort(function compare(a,b)
{
  return b-a;
});
document.writeln(result);
</script>
```

9. Explain about JavaScript Functions

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

Advantage of JavaScript function

There are mainly two advantages of JavaScript functions.

- Code reusability: We can call a function several times so it save coding.
- Less coding: It makes our program compact. We don't need to write many lines of code each time to perform a common task.

Syntax:

```
function functionName([arg1, arg2, ...argN]){
 //code to be executed
}
```

Example

```
<html>
<body>
<script>
function getcube(number){
alert(number*number*number);
}
</script>
<form>
<input type="button" value="click" onclick="getcube(4)"/>
</form>
</body>
</html>
```

10.Write about JavaScript Objects (in exam write any 4 or 5 objects)

A javaScript object is an entity having state and behavior (properties and method). For example: car, pen, bike, chair, glass, keyboard, monitor etc.

JavaScript is an object-based language. Everything is an object in JavaScript.

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

There are 2 ways to create string in JavaScript

- By string literal
- By string object (using new keyword)

By string literal

The string literal is created using double quotes. The syntax of creating string using string literal is given below:

Ex:-

```
<script>
var str="This is string literal";
document.write(str);
</script>
```

By string object (using new keyword)

The syntax of creating string object using new keyword is given below:

```
var stringname=new String("string literal");
```

Ex:-

```
<script>
var stringname=new String("hello javascript string");
document.write(stringname);
</script>
```

Methods	Description
charAt()	It provides the char value present at the specified index.
charCodeAt()	It provides the Unicode value of a character 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.
search()	It searches a specified regular expression in a given string and returns its position if a match occurs.
match()	It searches a specified regular expression in a given string and returns that regular expression if a match occurs.
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.
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.
toLocaleLowerCase()	It converts the given string into lowercase letter on the basis of

	host?s current locale.
toUpperCase()	It converts the given string into uppercase letter.

B)JavaScript Date Object

The JavaScript date object can be used to get year, month and day. You can display a timer on the webpage by the help of JavaScript date object.

You can use different Date constructors to create date object. It provides methods to get and set day, month, year, hour, minute and seconds.

Constructor

You can use 4 variant of Date constructor to create date object.

Date()
Date(milliseconds)
Date(dateString)
Date(year, month, day, hours, minutes, seconds, milliseconds)

JavaScript Date Methods

Let's see the list of JavaScript date methods with their description.

Methods	Description
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.
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.
setDate()	It sets the day value for the specified date on the basis of local time.
setDay()	It sets the particular day of the week on the basis of local time.
setFullYears()	It sets the year value for the specified date on the basis of local time.
setHours()	It sets the hour value for the specified date on the basis of local time.

setMinutes()	It sets the minute value for the specified date on the basis of local time.
setMonth()	It sets the month value for the specified date on the basis of local time.
setSeconds()	It sets the second value for the specified date on the basis of local time.
setUTCMonth()	It sets the month value for the specified date on the basis of universal time.
setUTCSeconds()	It sets the second value for the specified date on the basis of universal time.
toDateString()	It returns the date portion of a Date object.
toISOString()	It returns the date in the form ISO format string.
toString()	It returns the date in the form of string.
toTimeString()	It returns the time portion of a Date object.
valueOf()	It returns the primitive value of a Date object.

Example:-

```
<script>
var date=new Date();
var day=date.getDate();
var month=date.getMonth()+1;
var year=date.getFullYear();
document.write("<br>Date is: "+day+"/"+month+"/"+year);
</script>
```

To Display digital Timer in a Browser

```
Current Time: <span id="txt"></span>
<script>
window.onload=function(){getTime();}
function getTime(){
var today=new Date();
var h=today.getHours();
var m=today.getMinutes();
var s=today.getSeconds();
// add a zero in front of numbers<10
m=checkTime(m);
s=checkTime(s);
document.getElementById('txt').innerHTML=h+":"+m+":"+s;
setTimeout(function(){getTime()},1000);
}
//setInterval("getTime()",1000);//another way
function checkTime(i){
if (i<10){
  i="0" + i;
 }
return i;
}
</script>
```

c)JavaScript Math object

The JavaScript math object provides several constants and methods to perform mathematical operation. Unlike date object, it doesn't have constructors.

JavaScript Math Methods
Let's see the list of JavaScript Math methods with description.

Methods Description
abs() It returns the absolute value of the given number.

acos() It returns the arccosine of the given number in radians.
asin() It returns the arcsine of the given number in radians.
atan() It returns the arc-tangent of the given number in radians.
cbrt() It returns the cube root of the given number.
ceil() It returns a smallest integer value, greater than or equal to the given number.
cos() It returns the cosine of the given number.
cosh() It returns the hyperbolic cosine of the given number.
exp() It returns the exponential form of the given number.
floor() It returns largest integer value, lower than or equal to the given number.
hypot() It returns square root of sum of the squares of given numbers.
log() It returns natural logarithm of a number.
max() It returns maximum value of the given numbers.
min() It returns minimum value of the given numbers.
pow() It returns value of base to the power of exponent.

Example:

```
Random Number is: <span id="p2"></span>
<script>
document.getElementById('p2').innerHTML=Math.random();
</script>
```

D) Number Object

The **JavaScript number** object *enables you to represent a numeric value*. It may be integer or floating-point. JavaScript number object follows IEEE standard to represent the floating-point numbers.

JavaScript Number Methods

Let's see the list of JavaScript number methods with their description.

Methods	Description
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.
toExponential()	It returns the string that represents exponential notation of the given number.
toFixed()	It returns the string that represents a number with exact digits after a decimal point.
toPrecision()	It returns the string representing a number of specified precision.
toString()	It returns the given number in the form of string.

E) Boolean object

JavaScript Boolean is an object that represents value in two states: true or false. You can create the JavaScript Boolean object by Boolean() constructor as given below.

```
Boolean b=new Boolean(value);
```

JavaScript Boolean

JavaScript Boolean is an object that represents value in two states: true or false. You can create the JavaScript Boolean object by Boolean() constructor as given below.

```
Boolean b=new Boolean(value);
```

The default value of JavaScript Boolean object is false.

JavaScript Boolean Example

```
<script>
document.write(10<20);//true
```

```
document.write(10<5);//false
</script>
```

JavaScript Boolean Methods

Method	Description
toSource()	returns the source of Boolean object as a string.
toString()	converts Boolean into String.
valueOf()	converts other type into Boolean.

f) Window Object

Window Object

The window object represents a window in browser. An object of window is created automatically by the browser.Window is the object of browser, it is not the object of javascript. The javascript objects are string, array, date etc.

Method	Description
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.

11 Describe about javascript events.

The change in the state of an object is known as an Event. In html, there are various events which represents that some activity is performed by the user or by the browser. When javascript code is included in HTML, js react over these events and allow the execution. This process of reacting over the events is called Event Handling. Thus, js handles the HTML events via Event Handlers.

Mouse events:

Event Performed	Event Handler	Description
click	onclick	When mouse click on an element
mouseover	onmouseover	When the cursor of the mouse comes over the element
mouseout	onmouseout	When the cursor of the mouse leaves an element
mousedown	onmousedown	When the mouse button is pressed over the element

mouseup	onmouseup	When the mouse button is released over the element
mousemove	onmousemove	When the mouse movement takes place.

Keyboard events:

Event Performed	Event Handler	Description
Keydown & Keyup	onkeydown & onkeyup	When the user press and then release the key

Form events:

Event Performed	Event Handler	Description
focus	onfocus	When the user focuses on an element
submit	onsubmit	When the user submits the form
blur	onblur	When the focus is away from a form element
change	onchange	When the user modifies or changes the value of a form element

Window/Document events

Event Performed	Event Handler	Description
load	onload	When the browser finishes the loading of the page
unload	onunload	When the visitor leaves the current webpage, the browser unloads it
resize	onresize	When the visitor resizes the window of the browser

Example:-

```
<html>
<head> Javascript Events </head>
<body>
<script language="Javascript" type="text/Javascript">
   <!--
   function clickevent()
   {
      document.write("This is JavaTpoint");
   }
   //-->
</script>
<form>
<input type="button" onclick="clickevent()" value="Who's this?"/>
</form>
```

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

Example-2 to find simple interest

```
<html>

<form name="form1">
Enter Principle:<input type="text" name="pr"/><br>
Enter Time:<input type="text" name="tm"/><br>
Enter Rate:<input type="text" name="rate"/>    <br>
SI Value:<input type="text" name="simp"/>
<input type="button" onclick="printvalue()" value="Find SI"/>
</form>

<script type="text/javascript">
function printvalue(){
var p=parseFloat(document.form1.pr.value);
var t=parseFloat(document.form1.tm.value);
var r=parseFloat(document.form1.rate.value);
 var si=(p*t*r)/100.0;
document.form1.simp.value=si;
}
</script>
</html>
```

Example-3

```
<html>
<head> Javascript Events</head>
<body>
<h2> Enter something here</h2>
<input type="text" id="input1" onfocus="focusevent()"/>
<script>
<!--
   function focusevent()
   {
      document.getElementById("input1").style.background=" aqua";
   }
//-->
</script>
</body>
</html>
```

JQuery

12. Explain about features of JQuery and give a sample program .

jQuery is a small and lightweight JavaScript library created by John Resig in 2006 at google with a nice motto: **Write less, do more**. jQuery simplifies HTML document traversing, event handling, animating, and Ajax interactions for rapid web development. jQuery is a JavaScript toolkit designed to simplify various tasks by writing less code. Here is the list of important core features supported by jQuery −

- **DOM manipulation** − The jQuery made it easy to select DOM elements, negotiate them and modifying their content by using cross-browser open source selector engine called **Sizzle**.
- **Event handling** − The jQuery offers an elegant way to capture a wide variety of events, such as a user clicking on a link, without the need to clutter the HTML code itself with event handlers.
- **AJAX Support** − The jQuery helps you a lot to develop a responsive and featurerich site using AJAX technology.
- **Animations** − The jQuery comes with plenty of built-in animation effects which you can use in your websites.
- **Lightweight** − The jQuery is very lightweight library - about 19KB in size (Minified and gzipped).
- **Cross Browser Support** − The jQuery has cross-browser support, and works well in IE 6.0+, FF 2.0+, Safari 3.0+, Chrome and Opera 9.0+
- **Latest Technology** − The jQuery supports CSS3 selectors and basic XPath syntax.

to use jQuery

- Local Installation − You can download jQuery library on your local machine and include it in your HTML code.

- Go to the https://jquery.com/download/ to download the latest version available.
- Now put downloaded **jquery-2.1.3.min.js file** in a directory of your website, e.g. /jquery.

Many of the biggest companies on the web use jQuery.
Some of these companies are:

- Microsoft
- Google
- IBM
- Netflix

JQuery Sample Code:

```
<html>
  <head>
    <title>The jQuery Example</title>
    <script type = "text/javascript" src = "/jquery/jquery-2.1.3.min.js">
    </script>

    <script type = "text/javascript">
      $(document).ready(function() {
        document.write("Hello, World!");
      });
    </script>
  </head>

  <body>
    <h1>Hello</h1>
  </body>
</html>
```

```
<html>
<head>
 <title>First jQuery Example</title>
 <script type="text/javascript"
  src="http://ajax.googleapis.com/ajax/libs/jquery/2.1.3/jquery.min.js">
(directly call from google library if we have online connection)
 </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>
```

13. Discuss about JQuery Syntax

jQuery Syntax
The jQuery syntax is tailor-made for selecting HTML elements and performing some action on the element(s).

Basic syntax is: $(selector).action()

- A $ sign to define/access jQuery
- A (selector) to "query (or find)" HTML elements
- A jQuery action() to be performed on the element(s)

Examples:
$(this).hide() - hides the current element.
$("p").hide() - hides all <p> elements.
$(".test").hide() - hides all elements with class="test".
$("#test").hide() - hides the element with id="test"

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

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

$(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");
}
);
```

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

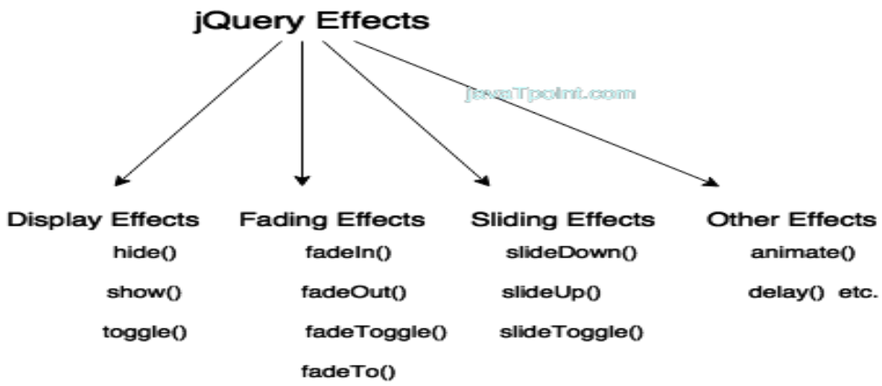
Different jQuery Selectors

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

14. Discuss about jQuery Effects

jQuery enables us to add effects on a web page. jQuery effects can be categorized into fading, sliding, hiding/showing and animation effects



jQuery provides many methods for effects on a web page. A complete list of jQuery effect methods are given below:

No.	Method	Description
1)	animate()	performs animation.
2	clearQueue()	It is used to remove all remaining queued functions from the selected elements.
3)	delay()	sets delay execution for all the queued functions on the selected elements.
4	dequeue()	It is used to remove the next function from the queue, and then execute the function.
5)	fadein()	shows the matched elements by fading it to opaque. In other words, it fades in the selected elements.
6)	fadeout()	shows the matched elements by fading it to transparent. In other words, it fades out the selected elements.
7)	fadeto()	adjusts opacity for the matched element. In other words, it

		fades in/out the selected elements.
8)	fadetoggle()	shows or hides the matched element. In other words, toggles between the fadeIn() and fadeOut() methods.
9)	finish()	It stops, removes and complete all queued animation for the selected elements.
10)	hide()	hides the matched or selected elements.
11)	queue()	shows or manipulates the queue of methods i.e. to be executed on the selected elements.
12)	show()	displays or shows the selected elements.
13)	slidedown()	shows the matched elements with slide.
14)	slidetoggle()	shows or hides the matched elements with slide. In other words, it is used to toggle between the slideUp() and slideDown() methods.
15)	slideup()	hides the matched elements with slide.
16)	stop()	stops the animation which is running on the matched elements.
17)	toggle()	shows or hides the matched elements.

Ex:- to hide and show effects.

```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
  $("#hide").click(function(){
    $("p").hide();
  });
  $("#show").click(function(){
    $("p").show();
  });
});
</script>
</head>
<body>

<p>If you click on the "Hide" button, I will disappear.</p>

<button id="hide">Hide</button>
<button id="show">Show</button>

</body>
</html>
```

Ex:- Toggle Effect

```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"></script>
<script>
$(document).ready(function(){
    $("#flip").click(function(){
        $("#panel").slideToggle("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 toggle panel</div>
<div id="panel">Hello javatpoint.com!
It is the best tutorial website to learn jQuery and other languages.</div>
</body>
</html>
```

// animate effect

```
<!DOCTYPE html>
<html>
<head>
<script src="https://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>
```

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.

- $(selector).html() It is used to return content.
- $(selector).html(content) It is used to set content.
- $(selector).html(function (index, currentcontent))

```
<!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>
```