**Theory Assignment 1**

**Q1 - What is a CSS Selector ? provide example id, class and element?**

Ans – A CSS selector is a way to select html elements so you can style them with CSS.

1 – Elements Selector

* Selects all elements of a type.
* Targets all of the tags(all the p tag in your page)

Ex-

P{

Color: blue;

}

This will make all <p> tags blue.

2 – Class Selector

* Select elements with specific class.
* Use . before Class name.
* It targets Multiple elements with the same class.

Ex-

Css

. para1 {

Color: red;

}

<p class=”para1”>This is Paragraph </p>

3 – Id Selector

* Selects a single element with specific Id.
* Use # before Id name.
* Targets only one Unique element.

Ex-

Css

#para1{

Color: green;

}

<p id=”para1”>This is Paragraph </p>

**Q2 – Explain the concept CSS specificity. How do conflicts between multiple styles get resolved?**

ANS – CSS specificity is the rule that decides which CSS style will be applied when there are multiple styles targeting the same element.

When two or more CSS rules conflict, the browser look at the specificity strength of each selector and applies the one with higher priority.

Order of specificity(From Highest to Lowest)

**1** Inline Styles (styles with directly inside an element using the style attribute).

**2** Id Selector (#id).

**3** Class Selector, Attribute selectors, and Pseudo-Classes (.Class, [type=”text”], :hover)

**4** Element Selectors (p, h1, div) and pseudo-elements (::before, ::after).

* If two Rules have the same specificity, the browser will apply the one that comes last in the CSS file.

**Q3 – What is Difference between Internal, Inline, and External CSS ? Discuss the advantages and dis-advantages of each approach .**

Ans –

**1** – Inline CSS : CSS written directly inside an element using a style attribute.

Ex- <p style = “color : red ”>This is Inline CSS </p>

Advantages :

* Quick and easy for small changes.
* Styles apply immediately to that element only.

Dis-advantages :

* Not reusable.
* Makes html code messy.
* Hard to manage for large project.

**2** – Internal CSS : CSS written inside the <style> tag in the <head> section of an html file.

Ex- <style>

P { color : blue }

</style>

Advantages :

* Useful for single page styling.
* Styles are kept separate from element tags.

Dis-advantages:

* Cannot be reused across multiple pages.
* Increase page size if styles are large.

**3** – External CSS – CSS written in a separate .CSS file and linked using <link rel = ”stylesheet”>.

Ex- <link rel = ”stylesheet” href=”Style.css ”>

Advantages:

* Styles are usable across multiple pages.
* Make code cleaner and easier to manage.
* Reduce Html file size.

Dis-advantages:

* Requires extra HTTP request to load the CSS file.
* If CSS file is missing, the page may look unstyled.

**Theory Assignment 2**

**Q1 Explain the CSS box model and its components (content, padding, border, margin). How does each affect the size of an element?**

Ans – The Box model describes how the size of this box is calculated and how elements interact with each other.

It has four main components

1. Content
2. Padding
3. Border
4. Margin
5. Content – The actual text, images, or inside the element.

* Control by Properties like Width & Height.
* This Example defines the size of content area only.

Ex-

Div{

Width: 500px;

Height: 500px;

}

1. Padding – Space between the content and the border.

* Makes the box look bigger, but keeps the background color visible.

Ex-

Div{

Padding: 20px;

}

1. Border– A line that surrounds the padding and content.

* Controlled by border-width , border-style, and border-color.

Ex-

Div{

Border: 5px solid black;

}

1. Margin – Space outside the border, between the border.

* It creates separation between boxes.

Ex-

Div{

Margin: 25px;

}

**Q2 What is the difference between border-box and content-box-sizing in CSS? which is the default?**

Ans – In CSS, the Box-sizing property controls how the width and height of an element are calculated.

1. Content-box – This is default property of CSS. width and height apply only to the content.

- Padding and border are added on top of the defined width/height.

- This means the actual visible size of the element increases if you add padding or border.

1. Border-box – Width & height include content, padding and border.

- The defined width/height is the total size, and content space shrinks to accommodate padding & border.

**Theory Assignment 3**

**O1 – What is CSS Flexbox and how is it useful for layout design? Explain the terms flex-container and flex-item.**

Ans – Flexbox (flexible Box Layout) is a CSS layout system that makes it easier to arrange elements in row and columns, and to distribute space between them.

Its very useful when you want your layout to adjust automatically for different screen sizes.

1 – Flex-Container – The parent element where you apply display: flex;

It acts like a box that controls how its flex-item behave.

Ex-

.container{

display: flex;

}

.container is the Flex-container.

2 – Flex-items – the child elements inside the flex-container.

These automatically align & adjust according to the container`s flex rules.

Ex-

<div class=”container”>

<div class=”item”>one</div>

<div class=”item”>two</div>

<div class=”item”>three</div>

</div>

.item elements are the flex-items.

**Q2 – Describe the properties justify-content, align-items, and flex-direction used in flexbox?**

Ans -

1. flex-direction – Defines the direction in which the flex-items are placed inside the container.

Options:

* row – Items go left to right(default).
* row-reverse – Items go right to left.
* Column – Items go top to bottom.
* Column – Items go bottom to top.

1. Justify-content – Controls how flex-items are aligned along the main axis.

Options:

* Flex-start – Items start at the beginning (left or top).
* Flex-end – Items go to end (right or bottom)
* Center – Items are centered in the line.
* Space-around – equal space around each item.
* Space-evenly – Equal space between all items, including edges.

1. Align-items – Controls how flex-items are aligned along the cross-axis.

Options :

* Flex-start – Items stick at the top (if row) or left(if column).
* Flex-end – Items stick at the bottom (if row) or to the right(if in a column).
* Center – Items sit in the middle line.
* Stretch(default) – Items grow taller(or wider) to fill the container.

**Theory Assignment 4**

**Q1 – Explain CSS grid and how it differs from flexbox. When would you use grid over flexbox?**

Ans **–** CSS Grid is a layout system that lets you arrange elements in roes & Columns at the same time.

It gives the **2D** control (both horizontal and vertical).

Flexbox – Works in one direction (row or column).

Grid **–** Works in two direction (rows and columns).

Flexbox **–** Content drives the layout (items adjust and flow).

Grid **–** Layout defines the content (you design the grid first, then place items).

**When would you use grid over flexbox –**

Use grid when you need a full page layout (header, sidebar, content, footer).

Use grid when you need items arranged in rows and columns together.

**Q2 – Describe the grid-template-columns, grid-template-rows, and grid-gap properties. Provide example of how to use them.**

Ans **–**

1. Grid-template-columns **–** Defines the number of columns and their sizes in grid.

Ex **–**

.container {

display: grid ;

grid-template-columns : 100px 200px auto;

}

This creates 3 columns  
 1st column = 100px

2nd column = 200px

3rd column = takes the remaining space(auto).

1. Grid-template-rows **–** Defines the number of rows and their sizes in a grid.

Ex **–**

.container {

display: grid ;

grid-template-rows : 50px 100px auto;

}

This creates 3 columns  
 1st column = 50px

2nd column = 100px

3rd column = takes the remaining height(auto).

1. Grid-gap **–** Defines the gap between the grid cells(row and columns).

Ex **– .**container {

display: grid;

grid-template-columns: auto auto;

grid-template-rows:auto auto auto;

grid-gap: 20px;

}

**Theory Assignment 5**

**Q1 – What are media queries in CSS, and why are they important for responsive design?**

Ans **–** Media Queries are special CSS rules that let your website change its style depending on the device or screen size.

They check things like:

* Screen width / height.
* Device type (Phone, tablet, desktop).
* Screen orientation (portrait / landscape).

Why are media Queries imp.. for responsive design **–**

* Different devices have different screen sizes (mobile, tablet, desktop).
* Without Media queries your website might look good on desktop but broken on mobile.
* With media queries, you can:
  + Resize text for smaller screens.
  + Change layouts(eg 3 columns **–** 1 columns in mobile).
  + Hide or show elements depending on device.
  + Make your site responsive (works well everywhere).

**Q2 – Write a basic media query that adjusts the font size of a webpage for screens smaller than 600px?**

Ans **–**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Media Query Example</title>

<style>

body {

font-size: 18px;

line-height: 1.6;

font-family: Arial, sans-serif;

}

h1 {

font-size: 36px;

color: darkblue;

}

p {

font-size: 18px;

color: #333;

}

@media (max-width: 600px) {

body {

font-size: 14px;

}

h1 {

font-size: 24px;

color: darkred;

}

p {

font-size: 14px;

}

}

</style>

</head>

<body>

<h1>Welcome to Example of Media Query</h1>

<p>This is an example of how media queries adjust font sizes depending on screen size.</p>

</body>

</html>