

Program No-1

Design a simple webpage using HTML and DHML

HTML stands for **H**yper **T**ext **M**arkup **L**anguage, which is the most widely used language on Web to develop web pages.

HTML was created by Berners-Lee in late 1991 but "HTML 2.0" was the first standard HTML specification which was published in 1995. HTML 4.01 was a major version of HTML and it was published in late 1999. Though HTML 4.01 version is widely used but currently we are having HTML-5 version which is an extension to HTML 4.01, and this version was published in 2012.

DHTML

DHTML stands for **D**ynamic **H**TML.

DHTML is not a language or a web standard.

To most people DHTML means the combination of HTML, JavaScript, DOM and CSS.

JAVASCRIPT

JavaScript is the most popular scripting language on the internet, and it works in all major browsers.

DHTML is about using JavaScript to control access and manipulate HTML elements.

HTML DOM

The HTML DOM is a W3C standard. It describes the **Document Object Model** for HTML.

The HTML DOM defines a standard way for accessing and manipulating HTML documents.

DHTML is about using the DOM to access and manipulate HTML elements.

HTML Events

HTML events are a part of the HTML DOM.

DHTML is about creating web pages that reacts to (user) events.

CSS

CSS defines how to display HTML elements.

DHTML is about using JavaScript and the HTML DOM to change the style and positioning of HTML elements.

Program

```
<html>
```

```
<head>
```

```
    <title>XYZ Page</title>
```

```
</head>
```

```
<body><h2 id="top"><ol>
```

```
    <li><a href="#ch1">Chapter-1</a></li>
```

```
    <li><a href="#ch2">Chapter-2</a></li>
```

```
    <li><a href="#ch3">Chapter-3</a></li>
```

Chapter-4

Chapter-5

</h2>

<h1 id="ch1">Chapter-1</h1>

<p>This is chapter. This is chapter. This is chapter. </p>

<p>This is chapter. This is chapter. This is chapter. </p>

<p>This is chapter. This is chapter. This is chapter. </p>

<p>This is chapter. This is chapter. This is chapter. </p>

<p>This is chapter. This is chapter. This is chapter. </p>

<i>Back</i>

<h2 id="ch2">Chapt-2</h2>

<p>This is chapter. This is chapter. This is chapter. </p>

<p>This is chapter. This is chapter. This is chapter. </p>

<p>This is chapter. This is chapter. This is chapter. </p>

<p>This is chapter. This is chapter. This is chapter. </p>

<p>This is chapter. This is chapter. This is chapter. </p>

<i>Back</i>

<h2 id="ch3">Chapt-3</h2>

<p>This is chapter. This is chapter. This is chapter. </p>

<p>This is chapter. This is chapter. This is chapter. </p>

<p>This is chapter. This is chapter. This is chapter. </p>

<p>This is chapter. This is chapter. This is chapter. </p>

<i>Back</i>

<h2 id="ch4">Chapt-4</h2>

<p>This is chapter. This is chapter. This is chapter. </p>

<p>This is chapter. This is chapter. This is chapter. </p>

<p>This is chapter. This is chapter. This is chapter. </p>

<p>This is chapter. This is chapter. This is chapter. </p>

<i>Back</i>

<h2 id="ch5">Chapt-5</h2>

<table border="1">

<tr>

<th>Serial No</th>

<th>First Name</th>

<th>Last Name</th>

<th>Location</th>

</tr>

<tr>

<td>1</td>

	<td>Akhtar</td>
	<td>Salim</td>
	<td>Delhi, India</td>
</tr>	
<tr>	
	<td>2</td>
	<td>Joe</td>
	<td>Hany</td>
	<td>United State</td>
</tr>	
<tr>	
	<td>3</td>
	<td>John</td>
	<td>Milton</td>
	<td>London</td>
</tr>	
<tr>	
	<td>4</td>
	<td>Narenr</td>
	<td>Modi</td>

<td>Gujrat, India</td>

</tr>

</table>

<i>Back</i>

</body>

</html>

Program No-2

Design a simple webpage using HTML5

What is New in HTML5?

The DOCTYPE declaration for HTML5 is very simple:

```
<!DOCTYPE html>
```

The character encoding (charset) declaration is also very simple:

```
<meta charset="UTF-8">
```

Example:

```
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>

<h1>My First Heading</h1>
<p>My first paragraph.</p>

</body>
</html>
```

Explanation:

1. The **DOCTYPE** declaration defines the document type to be HTML
2. The text between **<html>** and **</html>** describes an HTML document

3. The text between **<head>** and **</head>** provides information about the document
4. The text between **<title>** and **</title>** provides a title for the document
5. The text between **<body>** and **</body>** describes the visible page content
6. The text between **<h1>** and **</h1>** describes a heading
7. The text between **<p>** and **</p>** describes a paragraph

New HTML5 Elements

The most interesting new elements are:

New **semantic** elements like `<header>`, `<footer>`, `<article>`, and `<section>`.

New form **control attributes** like `number`, `date`, `time`, `calendar`, and `range`.

New **graphic** elements: `<svg>` and `<canvas>`.

New **multimedia** elements: `<audio>` and `<video>`.

HTML5 File (mypage.html with Internal CSS)

```
<!Doctype html>
```

```
<html>
```

```
<head>
```

```
<title>HTML 5</title>
```

```
<style>
```



```
img, html,body,h1,h2,h3,p,ul,li,a, main,article,section,header,footer,nav,  
aside{ margin: 0px; padding: 0px;}
```

```
body{
```

```
    width: 960px;
```

```
    margin: 0px auto;
```

```
}
```

```
    main{
```

```
        float: left;
```

```
        width: 660px;
```

```
    }
```

```
    aside{
```

```
        float: right;
```

```
        width: 295px;
```

```
        color: white;
```

```
        background: gray;
```

```
}
```

```
footer{
```

```
    clear: both;
```

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

```
    background: green;
```

```
    color: white;
```

```
    padding: 15px;
```

```
}
```

```
#img1{
```

```
    width: 100%;
```

```
    height: 200px;
```

```
}
```

```
section{
```

```
    background: lightgray;
```

```
padding: 5px;
```

```
}
```

```
article{
```

```
background: white;
```

```
margin-top: 5px;
```

```
padding: 10px 15px;
```

```
}
```

```
nav{
```

```
background: green;
```

```
}
```

```
li a{
```

```
color: white;
```

```
text-decoration: none;
```

```
}
```

```

    li{

        display: inline-block;

        padding: 15px;

        font-size: 20px;

        font-weight: bold;

    }

</style>

</head>

<body>

    <header>

    </header>

    <nav>

        <ul>

            <li><a href="">HOME</a> </li>

```

HTML

CSS

PHP

</nav>

<main>

<section>

<h2>HTML</h2>

<article>

<h3>Basics</h3>

<p>This is paragraph. This is paragraph. This is paragraph. This is paragraph. </p>

<p>This is paragraph. This is paragraph. This is paragraph.
This is paragraph. </p>

<p>This is paragraph. This is paragraph. This is paragraph.
This is paragraph. </p>

<p>This is paragraph. This is paragraph. This is paragraph.

This is paragraph. </p>

</article>

</section>

<section>

<h2>HTML</h2>

<article>

<h3>Introduction</h3>

<p>This is paragraph. This is paragraph. This is paragraph. This is paragraph. </p>

<p>This is paragraph. This is paragraph. This is paragraph.
This is paragraph. </p>

<p>This is paragraph. This is paragraph. This is paragraph.
This is paragraph. </p>

<p>This is paragraph. This is paragraph. This is paragraph.
This is paragraph. </p>

</article>

</section>

</main>

<aside style="color:lightgray;">

<h2 style="color:White;">Write News Here </h2>

<p>This is news. This is news. This is news. This is news. This is news. This is news. </p>

<p>This is news. This is news. This is news. This is news. This is news. This is news. </p>

<p>This is news. This is news. This is news. This is news. This is news. This is news. </p>

<p>This is news. This is news. This is news. This is news. This is news. This is news. </p>

<p>This is news. This is news. This is news. This is news. This is news. This is news. </p>

</aside>

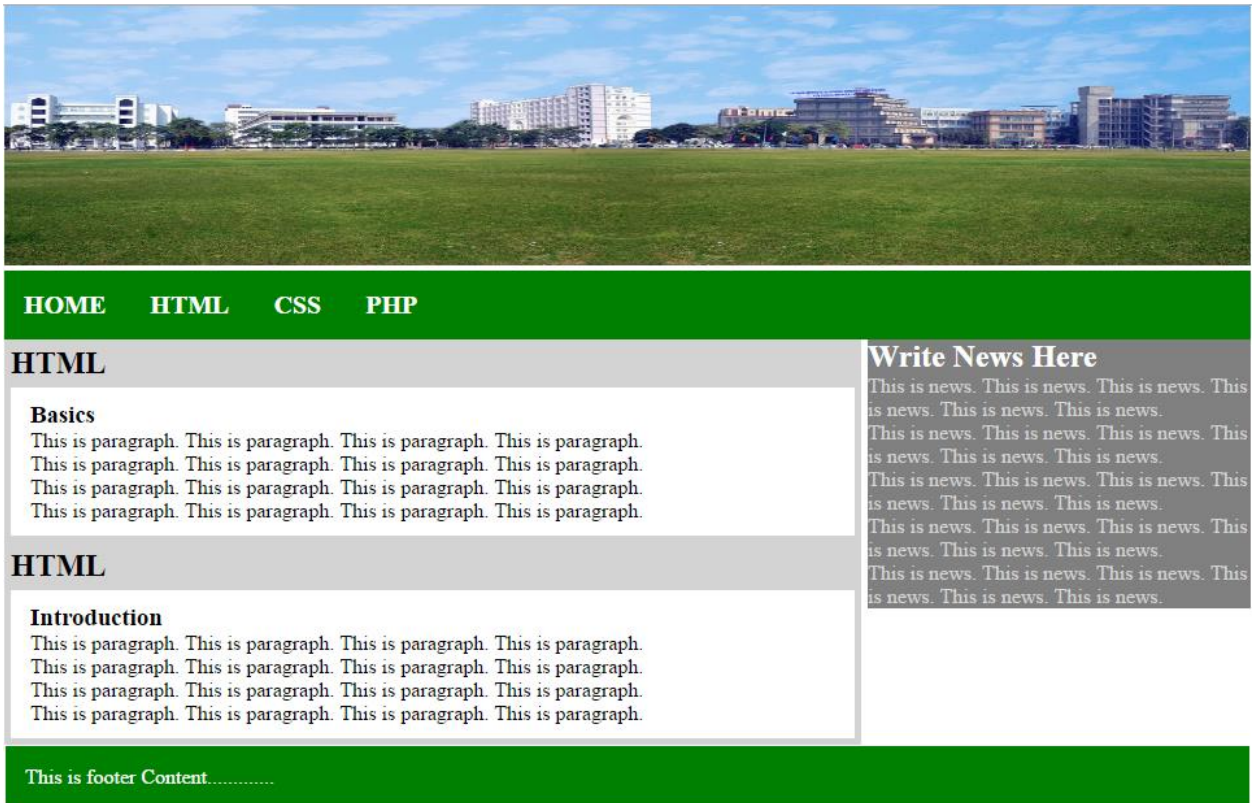
<footer>

<p>This is footer Content.....</p>

</footer>

</body>

</html>



Program No-3

Write inline, internal and external CSS for a Web Page

What is CSS?

Cascading Style Sheets, fondly referred to as CSS, is a simple design language intended to simplify the process of making web pages presentable.

CSS handles the look and feel part of a web page. Using CSS, you can control the color of the text, the style of fonts, the spacing between paragraphs, how columns are sized and laid out, what background images or colors are used, layout designs, variations in display for different devices and screen sizes as well as a variety of other effects.

CSS is easy to learn and understand but it provides powerful control over the presentation of an HTML document. Most commonly, CSS is combined with the markup languages HTML or XHTML.

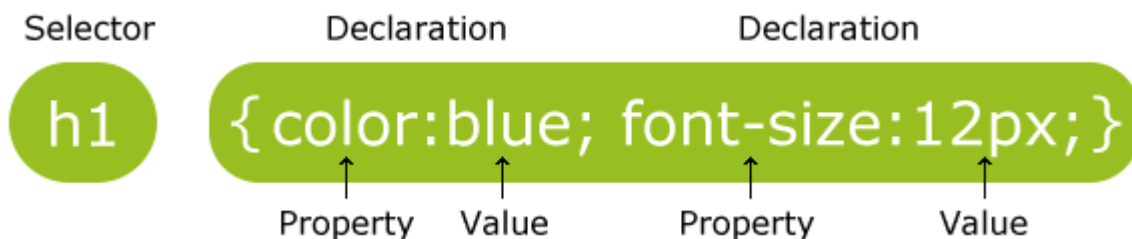
Advantages of CSS

- **CSS saves time** – You can write CSS once and then reuse same sheet in multiple HTML pages. You can define a style for each HTML element and apply it to as many Web pages as you want.
- **Pages load faster** – If you are using CSS, you do not need to write HTML tag attributes every time. Just write one CSS rule of a tag and apply it to all the occurrences of that tag. So less code means faster download times.
- **Easy maintenance** – To make a global change, simply change the style, and all elements in all the web pages will be updated automatically.
- **Superior styles to HTML** – CSS has a much wider array of attributes than HTML, so you can give a far better look to your HTML page in comparison to HTML attributes.

- **Multiple Device Compatibility** – Style sheets allow content to be optimized for more than one type of device. By using the same HTML document, different versions of a website can be presented for handheld devices such as PDAs and cell phones or for printing.
- **Global web standards** – Now HTML attributes are being deprecated and it is being recommended to use CSS. So its a good idea to start using CSS in all the HTML pages to make them compatible to future browsers.
- **Offline Browsing** – CSS can store web applications locally with the help of an offline cache. Using of this, we can view offline websites. The cache also ensures faster loading and better overall performance of the website.
- **Platform Independence** – The Script offer consistent platform independence and can support latest browsers as well.

CSS Syntax:

A CSS rule-set consists of a selector and a declaration block:



The selector points to the HTML element you want to style.

The declaration block contains one or more declarations separated by semicolons.

Each declaration includes a CSS property name and a value, separated by a colon.

A CSS declaration always ends with a semicolon, and declaration blocks are surrounded by curly braces.

In the following example all `<p>` elements will be center-aligned, with a red text color:

Example:

```
p {  
    color: red;  
    text-align: center;  
}
```

How to use inline CSS

An inline style may be used to apply a unique style for a single element.

To use inline styles, add the style attribute to the relevant element. The style attribute can contain any CSS property.

The example below shows how to change the color and the left margin of a `<h1>` element:

Example:

```
<h1 style="color:blue;margin-left:30px;">This is a heading.</h1>
```

Internal Style Sheet

An internal style sheet may be used if one single page has a unique style.

Internal styles are defined within the `<style>` element, inside the `<head>` section of an HTML page:

Example:

```
<head>

<style>

body {

    background-color: linen;

}

h1 {

    color: maroon;

    margin-left: 40px;

}

</style>

</head>
```

External Style Sheet

With an external style sheet, you can change the look of an entire website by changing just one file!

Each page must include a reference to the external style sheet file inside the <link> element. The <link> element goes inside the <head> section:

Example:

```
<head>  
  
<link rel="stylesheet" type="text/css" href="mystyle.css">  
  
</head>
```

An external style sheet can be written in any text editor. The file should not contain any html tags. The style sheet file must be saved with a .css extension.

Here is how the "myStyle.css" looks:

```
body {  
  
    background-color: lightblue;  
  
}
```

```
h1 {  
  
    color: navy;  
  
    margin-left: 20px;  
  
}
```

Program No-4

Design Navigation Menu using HTML and CSS

Horizontal Navigation

```
<!DOCTYPE html>

<html>

<head>

<style>

ul {

    list-style-type: none;

    margin: 0;

    padding: 0;

    overflow: hidden;

    background-color: #333;

}

li {

    float: left;

}

li a {

    display: block;

    color: white;
```

```

        text-align: center;

        padding: 14px 16px;

        text-decoration: none;
    }

    li a:hover:not(.active) {

        background-color: #111;

    }

    .active {

        background-color: #4CAF50;

    }

</style>

</head>

<body>

<ul>

    <li><a class="active" href="#home">Home</a></li>

    <li><a href="#news">News</a></li>

    <li><a href="#contact">Contact</a></li>

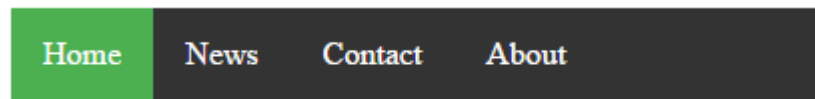
    <li><a href="#about">About</a></li>

</ul>

</body>

```

</html>



Vertical Navigation

<!DOCTYPE html>

<html>

<head>

<style>

ul {

list-style-type: none;

margin: 0;

padding: 0;

width: 200px;

background-color: #f1f1f1;

}

li a {

display: block;

color: #000;

padding: 8px 0 8px 16px;


```
        text-decoration: none;
    }

    li a.active {

        background-color: #4CAF50;

        color: white;

    }
```

```
    li a:hover:not(.active) {

        background-color: #555;

        color: white;

    }
```

```
</style>
```

```
</head>
```

```
<body>
```

```
<h2>Vertical Navigation Bar</h2>
```

<p>In this example, we create an "active" class with a green background color and a white text. The class is added to the "Home" link.</p>

```
<ul>
```

```
<li><a class="active" href="#home">Home</a></li>
```

```
<li><a href="#news">News</a></li>
```

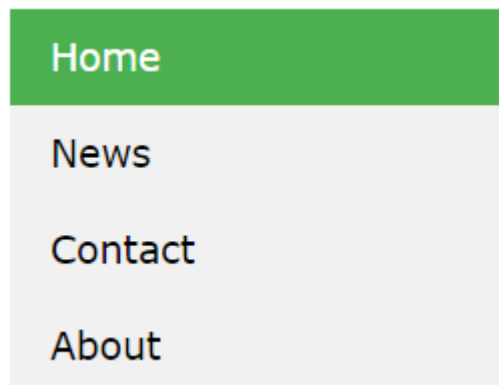
```
<li><a href="#contact">Contact</a></li>
```

```
<li><a href="#about">About</a></li>
```

```
</ul>
```

```
</body>
```

```
</html>
```



Program No-5

Design a dynamic web page with validation using JavaScript

JavaScript Introduction:

JavaScript started life as Live Script, but Netscape changed the name, possibly because of the excitement being generated by Java.to JavaScript. JavaScript made its first appearance in Netscape 2.0 in 1995 with a name *Live Script*.

JavaScript is a lightweight, interpreted programming language with object-oriented capabilities that allows you to build interactivity into otherwise static HTML pages.

JavaScript is:

- JavaScript is a lightweight, interpreted programming language
- Designed for creating network-centric applications
- Complementary to and integrated with Java
- Complementary to and integrated with HTML
- Open and cross-platform

Client-side JavaScript:

Client-side JavaScript is the most common form of the language. The script should be included in or referenced by an HTML document for the code to be interpreted by the browser.

It means that a web page need no longer be static HTML, but can include programs that interact with the user, control the browser, and dynamically create HTML content.

The JavaScript client-side mechanism features many advantages over traditional CGI server-side scripts. For example, you might use JavaScript to check if the user has entered a valid e-mail address in a form field.

The JavaScript code is executed when the user submits the form, and only if all the entries are valid they would be submitted to the Web Server.

JavaScript can be used to trap user-initiated events such as button clicks, link navigation, and other actions that the user explicitly or implicitly initiates.

Advantages of JavaScript:

The merits of using JavaScript are:

- **Less server interaction:** You can validate user input before sending the page off to the server. This saves server traffic, which means less load on your server.
- **Immediate feedback to the visitors:** They don't have to wait for a page reload to see if they have forgotten to enter something.
- **Increased interactivity:** You can create interfaces that react when the user hovers over them with a mouse or activates them via the keyboard.
- **Richer interfaces:** You can use JavaScript to include such items as drag-and-drop components and sliders to give a Rich Interface to your site visitors.

Limitations with JavaScript:

We can not treat JavaScript as a full fledged programming language. It lacks the following important features:

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

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

JavaScript Development Tools:

One of JavaScript's strengths is that expensive development tools are not usually required. You can start with a simple text editor such as Notepad.

Since it is an interpreted language inside the context of a web browser, you don't even need to buy a compiler.

To make our life simpler, various vendors have come up with very nice JavaScript editing tools. Few of them are listed here:

- **Microsoft FrontPage:** Microsoft has developed a popular HTML editor called FrontPage. FrontPage also provides web developers with a number of JavaScript tools to assist in the creation of an interactive web site.
- **Macromedia Dreamweaver MX:** Macromedia Dreamweaver MX is a very popular HTML and JavaScript editor in the professional web development crowd. It provides several handy prebuilt JavaScript components, integrates well with databases, and conforms to new standards such as XHTML and XML.
- **Macromedia HomeSite 5:** This provided a well-liked HTML and JavaScript editor, which will manage their personal web site just fine.

JavaScript Syntax

A JavaScript consists of JavaScript statements that are placed within the `<script>... </script>` HTML tags in a web page.

You can place the `<script>` tag containing your JavaScript anywhere within you web page but it is preferred way to keep it within the `<head>` tags.

The `<script>` tag alert the browser program to begin interpreting all the text between these tags as a script. So simple syntax of your JavaScript will be as follows

```
<script ...>
```

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

The script tag takes two important attributes:

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

So your JavaScript segment will look like:

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

Your First JavaScript Script:

Let us write our class example to print out "Hello World".

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

We added an optional HTML comment that surrounds our Javascript code. This is to save our code from a browser that does not support Javascript. The comment ends with a `"//-->"`. Here `"//"` signifies a comment in Javascript, so we

add that to prevent a browser from reading the end of the HTML comment in as a piece of Javascript code.

Next, we call a function *document.write* which writes a string into our HTML document. This function can be used to write text, HTML, or both. So above code will display following result:

Hello World!

Calculator's Program

```
<html>
<head>
<title>Calculator</title>
<form name="calc">
<table border=4><tr>
<td>
<input type="text" name="input" size="11">
<br></td>
</tr><tr>
<td>
<input type="button" name="one" value="1" onclick="calc.input.value+='1'">
<input type="button" name="two" value="2" onclick="calc.input.value+='2'">
<input type="button" name="three" value="3" onclick="calc.input.value+='3'">
<input type="button" name="plus" value="+" onclick="calc.input.value+='+'"><br/>
<input type="button" name="four" value="4" onclick="calc.input.value+='4'">
<input type="button" name="five" value="5" onclick="calc.input.value+='5'">
<input type="button" name="six" value="6" onclick="calc.input.value+='6'">
<input type="button" name="minus" value="-" onclick="calc.input.value+='-'">
<br>
<input type="button" name="seven" value="7" onclick="calc.input.value+='7'">
<input type="button" name="eight" value="8" onclick="calc.input.value+='8'">
<input type="button" name="nine" value="9" onclick="calc.input.value+='9'">
<input type="button" name="times" value="x" onclick="calc.input.value+='*'">
</br>
<input type="button" name="clear" value="c" onclick="calc.input.value=''>
<input type="button" name="zero" value="0" onclick="calc.input.value+='0'">
<input type="button" name="dolt" value="=" onclick="calc.input.value=eval(calc.input.value)">
<input type="button" name="div" value="/" onclick="calc.input.value+='/'">
<br>
</td>
</tr>
</table>
</form>
</head>
```

</html>

Simple Program

```
<!DOCTYPE html>
<html>
<head>
<script>
function validateForm() {
    var x = document.forms["myForm"]["fname"].value;
    if (x == null || x == "") {
        alert("Name must be filled out");
        return false;
    }
}
</script>
</head>
<body>
<form name="myForm" action="demo_form.asp"
onsubmit="return validateForm()" method="post">
Name: <input type="text" name="fname">
<input type="submit" value="Submit">
</form>
</body>
</html>
```

Output

Name:

If the name field blank then message "Name must be filled out" will display.

Name must be filled out

☐ Prevent this page from creating additional dialogs.

OK

How To Check Password?

```
<html>
```

```
<head>
```

```
  <title>Password Checker</title>
```

```
  <link type="text/css" rel="stylesheet" href="mystyle.css">
```

```
  <script type="text/javascript" src="myjs.js"></script>
```

```
</head>
```

```
<body>
```

```
  <div id="div1">
```

```
    <h1>Password Checker</h1>
```

<hr>

<form>

Password

<input type="password" placeholder="Password" required
id="pass1">

Re-Password

<input type="password" onkeyup="checkPass(); return false;"
placeholder="Confirm Password" required id="pass2">

</form>

</div>

</body>

</html>

Password Checker

Password

Re-Password

Myjs.js File

```
function checkPass()
{
    //Store the password field objects into variables ...
    var pass1 = document.getElementById('pass1');
    var pass2 = document.getElementById('pass2');

    //Store the Confirmation Message Object ...
    var message = document.getElementById('confirmMessage');

    //Set the colors we will be using ...
    var goodColor = "#66cc66";
```

```

var badColor = "#ff6666";

//Compare the values in the password field

//and the confirmation field

if(pass1.value == pass2.value){

    //The passwords match.

    //Set the color to the good color and inform

    //the user that they have entered the correct password

    pass2.style.backgroundColor = goodColor;

    message.style.color = goodColor;

    message.innerHTML = "Passwords Match!"

}else{

    //The passwords do not match.

    //Set the color to the bad color and

    //notify the user.

    pass2.style.backgroundColor = badColor;

    message.style.color = badColor;

    message.innerHTML = "Passwords Do Not Match!"

}

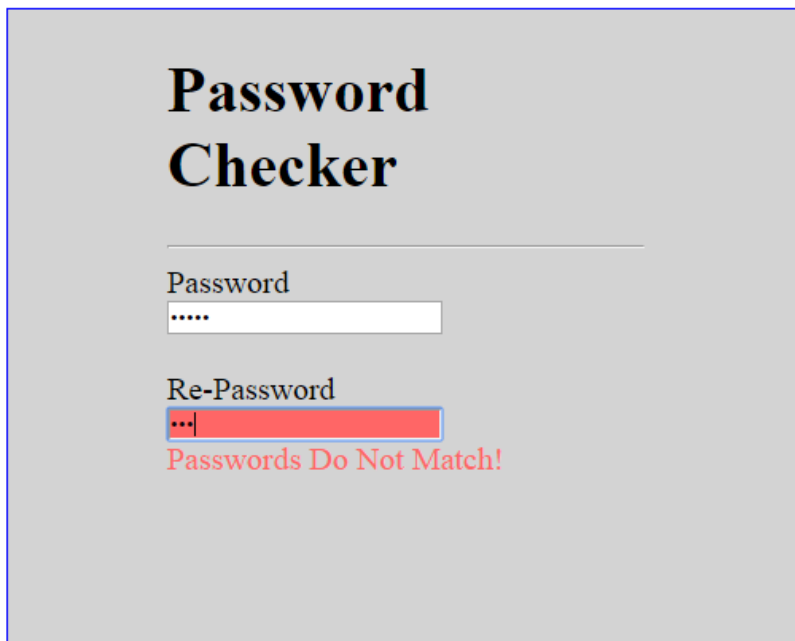
}

```

Mystyle.css File

```
#div1{  
  
    width: 300px;  
  
    height: 400px;  
  
    background: lightgray;  
  
    border: 1px solid blue;  
  
    margin-left: 30%;  
  
    padding: 0px 100px;  
  
    font-size: 20px;  
  
}
```

If Password does not match then output is...



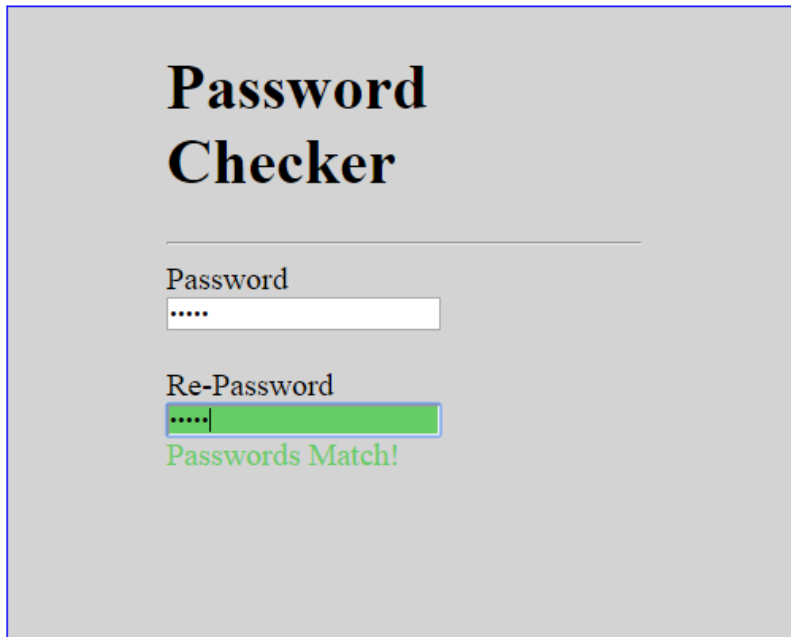
Password Checker

Password
.....

Re-Password
...|

Passwords Do Not Match!

If Password matched then output is...



The image shows a web form titled "Password Checker" on a light gray background. Below the title is a horizontal line. There are two input fields: "Password" and "Re-Password". The "Password" field is a white box with "....." inside. The "Re-Password" field is a green box with "....." inside. Below the "Re-Password" field, the text "Passwords Match!" is displayed in green.

Password Checker

Password
.....

Re-Password
.....

Passwords Match!

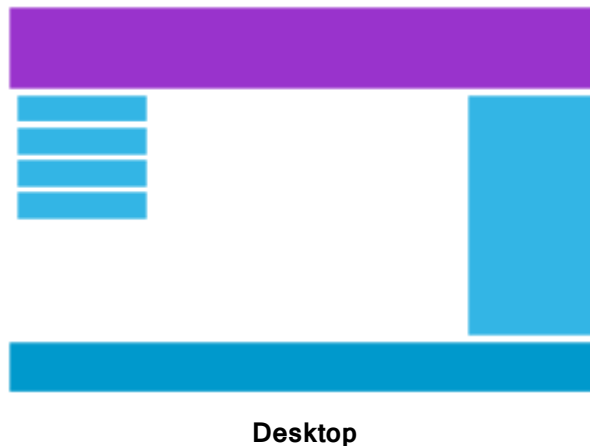
Program No-6

Design Responsive Web Page

Responsive Web Page

Web pages can be viewed using many different devices: desktops, tablets, and phones. Your web page should look good, and be easy to use, regardless of the device.

Web pages should not leave out information to fit smaller devices, but rather adapt its content to fit any device:





Phone

It is called responsive web design when you use CSS and HTML to resize, hide, shrink, enlarge, or move the content to make it look good on any screen.

```
<!DOCTYPE html>
```

```
<html lang="en-us">
```

```
<head>
```

```
<style>
```

```
.city {
```

```
    float: left;
```

```
    margin: 10px;
```

```
    padding: 10px;
```

```
    max-width: 300px;
```

```
    height: 300px;
```

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

```
}
```

```
</style>
```


</head>

<body>

<h1>Responsive Web Design Demo</h1>

<h2>Resize this responsive page!</h2>

<div class="city">

<h2>London</h2>

<p>London is the capital of England.</p>

<p>It is the most populous city in the United Kingdom, with a metropolitan area of over 13 million inhabitants.</p>

</div>

<div class="city">

<h2>Paris</h2>

<p>Paris is the capital of France.</p>

<p>The Paris area is one of the largest population centers in Europe, with more than 12 million inhabitants.</p>

</div>

<div class="city">

<h2>Tokyo</h2>

<p>Tokyo is the capital of Japan.</p>

<p>It is the center of the Greater Tokyo Area, and the most populous metropolitan area in the world.</p>

</div>

<div class="city">

<h2>New York</h2>

<p>The City of New York is the most populous city in the United States.</p>

<p>New York is an important center for international diplomacy and has been described as the cultural and financial capital of the world.</p>

</div>

</body>

</html>

Program No-7

Design a complete student registration form using Server Side Scripting Language

Signup.php File:

```
<!DOCTYPE HTML>
```

```
<html>
```

```
    <head>
```

```
        <title>Sign-Up</title>
```

```
        <link type="text/css" rel="stylesheet" href="mystyle.css">
```

```
        <script type="text/javascript" src="myjs.js"></script>
```

```
    </head>
```

```
    <body>
```

```
        <div id="main">
```

```
            <h1 style="text-align:center">Student Registration</h1>
```

```
            <hr>
```

```
<?php
```

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

```
{
```

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

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

```

$email=$_POST['email'];

$psw=$_POST['psw'];

$repsw=$_POST['repsw'];

$contact=$_POST['contact'];

//use here Database(MySql) program.

}

?>

<form action="signup.php" name="myForm" method="post">

    <div id="div1">

        First Name<br>

        <input type="text" required name="fname" placeholder="First
Name"><br><br>

        Last Name<br>

        <input type="text" required name="lname" placeholder="Last
Name"><br><br>

        Email<br>

        <input type="email" required name="email" placeholder="Email
Address"><br><br>

        Password<br>

```

```

        <input type="password" required id="pass1" name="psw"
placeholder="Password"><br><br>

        Re-Password<br>

        <input type="password" required id="pass2"
onkeyup="checkPass(); return false;" name="repsw" placeholder="Re-Password">

        <span id="confirmMessage"
class="confirmMessage"></span><br><br>

        Contact<br>

        <input type="tel" name="contact" required placeholder="Contact
Number" ><br><br>

        <input type="submit" name="submit" value="Sign-Up" >

    </div>

</form>

<hr>

<p>This is Footer...</p>

</div>

</body>

</html>

```

Mysyle.css file(This is external CSS)

```

#main{

    width: 40%;

```

```
background: #b3b3ff;

height: auto;

margin: 0px 30%;

border: 1px solid green;

font-size: 25px;

}

#div1{

padding: 0px 100px;

}

input[type="text"],

input[type="email"],

input[type="password"],

input[type="tel"],

input[type="submit"]{

padding: 10px;

width: 250px;

font-size: 18px;

border: none;

background: #e6e6ff;

}
```

```
input[type="submit"]{  
  
    background: #333399;  
  
    color: white;  
  
    width: 150px;  
  
    height: 45px;  
  
}
```

Myjs.js File

```
function checkPass()  
{  
  
    //Store the password field objects into variables ...  
  
    var pass1 = document.getElementById('pass1');  
  
    var pass2 = document.getElementById('pass2');  
  
    //Store the Confirmation Message Object ...  
  
    var message = document.getElementById('confirmMessage');  
  
    //Set the colors we will be using ...  
  
    var goodColor = "#66cc66";  
  
    var badColor = "#ff6666";  
  
    //Compare the values in the password field
```

```
//and the confirmation field

if(pass1.value == pass2.value){

    //The passwords match.

    //Set the color to the good color and inform

    //the user that they have entered the correct password

    pass2.style.backgroundColor = goodColor;

    message.style.color = goodColor;

    message.innerHTML = "Passwords Match!"

}else{

    //The passwords do not match.

    //Set the color to the bad color and

    //notify the user.

    pass2.style.backgroundColor = badColor;

    message.style.color = badColor;

    message.innerHTML = "Passwords Do Not Match!"

}

}
```


Student Registration

First Name

Last Name

Email

Password

Re-Password

Contact

Sign-Up

This is Footer...

Program No-8

Design a Login module using Server Side Scripting Language

PHP File

```
<?php

session_start();

?>

<!doctype html>

<html>

<head>

<title>Login Page</title>

    <meta http-equiv="Content-Type" content="text/html; charset=utf-8">

    <style>

        #header1{

            width:100%;

            height:10%;

            background-color:green;

            margin-top:8%;

            margin-left:0px;

            margin-right:0px;

            font-size:30px;
```

```
color:#CF3;

text-align:center;

padding-top:2px;

    border:2px solid #CF0;

}

footer{

    width:100%

    height:100px;

    background-color:green;

    border:1px solid #FF0;

}

section{

    background-color:#FF9;

}

#login{

    border:2px solid #FF0;

    width:400px;

    margin:0px;
```

```

padding:20px;

font-size:24px;

color:green;

font-weight:bolder;

background-color:#FFC;

    }

</style>

</head>

<body style="background-color:#CCC">

    <header id="header1">

        <h1>Welcome to Login Page</h1>

    </header>

    <section ><center>

        <div id="login">

<?php

require("connection.php");

if(isset($_POST['submit']))

{

```

```
$sql="select userid from users where username='".$_POST['username']."'
and password='".$_POST['password']."'";
```

```
if($result=mysql_query($sql))
```

```
{
```

```
    if($rows=mysql_num_rows($result))
```

```
    {
```

```
        if($rows==1)
```

```
        {
```

```
            $_SESSION['username']=$_POST['username'];
```

```
            header('Location: home.php');
```

```
        }
```

```
    else
```

```
    {
```

```
        echo "Invalid login details";
```

```
    }
```

```
}
```

```
else
```

```
{
```

```
    echo "Invalid login details";
```

```
}
```

}

}

?>

```
<form action="####" name="####" method="post">
```

```
<fieldset> <legend>Login Details</legend>
```

```
    Username <input type="text" size="25" width="10" name="username"
placeholder="Username" autofocus required><br><br>
```

```
    Password <input type="password" size="25" name="password"
placeholder="Password" required><br><br>
```

```
    <input type="submit" name="submit" value="Login" >
```

```
</fieldset>
```

```
</form>
```

```
</div></center>
```

```
</section>
```

```
<footer>
```

```
<h1></h1>
```

```
</footer>
```

```
</body>
```

</html>

Connection.php File

```
<?php

if(!mysql_connect("Host_Name","User_Name","Password") ||
!mysql_select_db("Database_Name"))

{

    die(mysql_error());

}

?>
```

Note:

At client machine by default Host_Name is “localhost”, User_Name is “root” and Password is null.

Welcome to Login Page

Login Details

Username

Password

Login

Program No-9

Create a Simple program to demonstrate XML

XML

XML is a **file** extension for an Extensible Markup Language (**XML**) **file** format used to create common information formats and share both the format and the data on the World Wide Web, intranets, and elsewhere using standard ASCII text. **XML** is similar to HTML.

XML Simplifies Things

- It simplifies data sharing
- It simplifies data transport
- It simplifies platform changes
- It simplifies data availability

Many computer systems contain data in incompatible formats. Exchanging data between incompatible systems (or upgraded systems) is a time-consuming task for web developers. Large amounts of data must be converted, and incompatible data is often lost.

XML stores data in plain text format. This provides a software- and hardware-independent way of storing, transporting, and sharing data.

XML also makes it easier to expand or upgrade to new operating systems, new applications, or new browsers, without losing data.

With XML, data can be available to all kinds of "reading machines" like people, computers, voice machines, news feeds, etc.

XML Syntax Rules

The syntax rules of XML are very simple and logical. The rules are easy to learn, and easy to use.

XML Documents Must Have a Root Element

XML documents must contain one **root** element that is the **parent** of all other elements:

```
<root>
  <child>
    <subchild>.....</subchild>
  </child>
</root>
```

Example:

In this example **<note>** is the root element:

```
<?xml version="1.0" encoding="UTF-8"?>

<note>

  <to>Tove</to>

  <from>Jani</from>

  <heading>Reminder</heading>

  <body>Don't forget me this weekend!</body>

</note>
```

XML File

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
<breakfast_menu>
```

```
<food>
```

```
<name>Belgian Waffles</name>
```

```
<price>$5.95</price>
```

```
<description>
```

Two of our famous Belgian Waffles with plenty of real maple syrup

```
<breakfast_menu>
```

```
<food>
```

```
<name>Belgian Waffles</name>
```

```
<price>$5.95</price>
```

```
<description>
```

Two of our famous Belgian Waffles with plenty of real maple syrup

```
</description>
```

```
<calories>650</calories>
```

```
</food>
```

```
<food>
```

```
<name>Strawberry Belgian Waffles</name>
```

```
<price>$7.95</price>
```

```
<description>
```

Light Belgian waffles covered with strawberries and whipped cream

</description>

<calories>900</calories>

</food>

<food>

<name>Berry-Berry Belgian Waffles</name>

<price>\$8.95</price>

<description>

Light Belgian waffles covered with an assortment of fresh berries and whipped cream

</description>

<calories>900</calories>

</food>

<food>

<name>French Toast</name>

<price>\$4.50</price>

<description>

Thick slices made from our homemade sourdough bread

</description>

<calories>600</calories>

</food>

<food>

<name>Homestyle Breakfast</name>

<price>\$6.95</price>

<description>

Two eggs, bacon or sausage, toast, and our ever-popular hash browns

</description>

<calories>950</calories>

</food>

</breakfast_menu>