**CSS Grid System**

**1. What is CSS Grid?**

**Definition:**

* CSS Grid is a **two-dimensional layout system** in CSS.
* It allows you to **design web layouts in rows and columns simultaneously**.
* Unlike Flexbox (1D), Grid handles **both horizontal and vertical alignment**.

**Key Idea:**

Grid provides a **container → grid items** structure to precisely place elements in rows and columns.

**2. Basic Structure**

* **Grid Container:** Parent element with display: grid or display: inline-grid.
* **Grid Items:** Direct children of the grid container.

**Example:**

<div class="grid-container">

<div>Item 1</div>

<div>Item 2</div>

<div>Item 3</div>

<div>Item 4</div>

</div>

.grid-container {

display: grid;

}

* All children are now **grid items**.

**3. Grid Container Properties**

| **Property** | **Description** |
| --- | --- |
| display: grid | Defines a block-level grid container |
| display: inline-grid | Defines an inline-level grid container |
| grid-template-columns | Defines **column sizes and count** |
| grid-template-rows | Defines **row sizes and count** |
| grid-template-areas | Names areas in the grid for placement |
| gap / row-gap / column-gap | Spacing between rows and columns |
| justify-items | Horizontal alignment of items in their cells |
| align-items | Vertical alignment of items in their cells |
| justify-content | Alignment of the entire grid along main axis |
| align-content | Alignment of entire grid along cross axis |

**4. Grid Item Properties**

| **Property** | **Description** |
| --- | --- |
| grid-column-start / grid-column-end | Start and end positions for columns |
| grid-row-start / grid-row-end | Start and end positions for rows |
| grid-column | Shorthand for start/end column |
| grid-row | Shorthand for start/end row |
| grid-area | Assign item to a named area |
| justify-self | Align single item horizontally in its cell |
| align-self | Align single item vertically in its cell |

**5. Defining Columns & Rows**

**A. Fixed Sizes**

.grid-container {

display: grid;

grid-template-columns: 100px 200px 150px; /\* 3 columns with fixed width \*/

grid-template-rows: 50px 100px; /\* 2 rows \*/

}

**B. Fractional Units (fr)**

grid-template-columns: 1fr 2fr 1fr;

/\* middle column gets double space \*/

**C. Auto & Minmax**

grid-template-columns: auto 200px minmax(100px, 300px);

**6. Using Gap Between Grid Items**

grid-container {

display: grid;

grid-template-columns: repeat(3, 1fr);

gap: 20px; /\* horizontal and vertical \*/

row-gap: 10px; /\* optional row spacing \*/

column-gap: 15px; /\* optional column spacing \*/

}

* gap → shorthand for row + column spacing

**7. Placing Items in Grid**

**A. Using Line Numbers**

.item1 {

grid-column: 1 / 3; /\* spans columns 1 and 2 \*/

grid-row: 1 / 2; /\* row 1 \*/

}

**B. Using Named Areas**

.grid-container {

display: grid;

grid-template-areas:

"header header"

"sidebar content"

"footer footer";

}

.header { grid-area: header; }

.sidebar { grid-area: sidebar; }

.content { grid-area: content; }

.footer { grid-area: footer; }

**8. Repeat Function**

grid-template-columns: repeat(4, 1fr);

/\* creates 4 equal columns \*/

* Saves typing for repeated columns/rows.

**9. Auto-fill / Auto-fit with minmax**

* **Responsive grid with minimum and maximum size**

grid-template-columns: repeat(auto-fill, minmax(150px, 1fr));

* Fills as many columns as fit in the container.
* Auto-fit collapses empty tracks.

**11. Quick Tips / Interview Points**

1. Grid is **2D** (row + column), Flexbox is 1D.
2. Use **fr units** for flexible layout.
3. Use **repeat()** and **minmax()** for responsive grids.
4. grid-template-areas → makes layout readable and semantic.
5. **gap** is preferred over using margin for spacing between items.
6. Can **combine Grid & Flexbox** for complex layouts.

**12. Common Interview Questions**

1. Difference between **Flexbox and Grid**?
2. How to **create a 3x3 grid** using Grid?
3. How to **span an item across multiple columns**?
4. Explain **grid-template-areas**.
5. How to make a **responsive grid** with minmax() and auto-fill?
6. What is the difference between **justify-items, align-items, justify-content, and align-content** in Grid?