[CSS GRID]

Niraj _Vadhiya

[WD - CSS AND CSS 3]

[CSS Grid]

Question 1: Explain CSS Grid and how it differs from Flexbox. When would you use Grid over Flexbox?

Ans: - CSS Grid is a two-dimensional layout system for the web. It allows you to create complex and responsive grid-based layouts with ease. Unlike Flexbox, which is primarily one-dimensional (either row or column), CSS Grid can handle both rows and columns simultaneously.

Key Differences Between CSS Grid and Flexbox:

1. **Layout Dimension:**

Flexbox: One-dimensional layout model. It deals with either rows or columns, but not both at the same time.

Grid: Two-dimensional layout model. It can handle both rows and columns, making it suitable for more complex layouts.

2. Use Cases:

Flexbox: Best for aligning items in a single direction (either horizontally or vertically). Ideal for components like navigation bars, form controls, and other UI elements that need to be aligned in a row or column.

Grid: Best for creating overall page layouts and complex grid-based designs. Ideal for layouts that require precise control over both rows and columns, such as web page layouts, image galleries, and dashboards.

Parent-Child Relationship:

Flexbox: The relationship is between the flex container (parent) and flex items (children).

Grid: The relationship is between the grid container (parent) and grid items (children), but it also allows for more complex nesting and positioning within the grid.

When to Use Grid Over Flexbox:

Use Grid when you need a two-dimensional layout, such as a web page layout with both rows and columns.

Use Flexbox when you need a one-dimensional layout, such as aligning items in a single row or column.

Question 2: Describe the grid-template-columns, grid-template-rows, and grid-gap properties. Provide examples of how to use them.

Ans: - grid-template-columns:

Defines the column structure of the grid container.

Specifies the number and size of the columns.

Example:

```
.container {
   display: grid;
   grid-template-columns: 100px 200px 100px;
}
```

This creates a grid with three columns: the first and third columns are 100px wide, and the second column is 200px wide.

grid-template-rows:

Defines the row structure of the grid container.

Specifies the number and size of the rows.

Example:

```
.container {
   display: grid;
   grid-template-rows: 50px 100px 50px;
}
```

This creates a grid with three rows: the first and third rows are 50px high, and the second row is 100px high.

grid-gap:

[CSS Grid]

Defines the space between the rows and columns.

Can be set using grid-row-gap for row gaps, grid-column-gap for column gaps, or grid-gap for both.

Example:

```
.container {
   display: grid;
   grid-template-columns: 1fr 1fr 1fr;
   grid-template-rows: auto;
   grid-gap: 10px;
}
```

This creates a grid with three equal-width columns and a 10px gap between both rows and columns.

Example of CSS Grid in Action:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width,
initial-scale=1.0">
  <title>CSS Grid Example</title>
  <style>
    .container {
      display: grid;
      grid-template-columns: 1fr 2fr 1fr;
      grid-template-rows: 100px 200px;
      grid-gap: 20px;
      background-color: #f0f0f0;
      padding: 20px;
    }
    .item {
            background-color: #007bff;
            color: white;
            padding: 20px;
            border-radius: 5px;
        }
    </style>
</head>
<body>
    <div class="container">
        <div class="item">Item 1</div>
        <div class="item">Item 2</div>
        <div class="item">Item 3</div>
        <div class="item">Item 4</div>
        <div class="item">Item 5</div>
        <div class="item">Item 6</div>
    </div>
</body>
</html>
```

[CSS Grid]

In this example:

The container is a grid container with display: grid.

grid-template-columns: 1fr 2fr 1fr creates three columns with the middle column being twice as wide as the others.

grid-template-rows: 100px 200px creates two rows with the second row being twice as tall as the first.

grid-gap: 20px adds a 20px gap between both rows and columns.

The items are grid items that are placed within the grid container.