CSS Grid Layout

CSS Grid Layout Module

The **CSS Grid Layout Module** provides a **grid-based layout system** with **rows and columns**.
It allows developers to design **complex and responsive web layouts** easily, without using floats or positioning.

CSS Grid is mainly used to create **page-level layouts** such as headers, footers, sidebars, and content areas.

Grid vs Flexbox

CSS Grid	CSS Flexbox
Two-dimensional (rows and columns)	One-dimensional (row or column)
Used for full page layout	Used for components
Controls both width and height	Controls layout in one direction

---

CSS Grid Components

A CSS Grid layout consists of:

1. Grid Container

The parent element where:

display: grid;

or

display: inline-grid;

is applied.

2. Grid Items

The **direct children** of the grid container automatically become grid items.

Simple Grid Example

HTML

```
<div class="container">
  <div>1</div>
  <div>2</div>
  <div>3</div>
  <div>4</div>
  <div>5</div>
</div>
```

CSS

```
.container {
  display: grid;
  grid-template-columns: auto auto auto;
  gap: 10px;
  background-color: dodgerblue;
  padding: 10px;
}

.container div {
  background-color: #f1f1f1;
  border: 1px solid black;
  padding: 10px;
  font-size: 20px;
  text-align: center;
}
```

Result:
The container is divided into **three columns**, and grid items are placed automatically.

Important CSS Grid Properties

grid-template-columns

Defines the **number and width of columns**.

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

Creates three equal-width columns.

---

grid-template-rows

Defines the height of rows.

grid-template-rows: 100px auto;

---

gap

Sets space between rows and columns.

gap: 20px;

---

column-gap

Sets space between columns only.

column-gap: 30px;

---

grid-column

Specifies how many columns an item should span.

```
.item1 {
  grid-column: 1 / 3;
```

}

---

grid-row

Controls row placement.

grid-row: 1 / 3;

}

---

justify-content

Aligns the entire grid **horizontally** inside the container.

justify-content: center;

---

align-content

Aligns the entire grid **vertically** inside the container.

align-content: center;

Real-World Layout Example

CSS Grid is commonly used for:

- Website headers
- Sidebars
- Content sections
- Footers
- Responsive page layouts

Why Use CSS Grid?

- Easy layout control

- Clean and readable code
- Responsive design support
- No need for floats or positioning
- Ideal for full website layouts

CSS Grid Concepts Explained

---

1. Grid Container

A **Grid Container** is the parent element where the CSS Grid layout is applied.

To make an element a grid container, we use:

display: grid;

or

display: inline-grid;

Example

```
.container {
  display: grid;
}
```

Use:
The grid container holds all grid items and controls the overall grid structure.

---

2. Grid Tracks

Grid Tracks are the **rows and columns** of a grid.

- Columns are created using grid-template-columns
- Rows are created using grid-template-rows

Example

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

Use:
Grid tracks define the size and layout structure of the grid.

---

3. Grid Gaps

Grid Gaps define the space between rows and columns in a grid.

Properties:

- row-gap
- column-gap
- gap (shorthand)

Example

```
.container {
  display: grid;
  gap: 20px;
}
```

Use:
Creates spacing between grid items without using margins.

---

4. Grid Align

Grid Align properties control the alignment of grid items and the grid itself.

Common Properties:

Property	Purpose
justify-items	Align items horizontally
align-items	Align items vertically
justify-content	Align entire grid horizontally
align-content	Align entire grid vertically

Example

```
.container {
  justify-items: center;
  align-items: center;
}
```

Use:
Helps position items neatly inside the grid.

---

5. Grid Items

Grid Items are the **direct child elements** of a grid container.

They can be placed or sized individually using grid properties.

Example

```
.item1 {
  grid-column: 1 / 3;
  grid-row: 1 / 2;
}
```

Use:
Allows precise placement of content inside the grid.

---

6. Grid 12-Column Layout

A **12-column grid layout** is commonly used in modern websites (Bootstrap-style layouts).

Example

```
.container {
  display: grid;
  grid-template-columns: repeat(12, 1fr);
  gap: 10px;
}
.header {
  grid-column: 1 / 13;
}

.sidebar {
  grid-column: 1 / 4;
}

.content {
  grid-column: 4 / 13;
}
```

Use:
Creates flexible and responsive layouts for web pages.

---

7. CSS @supports

The **@supports** rule checks whether a browser supports a specific CSS feature.

Syntax

```
@supports (display: grid) {
  .container {
    display: grid;
  }
}
```

Example

```
@supports not (display: grid) {
  .container {
    display: block;
  }
}
```

Use:
Ensures browser compatibility and graceful fallbacks.
