

# CSS Grid Area Explained

You said:



explain grid area in detail with examples

ChatGPT said:



The `grid-area` property in CSS is a powerful tool used in CSS Grid Layouts to define where an element should be placed within a grid. It can be used to both place an item within a grid container and define the span of rows and columns the item should occupy.

## Basic Syntax

The `grid-area` property has a shorthand form and a longhand form.

### Longhand Form

The longhand form of `grid-area` consists of four properties:

- `grid-row-start`
- `grid-column-start`
- `grid-row-end`
- `grid-column-end`

These properties define the starting and ending positions of the grid item along the grid's rows and columns.

### Shorthand Form

The shorthand syntax for `grid-area` is:

css

```
grid-area: row-start / column-start / row-end / column-end;
```

Where:

- `row-start`: Specifies the starting row line.
- `column-start`: Specifies the starting column line.
- `row-end`: Specifies the ending row line.
- `column-end`: Specifies the ending column line.

## Example 1: Basic Grid Area

## html

```
<div class="grid-container">
  <div class="item1">1</div>
  <div class="item2">2</div>
  <div class="item3">3</div>
  <div class="item4">4</div>
</div>
```

## css

```
.grid-container {
  display: grid;
  grid-template-columns: repeat(4, 100px);
  grid-template-rows: repeat(2, 100px);
  gap: 10px;
}

.item1 {
  grid-area: 1 / 1 / 2 / 3; /* Spans from row 1 to 2, column 1 to 3 */
}

.item2 {
  grid-area: 1 / 3 / 3 / 5; /* Spans from row 1 to 3, column 3 to 5 */
}

.item3 {
  grid-area: 2 / 1 / 3 / 2; /* Spans from row 2 to 3, column 1 to 2 */
}

.item4 {
  grid-area: 2 / 2 / 3 / 4; /* Spans from row 2 to 3, column 2 to 4 */
}
```

## Explanation

- `.item1`: Starts at row 1 and column 1, ends at row 2 and column 3. It spans 1 row and 2 columns.
- `.item2`: Starts at row 1 and column 3, ends at row 3 and column 5. It spans 2 rows and 2 columns.
- `.item3`: Starts at row 2 and column 1, ends at row 3 and column 2. It spans 1 row and 1 column.
- `.item4`: Starts at row 2 and column 2, ends at row 3 and column 4. It spans 1 row and 2 columns.

## Example 2: Named Grid Areas

You can also use named grid areas, which can simplify layout management.

## html

```
<div class="grid-container">
  <div class="header">Header</div>
  <div class="sidebar">Sidebar</div>
  <div class="main">Main Content</div>
  <div class="footer">Footer</div>
</div>
```

css

```
.grid-container {
  display: grid;
  grid-template-columns: 150px 1fr;
  grid-template-rows: auto 1fr auto;
  grid-template-areas:
    "header header"
    "sidebar main"
    "footer footer";
}

.header {
  grid-area: header;
}

.sidebar {
  grid-area: sidebar;
}

.main {
  grid-area: main;
}

.footer {
  grid-area: footer;
}
```

## Explanation

- `.grid-template-areas`: Defines the overall layout of the grid using named areas.
- `.header`, `.sidebar`, `.main`, `.footer`: Assigns each element to a specific named area.

This approach allows for a more semantic and easy-to-read layout definition.

## Key Points:

- Grid Lines:** The numbers in `grid-area` refer to the grid lines, not the grid cells. Lines are numbered starting from 1.
- Spanning:** You can make an item span multiple rows or columns by specifying different start and end points.
- Named Areas:** Using named areas makes your grid layout more readable and maintainable.

The `grid-area` property provides a flexible and powerful way to define complex layouts with minimal code.

ChatGPT can make mistakes. Check important info.