CSS Grid cheatsheet

Container

```
.grid-container {
```

```
  /* Display properties */
  display: grid;
  display: inline-grid;
  display: subgrid;
```

```
  /* Columns and rows */
  grid-template-columns: 1rem 2rem 1rem; /* Measurement units */
  grid-template-columns: 25% 50% 25%; /* Percentage units */
  grid-template-columns: 1rem auto 1rem 2fr; /* Fill remaining widths with auto or fr units */
  grid-template-columns: repeat(12, 1fr); /* Repeat columns without needing to write them */

  grid-template-rows: 1rem 10% auto repeat(5, 10px); /* Mix any group, same rules work for rows */
```

```
  /* Automatic columns and rows */
```

```
grid-auto-columns: 10px; /* No matter how many columns of content end up in the grid, each column will be this s
grid-auto-rows: 1rem; /* No matter how many rows of content end up in the grid, each row will be this same heigh
```

```
/* Areas */
grid-template-areas:
  "header header"
  "main aside"
  "footer footer"; /* Grid-style */

grid-template-areas: "header header" "main aside" "footer footer"; /* Inline-style */
```

```
/* Template shorthand */
grid-template:
  "header header" auto
  "main aside" 100vh
  "footer footer" 10rem
  / 80% 20%;

/* The above is the same as below long-hand */
grid-template-columns: 80% 20%;
grid-template-rows: auto 100vh 10rem;
grid-template-areas:
  "header header"
  "main aside"
  "footer footer";
```

```
/* Gaps */
grid-row-gap: 1rem;
grid-column-gap: 0.5rem; /* Define values separately */

grid-gap: 1rem 0.5rem; /* Short-hand for row / column */
grid-gap: 1rem; /* Gap in both dimensions */
```

```
/* Item justification (horizontal or column alignment) */
justify-items: start; /* Align items to the left */
justify-items: center; /* Align items centered within its column */
justify-items: end; /* Align items to the right */
justify-items: stretch; /* (default) Fills available area (horizontally) */
```

```
/* Item alignment (vertical or row alignment) */
align-items: start; /* Align items to the top */
align-items: center; /* Align items centered within its row */
align-items: end; /* Align items to the bottom */
align-items: stretch; /* (default) Fills available area (vertically) */
```

```
/* Place item shorthand */
place-items: start stretch;

/* The above is the same as below long-hand */
align-items: start;
justify-items: stretch;
```

```
/* Content justification (horizontal or column alignment) */
justify-content: start; /* Align content to the left */
justify-content: center; /* Align content centered horizontally within the grid */
justify-content: end; /* Align content to the right */
justify-content: stretch; /* (default) Fills available area (horizontally) */

justify-content: space-around; /* Chooses a space for both sides of the columns like a left and right margin */
justify-content: space-between; /* Chooses a space to go between columns, no margins on outside of content */
justify-content: space-evenly; /* Chooses a space that goes between all columns and edges consistently */
```

```
/* Content alignment (horizontal or column alignment) */
align-content: start; /* Align content to the top */
```

```
align-content: center; /* Align content centered vertically within the grid */
align-content: end; /* Align content to the bottom */
align-content: stretch; /* (default) Fills available area (vertically) */

align-content: space-around; /* Chooses a space for the top and bottom of the rows like a top and bottom margin
align-content: space-between; /* Chooses a space to go between rows, no margins on outside of content */
align-content: space-evenly; /* Chooses a space that goes between all rows and edges consistently */
```

```
/* Place item shorthand */
place-content: center start;

/* The above is the same as below long-hand */
align-content: center;
justify-content: start;
```

```
/* Automatic grid positioning */

grid-auto-flow: row; /* Left-to-right rows, then top-to-bottom*/
grid-auto-flow: column; /* Top-to-bottom columns, then left-to-right */
grid-auto-flow: dense; /* Responds with best-guess on left-to-right, top-to-bottom order with advanced layouts *
```

```
/* There is one final shorthand for all container properties in one */

/* Explicit grid columns, rows, and areas */
grid:
  "header header" auto
  "main aside" 100vh
  "footer footer" 10rem
  / 80% 20%; /* You can include a template as the only value, which is equivalent to below */
grid-template:
  "header header" auto
```

```
    "main aside" 100vh
    "footer footer" 10rem
    / 80% 20%; /* Which is again equivalent to below */
  grid-template-columns: 80% 20%;
  grid-template-rows: auto 100vh 10rem;
  grid-template-areas:
    "header header"
    "main aside"
    "footer footer";

  /* Automatic grid flows */
  grid: 1rem / auto-flow dense 1fr; /* You can include rows, a flow, and automatic columns, which is equivalent to
  grid-template-rows: 1rem;
  grid-auto-flow: dense;
  grid-auto-columns: 1fr;

  grid: auto-flow dense 1rem / repeat(10, 10%); /* Conversely, you can do the same thing with automatic rows, and
  grid-auto-flow: dense;
  grid-auto-rows: 1rem;
  grid-template-columns: repeat(10, 10%);
```

```
}
```

Child

```
.grid-child {
```

```
  /* Column position */
  grid-column-start: 1;
  grid-column-end: 2;
```

```
grid-column: 1 / 2; /* Short hand */
grid-column: 1 / span 2; /* Span 2 columns without explicitly defining an endpoint */
grid-column: 1; /* Start in and occupy a single column */
```

```
/* Row position */
grid-row-start: 2;
grid-row-end: 4;

grid-row: 2 / 4; /* Short hand */
grid-row: 2 / span 3;/* Span 3 rows without explicitly defining an endpoint */
grid-row: 1; /* Start in and occupy a single row */
```

```
/* Area positioning */
grid-area: header; /* You can use a named grid area from the container */

grid-area: 2 / 1 / 4 / 2; /* Or you can use positioning. This is equivalent to... */
grid-row-start: 2;
grid-column-start: 1;
grid-row-end: 4;
grid-column-end: 2;
```

```
/* Self justification (horizontal or column alignment) */
justify-self: start; /* Align item to the left */
justify-self: center; /* Align item centered within its column */
justify-self: end; /* Align item to the right */
justify-self: stretch; /* (default) Fills available area (horizontally) */
```

```
/* Self alignment (vertical or row alignment) */
align-self: start; /* Align item to the top */
align-self: center; /* Align item centered within its row */
align-self: end; /* Align item to the bottom */
align-self: stretch; /* (default) Fills available area (vertically) */
```

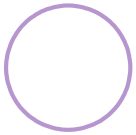
```
  /* Placement shorthand */
  place-self: start stretch;

  /* The above is the same as below long-hand */
  align-self: start;
  justify-self: stretch;

}
```

References

- GRID: A simple visual cheatsheet
- CSS Tricks: A Complete Guide to Grid
- Browser support



Over 383 curated cheatsheets, by developers for developers.

Devhints home

Other CSS cheatsheets

- Sass cheatsheet
- CSS flexbox cheatsheet
- CSS system fonts cheatsheet
- cssnext cheatsheet
- Stylus cheatsheet
- Bootstrap cheatsheet

Top cheatsheets

- Elixir cheatsheet
- ES2015+ cheatsheet
- React.js cheatsheet
- Vimdiff cheatsheet
- Vim cheatsheet
- Vim scripting cheatsheet