Date 30/7/24
Page No.

Practical No. 6

Aim- Design a webpage using external CSS.

Theory-

External CSS is used to style multiple HTML pages with a single style sheet. External CSS contains a separate css file with .css extension. The CSS file contains style properties added on selectors.

To link a CSS file to an HTML file, use the <link> element within the HTML file's <head> section with the rel attribute set to "stylesheet" and the href attribute specifying the CSS file's path.

CSS property is written in a separate file with a .css extension and should be linked to the HTML document using a link tag. External CSS sets styles for elements once and applies them consistently across multiple web pages, ensuring a unified design.

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

Date
Page No.

Advantages of External CSS:
* Improved maintainability and code organization.
* Enhanced reusability across multiple HTML files.
* Efficient caching and faster page load times.

Disadvantages of External CSS:
* Pages may not render correctly until the external CSS is loaded.
* Uploading or linking to multiple CSS files may increase your site's download time, affecting its overall performance.
* Large-scale projects may face versioning and caching challenges when using external CSS.

Example:

```
<!DOCTYPE html>
<html>
<head>
<link rel="stylesheet" href="styles.css">
</head>
<body>
<h1> This is a heading </h1>
<p> This is a paragraph. </p>
```

Date

Page No.

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

Result:

Hence, a webpage using external CSS has been designed successfully.

P	T	D	K	Total
3M	3M	3M	6M	15M
3	3	3	5	14.
Sign With Date				

14/8/24

Date 13/8/24
Page No.

Practical No. 7

Aim- Study and design the webpage using various CSS selectors.

Theory-
CSS selectors are used to select the content you want to style. Selectors are the part of CSS rule set. CSS selectors select HTML elements according to its id, class, type, attribute etc.

There are several different types of selectors in CSS:

1) CSS Element Selector
The element selector selects the HTML element by name.


```
<style>
p{
  text-align :center;
  color :blue;
}
</style>
<p> This style will be applied on every paragraph.
</p>
<p id="para1">Me too!</p>
<p> And me! </p>
```

Date
Page No.

2) CSS Id Selector

The id selector selects the id attribute of an HTML element to select a specific element. An id is always unique within the page so it is chosen to select a single, unique element. It is written with the hash character (#), followed by the id of the element.

```
<style>
#para1 {
text-align:center;
color:blue;
}
</style>
<p id="para1">Hello Javatpoint.com</p>
<p>This paragraph will not be affected.</p>
```

3) CSS Class Selector

The class selector selects HTML elements with a specific class attribute. It is used with a period character . (full stop symbol) followed by the class name.

Note: A class name should not be started with a number.

```
<style>
.center {
 text-align:center;
```

Date
Page No.

```
color: blue;
}
</style>
<h1 class = "center">This heading is blue and center aligned. </h1>
<p class = "center"> This paragraph is blue and center-aligned. </p>
```

CSS Class Selector for specific element

If you want to specify that only one specific HTML element should be affected then you should use the element name with class selector.

```
<style>
p.center {
 text-align: center;
 color: blue;
}
</style>
<h1 class = "center">This heading is not affected </h1>
<p class = "center">This paragraph is blue and center-aligned. </p>
```

4) CSS Universal Selector

The universal selector is used as a wildcard character. It selects all the elements on the pages.

Date
Page No.

```
<style>
*{
color :green;
font-size:20px;
}
</style>
<h2> This is heading </h2>
<p> This style will be applied on every paragraph.
</p>
<p id="para1"> Me too! </p>
<p> And me! </p>
```

5) CSS Group Selector

- The grouping selector is used to select all the elements with the same style definitions.
- Grouping selector is used to minimize the code. Commas are used to separate each selector in grouping.

```
h1 {
  text-align :center;
  color : blue;
  }
h2 {
  text-align: center;
  color : blue;
}
p {
```

Date
Page No.

```
text-align: center;
color: blue;
}
```

As you can see, you need to define CSS properties for all the elements. It can be grouped in following ways:

```
h1, h2, p {
text-align: center;
color: blue;
}
<style>
h1, h2, p {
text-align: center;
color: blue;
}
</style>
```


6) CSS Descendant Selector

The descendant selector is used to apply styles to elements that are nested within another elements. It targets elements that are inside a specified ancestor element at any level.

```
CSS
Copy code
ancestor descendant {
/* CSS properties */ }
```

Date
Page No.

7) CSS Child Selector
The child selector is more specific than the descendant selector. It targets only the direct children of a specified element. This means it will only style elements that are immediate children of a parent element, not deeper nested elements.
CSS
parent > child {
/* CSS properties */ }

8) CSS Adjacent Sibling Selector
The adjacent sibling selector targets an element that is immediately following another element. It is used when you want to style an element based on its relationship to a preceding element
element1 + element2 {
/* CSS properties */ }

9) CSS Attribute Selector
The attribute selector is used to style HTML elements based on the presence or value of their attributes.
element[attribute] {
/* CSS properties */
}
element[attribute="value"] {

Date
Page No.

```
/* css properties */ }
element [attribute = "value"] {
/* css properties */ }
```

10) CSS Pseudo-Element Selector

Pseudo-elements are used to style specific parts of an element's content, such as the first letter, first line or to insert content before or after an element's actual content.

```
element::pseudo-element {
/* css properties */
}
```


Result:

Hence a webpage has been designed using CSS Selector successfully.

P	T	D	K	Total
3M	3M	3M	6M	15M
3	3	3	5	14

Sign With Date

2/8/24.

Date 27/8/24
Page No.

Practical No. 8

Aim- Design and Implement all types of popup boxes using Java Script.

Theory-

HTML Popup boxes, otherwise called modal windows or discourse boxes, are a fundamental component in web advancement. These elements assume an urgent part in improving user experience by giving extra information, alerts, or looking for user input without exploring away from the ongoing page.

Popup Boxes:

1) Alert Boxes:

Alert boxes are straightforward notices that show a message to the user. They are made utilizing the alert() capability in JavaScript.

Example:

```
<Script>
alert("This is an alert box!");
</Script>
```

Date
Page No.

2) Confirm Boxes:
Confirm boxes prompt users for a yes or no reaction. They are valuable while looking for user confirmation prior to continuing with an activity.

Example:
```
<script>
var result = confirm("Do you want to proceed?");
if (result) {
//user clicked "OK"
//Perform the desired action
} else {
//user clicked "Cancel"
//Take appropriate action or do nothing
}
</script>
```

3) Prompt Boxes:
Prompt boxes permit users to enter information. They are regularly utilized while gathering user information or reactions.

Example:
```
<script>
var userInput = prompt("Please enter your name:
```

```
if (userInput !== null) {
// User input
// Process input as needed
} else {
// User clicked "Cancel" or closed the prompt
// Handle accordingly
}
</script>
```

4) Modal Windows:
Modal windows are more customizable popup boxes that can contain HTML, CSS and JavaScript. They are frequently utilized for showing nitty gritty information, forms, or intuitive content.

Example:-

```
<div id="myModal" class="modal">
<div class="modal-content">
<span class="close">×</span>
<p>This is a modal window</p>
</div>
</div>
```

10/9/24

Practical No. 9

Aim - Design a calculator in HTML using JAVA Script taking inputs from user

Theory -

1. HTML Layout:
HTML (Hypertext Markup Language) provides the structure of the calculator. We will create buttons for numbers, operations (like +, -, /, *) and a display to show input/output.

Key Elements:
- Input Field: To display the numbers and results.
- Buttons: For digits (0-9), operations (+, -, *, /) and the result button (=).
- Clear Button: To reset the calculator display.

2. JavaScript Functionality:
JavaScript will handle the logic behind calculator. You need functions to:
- Display numbers/operations: Shows the input in the display area when the user presses a button.
- Perform calculations: Using eval() or custom logic to compute the result.
- Clear the screen: Reset the display.

3. CSS Styling for the Calculator:
- Basic Layout: CSS Grid to align the buttons and the display properly.
- Button Styling: Each button should have a distinct size and color scheme to make it easy to use.

How it All Works Together:

1) HTML creates the basic structure of the calculator: the buttons, display area and other elements.

2) CSS styles the buttons, input field and layout to make it look neat and user-friendly.

3) JavaScript adds interactivity by responding to button clicks, performing calculations and updating the display.

Example:
The HTML button `<button onclick="display('7')">7</button>` is clicked.

JavaScript function display('7') is executed and it updates the input field with the number "7". Similarly, click the operator button and another operand button.

The calculate() function is called when '=' button is clicked. It evaluates the expression using eval(), gets the result and updates the display.

Result:
Hence a calculator using JavaScript has been designed Successfully.

P	T	D	K	Total
3M	3M	3M	6M	15M
3	3	3	3	12
Sign With Date				

19/9/24

Practical No. 10

Aim- Design & host a dynamic website using combinations of HTML, CSS & JAVA Script.

Theory-

Web Hosting is a service that allows individuals or businesses to store and serve website files on a server, making them accessible on the internet. These servers are maintained by hosting providers.

Types of Web Hosting:

1) Shared Hosting: Multiple websites share one server; affordable, but slower performance.

2) Virtual Private Server: Virtual Partition on a physical server; better performance than shared hosting.

3) Dedicated Hosting: Full server dedicated to one website; High performance, expensive.

4) Cloud Hosting: Uses multiple connected servers; scalable and reliable.

5) Managed Hosting: Hosting provider manages technical aspects like updates, security, etc.

6) Colocation: You own the hardware, but rent space in a data center.

Hosting vs Publishing:

* Hosting: Storing website files on a server so they can be accessed via the internet.
* Publishing: Uploading and deploying website content (e.g. text, images, code) to the hosting server.

Web Hosting Platforms:

1) GitHub Pages
2) Netlify
3) Vercel
4) Hostinger

GitHub is a cloud based platform that uses Git for version control, enabling developers to store, manage and collaborate on code.

GitHub Pages:
Github Pages allows users to host static websites directly from a GitHub repository.

Hosting Steps on Github:

1) Create a GitHub repository.
2) Add your website files (HTML, CSS, JS) to the repository.
3) Go to repository settings > pages.
4) Set the source branch to "main".
5) Your site will be published at:
https://username.github.io/repository-name.

Result:
A dynamic website has been designed and hosted on Github Pages successfully.

P	T	D	K	Total
3M	3M	3M	6M	15M
3	3	3	5	14
Sign With Date				

21/9/24