

# SJB INSTITUTE OF TECHNOLOGY



[As per Choice Based Credit System (CBCS) scheme]

(Effective from the academic year 2016 -2017)

## “WEB TECHNOLOGY LABORATORY WITH MINI PROJECT”

Subject Code: 15CSL77



BY

Assistant Professor  
CHAITRA H K  
MANJULA H S  
SRINIDHI K S

Supporting Staff  
YASHWANTH KUMAR  
LOKESH K

Speakers

SANTHOSH K  
Software Developer

Gowtham B C  
Software Developer

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SJB INSTITUTE OF TECHNOLOGY

BGS HEALTH AND EDUCATION CITY

KENGERI, BANGALORE-5600 60

2018-19

**WEB TECHNOLOGY LABORATORY WITH MINI PROJECT**  
**[As per Choice Based Credit System (CBCS) scheme] (Effective**  
**from the academic year 2016 -2017)**  
**SEMESTER – VII**

<b>Subject Code</b>	<b>15CSL77</b>	<b>IA Marks</b>	<b>20</b>
<b>Number of Lecture Hours/Week</b>	<b>01I + 02P</b>	<b>Exam Marks</b>	<b>80</b>
<b>Total Number of Lecture Hours</b>	<b>40</b>	<b>Exam Hours</b>	<b>03</b>

**PART A**

1. Write a JavaScript to design a simple calculator to perform the following operations:  
sum, product, difference and quotient.
2. Write a JavaScript that calculates the squares and cubes of the numbers from 0 to 10 and outputs HTML text that displays the resulting values in an HTML table format.
3. Write a JavaScript code that displays text “TEXT-GROWING” with increasing font size in the interval of 100ms in RED COLOR, when the font size reaches 50pt it displays “TEXT-SHRINKING” in BLUE color. Then the font size decreases to 5pt.
4. Develop and demonstrate a HTML5 file that includes JavaScript script that uses functions for the following problems:
  - a. Parameter: A string
  - b. Output: The position in the string of the left-most vowel
  - c. Parameter: A number
  - d. Output: The number with its digits in the reverse order
5. Design an XML document to store information about a student in an engineering college affiliated to VTU. The information must include USN, Name, and Name of the College, Branch, Year of Joining, and email id. Make up sample data for 3 students. Create a CSS style sheet and use it to display the document.
6. Write a PHP program to keep track of the number of visitors visiting the web page and to display this count of visitors, with proper headings.
7. Write a PHP program to display a digital clock which displays the current time of the server.
8. Write the PHP programs to do the following:
  - a. Implement simple calculator operations.
  - b. Find the transpose of a matrix.
  - c. Multiplication of two matrices.
  - d. Addition of two matrices.
9. Write a PHP program named states.py that declares a variable states with value "Mississippi Alabama Texas Massachusetts Kansas". write a PHP program that does the following:
  - a. Search for a word in variable states that ends in xas. Store this word in element 0 of a list named states List.
  - b. Search for a word in states that begins with k and ends in s. Perform a case- insensitive comparison. [Note: Passing re.I as a second parameter to method compile performs a case-insensitive comparison.] Store this word in element1 of statesList.
  - c. Search for a word in states that begins with M and ends in s. Store this word in element 2 of the list.
  - d. Search for a word in states that ends in a. Store this word in element 3 of the list.
10. Write a PHP program to sort the student records which are stored in the database using selection sort.

**Study Experiment / Project:**

Develop a web application project using the languages and concepts learnt in the theory and exercises listed in part A with a good look and feel effects. You can use any web technologies and frameworks and databases.

Note:

1. In the examination each student picks one question from part A.
2. A team of two or three students must develop the mini project. However during the examination, each student must demonstrate the project individually.
3. The team must submit a brief project report (15-20 pages) that must include the following
  - a. Introduction
  - b. Requirement Analysis
  - c. Software Requirement Specification
  - d. Analysis and Design
  - e. Implementation
  - f. Testing

**Course outcomes:** The students should be able to:

Design and develop dynamic web pages with good aesthetic sense of designing and latest technical know-how's.

Have a good understanding of Web Application Terminologies, Internet Tools other web services.

Learn how to link and publish web sites

**Conduction of Practical Examination:**

1. All laboratory experiments from part A are to be included for practical examination.
2. Mini project has to be evaluated for 30 Marks.
3. Report should be prepared in a standard format prescribed for project work.
4. Students are allowed to pick one experiment from the lot.
5. Strictly follow the instructions as printed on the cover page of answer script.
6. Marks distribution:
  - a) Part A: Procedure + Conduction + Viva: 10 + 35 + 5 = 50 Marks
  - b) Part B: Demonstration + Report + Viva voce = 15 + 10 + 05 = 30 Marks Change of experiment is allowed only once and marks allotted to the procedure part to be made zero.

**1. Write a JavaScript to design a simple calculator to perform the following operations: sum, product, difference and quotient.**

```
<!DOCTYPE html>
<html>
  <head>
    <title>Calculator - JavaScript and HTML </title>
  </head>
  <body>

    <div id='calc-contain'>

      <form name="calculator">

        <input type="text" name="answer" />
        <br>

        <input type="button" value=" 1 " onclick="calculator.answer.value += '1'" />
        <input type="button" value=" 2 " onclick="calculator.answer.value += '2'" />
        <input type="button" value=" 3 " onclick="calculator.answer.value += '3'" />
        <input type="button" value=" + " onclick="calculator.answer.value += '+'" />
        <br/>

        <input type="button" value=" 4 " onclick="calculator.answer.value += '4'" />
        <input type="button" value=" 5 " onclick="calculator.answer.value += '5'" />
        <input type="button" value=" 6 " onclick="calculator.answer.value += '6'" />
        <input type="button" value=" - " onclick="calculator.answer.value += '-'" />
        <br/>

        <input type="button" value=" 7 " onclick="calculator.answer.value += '7'" />
        <input type="button" value=" 8 " onclick="calculator.answer.value += '8'" />
        <input type="button" value=" 9 " onclick="calculator.answer.value += '9'" />
        <input type="button" value=" x " onclick="calculator.answer.value += '*'" />
        <br/>

        <input type="button" value=" c " onclick="calculator.answer.value = "" />
        <input type="button" value=" 0 " onclick="calculator.answer.value += '0'" />
        <input type="button" value=" = " onclick="calculator.answer.value =
eval(calculator.answer.value)" />
        <input type="button" value=" / " onclick="calculator.answer.value += '/'" />
        <br/>

      </form>
      <div id="agh">
        <p>SJB Institue of Technology
        </div>
```

```
</div>

</body>
</html>
```

**CSS File:**

```
#calc-contain{
  position: relative;
  width: 400px;
  border: 2px solid black;
  border-radius: 12px;
  margin: 0px auto;
  padding: 20px 20px 100px 20px;
}
#agh{
  position: relative;
  float: right;
  margin-top: 15px;
}
#agh p{
  font-size: 20px;
  font-weight: 900;
}
input[type=button] {
  background: lightGray;
  width: 20%;
  font-size: 20px;
  font-weight: 900;
  border-radius: 7px;
  margin-left: 13px;
  margin-top: 10px;
}
input[type=button]:active {
  background-color: #3e8e41;
  box-shadow: 0 5px #666;
  transform: translateY(4px);
}
input[type=button]:hover {
  background-color: #003300;
  color: white;
}

input[type = text] {
  position: relative;
  display: block;
  width: 90%;
  margin: 5px auto;
  font-size: 20px;
  padding: 10px;
```

```

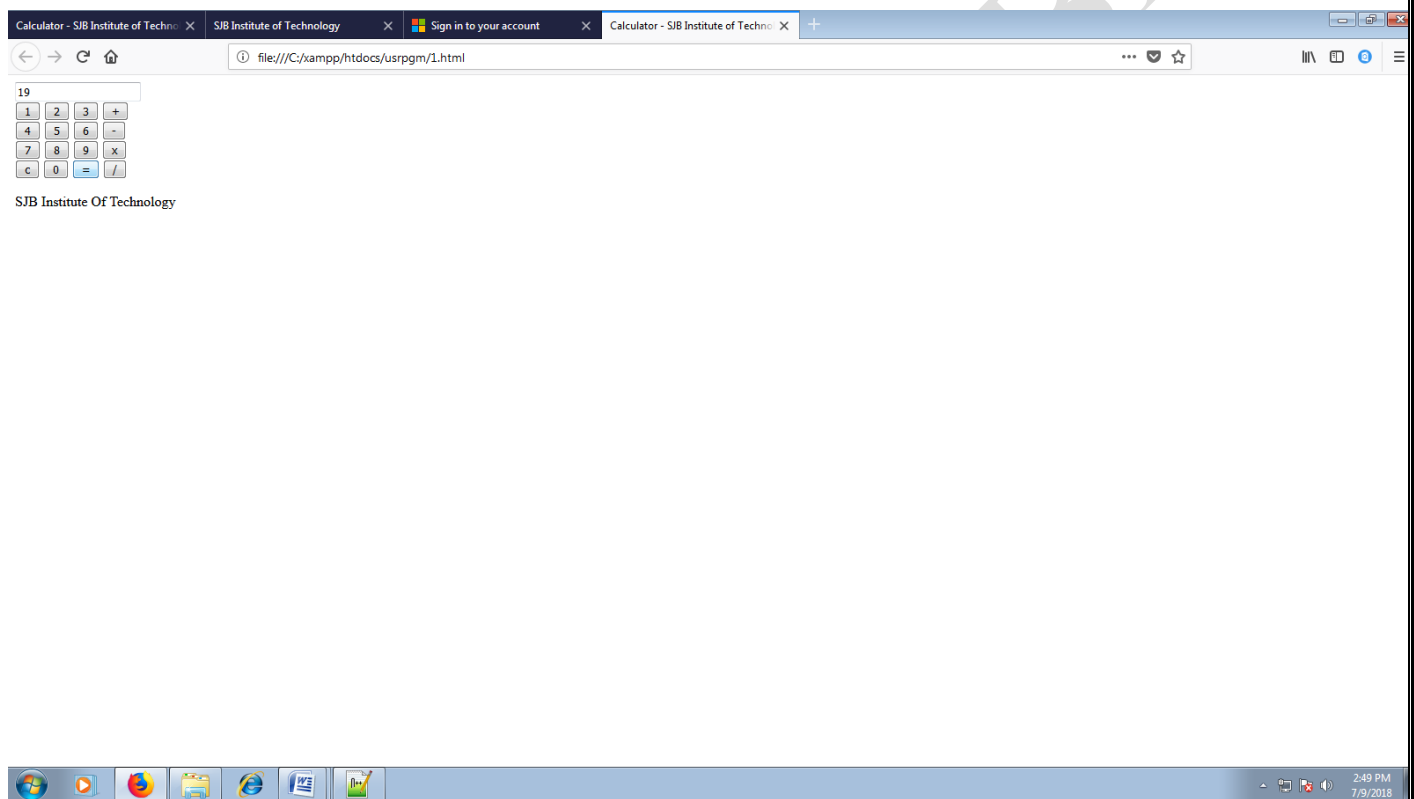
box-shadow: 4px 0px 12px black inset;
}

```

### Steps to Run:

1. open blank page in Notepad or Notepad++
2. Save the XHTML file with file extension as .html and select file type as all types or all files.
3. Save the CSS file with file extension as .css and select file type as all types or all files.
4. Open the file in the browser using the URL <http://localhost/filename.html>
5. If you are using Notepad++
  - Click on Run
  - Select launch in any browser.( Chrome, Internet Explorer, Safari, Firefox)

### Output:



2. Write a JavaScript that calculates the squares and cubes of the numbers from 0 to 10 and outputs HTML text that displays the resulting values in an HTML table format.

```

<html>
<head>
<script>
document.write('<h1 align="right">Squares and Cubes of the numbers from 0 to 10</h1>');
document.write('<center><table width="30%" border="1" bgcolor="white">');
document.write( "<tr> <th>Number</th> <th>Square</th> <th>Cube</th> </tr>" );
for(var n=0; n<=10; n++)
{
document.write( "<tr><td>" + n + "</td><td>" + n*n + "</td><td>" + n*n*n + "</td></tr>" );

```

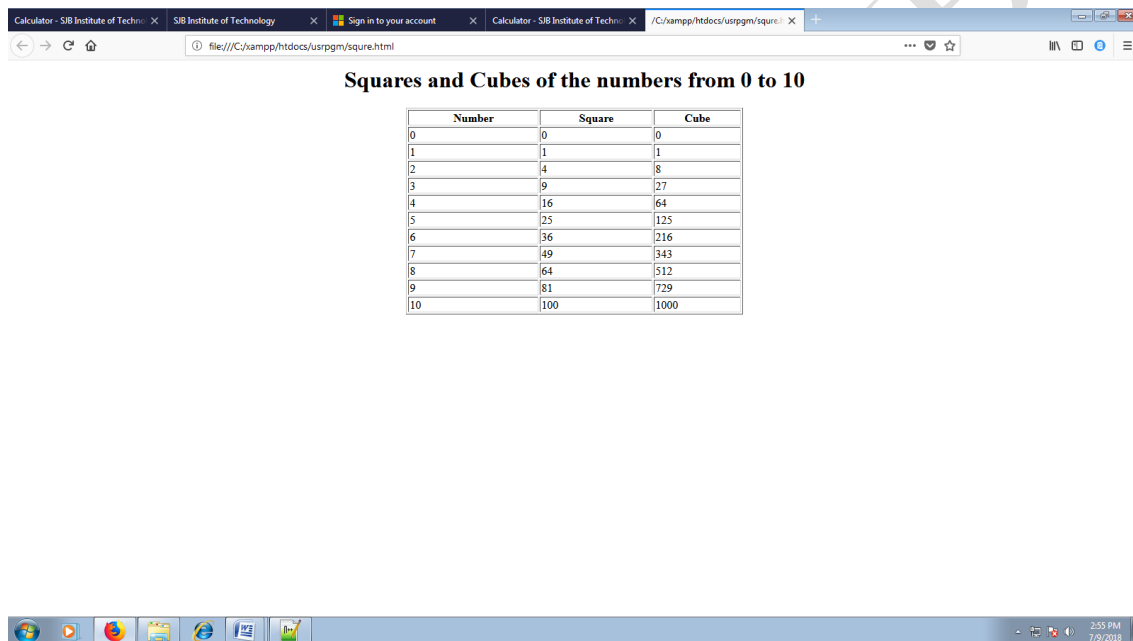
```

}
document.write( "</table>" );
</script>
</head>
</html>

```

**Steps to Run:**

1. Open blank page in Notepad or Notepad++
2. Save the XHTML file with file extension as .html and select file type as all types or all files.
3. Open the file in the browser using the URL <http://localhost/filename.html>
4. If you are using Notepad++
  - Click on Run
  - Select launch in any browser.( Chrome, Internet Explorer, Safari, Firefox)

**Output:**

**3. Write a JavaScript code that displays text “TEXT-GROWING” with increasing font size in the interval of 100ms in RED COLOR, when the font size reaches 50pt it displays “TEXT-SHRINKING” in BLUE color. Then the font size decreases to 5pt.**

Explanation:

The window object allows execution of code at specified time intervals.

These time intervals are called timing events.

The two key methods to use with JavaScript are:

`setInterval(function, milliseconds)`

Executes a function, after waiting a specified number of milliseconds, but repeats the execution of the function continuously. This function can be used as for loop.

`clearInterval(timerVariable)` The `clearInterval()` method stops the executions of the function specified in the `setInterval()` method.

**<!DOCTYPE html>**

```
<html>
<body>
<p id="myP1">TEXT-GROWING.</p>
<p id="myP2">TEXT-SHRININKING</p>
</body>
<script>
//Global declarations
var size = 10;
vari =0;
var myWait1 = setInterval(GrowText1, 100);

function GrowText1()
{
  if(size<51)
  {
    size = size + 1;
    document.getElementById("myP1").style.fontSize = (size+'pt');
    document.getElementById("myP1").style.color = "red";

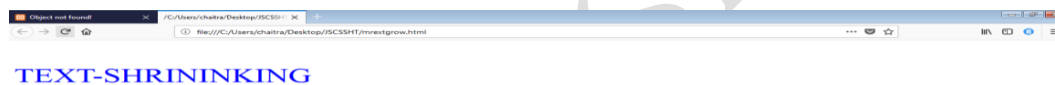
    //Hide the paragraph "text-shrinkning"
    document.getElementById("myP2").style.visibility = "hidden";
  }
  else
  {
clearInterval(myWait1);
    myWait1 = setInterval(ShrinkText1, 100);

    //Now hide the 1st paragraph and display the second paragraph
    document.getElementById("myP1").style.visibility = "hidden";
    document.getElementById("myP1").style.fontSize = '1pt';
    document.getElementById("myP2").style.visibility = "visible";
  }
}
function ShrinkText1()
{
  if(size>5)
  {
    size = size - 1;
    document.getElementById("myP2").style.fontSize = (size+'pt');
    document.getElementById("myP2").style.color = "blue";
  }
  else
  {
clearInterval(myWait1);
  }
}
</script>
</html>
```



**Steps to Run:**

1. Open blank page in Notepad or Notepad++
2. Save the XHTML file with file extension as .html and select file type as all types or all files.
3. Open the file in the browser using the URL <http://localhost/filename.html>
4. If you are using Notepad++
  - a. Click on Run
  - b. Select launch in any browser.( Chrome, Internet Explorer, Safari, Firefox)

**Output:****4. Develop and demonstrate a HTML5 file that includes JavaScript script that uses functions for the following problems:**

- a. **Parameter:** A string
- b. **Output:** The position in the string of the left-most vowel
- c. **Parameter:** A number
- d. **Output:** The number with its digits in the reverse order.

```
4a)<html>
<head><title>3A PROGRAM</title>
<SCRIPT>
function vow(st)
{
var pos;
pos=st.search(/[aeiouAEIOU]/);
```

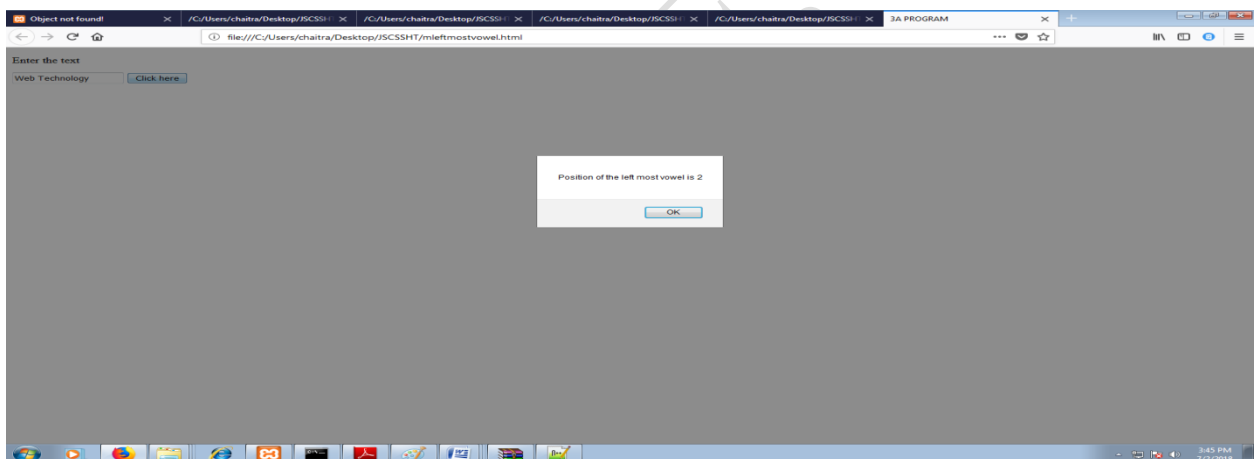
```

if(pos<0)
alert("pattern not found\n");
else
alert("Position of the left most vowel is "+(pos+1));
}
</SCRIPT>
</head>
<body>
<FORM><p>Enter the text</p>
<input type="text" id="voweltext"/>
<input type="button" value="Click here" onclick="vow(voweltext.value);"/>
</FORM></body>
</html>

```

**Steps to Run:**

1. Open blank page in Notepad or Notepad++
2. Save the XHTML file with file extension as .html and select file type as all types or all files.
3. Open the file in the browser using the URL <http://localhost/filename.html>
4. If you are using Notepad++
  - a. Click on Run
  - b. Select launch in any browser.( Chrome, Internet Explorer, Safari, Firefox)

**Output:****4b)**

```

<html>
<title>Reverse Number</title>
<script>
function rev()
{
    var n=prompt("Enter Number", " ");
    n=parseInt(n);
    var temp=0,rev=0;
    while(n>0)
    {
        temp=n%10;
        rev=rev*10+temp;
        n=n/10;
        n=parseInt(n);
    }
    document.write("The Reverse number is:",rev);
}

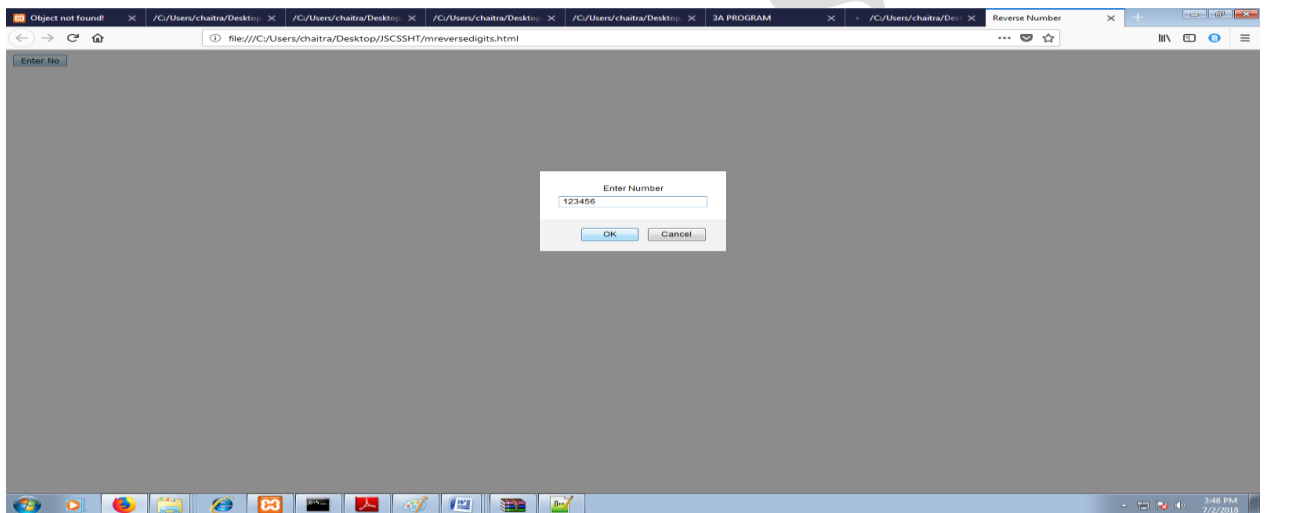
```

```
    }  
</script>  
<body>  
<form>  
<input type="button" value="Enter No" onclick="rev()";>  
</form>  
</body>  
</html>
```

#### Steps to Run:

1. Open blank page in Notepad or Notepad++
2. Save the XHTML file with file extension as .html and select file type as all types or all files.
3. Open the file in the browser using the URL <http://localhost/filename.html>
4. If you are using Notepad++
  - a. Click on Run
  - b. Select launch in any browser.( Chrome, Internet Explorer, Safari, Firefox)

#### Output:



5. Design an XML document to store information about a student in an engineering college affiliated to VTU. The information must include USN, Name, and Name of the College, Branch, Year of Joining, and email id. Make up sample data for 3 students. Create a CSS style sheet and use it to display the document.

```

<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE student_information[
<!ELEMENT student_information (ad+)>
<!ELEMENT ad (usn,name,college,branch,year,email)>
<!ELEMENT usn (#PCDATA)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT college (#PCDATA)>
<!ELEMENT branch (#PCDATA)>
<!ELEMENT year (#PCDATA)>
<!ELEMENT email (#PCDATA)>
]>
<?xml-stylesheet type="text/css" href="4a.css"?>
<student_information>
<h1>First Student Information</h1>
<ad><label>USN:</label><usn>1JB11CS001</usn></ad>
<ad><label>NAME:</label><name>ABC</name></ad>
<ad><label>COLLEGE:</label><college>SJBIT</college></ad>
<ad><label>BRANCH:</label><branch>CSE</branch></ad>
<ad><label>YEAR:</label><year>2001</year></ad>
<ad><label>EMAIL:</label><email>abc@gmail.com</email></ad>
<h1>Second Student Information</h1>
<ad><label>USN:</label><usn>1JB11CS002</usn></ad>
<ad><label>NAME:</label><name>DEF</name></ad>
<ad><label>COLLEGE:</label><college>SJBIT</college></ad>
<ad><label>BRANCH:</label><branch>CSE</branch></ad>
<ad><label>YEAR:</label><year>2001</year></ad>
<ad><label>EMAIL:</label><email>def@gmail.com</email></ad>
<h1>Third Student Information</h1>
<ad><label>USN:</label><usn>1JB11CS003</usn></ad>
<ad><label>NAME:</label><name>GHI</name></ad>
<ad><label>COLLEGE:</label><college>SJBIT</college></ad>
<ad><label>BRANCH:</label><branch>CSE</branch></ad>
<ad><label>YEAR:</label><year>2001</year></ad>
<ad><label>EMAIL:</label><email>ghi@gmail.com</email></ad>
</student_information>

```

### CSS Document for the above XML Document

```

ad{ display:block;}
  label{ font-weight:bold;color:blue;}
  usn{ font-size:14pt;color:red;}
  name{ font-size:14pt;color:red;}
  college{ font-size:14pt;color:red;}
  branch{ font-size:14pt;color:red;}
  year{ font-size:14pt;color:red;}
  email{ font-size:14pt;color:red;}

```

### Steps to Run:

1. open blank page in Notepad or Notepad++
2. Save the XHTML file with file extension as .html and select file type as all types or all files.
3. Save the CSS file with file extension as .css and select file type as all types or all files.
4. Open the file in the browser using the URL <http://localhost/filename.html>

**5. If you are using Notepad++**

- Click on Run
- Select launch in any browser.( Chrome, Internet Explorer, Safari, Firefox)

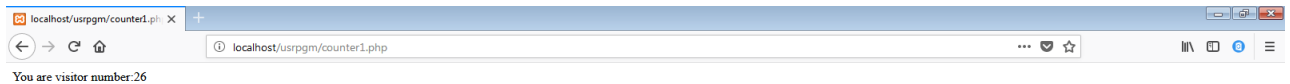
**OUTPUT:****First Student Information****USN: 1JB11CS001****NAME:ABC****COLLEGE: SJBIT****BRANCH:CSE****YEAR:2001****EMAIL:abc@gmail.com****Second Student Information****USN: 1JB11CS002****NAME:DEF****COLLEGE: SJBIT****BRANCH:CSE****YEAR:2001****EMAIL:def@gmail.com****Third Student Information****USN: 1JB11CS003****NAME: GHI****COLLEGE: SJBIT****BRANCH:CSE****YEAR:2001****EMAIL:ghi@gmail.com****6. Write a PHP program to keep track of the number of visitors visiting the web page and to display this count of visitors, with proper headings.**

```
<?php
$file = 'count.txt';
$count = strval(file_get_contents($file));
file_put_contents($file, $count + 1);
echo("You are visitor number:". $count);
?>
```

**Steps to run PHP File in XAMPP:**

1. Place your PHP file in the "htdocs" folder located under the XAMPP folder in your C:drive
2. The file path is C:\xampp\htdocs for your web server.
3. Make sure your PHP files are saved as such ,they should have the .php file extension .
4. Create file name count with the extension .txt (count.txt) and initialize to 1
5. Open up any web browser on your desktop and enter "localhost" into the address box
6. <http://localhost/usrpgm/counter1.php>

**Output:**



### 7. Write a PHP program to display a digital clock which displays the current time of the server.

In this program, we will be using the xampp browser to simulate the environment as a local server.

Navigate to: C:\xampp\php\php.ini

Line no: 1045 states the time zone of the server

Default value will be:

date.timezone=Europe/Berlin

date.timezone=Asia/Kolkata

The JavaScript function (client side) renders the time stamp of the local system.

Whereas php (server side program ) renders the time stamp from the server.

```
<html><head>
<script type="text/javascript">
    function startTime()
    {
        var d= new Date();
        var h= d.getHours();
        var m= d.getMinutes();
        var s= d.getSeconds();
        document.getElementById("txt").innerHTML= h+" : "+m+" : "+s;
        setTimeout('startTime()', 1000);
    }
</script>
<style type="text/css">
    h1
```

```

        {
            font-size: 70px;
        }
    </style></head>

    <body bgcolor = "#349" text="white" onload="startTime()">
    <br>
        <h1 align= "center"> The time from the local system is:
            <span id= "txt"></span>
        </h1>
    </body>
    </html>
</br>
</br></br></br>

```

**//To display the time or server**

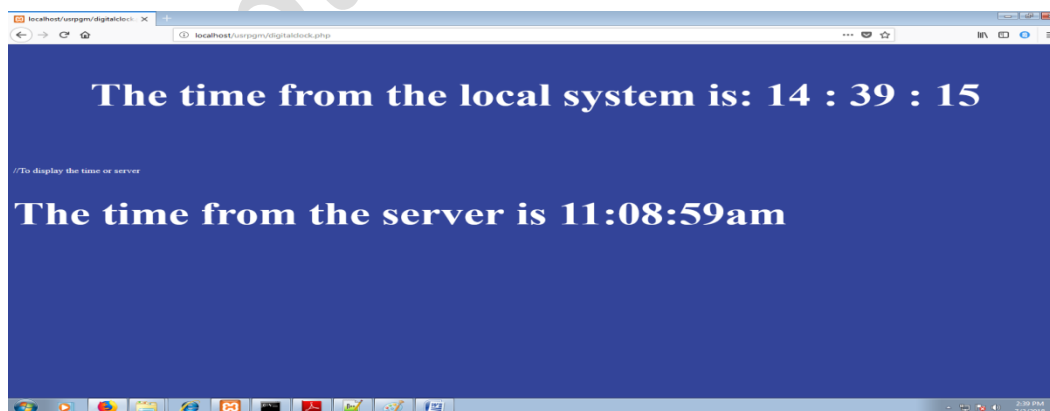
```

<?php
    $today = date("H:i:s");           // 17:16:18
    ?>
    <!DOCTYPE html>
<html>
<body>
<h1>
<?php echo "The time from the server is " . date("h:i:sa");?>
</h1>
</body>
</html>

```

**Steps to run PHP File in XAMPP:**

1. Place your PHP file in the "htdocs" folder located under the XAMPP folder in your C:drive
2. The file path is C:\xampp\htdocs for your web server.
3. Make sure your PHP files are saved as such ,they should have the .php file extension .
4. Open up any web browser on your desktop and enter "localhost" into the address box
5. <http://localhost/usrpgm/digitalclock.php>

**Output:**

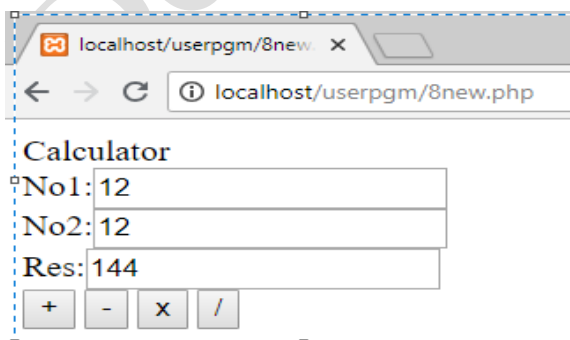
**8. Write the PHP Programs to do the following****a) Implement simple calculator operations.**

POST back to the same page, hence form action="" ( is kept empty)

```
<?php
if(isset($_POST['sub']))
{
    $txt1=$_POST['n1'];
    $txt2=$_POST['n2'];
    $oprnd=$_POST['sub'];

    if($oprnd=="+")
        $res=$txt1+$txt2;
    else if($oprnd=="-")
        $res=$txt1-$txt2;
    else if($oprnd=="x")
        $res=$txt1*$txt2;
    else if($oprnd=="/")
        $res=$txt1/$txt2;
}
?>

<html>
<form method="post" action="">
    Calculator
</br>
    No1:<input name="n1" value="<?php echo $txt1; ?>" >
</br>
    No2:<input name="n2" value="<?php echo $txt2; ?>" >
</br>
    Res:<input name="res" value="<?php echo $res; ?>" >
</br>
    <input type="submit" name="sub" value="+">
    <input type="submit" name="sub" value="-">
    <input type="submit" name="sub" value="x">
    <input type="submit" name="sub" value="/">
</form>
</html>
```

**Output:**







```

    {
        echo "The matrix multiplication is not possible.";
    }
?>

```

**Output:**

```

The order of the matrix A is:2x2
The order of the matrix B is:2x2
The input matrix A is:
1      2
4      5
The input matrix B is:
1      2
4      5

The output Transpose of matrix is:
1      4
2      5

The sum of matrix is:
2      4
8      10

The Multiplication of matrix is:
1      4
16     25

```

**8 Steps to run PHP File in XAMPP:**

1. Place your PHP file in the “htdocs” folder located under the XAMPP folder in your C:drive
2. The file path is C:\xampp\htdocs for your web server.
3. Make sure your PHP files are saved as such , they should have the .php file extension .
4. Open up any web browser on your desktop and enter “localhost” into the address box
5. <http://localhost/usrpgm/digitalclock.php>

9] Write a PHP program named states.py that declares a variable states with the value "Mississippi Alabama Texas Massachusetts Kansas". Write a php program that does the following:

- a) Search for a word in variable states that ends in xas. Store this word in element 0 of a list named statesList.
- b) Search for a word in states that begins with k and ends in s. Perform a case insensitive comparison. [Note: Passing re.I as s second parameter to method compile performs a case-insensitive comparison.]Store this word in element 1 of statesList.
- c) Search for a word in states that begins with M and ends in s. Store this element in 2 of the list.
- d) Search for a word in states that ends in a. Store this word in element 3 of the list.

```

<?php
header('Content-Type: text/plain');
$allTheStates = "Mississippi Alabama Texas Massachusetts Kansas tuxas";
$statesArray = [];
$states1 = explode(' ', $allTheStates);

    $i = 0;

```

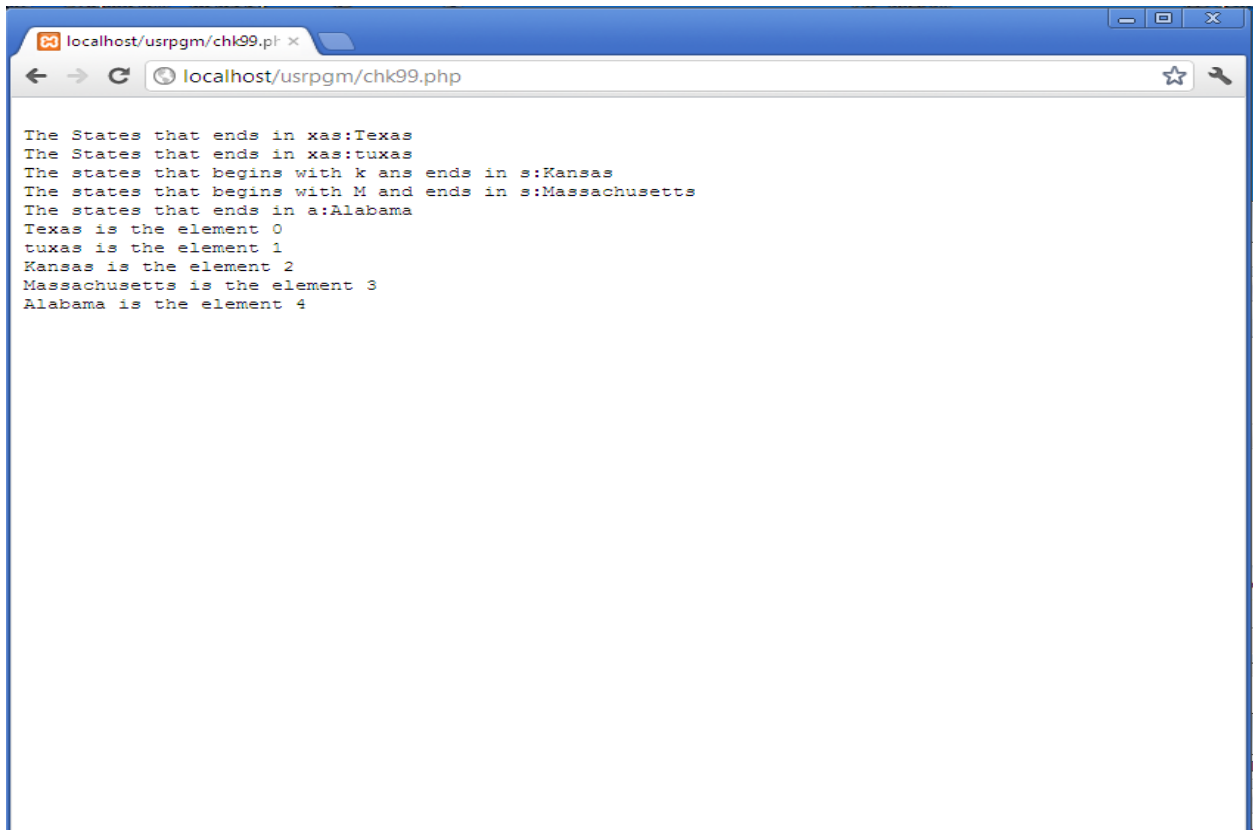
```
//states that ends in xas
foreach($states1 as $state) {
    if(preg_match( '/xas$/', ($state)))
    {
        $statesArray[$i] = ($state);
        $i = $i + 1;
        print "\n\nThe States that ends in xas:" . $state;
    }
}

//states that begins with k and ends in s
foreach($states1 as $state)
{
    if(preg_match('/^k.*s$/i', ($state)))
    {
        $statesArray[$i] = ($state);
        $i = $i + 1;
        echo "\n\nThe states that begins with k ans ends in s:" . $state;
    }
}

//states that begins with M and ends in s
foreach($states1 as $state) {
    if(preg_match('/^M.*s$/', ($state)))
    {
        $statesArray[$i] = ($state);
        $i = $i + 1;
        echo "\n\nThe states that begins with M and ends in s:" . $state;
    }
}

//states that ends in a
foreach($states1 as $state) {
    if(preg_match('/a$/', ($state)))
    {
        $statesArray[$i] = ($state);
        $i = $i + 1;
        echo "\n\nThe states that ends in a:" . $state;
    }
}

//}
foreach( $statesArray as $element => $value ){
print( "\n" . $value." is the element ". $element);
}
?>
```



```
localhost/usrpgm/chk99.php
localhost/usrpgm/chk99.php

The States that ends in xas:Texas
The States that ends in xas:texas
The states that begins with k ans ends in s:Kansas
The states that begins with M and ends in s:Massachusetts
The states that ends in a:Alabama
Texas is the element 0
texas is the element 1
Kansas is the element 2
Massachusetts is the element 3
Alabama is the element 4
```

**Output:**

The States that ends in xas:Texas  
The States that ends in xas:texas  
The states that begins with k ans ends in s:Kansas  
The states that begins with M and ends in s:Massachusetts  
The states that ends in a:Alabama  
Texas is the element 0  
texas is the element 1  
Kansas is the element 2  
Massachusetts is the element 3  
Alabama is the element 4

**10. Write a PHP program to sort the student records which are stored in the database using selection sort.**

```
<!DOCTYPE html>
<html>
<head>
  <title>Selection Sort</title>
</head>

<!-- jQuery 3 -->
<script src="jquery.min.js"></script>
<script type="text/javascript">
var globalData=null;
var tempData;
```

```
if(tempData===null||tempData===undefined){
    tempData={ };
}

tempData={
    saveRecord:function(){
        var url="ajaxInfo.php";
        var formEQData = new FormData($('#formRecord')[0]);
        formEQData.append("saveRecord","saveRecord");

        $.ajax({
            type:"POST",
            url:url,
            async: false,
            dataType: 'json',
            cache : false,
            processData: false,
            contentType: false,
            data:formEQData,
            success: function(obj) {
                alert(obj.msg);
                tempData.getRecord(); // Calling function to fetch the Record from DB.
            }
        });
    },
    getRecord:function(){
        var url="ajaxInfo.php";
        var myData = {getRecord:'getRecord'};

        $.ajax({
            type:"POST",
            url:url,
            async: false,
            dataType: 'json',
            data:myData,
            success: function(obj) {
                globalData=obj.studentArr; // Assigning to Global Variable

                var content="";
                $('#tableRow').html("");
                for(var i = 0; i<obj.studentArr.length; i++) {
                    content+='<tr><td>'+obj.studentArr[i].stu_id+'</td><td>'+obj.studentArr[i].stu_name+'</td>'
                    +'<td>'+obj.studentArr[i].stu_mobile+'</td><td>'+obj.studentArr[i].stu_email+'</td></tr>';
                }
                $('#tableRow').append(content);
            }
        });
    },
    selectionSort:function(){
        var url="ajaxInfo.php";
        var myData = {selectionSort:'selectionSort'};
    }
}
```

```
$.ajax({
  type:"POST",
  url:url,
  async: false,
  dataType: 'json',
  data:myData,
  success: function(obj) {
    var content="";
    $('#tableRow').html("");

    for(var i = 0; i<obj.sortedArr.length; i++) {
      for(var j = 0; j<globalData.length; j++) {
        if(obj.sortedArr[i]==globalData[j].stu_id){
          content+="|<td>'"+globalData[j].stu_id+'</td><td>'"+globalData[j].stu_name+'</td>'
+<td>'"+globalData[j].stu_mobile+'</td><td>'"+globalData[j].stu_email+'</td></tr>";
        }
      }
    }
    $('#tableRow').append(content);

  }
});
},
};

$(document).ready(function() {
  tempData.getRecord();
});
</script>
<body>

<center>
  <h1> Add Student Record </h1>
  <form id="formRecord">
    <table border="0">
      <tr>
        <td>Name</td>
        <td><input type="text" name="name" id="name"></td>
      </tr>
      <tr>
        <td>Mobile Number</td>
        <td><input type="number" name="mobile" id="mobile"></td>
      </tr>
      <tr>
        <td>Email ID</td>
        <td><input type="email" name="email" id="email"></td>
      </tr>
    </table><br>
|  |

```

```
<button type="button" onclick="tempData.saveRecord();" style="width:150px;">Add  
Student</button>
```

```
</form>  
<br/><br/>
```

```
<div style="overflow-x: scroll;height: 600px;width:60%;">
```

```
<button type="button" style="width:150px;float: right;" onclick="tempData.selectionSort();" >  
Selection Sort</button>
```

```
<br/><br/>
```

```
<table border=1 style="width:100%;">
```

```
<thead>
```

```
<tr style="text-align: left;">
```

```
<th>Student ID</th>
```

```
<th>Name</th>
```

```
<th>Mobile</th>
```

```
<th>Email</th>
```

```
</tr>
```

```
</thead>
```

```
<tbody id="tableRow">
```

```
</tbody>
```

```
</table>
```

```
</div>
```

```
</center>
```

```
</body>
```

```
</html>
```

```
<?php
```

```
require_once('db.php');
```

```
/* Fetching the initial data */
```

```
/*$Query = 'select * from info';
```

```
$fetchRec = mysqli_query($con,$Query) or die(mysqli_error());*/
```

```
/* Add the record to DATABASE */
```

```
if(isset($_POST['saveRecord']))
```

```
{
```

```
    $name = $_POST['name'];
```

```
    $mobile = $_POST['mobile'];
```

```
    $email = $_POST['email'];
```

```
if($name !=" && $mobile !=" && $email !=")
```

```
{
```

```
    $stu_id = rand(0,99999); //random number generation
```

```
    $Query ="insert into info(stu_id,stu_name,stu_mobile,stu_email)
```

```
    values($stu_id,$name,$mobile,$email)";
```

```
    mysqli_query($con,$Query) or die(mysqli_error());
```

```
    $msg = 'Record Saved Successfully !';
```



```
}
else{
    $msg = "Text Field is empty !";
}

$status['msg'] = $msg;
echo json_encode($status);
mysqli_close($con);
}

/* read all studrnt data from DATABASE */
if(isset($_POST['getRecord']))
{
    $sql = "select * from info";
    $fetchRec = mysqli_query($con,$sql) or die(mysqli_error());
    while ($row=mysqli_fetch_array($fetchRec))
    {
        $stu_id = $row['stu_id'];
        $stu_name = $row['stu_name'];
        $stu_mobile = $row['stu_mobile'];
        $stu_email = $row['stu_email'];

        $studentArr[]=array('stu_id' =>$stu_id,
                            'stu_name' =>$stu_name,
                            'stu_mobile' =>$stu_mobile,
                            'stu_email' =>$stu_email
                            );
    }

    $status['studentArr'] = $studentArr;
    echo json_encode($status);
    mysqli_close($con);
}

/* read data from DATABASE */
if(isset($_POST['selectionSort']))
{
    $sql = "select * from info";
    $fetchRec = mysqli_query($con,$sql) or die(mysqli_error());
    while ($row=mysqli_fetch_array($fetchRec)) {
        $getStuID[] = $row['stu_id'];
    }
    $selectionArr = selection_sort($getStuID); // calling selection Sort function

    $status['sortedArr'] = $selectionArr;
    echo json_encode($status);
    mysqli_close($con);
}

function selection_sort($data)
{
    for($i=0; $i<count($data)-1; $i++) {
```

```

    $min = $i;
    for($j=$i+1; $j<count($data); $j++) {
        if ($data[$j]<$data[$min]) {
            $min = $j;
        }
    }
    $data = swap_positions($data, $i, $min);
}
return $data;
}

function swap_positions($data1, $left, $right) {
    $backup_old_data_right_value = $data1[$right];
    $data1[$right] = $data1[$left];
    $data1[$left] = $backup_old_data_right_value;
    return $data1;
}
?>

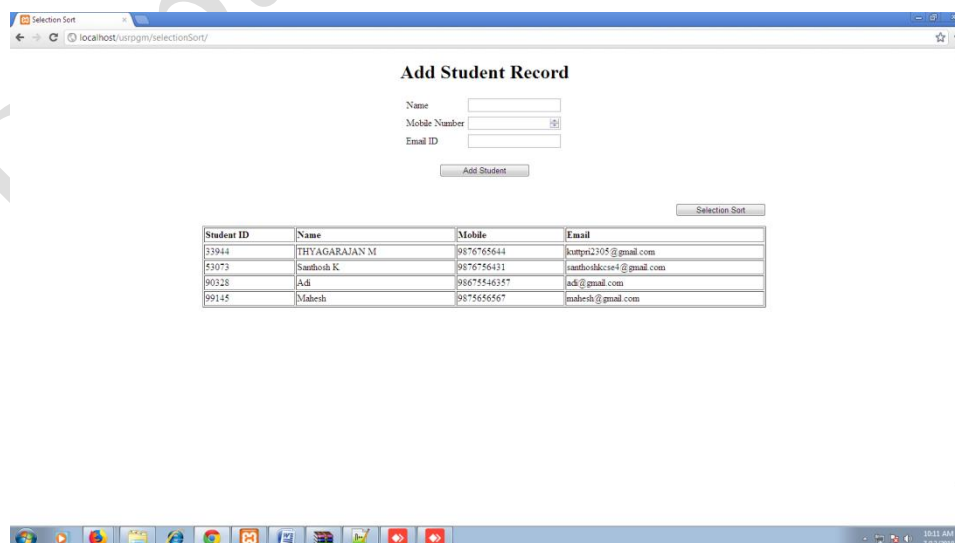
<?php
error_reporting(0);
session_start();

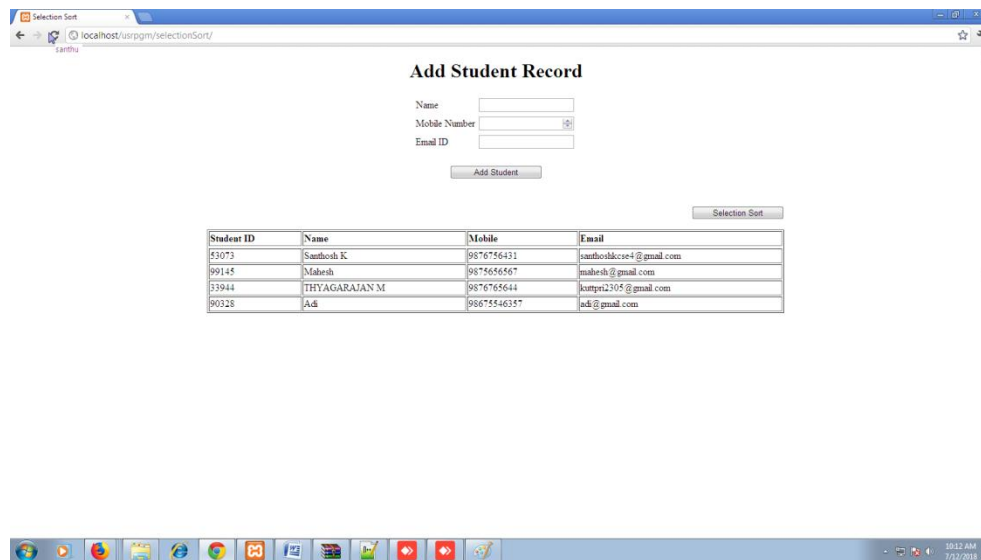
$con = mysqli_connect("localhost","root","","student_db");
if (mysqli_connect_errno()) {
    echo "Failed to connect to MySQL: " . mysqli_connect_error();
}
?>

```

**Steps to run**

1. Create Database in phpmyadmin. [refer student\_db.sql file]
2. run file browser.
1. ajaxInfo.php - Database related file [Save, Read, SelectionSort]
2. db.php - connect to database.
3. index.php - Home file [Ajax, JSon,]
4. jquery.min.js - [it will helps to run the ajax] [given by teechar]





### Study Experiment / Project: List of mini Project Examples

1. Shopping cart
2. CRM(customer relationship management )
3. Library Management
4. SIMS(Student information System)
5. Profile Websites
6. Garment Root
7. Dashboards

**WEB TECHNOLOGY Lab VIVA Questions :-****CSS :-****1. What are Cascading Style Sheets?**

Called (CSS) is a list of statements (or rules) that can assign various rendering properties to HTML elements. Style rules can be specified for a single element occurrence, multiple elements, and entire document, or even multiple documents at once.

**2. What is class?**

A group of instances of the same element to which a unique style can be attached.

**3. What is grouping?**

Gathering into a comma separated list two or more selectors that share the same style or into a semicolon separated list two or more declarations that are attached to the same selector.

**4. What is ID selector?**

ID selector is an individually identified (named) selector to which a specific style is declared. Using the ID attribute, the declared style can then be associated with one and only one HTML element per document as to differentiate it from all the other elements. They use the # character followed by a name.

**5. What is contextual selector?**

A contextual selector addresses a specific occurrence of an element. It is a string of individual selectors separated by white space, a search pattern, where only the last element in the pattern is addressed providing it matches the specified context. Example: “td li {color: red} “

**6. What does \ABCD (and \ABCDE) mean?**

CSS allows Unicode characters to be entered by number. For example, if a class value in some Russian document contains Cyrillic letters EL PE (Unicode numbers 041B and 041F) and you want to write a style rule for that class, you can put that letter into the style sheet by writing:

```
.\041B\041F {font-style: italic;}
```

This works on all keyboards, so you don't need a Cyrillic keyboard to write class names in Russian or another language that uses that script.

**7. What are the advantages / disadvantages of various style methods?****External Style Sheet Advantages**

- Can control styles for multiple documents at once.
- Classes can be created for use on multiple HTML element types in many documents.
- Selector and grouping methods can be used to apply styles under complex contexts.

**External Style Sheet Disadvantages**

- An extra download is required to import style information for each document.
- The rendering of the document may be delayed until the external style sheet is loaded.

**Embedded Style Sheet Advantages**

- Classes can be created for use on multiple tag types in the document.
- Selector and grouping methods can be used to apply styles under complex contexts
- No additional downloads necessary to receive style information.

**Inline Style Disadvantages**

- Does not distance style information from content.

- Cannot control styles for multiple documents at once.
- Author cannot create or control classes of elements to control multiple element types within the document.
- Selector grouping methods cannot be used to create complex element addressing scenarios.

**8. What is property?**

Property is a stylistic parameter (attribute) that can be influenced through CSS, e.g. font or width. There must always be a corresponding value or values set to each property.

**9. What is the CSS clear property?**

The clear property specifies which sides of an element where other floating elements are not allowed.

This method cannot control styles for multiple documents at once.

Inline Style Advantages

Useful for small quantities of style definitions. Can override other style specification methods at the local level so only exceptions need to be listed in conjunction with other style methods.

**10. What are the necessities of using HTML forms?**

1. Gathering user information
2. Conducting Surveys
3. Interactive services

**11. What are the sequences of steps for each HTTP request from a client to the server?**

1. Making the connection
2. Making a request
3. The response
4. Closing the connection

HTML :-

**1. What is HTML?**

HTML (HyperText Markup Language) is a Universal language which allows an individual using special code to create web pages to be viewed on the Internet.

**2. What is a tag?**

In HTML, a tag tells the browser what to do. When you write an HTML page, you enter tags for many reasons; to change the appearance of text, to show a graphic, or to make a link to another page.

**3. What is the simplest HTML page?**

HTML Code:

```
<HTML><HEAD><TITLE>This is my page title! </TITLE>
</HEAD>
<BODY> This is my message to the world! </BODY></HTML>
```

Browser Display: This is my message to the world!

**4. How do I create frames? What is a frameset?**

Frames allow an author to divide a browser window into multiple (rectangular) regions. Multiple documents can be displayed in a single window, each within its own frame. Graphical browsers allow these frames to be scrolled independently of each other, and links can update the document displayed in one frame without affecting the

others. You can't just "add frames" to an existing document. Rather, you must create a frameset document that defines a particular combination of frames, and then display your content documents inside those frames. The frameset document should also include alternative non-framed content in a NOFRAMES element.

**5. How can I include comments in HTML?**

An HTML comment begins with "<!--", ends with "-->", and does not contain "--" or ">" anywhere in the comment.

**6. What is a Hypertext link?**

A hypertext link is a special tag that links one page to another page or resource. If you click the link, the browser jumps to the link's destination.

**7. What is a DOCTYPE? Which one do I use?**

According to HTML standards, each HTML document begins with a DOCTYPE declaration that specifies which version of HTML the document uses. Many browsers use the document's DOCTYPE declaration to determine whether to use a stricter, more standards-oriented layout mode, or to use a "quirks" layout mode that attempts to emulate older, buggy browsers.

**8. Can I nest tables within tables?**

Yes, a table can be embedded inside a cell in another table. The main caveat about nested tables is that older versions of Netscape Navigator have problems with them if you don't explicitly close your TR, TD, and TH elements. To avoid problems, include closing tags for your TR, TD, and TH even though the HTML specifications don't require them.

**9. How do you align a table to the right (or left)?**

You use the <table align="right"> property to float a table to the right. Put left in place of right to float right.

**10. How can I use tables to structure forms?**

Small forms are sometimes placed within a TD element within a table. This can be useful for positioning a form relative to other content, but it doesn't help position the form-related elements relative to each other. The table must be within the form and then use the table to position the form elements.

**11. How do I open a link into a new window?**

Add target="\_blank" to your link syntax.

\*\*\*\*\*

## **Java Server Pages (JSP) ::**

**1. What are the advantages of JSP over Servlet?**

JSP is a serverside technology to make content generation a simple appear. The advantage of JSP is that they are document-centric. Servlets, on the other hand, look and act like programs. A Java Server Page can contain Java program fragments that instantiate and execute Java classes, but these occur inside an HTML template file and are primarily used to generate dynamic content. Some of the JSP functionality can be achieved on the client, using JavaScript. The power of JSP is that it is server-based and provides a framework for Web application development.

**2. What is the life-cycle of JSP?**

When a request is mapped to a JSP page for the first time, it translates the JSP page into a servlet class and compiles the class. It is this servlet that services the client requests.

A JSP page has seven phases in its lifecycle, as listed below in the sequence of occurrence:

Translation

Compilation

Loading the class

Instantiating the class

jspInit() invocation

\_jspService() invocation

jspDestroy() invocation

### 3. What is the jspInit() method?

The jspInit() method of the javax.servlet.jsp.JspPage interface is similar to the init() method of servlets. This method is invoked by the container only once when a JSP page is initialized. It can be overridden by a page author to initialize resources such as database and network connections, and to allow a JSP page to read persistent configuration data.

### 4. What is the \_jspService() method?

The \_jspService() method of the javax.servlet.jsp.HttpJspPage interface is invoked every time a new request comes to a JSP page.

This method takes the HttpServletRequest and HttpServletResponse objects as its arguments. A page author cannot override this method, as its implementation is provided by the container.

### 5. What is the jspDestroy() method?

The jspDestroy() method of the javax.servlet.jsp.JspPage interface is invoked by the container when a JSP page is about to be destroyed. This method is similar to the destroy() method of servlets. It can be overridden by a page author to perform any cleanup operation such as closing a database connection.

### 6. What JSP lifecycle methods can I override?

You cannot override the \_jspService() method within a JSP page. You can however, override the jspInit() and jspDestroy() methods within a JSP page. jspInit() can be useful for allocating resources like

database connections, network connections, and so forth for the JSP page. It is good programming practice to free any allocated resources within jspDestroy().

### 7. How can I override the jspInit() and jspDestroy() methods within a JSP page?

The jspInit() and jspDestroy() methods are each executed just once during the lifecycle of a JSP page and are typically declared as JSP

declarations:

```
<%!  
public void jspInit() {  
    ...  
}  
%>  
<%!  
public void jspDestroy() {  
    ...  
}  
%>
```

### 8. What are implicit objects in JSP?

Implicit objects in JSP are the Java objects that the JSP Container makes available to developers in each page. These objects need not be declared or instantiated by the JSP author. They are automatically instantiated by the container and are accessed using standard variables; hence, they are called implicit objects. The implicit objects available in JSP are as follows:

- request
- response
- pageContext
- session
- application
- out
- config
- page
- exception

The implicit objects are parsed by the container and inserted into the generated servlet code. They are available only within the `jspService` method and not in any declaration.

**9. What are the different types of JSP tags?**

The different types of JSP tags are as follows:

**10. What are JSP directives?**

JSP directives are messages for the JSP engine. i.e., JSP directives serve as a message from a JSP page to the JSP container and control the processing of the entire page. They are used to set global values such as a class declaration, method implementation, output content type, etc.

They do not produce any output to the client.

Directives are always enclosed within `<%@ ..... %>` tag.

Ex: page directive, include directive, etc.

**11. What is page directive?**

A page directive is to inform the JSP engine about the headers or facilities that page should get from the environment.

Typically, the page directive is found at the top of almost all of our JSP pages. There can be any number of page directives within a JSP page (although the attribute – value pair must be unique).

The syntax of the include directive is: `<%@ page attribute="value">`

Example: `<%@ include file="header.jsp" %>`

**12. What are the attributes of page directive?**

There are thirteen attributes defined for a page directive of which the important attributes are as follows:

- import: It specifies the packages that are to be imported.
- session: It specifies whether a session data is available to the JSP page.
- contentType: It allows a user to set the content-type for a page.
- isELIgnored: It specifies whether the EL expressions are ignored when a JSP is translated to a servlet.

**13. What is the include directive?**

There are thirteen attributes defined for a page directive of which the important attributes are as follows:

The include directive is used to statically insert the contents of a resource into the current JSP.



This enables a user to reuse the code without duplicating it, and includes the contents of the specified file at the translation time.

The syntax of the include directive is as follows:

```
<%@ include file = "FileName" %>
```

This directive has only one attribute called file that specifies the name of the file to be included.

#### 14. What are the JSP standard actions?

The JSP standard actions affect the overall runtime behavior of a JSP page and also the response sent back to the client.

They can be used to include a file at the request time, to find or instantiate a JavaBean, to forward a request to a new page, to generate a browser-specific code, etc.

Ex: include, forward, useBean, etc. object

#### 15. What are the standard actions available in JSP?

The standard actions available in JSP are as follows:

`<jsp:include>`: It includes a response from a servlet or a JSP page into the current page. It differs from an include directive in that it includes a resource at request processing time, whereas the include directive includes a resource at translation time.

`<jsp:forward>`: It forwards a response from a servlet or a JSP page to another page.

`<jsp:useBean>`: It makes a JavaBean available to a page and instantiates the bean.

`<jsp:setProperty>`: It sets the properties for a JavaBean.

`<jsp:getProperty>`: It gets the value of a property from a JavaBean component and adds it to the response.

`<jsp:param>`: It is used in conjunction with `<jsp:forward>`, `<jsp:useBean>`, or `<jsp:plugin>`; to add a parameter to a request. These parameters are provided using the name-value pairs.

`<jsp:plugin>`: It is used to include a Java applet or a JavaBean in the current JSP page.

#### 16. What is the `<jsp:useBean>` standard action?

The `<jsp:useBean>` standard action is used to locate an existing JavaBean or to create a JavaBean if it does not exist. It has attributes to identify the object instance, to specify the lifetime of the bean, and to specify the fully qualified classpath and type.

#### 17. What are the scopes available in `<jsp:useBean>`?

The scopes available in `<jsp:useBean>` are as follows:

page scope: It specifies that the object will be available for the entire JSP page but not outside the page.

request scope: It specifies that the object will be associated with a particular request and exist as long as the request exists.

application scope: It specifies that the object will be available throughout the entire Web application but not outside the application.

session scope: It specifies that the object will be available throughout the session with a particular client.

#### 18. What is the `<jsp:forward>` standard action?

- The `<jsp:forward>` standard action forwards a response from a servlet or a JSP page to another page.
- The execution of the current page is stopped and control is transferred to the forwarded page.
- The syntax of the `<jsp:forward>` standard action is :

```
<jsp:forward page="/targetPage" />
```

Here, targetPage can be a JSP page, an HTML page, or a servlet within the same context.

• If anything is written to the output stream that is not buffered

before `<jsp:forward>`, an `IllegalStateException` will be thrown.

Note : Whenever we intend to use `<jsp:forward>` or `<jsp:include>` in a page, buffering should be enabled. By default buffer is enabled.

**19.**What is the `<jsp:include>` standard action?

The `<jsp:include>` standard action enables the current JSP page to include a static or a dynamic resource at runtime. In contrast to the include directive, the include action is used for resources that change

frequently. The resource to be included must be in the same context. The syntax of the `<jsp:include>` standard action is as follows:

```
<jsp:include page="targetPage" flush="true"/>
```

Here, `targetPage` is the page to be included in the current JSP.

**20.**What is the difference between include directive and include action?

**Include directive** The include directive, includes the content of the specified file during the translation phase—when the page is converted to a servlet. The include action, includes the response generated by executing the specified page (a JSP page or a servlet) during the request processing phase—when the page is requested by a user. The include directive is used to statically insert the contents of a resource into the current JSP. The include standard action enables the current JSP page to include a static or a dynamic resource at runtime.

Use the include directive if the file changes rarely. It's the fastest mechanism. Use the include action only for content that changes often, and if which page to include cannot be decided until the main page is requested.

**21.**Differentiate between `pageContext.include` and `jsp:include`?

The `<jsp:include>` standard action and the `pageContext.include()` method are both used to include resources at runtime. However, the `pageContext.include()` method always flushes the output of the current page before including the other components, where as `<jsp:include>` flushes the output of the current page only if the value

of `flush` is explicitly set to true as follows:

```
<jsp:include page="/index.jsp" flush="true"/>
```

**22.**What is the `jsp:setProperty` action?

You use `jsp:setProperty` to give values to properties of beans that have been referenced earlier. You can do this in two contexts. First, you can use `jsp:setProperty` after, but outside of, a `jsp:useBean` element, as below:

```
<jsp:useBean id="myName" ... />
```

...

```
<jsp:setProperty name="myName" property="myProperty" ... />
```

In this case, the `jsp:setProperty` is executed regardless of whether a new bean was instantiated or an existing bean was found.

A second context in which `jsp:setProperty` can appear is inside the body of a `jsp:useBean` element, as below:

```
<jsp:useBean id="myName" ... >
```

...

```
<jsp:setProperty name="myName"
```

```
property="someProperty" ... />
```

```
</jsp:useBean>
```

Here, the `jsp:setProperty` is executed only if a new object was instantiated, not if an existing one was found.

**23. What is the `jsp:getProperty` action?**

The `<jsp:getProperty>` action is used to access the properties of a bean that was set using the `<jsp:setProperty>` action. The container converts the property to a String as follows:

- If it is an object, it uses the `toString()` method to convert it to a String.
- If it is a primitive, it converts it directly to a String using the `valueOf()` method of the corresponding Wrapper class.
- The syntax of the `<jsp:getProperty>` method is: `<jsp:getProperty name="Name" property="Property" />`

Here, `name` is the id of the bean from which the property was set. The property attribute is the property to get. A user must create or locate a bean using the `<jsp:useBean>` action before using the `<jsp:getProperty>` action.

**24. What is the `<jsp:param>` standard action?**

The `<jsp:param>` standard action is used with `<jsp:include>` or `<jsp:forward>` to pass parameter names and values to the target resource. The syntax of the `<jsp:param>` standard action is as follows: `<jsp:param name="paramName" value="paramValue"/>`

**25. What is the `jsp:plugin` action ?**

This action lets you insert the browser-specific OBJECT or EMBED element needed to specify that the browser run an applet using the Java plugin

**26. What are scripting elements?**

JSP scripting elements let you insert Java code into the servlet that will be generated from the current JSP page. There are three forms:

1. Expressions of the form `<%= expression %>` that are evaluated and inserted into the output,
2. Scriptlets of the form `<% code %>` that are inserted into the servlet's service method,
3. Declarations of the form `<%! code %>` that are inserted into the body of the servlet class, outside of any existing methods.

**27. What is a scriptlet?**

A scriptlet contains Java code that is executed every time a JSP is invoked. When a JSP is translated to a servlet, the scriptlet code goes into the `service()` method. Hence, methods and variables written in scriptlets are local to the `service()` method. A scriptlet is written between the `<%` and `%>` tags and is executed by the container at request processing time.

**28. What are JSP declarations?**

As the name implies, JSP declarations are used to declare class variables and methods in a JSP page. They are initialized when the class is initialized. Anything defined in a declaration is available for the whole JSP page. A declaration block is enclosed between the `<%!` and `%>` tags. A declaration is not included in the `service()` method when a JSP is translated to a servlet.

**29. What is a JSP expression?**

A JSP expression is used to write an output without using the `out.print` statement. It can be said as a shorthand representation for scriptlets. An expression is written between the `<%=` and `%>` tags. It is not required to end the expression with a semicolon, as it implicitly adds a semicolon to all the expressions within the expression tags.

**30. How is scripting disabled?**

Scripting is disabled by setting the `scripting-invalid` element of the deployment descriptor to true. It is a subelement of `jsp-propertygroup`. Its valid values are true and false. The syntax for disabling scripting is as follows:

```
<jsp-property-group>  
<url-pattern>*.jsp</url-pattern>  
<scripting-invalid>true</scripting-invalid>  
</jsp-property-group>
```

### 31. What is difference between custom JSP tags and beans?

Custom JSP tag is a tag you defined. You define how a tag, its attributes and its body are interpreted, and then group your tags into collections called tag libraries that can be used in any number of JSP files. Custom tags and beans accomplish the same goals —encapsulating complex behavior into simple and accessible forms.

There are several differences:

Custom tags can manipulate JSP content; beans cannot.

Complex operations can be reduced to a significantly simpler form with custom tags than with beans.

Custom tags require quite a bit more work to set up than do beans.

Custom tags usually define relatively self-contained behavior, whereas beans are often defined in one servlet and used in a different servlet or JSP page.

Custom tags are available only in JSP 1.1 and later, but beans can be used in all JSP 1.x versions.

JAVASCRIPT ::

#### 1.What's relationship between JavaScript and ECMAScript?

ECMAScript is yet another name for JavaScript (other names include LiveScript). The current JavaScript that you see supported in browsers is ECMAScript revision

#### 2.What are JavaScript types?

Number, String, Boolean, Function, Object, Null, Undefined.

#### 3.How do you convert numbers between different bases in JavaScript?

– Use the parseInt() function, that takes a string as the first parameter, and the base as a second parameter. So to convert hexadecimal 3F to decimal, use parseInt ("3F", 16);

#### 4.What does isNaN function do?

Return true if the argument is not a number.

#### 5.What is negative infinity?

It's a number in JavaScript, derived by dividing negative number by zero.

#### 6.What boolean operators does JavaScript support?

– &&, || and !

#### 7.What do "1"+2+4 evaluate to? –

Since 1 is a string, everything is a string, so the result is 124.

#### 8.How about 2+5+"8"?

Since 2 and 5 are integers, this is number arithmetic, since 8 is a string, it's concatenation, so 78 is the result.

#### 9.What looping structures are there in JavaScript? –

for, while, do-while loops, but no foreach.

#### 10.How do you create a new object in JavaScript? –

```
var obj = new Object(); or var obj = {};
```

**11.** How do you assign object properties? –  
`obj["age"] = 17` or `obj.age = 17`.

**12.** What's a way to append a value to an array? –

```
arr[arr.length] = value;
```

=>What is this keyword? –

It refers to the current object.

JavaScript gives HTML designers a programming tool – HTML authors are normally not programmers, but JavaScript is a scripting language with a very simple syntax! Almost anyone can put small

“snippets” of code into their HTML pages JavaScript can react to events – A JavaScript can be set to execute

when something happens, like when a page has finished loading or when a user clicks on an HTML element

JavaScript can read and write HTML elements – A JavaScript can read and change the content of an HTML element

JavaScript can be used to validate data – A JavaScript can be used to validate form data before it is submitted to a server. This saves the server from extra processing JavaScript can be used to detect the visitor's browser – A JavaScript

can be used to detect the visitor's browser, and – depending on the browser – load another page specifically designed for that browser JavaScript can be used to create cookies – A JavaScript can be used to store and retrieve information on the visitor's computer

**13.** How do I let people download a file from my page?

Once the file is uploaded to the server, you need to use an anchor reference tag to link to it.

Example: `<a href="../files/foo.zip">Download Foo Now!</a>`

**14.** What is an Empty HTML Tag?

Empty HTML tags are tags that do not need to be closed when one is creating a HTML document.

Example: `br (<br / )`

XML ::

1. What is XML?

- Extensible Markup Language (XML) is the universal language for data on the Web
- XML is a technology which allows us to create our own markup language.
- XML documents are universally accepted as a standard way of representing information in platform and language independent manner.
- XML is universal standard for information interchange.
- XML documents can be created in any language and can be used in any language.

2. What is the difference between XML and HTML?

XML is no way clashes with HTML, since they are for two different purposes.

HTML XML

HTML is for displaying purpose, whereas XML is for data representation.

HTML is used to mark up text so it can be displayed to users. XML is used to mark up data so it can be processed by computers.

HTML describes both structure (e.g. `<p>`, `<h2>`, `<em>`) and appearance (e.g. `<br>`, `<font>`, `<i>`) XML describes only content, or “meaning” HTML uses a fixed, unchangeable set of tags In XML, you make up your own tags

**3.** What are the benefits of XML?

There are many benefits of using XML on the Web :

- **Simplicity**- Information coded in XML is easy to read and understand, plus it can be processed easily by computers.
- **Openness**- XML is a W3C standard, endorsed by software industry market leaders.
- **Extensibility** – There is no fixed set of tags. New tags can be created as they are needed.
- **Self-description**- In traditional databases, data records require schemas set up by the database administrator. XML documents can be stored without such definitions, because they contain meta data in the form of tags and attributes.
- **Contains machine-readable context information**- Tags, attributes and element structure provide context information that can be used to interpret the meaning of content, opening up new possibilities for highly efficient search engines, intelligent data mining, agents, etc.
- **Separates content from presentation**- XML tags describe meaning not presentation. The motto of HTML is: “I know how it looks”, whereas the motto of XML is: “I know what it means, and you tell me how it should look.” The look and feel of an XML document can be controlled by XSL style sheets, allowing the look of a document to be changed without touching the content of the document. Multiple views or presentations of the same content are easily rendered.
- **Supports multilingual documents and Unicode**-This is important for the internationalization of applications.
- **Facilitates the comparison and aggregation of data** – The tree structure of XML documents allows documents to be compared and aggregated efficiently element by element.
- **Can embed multiple data types** – XML documents can contain any possible data type – from multimedia data (image, sound, video) to active components (Java applets, ActiveX).
- **Can embed existing data** – Mapping existing data structures like file systems or relational databases to XML is simple. XML supports multiple data formats and can cover all existing data structures and .
- **Provides a ‘one-server view’ for distributed data** – XML documents can consist of nested elements that are distributed over multiple remote servers. XML is currently the most sophisticated format for distributed data – the World Wide Web can be seen as one huge XML database.

#### 4. What is a well-formed XML document?

If a document is syntactically correct it can be called as well-formed XML documents. A well-formed document conforms to XML’s basic rules of syntax:

- Every open tag must be closed.
- The open tag must exactly match the closing tag: XML is casesensitive.
- All elements must be embedded within a single root element.
- Child tags must be closed before parent tags.
- A well-formed document has correct XML tag syntax, but the elements might be invalid for the specified document type.

#### 5. What is a valid XML document

If a document is structurally correct then it can be called as valid XML documents. A valid document conforms to the predefined rules of a specific type of document:

- These rules can be written by the author of the XML document or by someone else.
- The rules determine the type of data that each part of a document can contain.

Note:Valid XML document is implicitly well-formed, but well-formed may not be valid

#### 7. What is a Processing Instruction in XML?

A ProcessingInstruction is the information which we would like to give to application. Through a ProcessingInstruction an application would get idea about how to process the document.

A ProcessingInstruction can appear anywhere and any no. of times in a document.

#### 8. How does the XML structure is defined?

XML document will have a structure which has to be defined before we can create the documents and work with them. The structural rules can be defined using many available technologies, but the following are popular way of doing so

- Document Type Definition (DTD)
- Schema

### 9. What is DTD?

A Document Type Definition (DTD) defines the legal building blocks of an XML document. It defines **rules** for a specific type of document, including:

- Names of elements, and how and where they can be used
- The order of elements
- Proper nesting and containment of elements
- Element attributes

To apply a DTD to an XML document, you can:

- Include the DTD's element definitions within the XML document itself.
- Provide the DTD as a separate file, whose name you reference in the XML document.

### 10. What is XML Schema?

An XML Schema describes the structure of an XML instance document by defining what each element must or may contain. XML Schema is expressed in the form of a separate XML file.

- XML Schema provides much more control on element and attribute datatypes.
- Some datatypes are predefined and new ones can be created.
- <xsd:schema

```
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
```

```
<xsd:element name="test">
```

```
<xsd:complexType>
```

### 11. What are differences between DTDs and Schema?

Schema DTD Schema document is an XML document i.e., the structure of an XML document is specified by another XML document. DTDs follow SGML syntax.

Schema supports variety of dataTypes similar to programming language. In DTD everything is treated as text.

In Schema, It is possible to inherit and create relationship among elements. This is not possible in DTD without invalidating existing documents.

In Schema, It is possible to group elements and attributes so that they can be treated as single logical unit. Grouping of elements and attributes is not possible in DTD.

In Schemas, it is possible to specify an upper limit for the number of occurrences of an element It is not possible to specify an upper limit of an element in DTDs

### 12. What is a Complex Element?

A complex element is an XML element that contains other elements and/or attributes.

There are four kinds of complex elements:

- empty elements
- elements that contain only other elements
- elements that contain only text
- elements that contain both other elements and text

### 13. What is a Simple Element?

A simple element is an XML element that can contain only text.

- A simple element cannot have attributes
- A simple element cannot contain other elements
- A simple element cannot be empty
- However, the text can be of many different types, and may have various restrictions applied to it

**14. What are namespaces? Why are they important?**

A simple element is an XML element that can contain only text.

- Namespaces are a simple and straightforward way to distinguish names used in XML documents, no matter where they come from.
- XML namespaces are used for providing uniquely named elements and attributes in an XML instance
- They allow developers to qualify uniquely the element names and relationships and make these names recognizable, to avoid name collisions on elements that have the same name but are defined in different vocabularies.
- They allow tags from multiple namespaces to be mixed, which is essential if data is coming from multiple sources.

Example: a bookstore may define the <TITLE> tag to mean the title of a book, contained only within the <BOOK> element. A directory of people, however, might define <TITLE> to indicate a person's position, for instance: <TITLE>President</TITLE>. Namespaces help define this distinction clearly.

Note: a) Every namespace has a unique name which is a string. To maintain the uniqueness among namespaces a IRL is most preferred approach, since URLs are unique.

b) Except for no-namespace Schemas, every XML Schema uses at least two namespaces:

1. the target namespace.
2. The XMLSchema namespace (<http://w3.org/2001/XMLSchema>)

**15. What are the ways to use namespaces?**

There are two ways to use namespaces:

- Declare a default namespace
- Associate a prefix with a namespace, then use the prefix in the XML to refer to the namespace

**16. What is the relevance of Element Form Default attribute in the Schema?**

Element Form Default indicates whether or not locally declared elements must be qualified by the target namespace in an instance document. Element Form Default attribute in the Schema has the following relevance:

- Qualified: Each and every element of the Schema must be qualified with the namespace in the instance document.
- Unqualified: means only globally declared elements must be qualified with there namespace and not the local elements.

**17. What is XML parser?**

An XML parser is a piece of software which can do following:

- Check for well-formedness
- Validate the document
- Allows us to read, create or modify existing XML documents Note: Parser is piece of software provided by vendors. An XML parser is built in Java runtime from JDK 1.4 onwards

**18. What is DOM?**

The Document Object Model (DOM) is a platform and language independent standard object model for representing XML and related formats. DOM is standard API which is not specific to any programming language. DOM represents an XML document as a tree model. The tree model makes



the XML document hierarchical by nature. Each and every construct of the XML document is represented as a node in the tree.

**19. What is SAX?**

SAX-Simple API for XML processing. SAX provides a mechanism for reading data from an XML document. It is a popular alternative to the Document Object Model (DOM). SAX provides an event based processing approach unlike DOM which is tree based.

**20. What are the interfaces of SAX?**

The interfaces of SAX are:

- Document Handler- is used for getting event notification relating to a document.
- DTDHandler- is implemented to get the notifications related to declarations in DTD like entities and notations
- EntityResolver- is used for reading external entities.
- ErrorHandler- is used for handling error related notifications.

**21. What is the difference between SAX parser and DOM parser?****SAX DOM**

A SAX parser takes the occurrences of components of an input document as events (i.e., event based processing), and tells the client what it reads as it reads through the input document. A DOM parser creates a tree structure in memory from an input document and then waits for requests from client. No navigation possible (top to bottom only once) Whereas, we can navigate the DOM tree in any direction, any no. of times. We cannot modify the document content in SAX We can modify the document content in DOM .

A SAX parser serves the client application always only with pieces of the document at any given time. A DOM parser always serves the client application with the entire document no matter how much is actually needed by the client.

A SAX parser, however, is much more space efficient in case of a big input document A DOM parser is space inefficient when the document is huge.

Use SAX parser when

- Input document is too big for available memory.
- When only a part of the document is to be read and we create the data structures of our own.
- If you use SAX, you are using much less memory and performing much less dynamic memory allocation.

Use DOM when

- Your application has to access various parts of the document and using your own structure is just as complicated as the DOM tree.
- Your application has to change the tree very frequently and data has to be stored for a significant amount of time.

**22. What is a CDATA section in XML?**

CDATA Sections are used to escape blocks of text containing characters which would otherwise be recognized as markup. All tags and entity references are ignored by an XML processor that treats them just like any character data. CDATA blocks have been provided as a convenience measure when you want to include large blocks of special characters as character data, but you do not want to have to use entity references all the time.

**23. What is XSL?**

eXtensible Stylesheet Language(XSL) deals with most displaying the contents of XML documents.XSL consists of three parts:

- XSLT – a language for transforming XML documents
- XPath – a language for navigating in XML documents

- XSL-FO – a language for formatting XML documents

**24.**How is XSL different from Cascading Style Sheets? Why is a new Stylesheet language needed?

XSL is compatible with CSS and is designed to handle the new capabilities of XML that CSS can't handle. XSL is derived from Document Style Semantics and Specification Language (DSSSL), a complex Stylesheet language with roots in the SGML community. The syntax of XSL is quite different from CSS, which could be used to display simple XML data but isn't general enough to handle all the possibilities generated by XML. XSL adds the capability to handle these possibilities. For instance, CSS cannot add new items or generated text (for instance, to assign a purchase order number) or add a footer (such as an order confirmation). XSL allows for these capabilities.

**25.**What is XSLT?

eXtensible Stylesheet Language Transformation (XSLT) deals with transformation of one XML document into XHTML documents or to other XML documents. XSLT uses XPath for traversing an XML document and arriving at a particular node.

## SERVLET

**1.**What is the Servlet?

A servlet is a Java programming language class that is used to extend the capabilities of servers that host applications accessed by means of a request- response programming model.

**2.**What are the new features added to Servlet 2.5?

Following are the changes introduced in Servlet 2.5: • A new dependency on J2SE 5.0

- Support for annotations
- Loading the class
- Several web.xml conveniences
- A handful of removed restrictions
- Some edge case clarifications

**3.**What are the uses of Servlet?

Typical uses for HTTP Servlets include:

- Processing and/or storing data submitted by an HTML form.
- Providing dynamic content, e.g. returning the results of a database query to the client.
- A Servlet can handle multiple request concurrently and be used to develop high performance system
- Managing state information on top of the stateless HTTP, e.g. for an online shopping cart system which manages shopping carts for many concurrent customers and maps every request to the right customer.

**4.**What are the advantages of Servlet over CGI?

Servlets have several advantages over CGI:

- A Servlet does not run in a separate process. This removes the overhead of creating a new process for each request.
- A Servlet stays in memory between requests. A CGI program (and probably also an extensive runtime system or interpreter) needs to be loaded and started for each CGI request.
- There is only a single instance which answers all requests concurrently. This saves memory and allows a Servlet to easily manage persistent data.
- Several web.xml conveniences
- A handful of removed restrictions
- Some edge case clarifications

**5. What are the phases of the servlet life cycle?**

The life cycle of a servlet consists of the following phases:

- **Servlet class loading** : For each servlet defined in the deployment descriptor of the Web application, the servlet container locates and loads a class of the type of the servlet. This can happen when the servlet engine itself is started, or later when a client request is actually delegated to the servlet.
- **Servlet instantiation** : After loading, it instantiates one or more object instances of the servlet class to service the client requests.
- **Initialization (call the init method)** : After instantiation, the container initializes a servlet before it is ready to handle client requests. The container initializes the servlet by invoking its `init()` method, passing an object implementing the `ServletConfig` interface. In the `init()` method, the servlet can read configuration parameters from the deployment descriptor or perform any other one-time activities, so the `init()` method is invoked once and only once by the servlet container.
- **Request handling (call the service method)** : After the servlet is initialized, the container may keep it ready for handling client requests. When client requests arrive, they are delegated to the servlet through the `service()` method, passing the request and response objects as parameters. In the case of HTTP requests, the request and response objects are implementations of `HttpServletRequest` and `HttpServletResponse` respectively. In the `HttpServlet` class, the `service()` method invokes a different handler method for each type of HTTP request, `doGet()` method for GET requests, `doPost()` method for POST requests, and so on.

- **Removal from service (call the destroy method)** : A servlet container may decide to remove a servlet from service for various reasons, such as to conserve memory resources. To do this, the servlet container calls the `destroy()` method on the servlet. Once the `destroy()` method has been called, the servlet may not service any more client requests. Now the servlet instance is eligible for garbage collection. The life cycle of a servlet is controlled by the container in which the servlet has been deployed.

**6. Why do we need a constructor in a servlet if we use the init method?**

Even though there is an `init` method in a servlet which gets called to initialize it, a constructor is still required to instantiate the servlet.

Even though you as the developer would never need to explicitly call the servlet's constructor, it is still being used by the container (the container still uses the constructor to create an instance of the servlet). Just like a normal POJO (plain old java object) that might have an `init` method, it is no use calling the `init` method if you haven't constructed an object to call it on yet.

**7. How the servlet is loaded?**

A servlet can be loaded when:

- First request is made.
- Server starts up (auto-load).
- There is only a single instance which answers all requests concurrently. This saves memory and allows a Servlet to easily manage persistent data.
- Administrator manually loads.

**8. How a Servlet is unloaded?**

A servlet is unloaded when:

- Server shuts down.
- Administrator manually unloads.

**9. What is Servlet interface?**

The central abstraction in the Servlet API is the Servlet interface. All servlets implement this interface, either directly or, more commonly by extending a class that implements it.

Note: Most Servlets, however, extend one of the standard implementations of that interface, namely `javax.servlet.GenericServlet` and `javax.servlet.http.HttpServlet`.

**10. What is the GenericServlet class?**

`GenericServlet` is an abstract class that implements the `Servlet` interface and the `ServletConfig` interface. In addition to the methods declared in these two interfaces, this class also provides simple versions of the lifecycle methods `init` and `destroy`, and implements the `log` method declared in the `ServletContext` interface. Note: This class is known as generic servlet, since it is not specific to any protocol.

**11. What's the difference between GenericServlet and HttpServlet?**

`GenericServlet` `HttpServlet`

The `GenericServlet` is an abstract class that is extended by `HttpServlet` to provide HTTP protocol-specific methods. An abstract class that simplifies writing HTTP servlets. It extends the `GenericServlet` base class and provides a framework for handling the HTTP protocol.

The `GenericServlet` does not include protocol-specific methods for handling request parameters, cookies, sessions and setting response headers. The `HttpServlet` subclass passes generic service method requests to the relevant `doGet()` or `doPost()` method.

`GenericServlet` is not specific to any protocol. `HttpServlet` only supports HTTP and HTTPS protocol.

**12. Why is HttpServlet declared abstract?**

The `HttpServlet` class is declared abstract because the default implementations of the main service methods do nothing and must be overridden. This is a convenience implementation of the `Servlet` interface, which means that developers do not need to implement all service methods. If your servlet is required to handle `doGet()` requests for example, there is no need to write a `doPost()` method too.

**13. Can servlet have a constructor ?**

One can definitely have constructor in servlet. Even you can use the constructor in servlet for initialization purpose, but this type of approach is not so common. You can perform common operations with the constructor as you normally do. The only thing is that you cannot call that constructor explicitly by the `new` keyword as we normally do. In the case of servlet, servlet container is responsible for instantiating the servlet, so the constructor is also called by servlet container only.

**14. What are the types of protocols supported by HttpServlet ?**

It extends the `GenericServlet` base class and provides a framework for handling the HTTP protocol. So, `HttpServlet` only supports HTTP and HTTPS protocol.

**15. What is the difference between doGet() and doPost()?**

# `doGet()` `doPost()`

1 In `doGet()` the parameters are appended to the URL and sent along with header information. In `doPost()`, on the other hand will (typically) send the information through a socket back to the webserver and it won't show up in the URL bar.

2 The amount of information you can send back using a GET is restricted as URLs can only be 1024 characters. You can send much more information to the server this way – and it's not restricted to textual data either. It is possible to send files and even binary data such as serialized Java objects!

3 `doGet()` is a request for information; it does not (or should not) change anything on the server. (`doGet()` should be idempotent) `doPost()` provides information (such as placing an order for merchandise) that the server is expected to remember

4 Parameters are not encrypted Parameters are encrypted

5 `doGet()` is faster if we set the response content length since the same connection is used. Thus increasing the performance `doPost()` is generally used to update or post some information to the

server.doPost is slower compared to doGet since doPost does not write the content length

6 doGet() should be idempotent. i.e. doGet should be able to be repeated safely many times This method does not need to be idempotent. Operations requested through POST can have side effects for which the user can be held accountable.

7 doGet() should be safe without any side effects for which user is held responsible This method does not need to be either safe

8 It allows bookmarks. It disallows bookmarks.

**16. When to use doGet() and when doPost()?**

Always prefer to use GET (As because GET is faster than POST), except mentioned in the following reason:

- If data is sensitive
- Data is greater than 1024 characters
- If your application don't need bookmarks.

**17. How do I support both GET and POST from the same Servlet?**

The easy way is, just support POST, then have your doGet method call your doPost method:

```
public void doGet(HttpServletRequest request, HttpServletResponse
response)
throws ServletException, IOException
{
doPost(request, response);
}
```

**18. Should I override the service() method?**

We never override the service method, since the HTTP Servlets have already taken care of it . The default service function invokes the doXXX() method corresponding to the method of the HTTP request. For example, if the HTTP request method is GET, doGet() method is called by default. A servlet should override the doXXX() method for the HTTP methods that servlet supports. Because HTTP service method check the request method and calls the appropriate handler method, it is not necessary to override the service method itself. Only override the appropriate doXXX() method.

**19. How the typical servlet code look like ?**

**20. What is a servlet context object?**

A servlet context object contains the information about the Web application of which the servlet is a part. It also provides access to the resources common to all the servlets in the application. Each Web application in a container has a single servlet context associated with it.

**21. What are the differences between the ServletConfig interface and the ServletContext interface?**

ServletConfig ServletContext

The ServletConfig interface is implemented by the servlet container in order to pass configuration information to a servlet. The server passes an object that implements the ServletConfig interface to the servlet's init() method. A ServletContext defines a set of methods that a servlet uses to communicate with its servlet container.

There is one ServletConfig parameter per servlet. There is one ServletContext for the entire webapp and all the servlets in a webapp share it.

The param-value pairs for ServletConfig object are specified in the <init-param> within the <servlet> tags in the web.xml file The param-value pairs for ServletContext object are specified in the <context-param> tags in the web.xml file.

**22.**What's the difference between forward() and sendRedirect() methods?

forward() sendRedirect()

A forward is performed internally by the servlet. A redirect is a two step process, where the web application instructs the browser to fetch a second URL, which differs from the original.

The browser is completely unaware that it has taken place, so its original URL remains intact. The browser, in this case, is doing the work and knows that it's making a new request.

Any browser reload of the resulting page will simply repeat the original request, with the original URL. A browser reloads of the second URL, will not repeat the original request, but will rather fetch the second URL.

Both resources must be part of the same context (Some containers make provisions for cross-context communication but this tends not to be very portable) This method can be used to redirect users to resources that are not part of the current context, or even in the same domain. Since both resources are part of same context, the original request context is retained. Because this involves a new request, the previous request scope objects, with all of its parameters and attributes are no longer available after a redirect. (Variables will need to be passed by via the session object).

Forward is marginally faster than redirect. redirect is marginally slower than a forward, since it requires two browser requests, not one.

**23.**What is the difference between the include() and forward() methods?

include() forward()

The RequestDispatcher include() method inserts the the contents of the specified resource directly in the flow of the servlet response, as if it were part of the calling servlet. The RequestDispatcher forward() method is used to show a different resource in place of the servlet that was originally called. If you include a servlet or JSP document, the included resource must not attempt to change the response status code or HTTP headers, any such request will be ignored. The forwarded resource may be another servlet, JSP or static HTML document, but the response is issued under the same URL that was originally requested. In other words, it is not the same as a redirection.

The include() method is often used to include common "boilerplate" text or template markup that may be included by many servlets. The forward() method is often used where a servlet is taking a controller role; processing some input and deciding the outcome by returning a particular response page.

**24.**What's the use of the servlet wrapper classes??

The HttpServletRequestWrapper and HttpServletResponseWrapper classes are designed to make it easy for developers to create custom implementations of the servlet request and response types. The classes are constructed with the standard HttpServletRequest and HttpServletResponse instances respectively and their default behaviour is to pass all method calls directly to the underlying objects.

**26.**What is a deployment descriptor?

A deployment descriptor is an XML document with an .xml extension. It defines a component's deployment settings. It declares transaction attributes and security authorization for an enterprise bean. The information provided by a deployment descriptor is declarative and therefore it can be modified without changing the source code of a bean. The JavaEE server reads the deployment descriptor at run time and acts upon the component accordingly.

**27.**What is the difference between the getRequestDispatcher(String path) method of javax.servlet.ServletRequest interface and javax.servlet.ServletContext interface?

ServletRequest.getRequestDispatcher(String path)

ServletContext.getRequestDispatcher(String path)

The `getRequestDispatcher(String path)` method of `javax.servlet.ServletException` interface accepts parameter the path to the resource to be included or forwarded to, which can be relative to the request of the calling servlet. If the path begins with a “/” it is interpreted as relative to the current context root. The `getRequestDispatcher(String path)` method of `javax.servlet.ServletContext` interface cannot accept relative paths. All path must start with a “/” and are interpreted as relative to current context root.

**28. What is preinitialization of a servlet?**

A container does not initialize the servlets as soon as it starts up, it initializes a servlet when it receives a request for that servlet first time. This is called lazy loading. The servlet specification defines the element, which can be specified in the deployment descriptor to make the servlet container load and initialize the servlet as soon as it starts up. The process of loading a servlet before any request comes in is called preloading or preinitializing a servlet.

**29. What is the <load-on-startup> element?**

The <load-on-startup> element of a deployment descriptor is used to load a servlet file when the server starts instead of waiting for the first request. It is also used to specify the order in which the files are to be loaded. The <load-on-startup> element is written in the deployment descriptor as follows:

```
<servlet>
<servlet-name>ServletName</servlet-name>
<servlet-class>ClassName</servlet-class>
<load-on-startup>1</load-on-startup>
</servlet>
```

Note: The container loads the servlets in the order specified in the <load-on-startup> element.

**30. What is session?**

A session refers to all the requests that a single client might make to a server in the course of viewing any pages associated with a given application. Sessions are specific to both the individual user and the application. As a result, every user of an application has a separate session and has access to a separate set of session variables.

**31. What is Session Tracking?**

Session tracking is a mechanism that servlets use to maintain state about a series of requests from the same user (that is, requests originating from the same browser) across some period of time.

**32. What is the need of Session Tracking in web application?**

HTTP is a stateless protocol i.e., every request is treated as new request. For web applications to be more realistic they have to retain information across multiple requests. Such information which is part of the application is referred as “state”. To keep track of this state we need session tracking. Typical example: Putting things one at a time into a shopping cart, then checking out—each page request must somehow be associated with previous requests.

**33. What are the types of Session Tracking ?**

Sessions need to work with all web browsers and take into account the users security preferences. Therefore there are a variety of ways to send and receive the identifier:

- URL rewriting : URL rewriting is a method of session tracking in which some extra data (session ID) is appended at the end of each URL. This extra data identifies the session. The server can associate this session identifier with the data it has stored about that session. This method is used with browsers that do not support cookies or where the user has disabled the cookies.

- Hidden Form Fields : Similar to URL rewriting. The server embeds new hidden fields in every dynamically generated form page for the client. When the client submits the form to the server the hidden fields identify the client.
- Cookies : Cookie is a small amount of information sent by a servlet to a Web browser. Saved by the browser, and later sent back to the server in subsequent requests. A cookie has a name, a single value, and optional attributes. A cookie's value can uniquely identify a client.
- Secure Socket Layer (SSL) Sessions : Web browsers that support Secure Socket Layer communication can use SSL's support via HTTPS for generating a unique session key as part of the encrypted conversation.

### 34. How do I use cookies to store session state on the client?

In a servlet, the `HttpServletResponse` and `HttpServletRequest` objects passed to method `HttpServlet.service()` can be used to create cookies on the client and use cookie information transmitted during client requests. JSPs can also use cookies, in scriptlet code or, preferably, from within custom tag code.

- To set a cookie on the client, use the `addCookie()` method in class `HttpServletResponse`. Multiple cookies may be set for the same request, and a single cookie name may have multiple values.
- To get all of the cookies associated with a single HTTP request, use the `getCookies()` method of class `HttpServletRequest`

### 35. What are some advantages of storing session state in cookies?

- Cookies are usually persistent, so for low-security sites, user data that needs to be stored long-term (such as a user ID, historical information, etc.) can be maintained easily with no server interaction.
- For small- and medium-sized session data, the entire session data (instead of just the session ID) can be kept in the cookie.

### 36. What are some disadvantages of storing session state in cookies?

- Cookies are controlled by programming a low-level API, which is more difficult to implement than some other approaches.
- All data for a session are kept on the client. Corruption, expiration or purging of cookie files can all result in incomplete, inconsistent, or missing information.
- Cookies may not be available for many reasons: the user may have disabled them, the browser version may not support them, the browser may be behind a firewall that filters cookies, and so on. Servlets and JSP pages that rely exclusively on cookies for client-side session state will not operate properly for all clients. Using cookies, and then switching to an alternate client-side session state strategy in cases where cookies aren't available, complicates development and maintenance.
- Browser instances share cookies, so users cannot have multiple simultaneous sessions.
- Cookie-based solutions work only for HTTP clients. This is because cookies are a feature of the HTTP protocol. Notice that the `while` package `javax.servlet.http` supports session management (via class `HttpSession`), package `javax.servlet` has no such support.

### 37. What is URL rewriting?

URL rewriting is a method of session tracking in which some extra data is appended at the end of each URL. This extra data identifies the session. The server can associate this session identifier with the data it has stored about that session.

Every URL on the page must be encoded using method

`HttpServletResponse.encodeURL()`. Each time a URL is output, the servlet passes the URL to `encodeURL()`, which encodes session ID in the URL if the browser isn't accepting cookies, or if the session tracking is turned off.

E.g., <http://abc/path/index.jsp;jsessionid=123465hfhs> Advantages

- URL rewriting works just about everywhere, especially when cookies are turned off.
- Multiple simultaneous sessions are possible for a single user.



Session information is local to each browser instance, since it's stored in URLs in each page being displayed. This scheme isn't foolproof, though, since users can start a new browser instance using a URL for an active session, and confuse the server by interacting with the same session through two instances.

- Entirely static pages cannot be used with URL rewriting, since every link must be dynamically written with the session state. It is possible to combine static and dynamic content, using (for example) templating or server-side includes. This limitation is also a barrier to integrating legacy web pages with newer, servlet-based pages.

#### DisAdvantages

- Every URL on a page which needs the session information must be rewritten each time a page is served. Not only is this expensive computationally, but it can greatly increase communication overhead.
- URL rewriting limits the client's interaction with the server to HTTP GETs, which can result in awkward restrictions on the page.
- URL rewriting does not work well with JSP technology.
- If a client workstation crashes, all of the URLs (and therefore all of the data for that session) are lost.

#### 38. How can an existing session be invalidated?

An existing session can be invalidated in the following two ways:

- Setting timeout in the deployment descriptor: This can be done by specifying timeout between the <session-timeout> tags as follows:

```
<session-config>
<session-timeout>10</session-timeout>
</session-config>
```

This will set the time for session timeout to be ten minutes.

- Setting timeout programmatically: This will set the timeout for a specific session. The syntax for setting the timeout programmatically is as follows:

```
public void setMaxInactiveInterval(int interval)
```

The setMaxInactiveInterval() method sets the maximum time in seconds before a session becomes invalid.

Note :Setting the inactive period as negative(-1), makes the container stop tracking session, i.e, session never expires.

#### 39. How can the session in Servlet can be destroyed?

An existing session can be destroyed in the following two ways:

- Programmatically : Using session.invalidate() method, which makes the container abandon the session on which the method is called.
- When the server itself is shutdown.

#### 40. A client sends requests to two different web components. Both of the components access the session. Will they end up using the same session object or different session ?

Creates only one session i.e., they end up with using same session . Sessions is specific to the client but not the web components. And there is a 1-1 mapping between client and a session.

#### 41. What is servlet lazy loading?

- A container doesnot initialize the servlets ass soon as it starts up, it initializes a servlet when it receives a request for that servlet first time. This is called lazy loading.
- The servlet specification defines the <load-on-startup> element, which can be specified in the deployment descriptor to make the servlet container load and initialize the servlet as soon as it starts up.
- The process of loading a servlet before any request comes in is called preloading or preinitializing a servlet.

**42. What is Servlet Chaining?**

Servlet Chaining is a method where the output of one servlet is piped into a second servlet. The output of the second servlet could be piped into a third servlet, and so on. The last servlet in the chain returns the output to the Web browser.

**43. How are filters?**

Filters are Java components that are used to intercept an incoming request to a Web resource and a response sent back from the resource. It is used to abstract any useful information contained in the request or response. Some of the important functions performed by filters are as follows:

- Security checks
- Modifying the request or response
- Data compression
- Logging and auditing
- Response compression

Filters are configured in the deployment descriptor of a Web application. Hence, a user is not required to recompile anything to change the input or output of the Web application.

**44. What are the functions of an intercepting filter?**

The functions of an intercepting filter are as follows:

- It intercepts the request from a client before it reaches the servlet and modifies the request if required.
- It intercepts the response from the servlet back to the client and modifies the request if required.
- There can be many filters forming a chain, in which case the output of one filter becomes an input to the next filter. Hence, various modifications can be performed on a single request and response.

**45. What are the functions of the Servlet container?**

The functions of the Servlet container are as follows:

- Lifecycle management : It manages the life and death of a servlet, such as class loading, instantiation, initialization, service, and making servlet instances eligible for garbage collection.
- Communication support : It handles the communication between the servlet and the Web server.
- Multithreading support : It automatically creates a new thread for every servlet request received. When the Servlet service() method completes, the thread dies.
- Declarative security : It manages the security inside the XML deployment descriptor file.
- JSP support : The container is responsible for converting JSPs to servlets and for maintaining them.

\*\*\*\*\*