



Shree Pandurang Prathisthan Pandharpur's

Karmayogi Institute of Technology ShelvePandharpur

Affiliated to Dr.B.A.Technological University,
Lonere MaharashtraISO 9001:2015, 140001:2015

Certified, AICTE Approved

Department of Computer Science & Engineering LAB MANUAL

Programme (UG/PG) : UG
Year : Second Year
Semester : IV
Course Code : BTCOS407
Course Title : Seminar – II

Prepared By

D.P.Kulkarni

[Asst. Prof , Department of Computer Science & Engineering]

FOREWORD

It is my great pleasure to present this laboratory manual for **SECOND year** engineering students for the subject of Seminar-II.

As a student, many of you may be wondering with some of the questions in your mind regarding the subject and exactly what has been tried is to answer through this manual.

As you may be aware that KIT has already been awarded with ISO 9001:2015,140001:2015 certification and it is our endure to technically equip our students taking the advantage of the procedural aspects of ISO Certification.

Faculty members are also advised that covering these aspects in initial stage itself, will greatly relived them in future as much of the load will be taken care by the enthusiasm energies of the students once they are conceptually clear.

Dr. S. P. Patil
Principal

LABORATORY MANUAL CONTENTS

This manual is intended for the Second year students of Computer Science & Engineering in the subject of Seminar-II. This manual typically contains practical/Lab Sessions related to Seminar-II covering various aspects related the subject to enhanced understanding.

Students are advised to thoroughly go through this manual rather than only topics mentioned in the syllabus as practical aspects are the key to understanding and conceptual visualization of theoretical aspects covered in the books.

Good Luck for your Enjoyable Laboratory Sessions

Miss D P Kulkarni
Subject Teacher

Mr. Dipak Bhosale
HOD

LIST OF EXPERIMENTS

Course Code: BTCOS407

Course Title: Seminar-II.

Sr. No	Name of the Experiment	Page No
1	Introduction to HTML,CSS,javascript,php	
2	Design an html form for displaying information using interactive css including images, tables	
3	Create a webpage with HTML describing college department	
4	Write a JavaScript to design a simple calculator to perform the following operations: sum, product, difference and quotient.	
5	Write a JavaScript to validate the following fields of employee on html form: email, name, mobile no., address, salary	
6	Develop and demonstrate a HTML file that includes JavaScript to find length of String and Reverse given number	
7	Develop and demonstrate a HTML file that includes JavaScript to find prime number	
8	Write a PHP program to display a digital clock which displays the current time of the server.	
9	Write a PHP program to implement sign-In and Sign-out functionality	
10	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	
11	Create HTML page which contain Audio and Vedio files	

DOs and DON'Ts in Laboratory:

1. Make entry in the Log Book as soon as you enter the Laboratory.
2. All the students should sit according to their roll numbers starting from their left to right.
3. All the students are supposed to enter the terminal number in the log book.
4. Do not change the terminal on which you are working.
5. All the students are expected to get at least the algorithm of the program/concept to be implemented.
6. Strictly observe the instructions given by the teacher/Lab Instructor.
7. Do not disturb machine Hardware / Software Setup.

Instruction for Laboratory Teachers:

1. Submission related to whatever lab work has been completed should be done during the next lab session along with signing the index.
2. The promptness of submission should be encouraged by way of marking and evaluation patterns that will benefit the sincere students.
3. Continuous assessment in the prescribed format must be followed.

Shree Pandurang Prathisthan Pandharpur's



**Karmayogi Institute of Technology Shelve Pandharpur
Department of Computer Science and Engineering**

Vision of CSE Department

To develop computer engineers with necessary analytical ability and human values who can creatively design, implement a wide spectrum of computer systems for welfare of the society.

Mission of the CSE Department:

- 1.** Preparing graduates to work on multidisciplinary platforms associated with their professional position both independently and in a team environment.
- 2.** Preparing graduates for higher education and research in computer science and engineering enabling them to develop systems for society development.

Programme Educational Objectives

Graduates will be able to

- I.** To analyze, design and provide optimal solution for Computer Science & Engineering and multidisciplinary problems.
- II.** To pursue higher studies and research by applying knowledge of mathematics and fundamentals of computer science.
- III.** To exhibit professionalism, communication skills and adapt to current trends by engaging in lifelong learning.

Programme Outcomes (POs):

Engineering Graduates will be able to:

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. **Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. **Life-long learning:** Recognize the need for, and have the preparation and ability to engage independent and life-long learning in the broadest context of technological change.

LABORATORY OUTCOMES

The practical/exercises in this section are psychomotor domain Learning Outcomes (i.e. subcomponents of the COs), to be developed and assessed to lead to the attainment of the competency.

LO-1: Study and Implement Front End Technology HTML,CSS,Javascript.

LO-2: Study and Implement Back End Technology PHP.

Experiment no-1

Title:- Introduction to HTML,CSS,javascript,php.

Objective:- To understand HTML,CSS,javascript,php.

Theory:-

Web Development

Web development refers to the creating, building, and maintaining of websites. It includes aspects such as web design, web publishing, web programming, and database management. It is the creation of an application that works over the internet i.e. websites.

The word Web Development is made up of two words, that is:

- **Web:** It refers to websites, web pages or anything that works over the internet.
- **Development:** It refers to building the application from scratch.

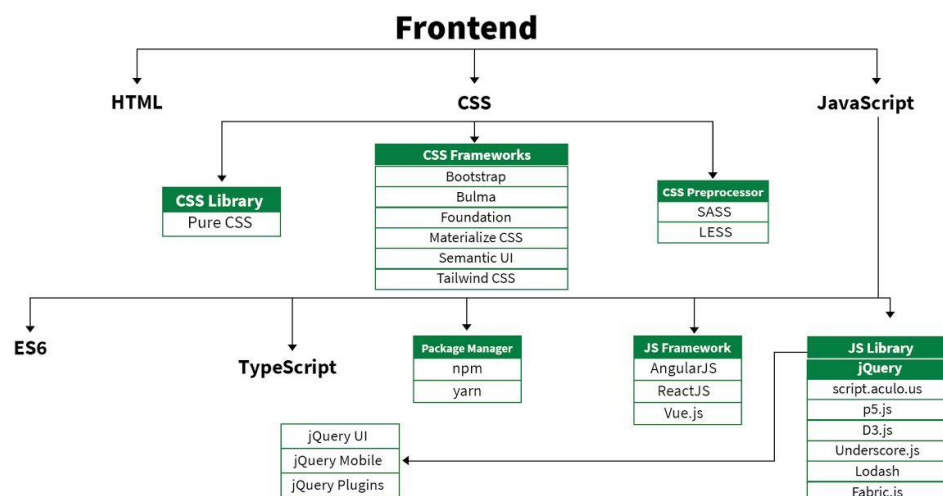
Web Development can be classified into two ways:

- Frontend Development
- Backend Development

Frontend Development

The part of a website that the user interacts directly is termed as front end. It is also referred to as the 'client side' of the application.

- **Frontend Roadmap:**



- **HTML:**HTML stands for HyperText Markup Language. It is used to design the front end portion of web pages using markup language. It acts as a skeleton for a website since it is used to make the structure of a website.
- **CSS:** Cascading Style Sheets fondly referred to as CSS is a simply designed language intended to simplify the process of making web pages presentable. It is used to style our website.
- **JavaScript:** JavaScript is a scripting language used to provide a dynamic behavior to our website.
- **Bootstrap:** Bootstrap is a free and open-source tool collection for creating responsive websites and web applications. It is the most popular CSS framework for developing responsive, mobile-first websites. Nowadays, the websites are perfect for all the browsers (IE, Firefox, and Chrome) and for all sizes of screens (Desktop, Tablets, Phablets, and Phones).
 - [Bootstrap 4](#)
 - [Bootstrap 5](#)

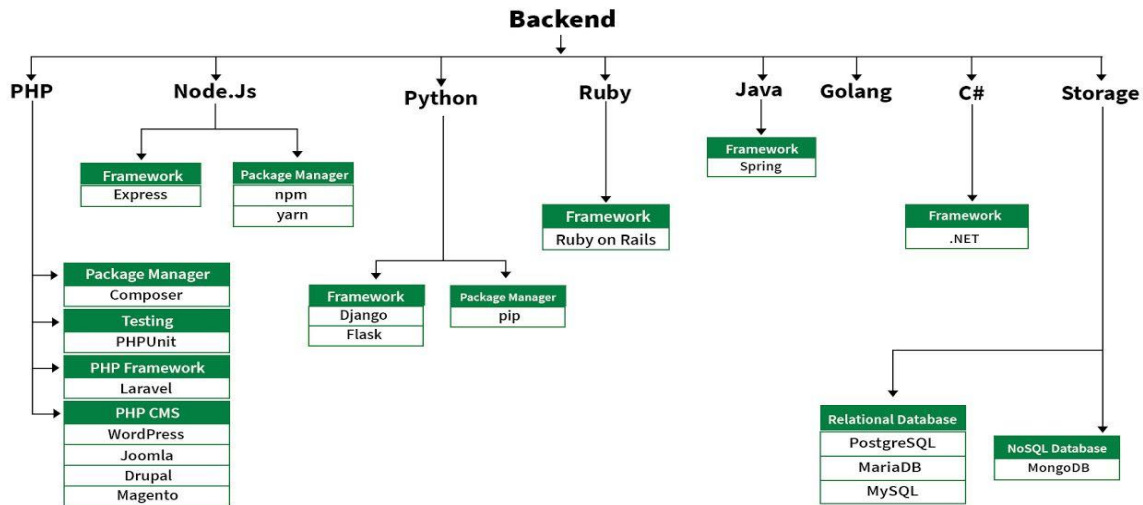
Frontend Frameworks and Libraries:

- [ngularJS](#)
- [React.js](#)
- [VueJS](#)
- [jQuery](#)
- [Bootstrap](#)
- [Material UI](#)
- [Tailwind CSS](#)
- [jQuery UI](#)
- Some other libraries and frameworks are: [Handlebar.js](#) [Backbone.js](#), [Ember.js](#) etc

• Backend Development

- Backend is the server side of a website. It is the part of the website that users cannot see and interact. It is the portion of software that does not come in direct contact with the users. It is used to store and arrange data.

Backend Roadmap:



- **PHP:** PHP is a server-side scripting language designed specifically for web development.
- **Java:** Java is one of the most popular and widely used programming language. It is highly scalable.
- **Python:** Python is a programming language that lets you work quickly and integrate systems more efficiently.
- **Node.js:** Node.js is an open source and cross-platform runtime environment for executing JavaScript code outside a browser.
- **Back End Frameworks:** The list of back end frameworks are: Express, Django, Rails, Laravel, Spring, etc.

Coclusion:-

Experiment no-2

Title:- Design an html form for displaying information using interactive css including images, tables

Objective:- To Understand css styles of Table and image

Theory:-

HTML Images Syntax

The HTML tag is used to embed an image in a web page.

Images are not technically inserted into a web page; images are linked to web pages. The tag creates a holding space for the referenced image.

The tag is empty, it contains attributes only, and does not have a closing tag.

The tag has two required attributes:

- src - Specifies the path to the image
- alt - Specifies an alternate text for the image

```

```

The src Attribute

The required src attribute specifies the path (URL) to the image.

Note: When a web page loads, it is the browser, at that moment, that gets the image from a web server and inserts it into the page. Therefore, make sure that the image actually stays in the same spot in relation to the web page, otherwise your visitors will get a broken link icon. The broken link icon and the alt text are shown if the browser cannot find the image.

The alt Attribute

The required alt attribute provides an alternate text for an image, if the user for some reason cannot view it (because of slow connection, an error in the src attribute, or if the user uses a screen reader).

The value of the alt attribute should describe the image:

Image Size - Width and Height

You can use the **style** attribute to specify the width and height of an image.

```

```

Define an HTML Table

A table in HTML consists of table cells inside rows and columns.

Table Cells

Each table cell is defined by a `<td>` and a `</td>` tag.

Everything between `<td>` and `</td>` are the content of the table cell.

Table Rows

Each table row starts with a `<tr>` and ends with a `</tr>` tag.

Table Headers

Sometimes you want your cells to be table header cells. In those cases use the `<th>` tag instead of the `<td>` tag:

HTML Table Borders

HTML tables can have borders of different styles and shapes.

How To Add a Border

When you add a border to a table, you also add borders around each table cell:

To add a border, use the CSS border property on table, th, and td elements:

```
<style>
```

```
table, th, td {
```

```
    border: 1px solid black;
```

```
}
```

```
</style>
```

Round Table Borders

With the border-radius property, the borders get rounded corners:









```
<style>
```

```
table, th, td {
    border: 1px solid black;
    border-radius: 10px;
}
</style>
```

Dotted Table Borders

With the border-style property, you can set the appearance of the border.

The following values are allowed:

- dotted 
- dashed 
- solid 
- double 
- groove 
- ridge 
- inset 
- outset 
- none
- hidden

Syntax-

```
<style>
th, td {
    border-style: dotted;
}
</style>
```

HTML Table Sizes

Use the style attribute with the width or height properties to specify the size of a table, row or column.

HTML Table Column Width

To set the size of a specific column, add the style attribute on a <th> or <td> element:

```
<table style="width:100%">
  <tr>
    <th style="width:70%">Firstname</th>
    <th>Lastname</th>
    <th>Age</th>
  </tr>
  <tr>
    <td>Jill</td>
    <td>Smith</td>
    <td>50</td>
  </tr>
  <tr>
    <td>Eve</td>
    <td>Jackson</td>
    <td>94</td>
  </tr>
</table>
```

HTML Table Row Height

To set the height of a specific row, add the style attribute on a table row element:

```
<table style="width:100%">
  <tr>
    <th>Firstname</th>
    <th>Lastname</th>
    <th>Age</th>
```

```
</tr>
<tr style="height:200px">
  <td>Jill</td>
  <td>Smith</td>
  <td>50</td>
</tr>
<tr>
  <td>Eve</td>
  <td>Jackson</td>
  <td>94</td>
</tr>
</table>
```

Cocclusion:-

Experiment no-3

Title:- Create a webpage with HTML describing college department

Objective:-

- a) Change the background color of the page. At the bottom create a link to take user to the top of the page.
- b) Insert an image and create a link such that clicking on image takes user to other page.
- c) Also apply font styling like italics, underline and two other fonts to words you find appropriate. Also use header tags.

Theory:-

Style background Property

The background property sets or returns up to eight separate background properties, in a shorthand form.

DOM Property	CSS Property
backgroundAttachment	background-attachment
backgroundClip	background-clip
backgroundColor	background-color
backgroundImage	background-image
backgroundOrigin	background-origin

backgroundPosition	background-position
--------------------	---------------------

backgroundRepeat	background-repeat
------------------	-------------------

backgroundSize	background-size
----------------	-----------------

With this property, you can set/return one or more of the following (in any order):

Syntax

Return the background property:

```
object.style.background
```

Set the background property:

```
object.style.background = "color image repeat attachment position size  
origin clip|initial|inherit"
```

Style backgroundColor Property

Syntax

Return the backgroundColor property:

```
object.style.backgroundColor
```

Set the backgroundColor property:

```
object.style.backgroundColor = "color|transparent|initial|inherit"
```

Image as a Link

To use an image as a link, put the tag inside the <a> tag:

```
<a href="default.asp">
```

```

```

```
</a>
```

Fonts

The CSS font-family property defines the font to be used for an HTML element:

```
<html>
```

```
<body>
```

```
<h1 style="font-family:verdana;">This is a heading</h1>
```

```
<p style="font-family:courier;">This is a paragraph.</p>
```

```
</body>
```

```
</html>
```

HTML Text Formatting-

HTML Formatting Elements-

Formatting elements were designed to display special types of text:

- - Bold text
- - Important text
- <i> - Italic text
- - Emphasized text
- <mark> - Marked text
- <small> - Smaller text
- - Deleted text
- <ins> - Inserted text
- <sub> - Subscript text
- <sup> - Superscript text

HTML <u> Tag

Definition and Usage

The <u> tag represents some text that is unarticulated and styled differently from normal text, such as misspelled words or proper names in Chinese text. The content inside is typically displayed with an underline.

```
<!DOCTYPE html>
```

```
<html>
```

```
<body>
```

```
<h1>The u element</h1>
```

```
<p>This is some <u>misperled</u> text.</p>
```

```
</body>
```

```
</html>
```

HTML <header> Tag

Definition and Usage

The <header> element represents a container for introductory content or a set of navigational links.

A <header> element typically contains:

- one or more heading elements (<h1> - <h6>)
- logo or icon
- authorship information

Note: You can have several <header> elements in one HTML document. However, <header> cannot be placed within a <footer>, <address> or another <header> element.

Conclusion:-

Experiment no-4

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

Objective:- Design a simple calculator to perform the following operations

- 1) Sum
- 2)Product
- 3)difference
- 4)quotient.

Theory:-

HTML Forms

An HTML form is used to collect user input. The user input is most often sent to a server for processing.

The <form> Element

The HTML <form> element is used to create an HTML form for user input:

The <input> Element

The HTML <input> element is the most used form element.

An <input> element can be displayed in many ways, depending on the type attribute.

Here are some examples:

Type	Description
<code><input type="text"></code>	Displays a single-line text input field
<code><input type="radio"></code>	Displays a radio button (for selecting one of many choices)
<code><input type="checkbox"></code>	Displays a checkbox (for selecting zero or more of many choices)
<code><input type="submit"></code>	Displays a submit button (for submitting the form)
<code><input type="button"></code>	Displays a clickable button

The Action Attribute

The action attribute defines the action to be performed when the form is submitted.

Usually, the form data is sent to a file on the server when the user clicks on the submit button.

In the example below, the form data is sent to a file called "action_page.php". This file contains a server-side script that handles the form data:

The Method Attribute

The method attribute specifies the HTTP method to be used when submitting the form data.

The form-data can be sent as URL variables (with method="get") or as HTTP post transaction (with method="post").

The default HTTP method when submitting form data is GET.

The <input> Element

One of the most used form element is the <input> element.

The <input> element can be displayed in several ways, depending on the type attribute.

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

<h2>The input Element</h2>

<form action="/action_page.php">
  <label for="fname">First name:</label><br>
  <input type="text" id="fname" name="fname"><br><br>
  <input type="submit" value="Submit">
</form>

</body>
</html>
```

The <button> Element

The <button> element defines a clickable button:

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

<h2>The button Element</h2>

<button type="button" onclick="alert('Hello World!')">Click Me!</button>

</body>
</html>
```

Conclusion:-

Experiment no-5

Title:- Write a JavaScript to validate the following fields of employee on html form: email, name, mobile no., address, salary

Objective:- validate the following fields of employee on html form:

email

name

mobile no.

address

salary

Theory:-

Forms are used in webpages for the user to enter their required details that further send it to the server for processing. A form is also known as a web form or HTML form. Examples of form use are prevalent in e-commerce websites, online banking, online surveys to name a few.

Validating a form: The data entered into a form needs to be in the right format and certain fields need to be filled in order to effectively use the submitted form. Username, password, contact information are some details that are mandatory in forms and thus need to be provided by the user.

HTML is used to create the form.

JavaScript to validate the form.

CSS to design the layout of the form.

Regular Expression for validation

var regEmail=/^[w+([.-]?w+)@[w+([.-]?w+)*(\.w{2,3})+\$/g;* *//Javascript*

reGex for Email Validation.

var regPhone=/^[d{10}\$/; *// Javascript reGex for Phone*

Number validation.

var regName = /^[d+\$/g; *// Javascript reGex for Name*
validation

Here we have used 3 Regular Expressions in *Javascript for Email Validation, Phone Number Validation and User's Name Validation*.

- **regEmail** = `/^\w+([\.-]?\w+)*@\w+([\.-]?\w+)*(\.\w{2,3})+$/g;`

Above regular expression checks for valid mail-id entered by user.

`\w` shows any word character including Characters, Numbers and Underscore

`+` Symbol shows one or more combination of characters.

Most common used special characters in mail-id's are Dot(`.`) ,Underscore(`_`) and (`-`) . Hence we have included them in `\.-` **expression**.

`*` Denotes Zero or more combinations of given characters.

`$` Denotes End of the string or regular expression.

`g` denotes all matches from starting to end of given string with regular expression. `g` is used to find all the occurrences of the pattern instead of stopping after the first match.

- **regPhone** = `/^d{10}$/;`

In the Phone Number validation :

`^` denotes starting of the string and it should be of type `\d`. That means it should contain only digits from 0 to 9. Any non-digit characters will not be accepted under `\d`.

`{10}` denotes that length of the Digit characters should be exactly 10 only. No extra or less characters will be permitted.

`$` Denotes end of the string.

`/ and /` denotes start and end of the regular expression.

- **regName** = `/\d+$/g;`

Under user's name validation :

User's original name should not contain any kind of Numbers, Hence to avoid it we have used again `\d` class of characters which will accept only any combinations of 0-9 digits(Here `+` denotes one or more combination of any digits from 0 to 9). After that verification takes place using **regName.test(name)** .

Where "name" is the variable value entered by user in form's name field. If it contains any kind of Digit value then it will prompt message of "Please enter your name properly."

- **Validation on Password length :**

If password entered by user is too small then browser should prompt message to user that password length is too small or like that .

Hence we have used this validation also in form field. If password entered by user have length less than 6 then browser will prompt message to user that "Password should be at least 6 character long".

Please note that `Regular_Expression.test(String)` will check user entered string with the standard regular expression,

If it matches then if condition will be satisfied and **!Regular_Expression.test(String)** will check for alternative condition.

- **Validation on Dropdown menu items :**

If we have not selected any items from Drop-Down menu then it will prompt message that “Please enter your course”. Because (-1) denotes that not any field is selected and subsequent option items are considered as indices 0,1,2,3,4, and so on..

For indexing purposes **.selectedIndex** method is being used. If we are clicking on BTECH then selectedIndex = 0, for BBA selectedIndex = 1 and so on..

And if user *doesn't select any option then by default it will take selectedIndex = (-1)* and it should not occur . Hence ***what.selectedIndex == (-1)*** is used in If condition.

Conclusion:-

Experiment no-6

Title Develop and demonstrate a HTML file that includes JavaScript

Objective:- a. Parameter: A string

Output: Length of the String

b. Parameter: A number

Output: The number with its digits in the reverse order

Theory:-

JavaScript String Methods-

i)JavaScript String Length-

The length property returns the length of a string:

```
<html>
```

```
<body>
```

```
<h1>JavaScript Strings</h1>
```

```
<h2>The length Property</h2>
```

```
<p>The length of the string is:</p>
```

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

```
<script>
```

```
let text = "ABCDEFGHJKLMNOPQRSTUVWXYZ";
```

```
document.getElementById("demo").innerHTML = text.length;
```

```
</script>
```

```
</body>
```

```
</html>
```

ii)Extracting String Characters

There are 3 methods for extracting string characters:

- `charAt(position)`

- `charCodeAt(position)`

JavaScript String `charAt()`

The `charAt()` method returns the character at a specified index (*position*) in a string:

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

<h2>JavaScript String Methods</h2>

<p>The charAt() method returns the character at a given position in a string:</p>

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

<script>
var text = "HELLO WORLD";
document.getElementById("demo").innerHTML = text.charAt(0);
</script>

</body>
</html>
```

JavaScript String `charCodeAt()`

The `charCodeAt()` method returns the unicode of the character at a specified index in a string:

The method returns a UTF-16 code (an integer between 0 and 65535).

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

<h2>JavaScript String Methods</h2>

<p>The charCodeAt() method returns the unicode of the character at a given position in a string:</p>

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

<script>
```

```
let text = "HELLO WORLD";  
document.getElementById("demo").innerHTML = text.charCodeAt(0);  
</script>
```

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

Conclusion:-

Experiment no-7

Title Develop and demonstrate a HTML file that includes JavaScript to find prime number.

Objective:- Develop and demonstrate a HTML file that includes JavaScript for the following problems:

- a. Input: A starting and ending number
- b. Output: find all the prime numbers between starting and ending number.

Theory:-

Conditional Statements

Very often when you write code, you want to perform different actions for different decisions.

You can use conditional statements in your code to do this.

In JavaScript we have the following conditional statements:

- Use **if** to specify a block of code to be executed, if a specified condition is true
- Use **else** to specify a block of code to be executed, if the same condition is false
- Use **else if** to specify a new condition to test, if the first condition is false

The if Statement

Use the if statement to specify a block of JavaScript code to be executed if a condition is true.

Syntax

```
if (condition) {  
    // block of code to be executed if the condition is true  
}
```

The else Statement

Use the else statement to specify a block of code to be executed if the condition is false.

```
if (condition) {  
    // block of code to be executed if the condition is true  
} else {  
    // block of code to be executed if the condition is false  
}
```

Loops can execute a block of code a number of times.

JavaScript Loops

Loops are handy, if you want to run the same code over and over again, each time with a different value.

Often this is the case when working with arrays:

The For Loop

The for statement creates a loop with 3 optional expressions:

```
for (expression 1; expression 2; expression 3) {  
    // code block to be executed  
}
```

Expression 1 is executed (one time) before the execution of the code block.

Expression 2 defines the condition for executing the code block.

Expression 3 is executed (every time) after the code block has been executed.

JavaScript break

Definition and Usage

The break statement breaks out of a switch or a loop.

In a switch, it breaks out of the switch block. This stops the execution of more code inside the switch.

In in a loop, it breaks out of the loop and continues executing the code after the loop (if any)

Using Lables

The break statement can use a label reference, to break out of any JavaScript code block (see "More Examples" below).

Without a label, break can only be used inside a loop or a switch.

Syntax

`break;`

Using the optional label reference:

`break labelname;`

Conclusion:-

Experiment no-8

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

Objective:- create digital clock which displays the current time of the server

Theory:-

HTML <meta> http-equiv Attribute

Definition and Usage

The http-equiv attribute provides an HTTP header for the information/value of the content attribute.

The http-equiv attribute can be used to simulate an HTTP response header.

Syntax

```
<meta http-equiv="content-security-policy|content-type|default-style|refresh">
```

Attribute Values

Value	Description
content-security-policy	Specifies a content policy for the document. Example: <meta http-equiv="content-security-policy" content="default-src 'self'">

content-type Specifies the character encoding for the document.

Example:

```
<meta http-equiv="content-type" content="text/html; charset=UTF-8">
```

default-style Specified the preferred style sheet to use.

Example:

```
<meta http-equiv="default-style" content="the document's preferred stylesheet">
```

Note: The value of the content attribute above must match the value of the title attribute on a link element in the same document, or it must match the value of the title attribute on a style element in the same document.

refresh Defines a time interval for the document to refresh itself.

Example:

```
<meta http-equiv="refresh" content="300">
```

Note: The value "refresh" should be used carefully, as it takes the control of a page away from the user. Using "refresh" will cause a failure in W3C's Web Content Accessibility Guidelines.

PHP Date and Time

Get a Time

Here are some characters that are commonly used for times:

- H - 24-hour format of an hour (00 to 23)
- h - 12-hour format of an hour with leading zeros (01 to 12)
- i - Minutes with leading zeros (00 to 59)
- s - Seconds with leading zeros (00 to 59)
- a - Lowercase Ante meridiem and Post meridiem (am or pm)

The example below outputs the current time in the specified format:

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

<?php
echo "The time is " . date("h:i:sa");
?>

</body>
</html>
```

Get Your Time Zone

If the time you got back from the code is not correct, it's probably because your server is in another country or set up for a different timezone.

So, if you need the time to be correct according to a specific location, you can set the timezone you want to use.

The example below sets the timezone to "America/New_York", then outputs the current time in the specified format:

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

<?php
date_default_timezone_set("America/New_York");
echo "The time is " . date("h:i:sa");
?>

</body>
</html>
```

Conclusion:-

Experiment no-9

Title :- Write a PHP program to implement sign-In and Sign-out functionality.

Objective:- PHP program to implement sign-In and Sign-out functionality.

Theory:-

PHP Form Handling

The PHP superglobals `$_GET` and `$_POST` are used to collect form-data.

PHP - A Simple HTML Form

GET vs. POST

Both GET and POST create an array (e.g. `array(key1 => value1, key2 => value2, key3 => value3, ...)`). This array holds key/value pairs, where keys are the names of the form controls and values are the input data from the user.

Both GET and POST are treated as `$_GET` and `$_POST`. These are superglobals, which means that they are always accessible, regardless of scope - and you can access them from any function, class or file without having to do anything special.

`$_GET` is an array of variables passed to the current script via the URL parameters.

`$_POST` is an array of variables passed to the current script via the HTTP POST method.

When to use GET?

Information sent from a form with the GET method is visible to everyone (all variable names and values are displayed in the URL). GET also has limits on the amount of information to send. The

limitation is about 2000 characters. However, because the variables are displayed in the URL, it is possible to bookmark the page. This can be useful in some cases.

GET may be used for sending non-sensitive data.

Note: GET should NEVER be used for sending passwords or other sensitive information!

When to use POST?

Information sent from a form with the POST method is invisible to others (all names/values are embedded within the body of the HTTP request) and has no limits on the amount of information to send.

Moreover POST supports advanced functionality such as support for multi-part binary input while uploading files to server.

However, because the variables are not displayed in the URL, it is not possible to bookmark the page.

PHP \$_SERVER

\$_SERVER is a PHP super global variable which holds information about headers, paths, and script locations.

The following table lists the most important elements that can go inside \$_SERVER:

Element/Code	Description
\$_SERVER['PHP_SELF']	Returns the filename of the currently executing script
\$_SERVER['GATEWAY_INTERFACE']	Returns the version of the Common Gateway Interface (CGI) the server is using
\$_SERVER['SERVER_ADDR']	Returns the IP address of the host server

`$_SERVER['SERVER_NAME']`

Returns the name of the host server (such as `www.w3schools.com`)

`$_SERVER['SERVER_SOFTWARE']`

Returns the server identification string (such as `Apache/2.2.24`)

`$_SERVER['SERVER_PROTOCOL']`

Returns the name and revision of the information protocol (such as `HTTP/1.1`)

`$_SERVER['REQUEST_METHOD']`

Returns the request method used to access the page (such as `POST`)

`$_SERVER['REQUEST_TIME']`

Returns the timestamp of the start of the request (such as `1377687496`)

`$_SERVER['QUERY_STRING']`

Returns the query string if the page is accessed via a query string

`$_SERVER['HTTP_ACCEPT']`

Returns the Accept header from the current request

`$_SERVER['HTTP_ACCEPT_CHARSET']`

Returns the Accept_Charset header from the current request (such as `utf-8,ISO-8859-1`)

`$_SERVER['HTTP_HOST']`

Returns the Host header from the current request

`$_SERVER['HTTP_REFERER']`

Returns the complete URL of the current page (not reliable because not all user-agents support it)

`$_SERVER['HTTPS']`

Is the script queried through a secure HTTP protocol

`$_SERVER['REMOTE_ADDR']`

Returns the IP address from where the user is viewing the current page

`$_SERVER['REMOTE_HOST']`

Returns the Host name from where the user is viewing the current page

`$_SERVER['REMOTE_PORT']`

Returns the port being used on the user's machine to communicate with the web server

`$_SERVER['SCRIPT_FILENAME']`

Returns the absolute pathname of the currently executing script

`$_SERVER['SERVER_ADMIN']`

Returns the value given to the `SERVER_ADMIN` directive in the web server configuration file (if your script runs on a virtual host, it will be the value defined for that virtual host) (such as `someone@w3schools.com`)

`$_SERVER['SERVER_PORT']`

Returns the port on the server machine being used by the web server for communication (such as 80)

`$_SERVER['SERVER_SIGNATURE']`

Returns the server version and virtual host name

which are added to server-generated pages

<code>\$_SERVER['PATH_TRANSLATED']</code>	Returns the file system based path to the current script
---	--

<code>\$_SERVER['SCRIPT_NAME']</code>	Returns the path of the current script
---------------------------------------	--

<code>\$_SERVER['SCRIPT_URI']</code>	Returns the URI of the current page
--------------------------------------	-------------------------------------

PHP header() Function

Definition and Usage

The header() function sends a raw HTTP header to a client.

It is important to notice that the header() function must be called before any actual output is sent!

Syntax

`header(header, replace, http_response_code)`

Parameter Values

Parameter	Description
<i>header</i>	Required. Specifies the header string to send
<i>replace</i>	Optional. Indicates whether the header should replace a previous similar header or add a new header of the same type. Default is TRUE (will replace). FALSE allows multiple headers of the same type

http_response_code Optional. Forces the HTTP response code to the specified value

Conclusion:-

Experiment no-10

Title :- 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

Objective:- PHP program to keep track of the number of visitors visiting the Web page and to display this count of visitors, with proper headings

Theory:-

PHP Sessions

A session is a way to store information (in variables) to be used across multiple pages.

Unlike a cookie, the information is not stored on the users computer.

What is a PHP Session?

When you work with an application, you open it, do some changes, and then you close it. This is much like a Session. The computer knows who you are. It knows when you start the application and when you end. But on the internet there is one problem: the web server does not know who you are or what you do, because the HTTP address doesn't maintain state.

Session variables solve this problem by storing user information to be used across multiple pages (e.g. username, favorite color, etc). By default, session variables last until the user closes the browser.

So; Session variables hold information about one single user, and are available to all pages in one application.

Start a PHP Session

A session is started with the `session_start()` function.

Session variables are set with the PHP global variable: `$_SESSION`.

```
<?php
// Start the session
session_start();
?>
```

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

<?php
// Set session variables
$_SESSION["favcolor"] = "green";
$_SESSION["favanimal"] = "cat";
echo "Session variables are set.";
?>

</body>
</html>
```

Get PHP Session Variable Values

Next, we create another page called "demo_session2.php". From this page, we will access the session information we set on the first page ("demo_session1.php").

Notice that session variables are not passed individually to each new page, instead they are retrieved from the session we open at the beginning of each page (session_start()).

```
<?php
session_start();
?>
<!DOCTYPE html>
<html>
<body>

<?php
print_r($_SESSION);
?>

</body>
</html>
```

Destroy a PHP Session

To remove all global session variables and destroy the session, use session_unset() and session_destroy():

```
<?php
session_start();
```

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

<?php
// remove all session variables
session_unset();

// destroy the session
session_destroy();

echo "All session variables are now removed, and the session is destroyed."
?>

</body>
</html>
```

Conclusion:-

Experiment no-11

Title :- Create HTML page which contain Audio and Vedio files

Objective:-

1]Adding Audio files in HTML page

2]Adding Vedio files in HTML page

Theory:-

HTML <audio> Tag

Definition and Usage

The <audio> tag is used to embed sound content in a document, such as music or other audio streams.

The <audio> tag contains one or more <source> tags with different audio sources. The browser will choose the first source it supports.

The text between the <audio> and </audio> tags will only be displayed in browsers that do not support the <audio> element.

There are three supported audio formats in HTML: MP3, WAV, and OGG.

HTML <video> Tag

Definition and Usage

The <video> tag is used to embed video content in a document, such as a movie clip or other video streams.

The <video> tag contains one or more <source> tags with different video sources. The browser will choose the first source it supports.

The text between the <video> and </video> tags will only be displayed in browsers that do not support the <video> element.

There are three supported video formats in HTML: MP4, WebM, and OGG.

Conclusion:-