DRBEALMAN

CSS GRID LAYOUTS

Learn with Real Examples

CSS GRID

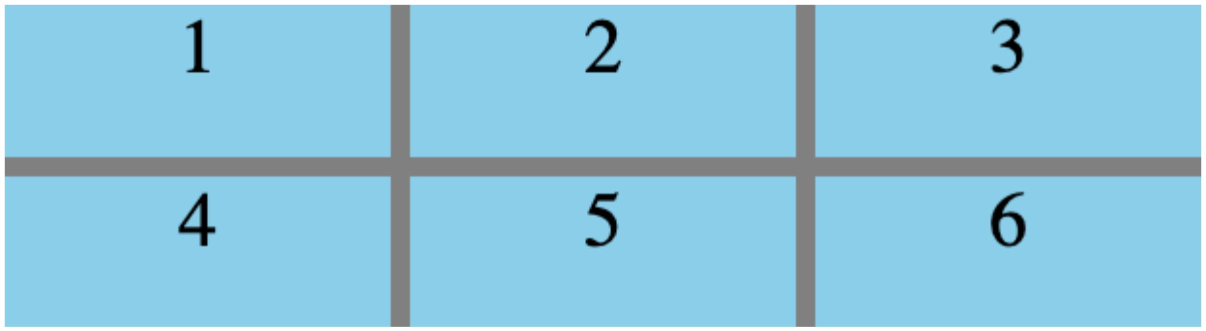
A two-dimensional (2D) grid-based layout system that completely changes the way of designing grid-based user interfaces (UI).

How does CSS Grid work?

- CSS Grid helps you design complex web layouts.

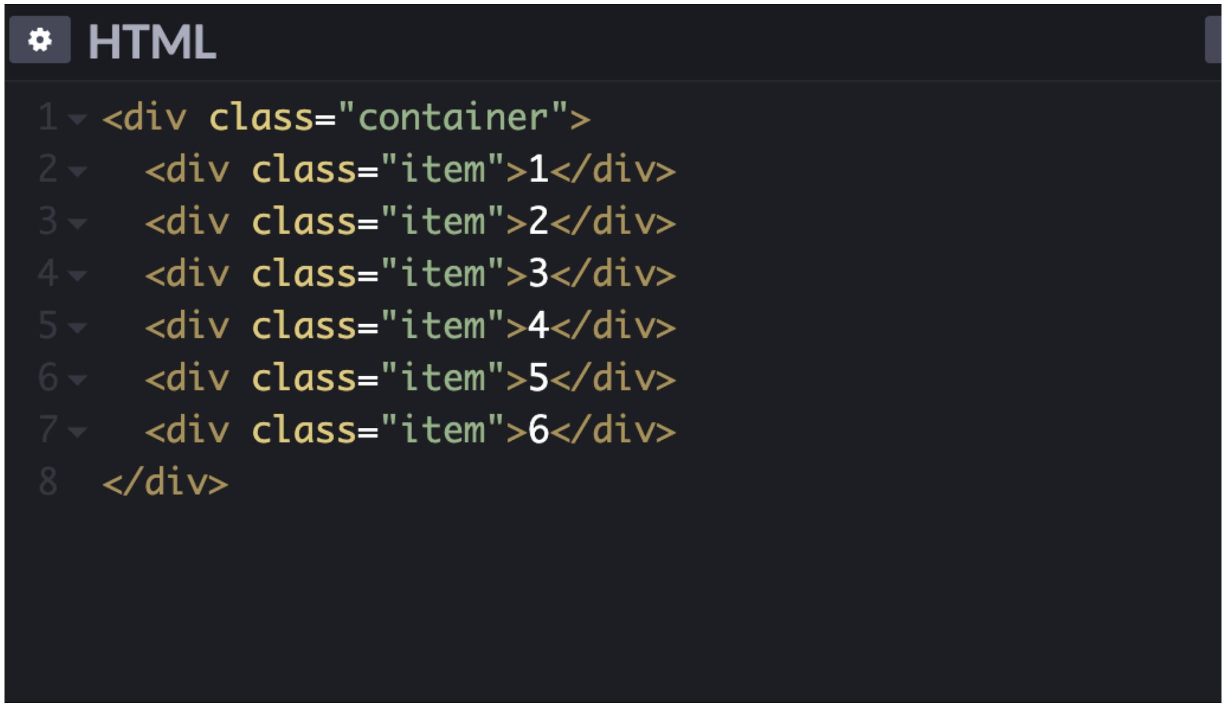
We will learn CSS Grid through layout problems and discuss in-depth solutions.

> *Problem 1: Create a grid container of 2 rows x 3 columns, and make each item size 100px x 40px with a 5px gap in between them.*



2x3 matrix of 100px x 40px and 5px gap in between

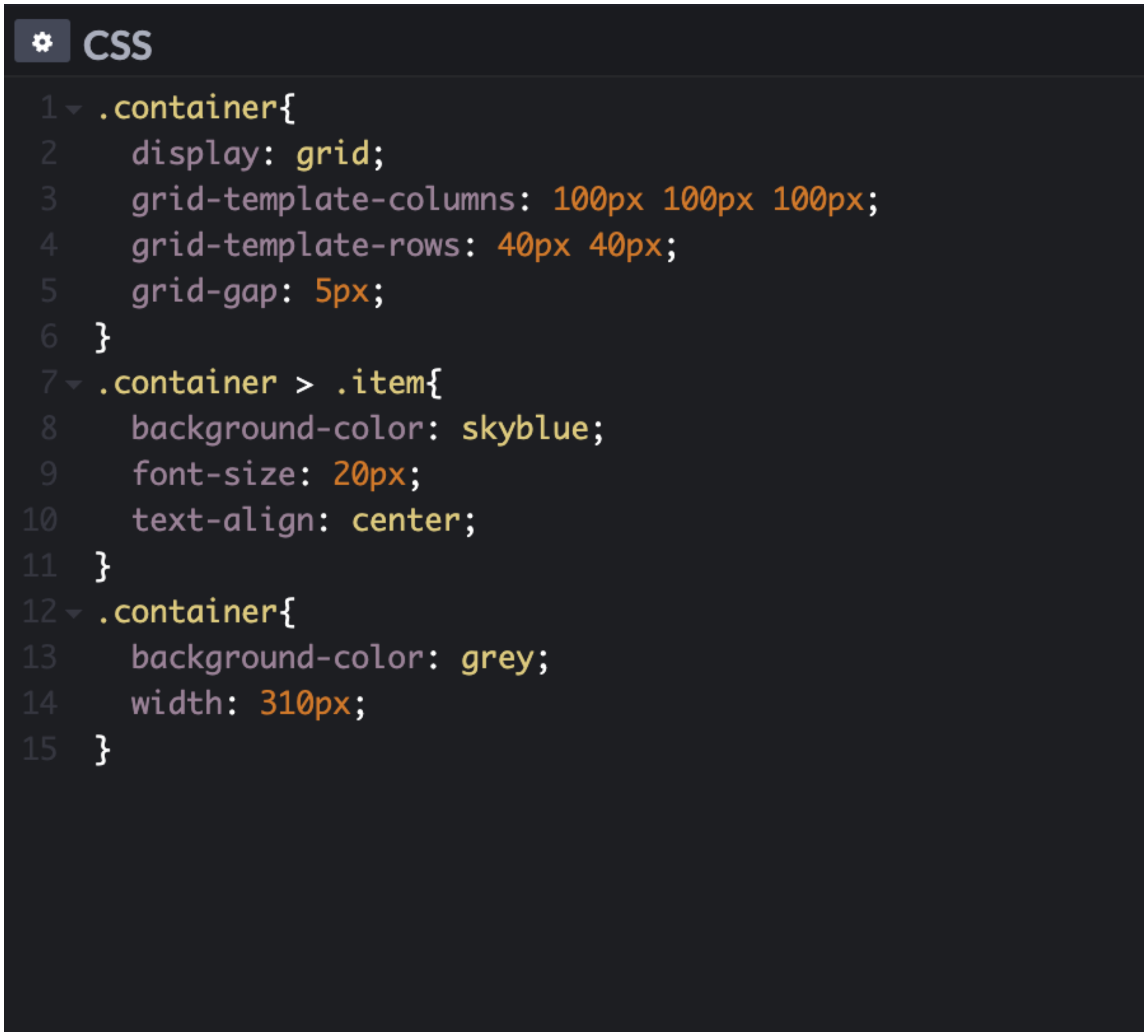
HTML Code:



```html
<div class="container">
  <div class="item">1</div>
  <div class="item">2</div>
  <div class="item">3</div>
  <div class="item">4</div>
  <div class="item">5</div>
  <div class="item">6</div>
</div>
```

HTML Code: 2x3 matrix of 100px x 40px and 5px gap in between

CSS Code:


CSS

```css
.container{
  display: grid;
  grid-template-columns: 100px 100px 100px;
  grid-template-rows: 40px 40px;
  grid-gap: 5px;
}
.container > .item{
  background-color: skyblue;
  font-size: 20px;
  text-align: center;
}
.container{
  background-color: grey;
  width: 310px;
}
```

CSS Code: 2x3 matrix of 100px x 40px and 5px gap in between

Grid container: The line of code below tells the browser to render the element as a grid container.

```
display: grid;
      Or
display: inline-grid;
```

Grid Container: 2x3 matrix of 100px x 40px and 5px gap in between

Grid rows and columns: This tells the browser how many rows or columns the grid container has and their sizes.

```
grid-template-rows: 40px 40px; // 2 rows of 40px each
grid-template-columns: 100px 100px 100px; // 3 columns of 100px each
```

Grid Rows & Columns: 2x3 matrix of 100px x 40px and 5px gap in between

Grid gap: This defines a gap or a gutter space between the rows and between the columns using the grid-gap property, as shown below.

```
grid-gap: 5px; // 5px between rows and 5px between columns
```

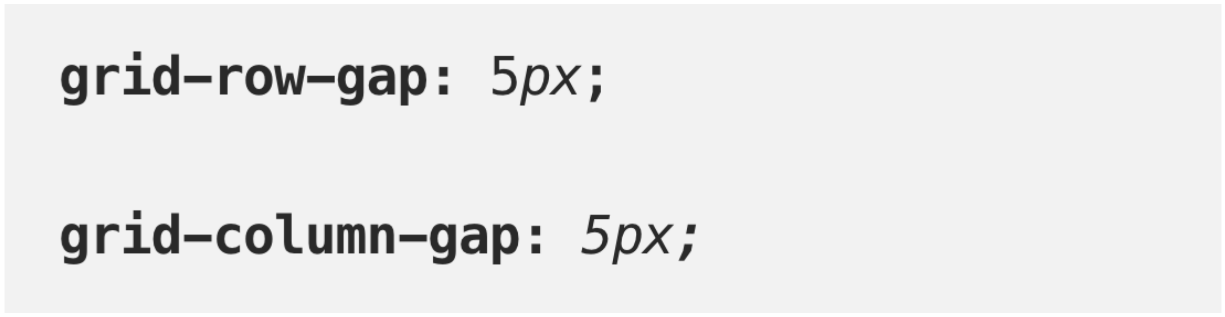
Grid gap: 2x3 matrix of 100px x 40px and 5px gap in between

Or with expanded syntax for row and column gaps:

```
grid-gap: 5px 5px; // row col
```

Expanded syntax: for row & column gaps

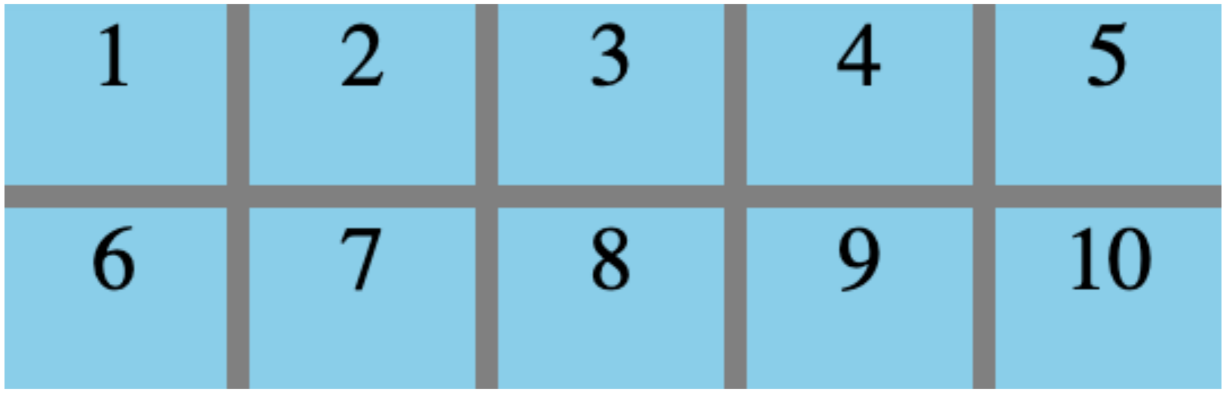
Or as the two separate properties for row and column gaps:



```
grid-row-gap: 5px;

grid-column-gap: 5px;
```

Separate properties: for row & column gaps

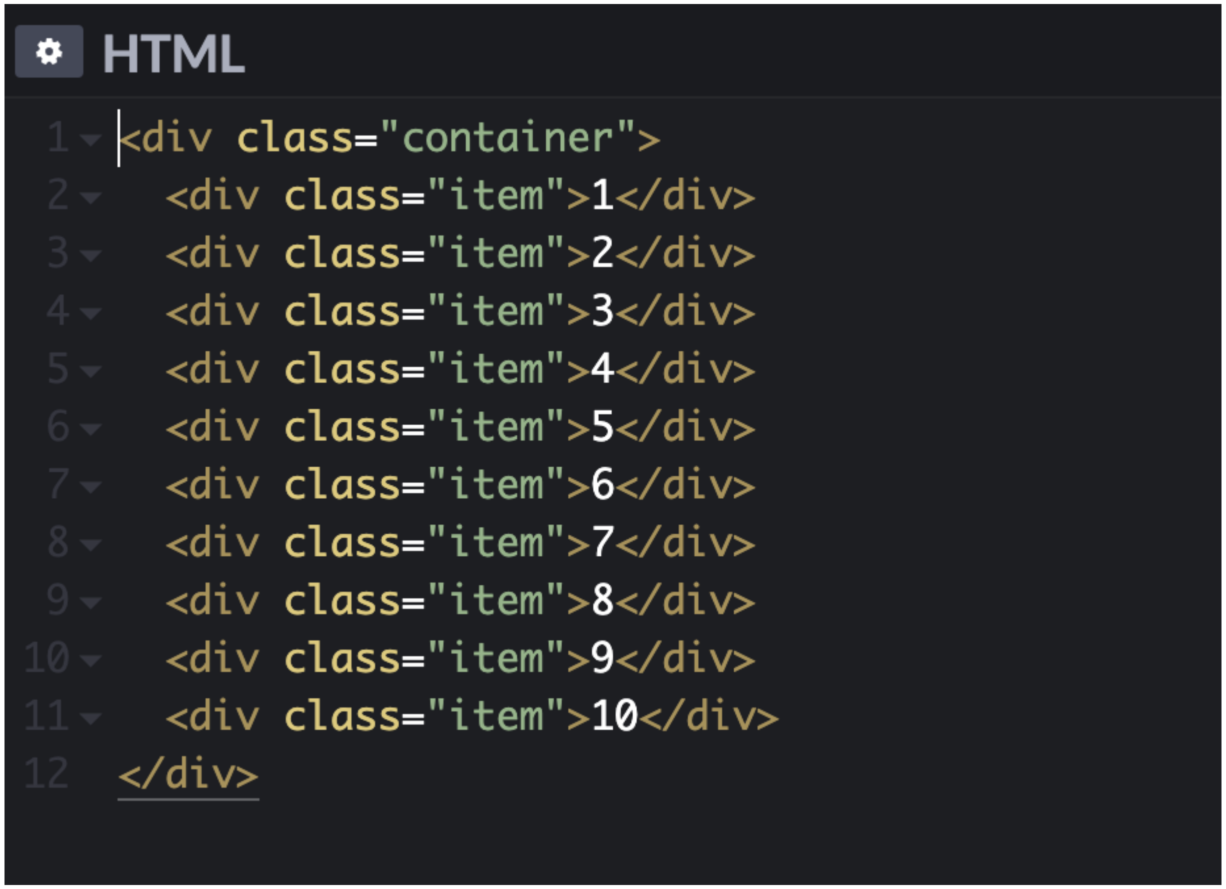
Quick try 1: Create a grid container of 2 rows x 5 columns, and make each item size 50px x 40px.



2 rows x 5 columns, each item size 50px x 40px | view on codepen

> *Problem 2: Create a grid container of 2 rows x 5 columns, and make the columns different widths.*

HTML Code:

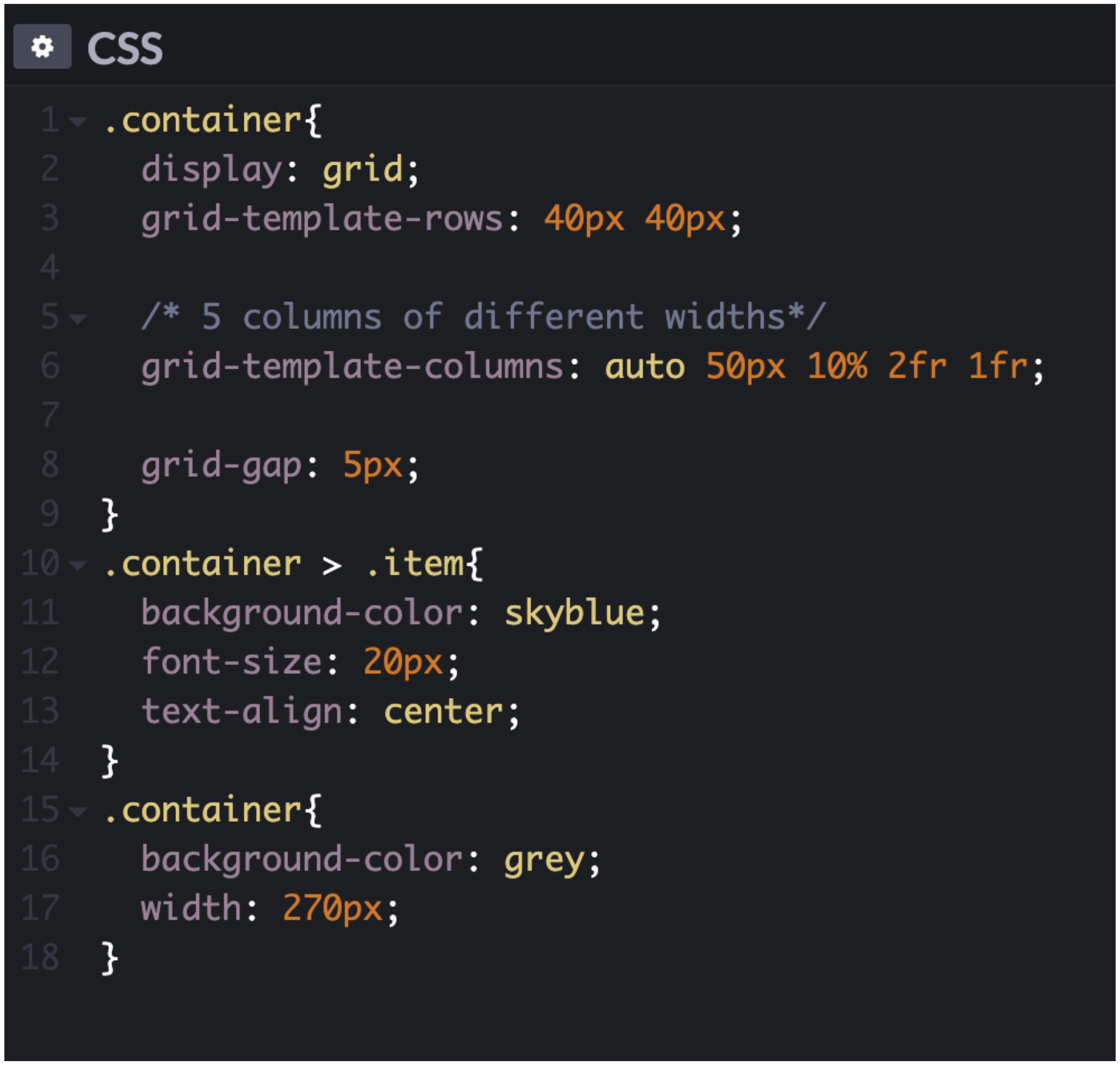


HTML

```html
<div class="container">
  <div class="item">1</div>
  <div class="item">2</div>
  <div class="item">3</div>
  <div class="item">4</div>
  <div class="item">5</div>
  <div class="item">6</div>
  <div class="item">7</div>
  <div class="item">8</div>
  <div class="item">9</div>
  <div class="item">10</div>
</div>
```

HTML Code: Grid container of 2 rows x 5 columns, with columns different widths

CSS Code:



```css
.container{
  display: grid;
  grid-template-rows: 40px 40px;

  /* 5 columns of different widths*/
  grid-template-columns: auto 50px 10% 2fr 1fr;

  grid-gap: 5px;
}
.container > .item{
  background-color: skyblue;
  font-size: 20px;
  text-align: center;
}
.container{
  background-color: grey;
  width: 270px;
}
```

CSS Code: Grid container of 2 rows x 5 columns, with columns different widths

This **CSS snippet** creates five columns:

```
grid-template-columns: auto 50px 10% 2fr 1fr;
```

CSS snippet: creates 5 columns

Have you noticed *four different units are supplied to grid-template-columns*?

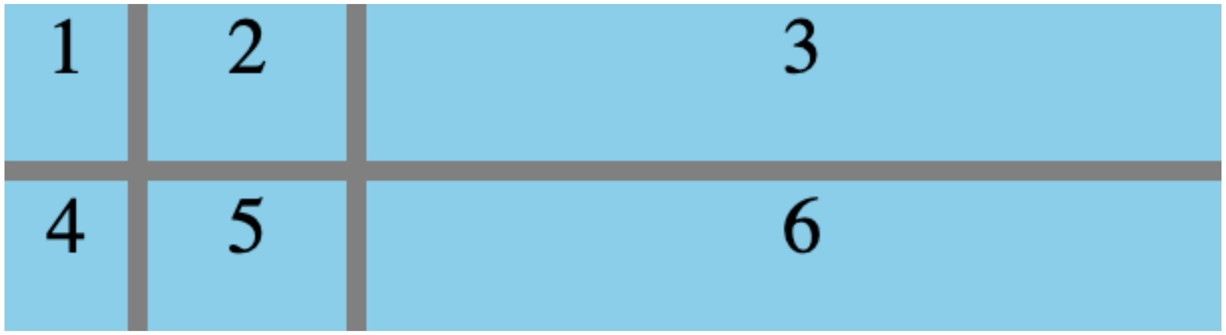
- The first column has auto width, which means it is as wide as its content.
- The second column is of 50px width.
- The third column has 10% width of the container width.
- The fourth and fifth columns are 2fr and 1fr, respectively. fr is a new unit, known as a fractional unit. Let’s see this in detail in the next section.

Fractional unit or fr unit

The fr unit is a fraction of the free space available in the grid container.

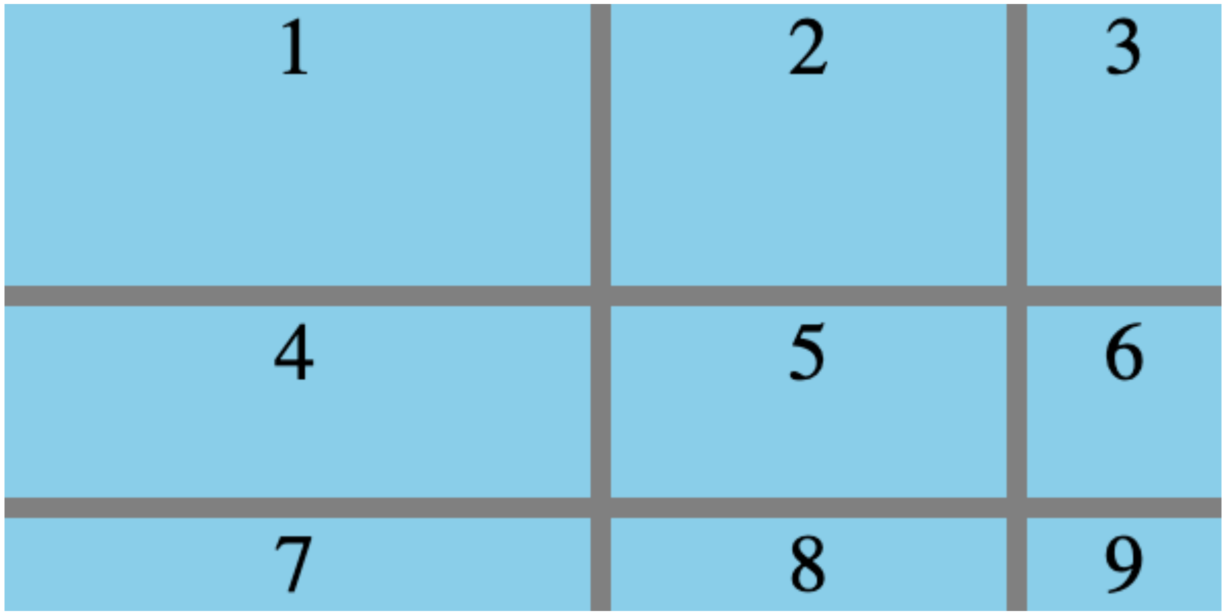
- So, in the above problem, the last two columns have widths 2fr and 1fr, which means the remaining space (after column 1, 2 and 3) is divided into three sections; two are allocated for the fourth column, and one for the fifth.

Quick try 2: Create a grid container of 2 rows x 3 columns, and make each column's width 10%, 50px and 1fr, respectively.



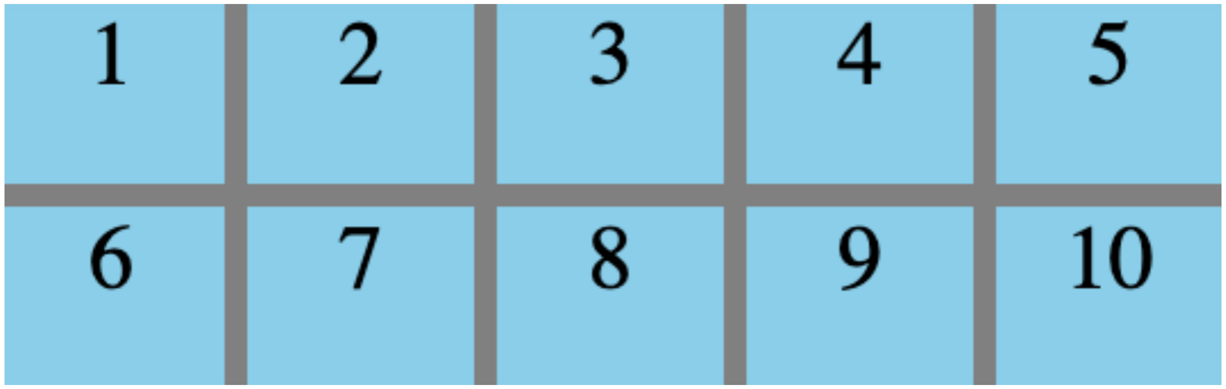
2 rows x 3 columns; 10%, 50px, 1fr | view on codepen

Quick try 3: Create a grid container of 3 rows x 3 columns, and make each column width and row height 3fr, 2fr and 1fr, respectively.



3 rows x 3 columns; 3fr, 2fr, 1fr | view on codepen

> *Problem 3: Create 2 rows x 5 columns using repeat function.*



2 rows x 5 columns; repeat function

You may want to create a grid container with all or some columns or rows of the same size (e.g. each 1fr) or repeated sizes (e.g. 1fr, 2fr, 1fr, 2fr). The solution to this problem is **repeat function**.

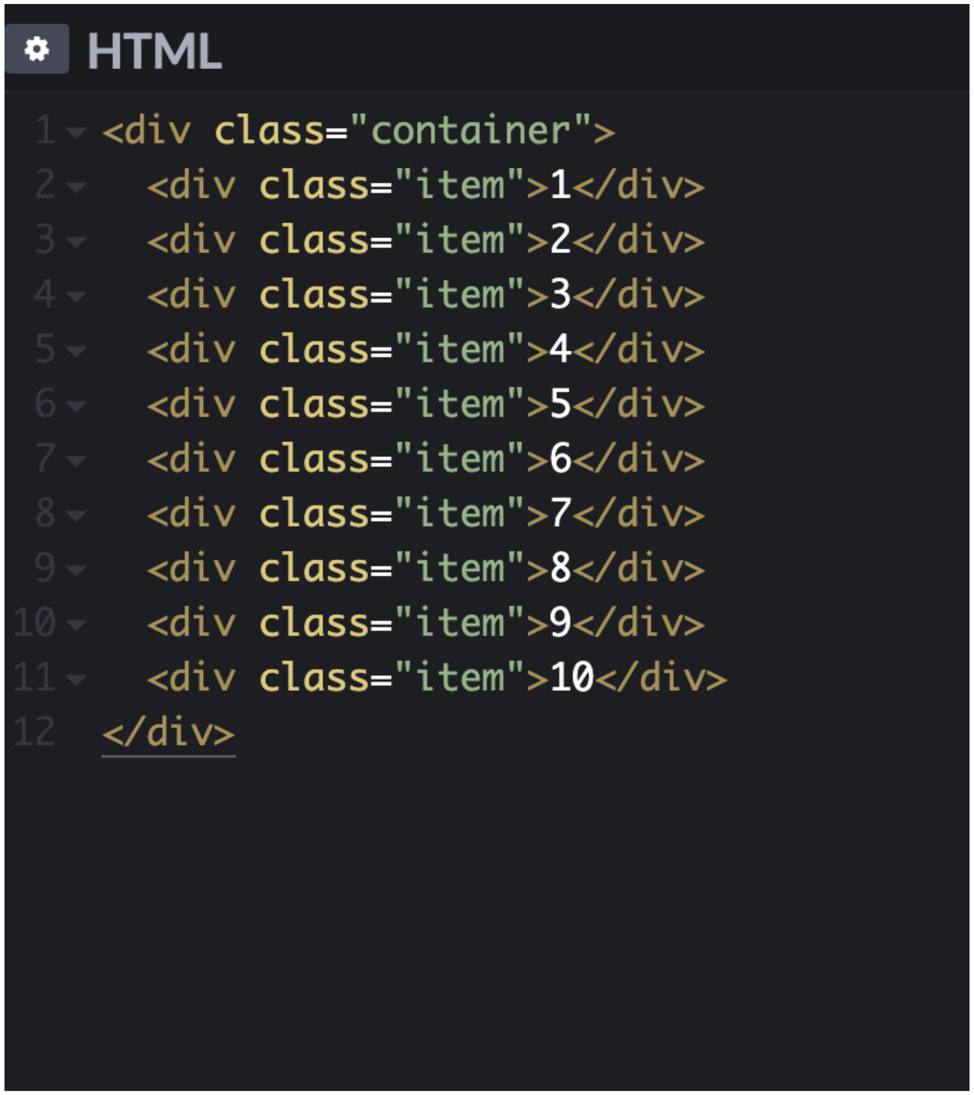
Repeat function syntax

```
repeat(number_of_times, space_separated_widths)
```

Reperat function syntax

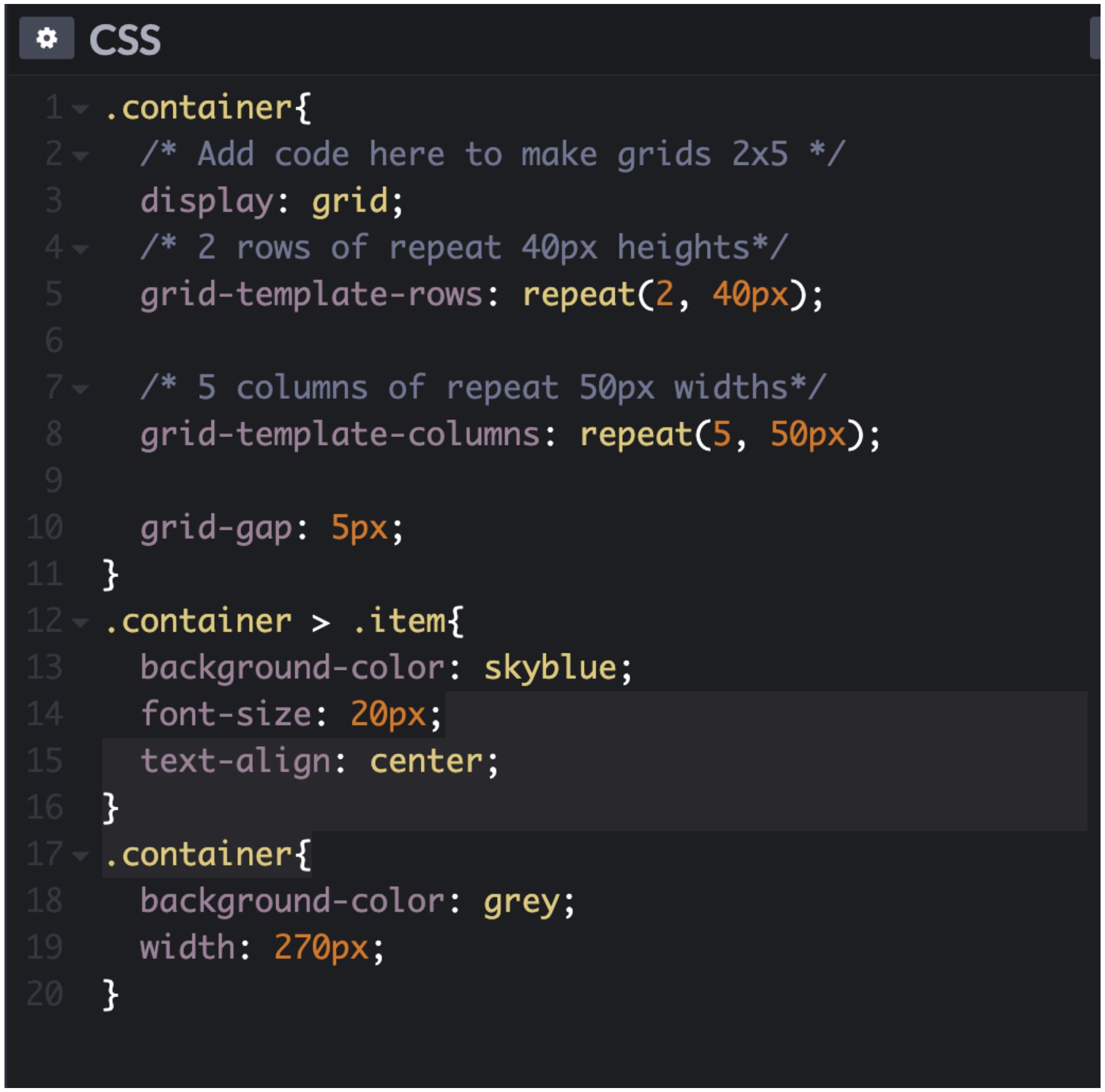
Let’s create five columns of 50px width using **repeat function** by updating the CSS code with the below:

```
grid-template-columns: repeat(5, 50px);
```



```html
<div class="container">
  <div class="item">1</div>
  <div class="item">2</div>
  <div class="item">3</div>
  <div class="item">4</div>
  <div class="item">5</div>
  <div class="item">6</div>
  <div class="item">7</div>
  <div class="item">8</div>
  <div class="item">9</div>
  <div class="item">10</div>
</div>
```

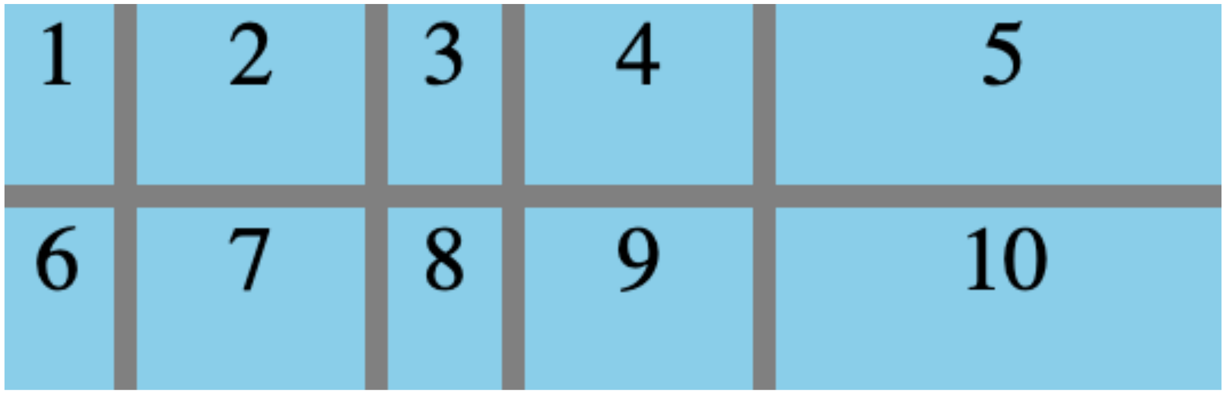
HTML Code 2 x 5 with repeat function


CSS

```css
.container{
  /* Add code here to make grids 2x5 */
  display: grid;
  /* 2 rows of repeat 40px heights*/
  grid-template-rows: repeat(2, 40px);

  /* 5 columns of repeat 50px widths*/
  grid-template-columns: repeat(5, 50px);

  grid-gap: 5px;
}
.container > .item{
  background-color: skyblue;
  font-size: 20px;
  text-align: center;
}
.container{
  background-color: grey;
  width: 270px;
}
```

Updated CSS code with repeat function

Quick try 4: Create a grid container of 5 columns with the following widths — 1fr, 50px, 1fr, 50px, and 100px.



2 rows x 5 columns; 1 fr, 50px, 1fr, 50px, 100px | view on codepen

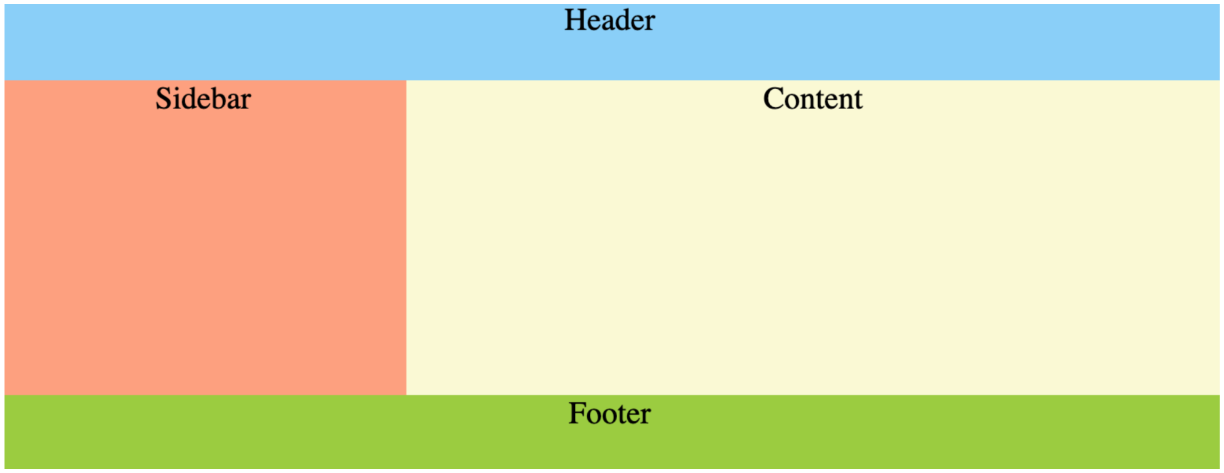
Hint: Update the above CSS code with the line given below.

```
grid-template-columns: repeat(2, 1fr 50px) 100px;
```

Until now, we have only discussed one item per grid. What if we have one item spanned over multiple grids?

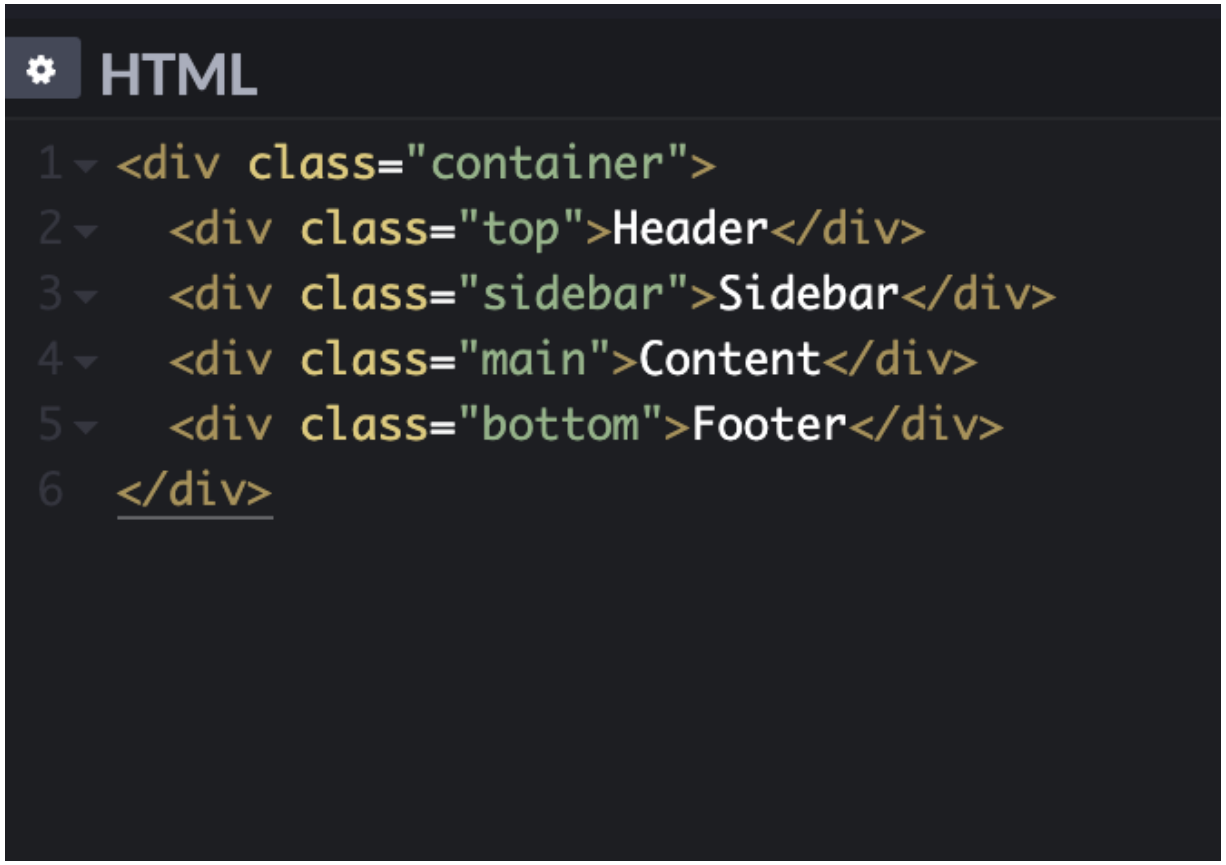
- Let’s explore this problem with a standard web page layout, which has four components: Header, Sidebar, Content, and Footer.

> *Problem 4: Create a standard web page layout on a 3x3 grid template.*



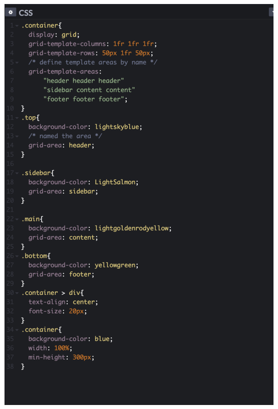
Standard Web Page: 3 x 3 grid

Code: Web page layout:



```html
<div class="container">
  <div class="top">Header</div>
  <div class="sidebar">Sidebar</div>
  <div class="main">Content</div>
  <div class="bottom">Footer</div>
</div>
```

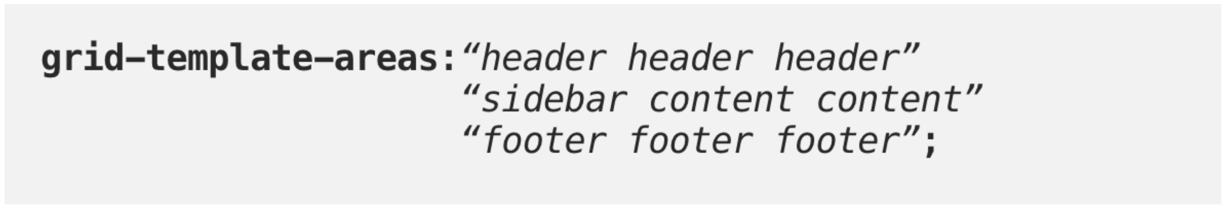
HTML Code Standrad 3 x 3 Web Page



```css
.container{
  display: grid;
  grid-template-columns: 1fr 1fr 1fr;
  grid-template-rows: 50px 1fr 50px;
  /* define template areas by name */
  grid-template-areas:
      "header header header"
      "sidebar content content"
      "footer footer footer";
}
.top{
  background-color: lightskyblue;
  /* named the area */
  grid-area: header;
}

.sidebar{
  background-color: LightSalmon;
  grid-area: sidebar;
}

.main{
  background-color: lightgoldenrodyellow;
  grid-area: content;
}
.bottom{
  background-color: yellowgreen;
  grid-area: footer;
}
.container > div{
  text-align: center;
  font-size: 20px;
}
.container{
  background-color: blue;
  width: 100%;
  min-height: 300px;
}
```

CSS Code Standard 3x3 Web Page

You can define a template in the container element by referencing the name of the component in each grid, as shown in the code snippet below. Each line with the quotes ("") represents a row, and each value between the quotes ("") represents a grid.



```
grid-template-areas:“header header header”
                    “sidebar content content”
                    “footer footer footer”;
```

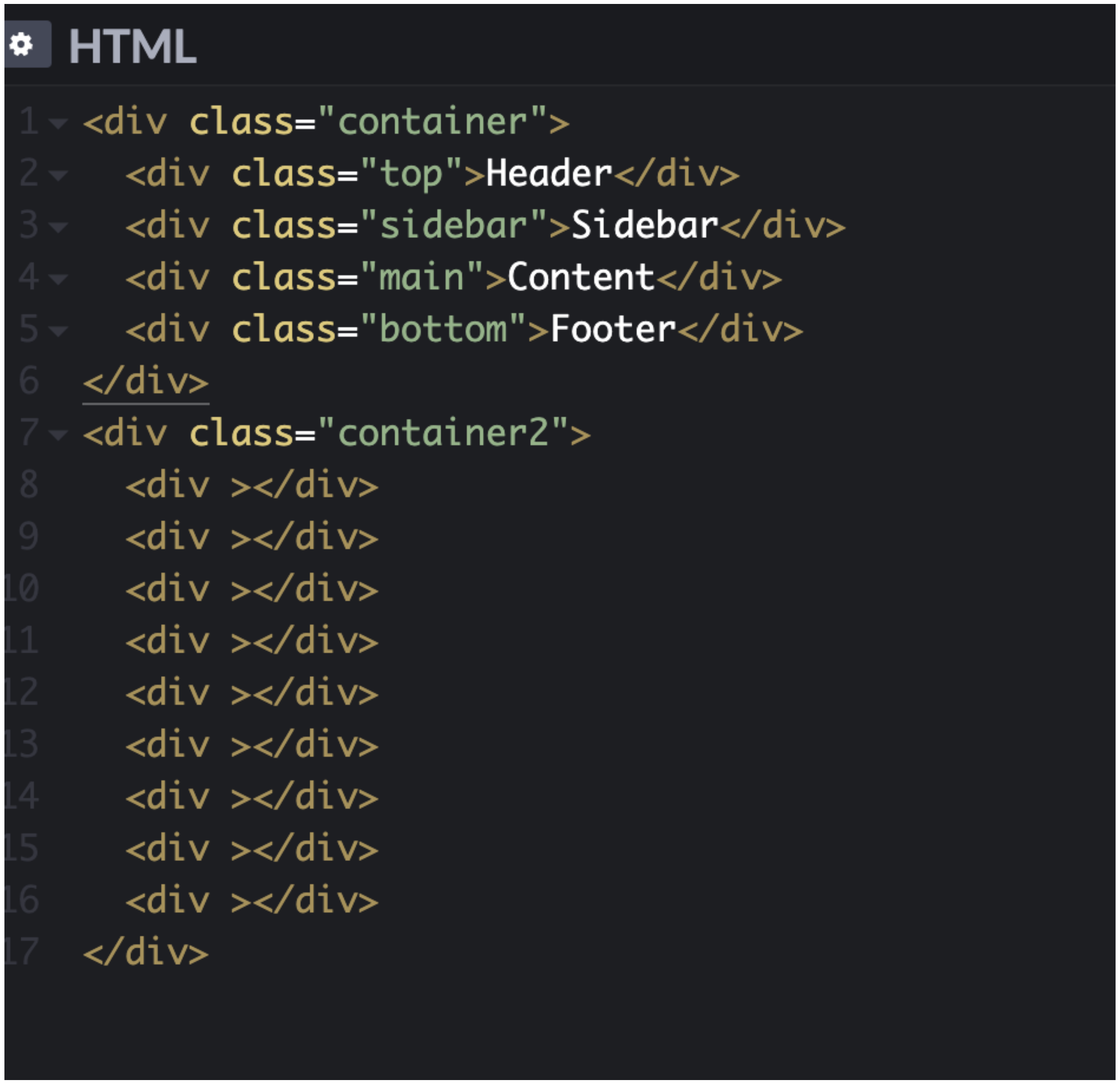
Define template in container element

So, the above code has nine grids, where all three grids of the first row are for the Header component. Similarly, all grids of the last row are for the Footer component. In the middle row, the first grid is for the Sidebar component, and the remaining two grids are for the Content component.

Please note that the names of the components are not CSS class names. You can name these components whatever you want.

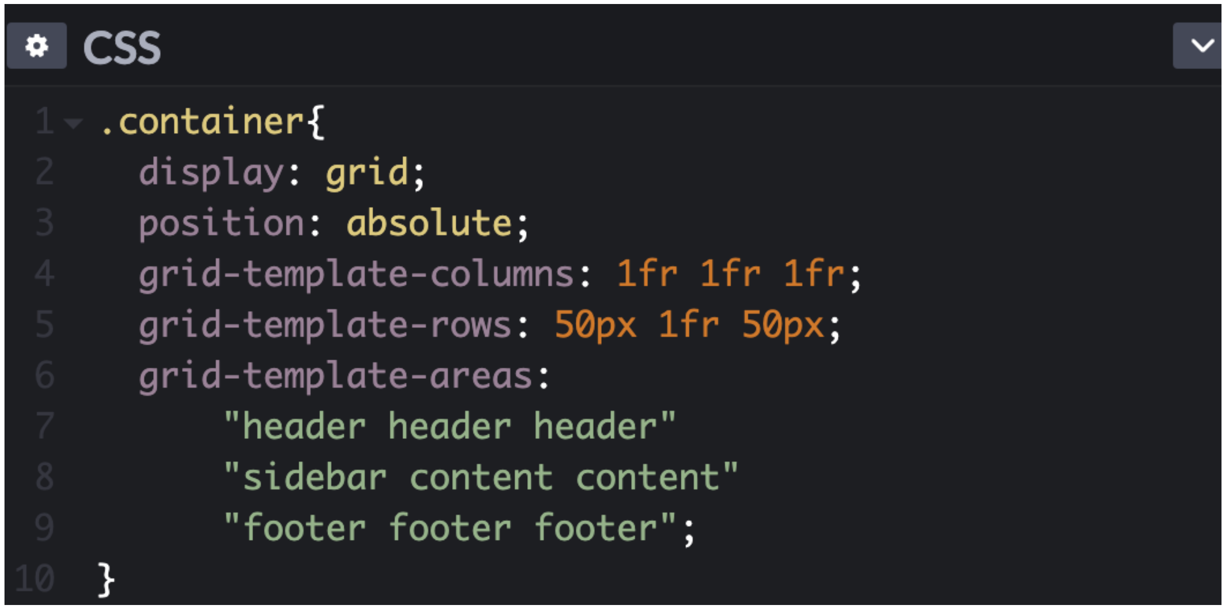
After defining grid-template-areas, you have to name the component with the below property.

```
grid-area: header;
```


HTML

```html
<div class="container">
  <div class="top">Header</div>
  <div class="sidebar">Sidebar</div>
  <div class="main">Content</div>
  <div class="bottom">Footer</div>
</div>
<div class="container2">
  <div ></div>
  <div ></div>
  <div ></div>
  <div ></div>
  <div ></div>
  <div ></div>
  <div ></div>
  <div ></div>
  <div ></div>
</div>
```

HTML: Components named



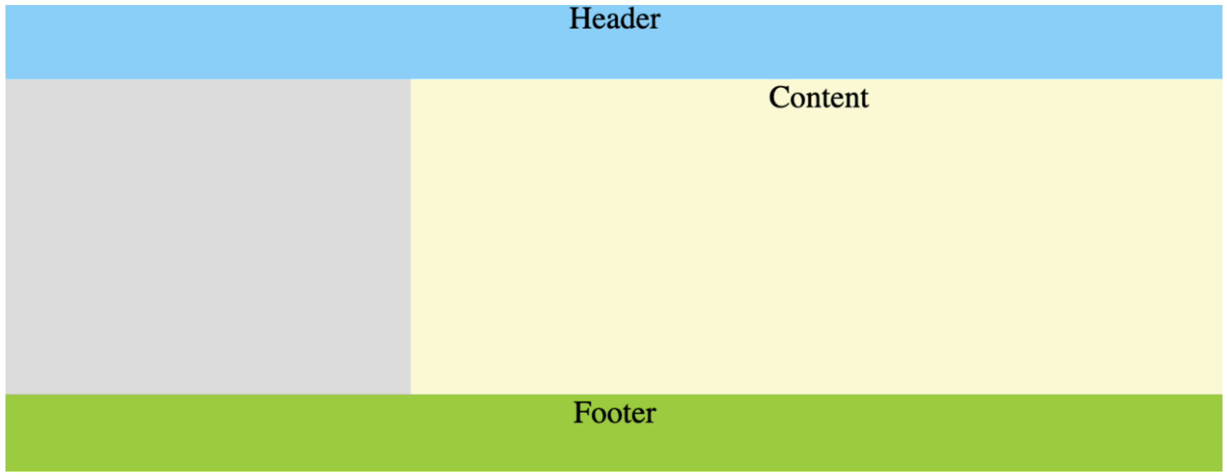
CSS

```css
.container{
  display: grid;
  position: absolute;
  grid-template-columns: 1fr 1fr 1fr;
  grid-template-rows: 50px 1fr 50px;
  grid-template-areas:
      "header header header"
      "sidebar content content"
      "footer footer footer";
}
```

CSS Code

Empty (.) area reference

In reference to the above problem, let's suppose you only want to have three components in your web page — Header, Content, and Footer — and want to remove Sidebar, as in the image below. Then, empty (.) area reference is coming to rescue.



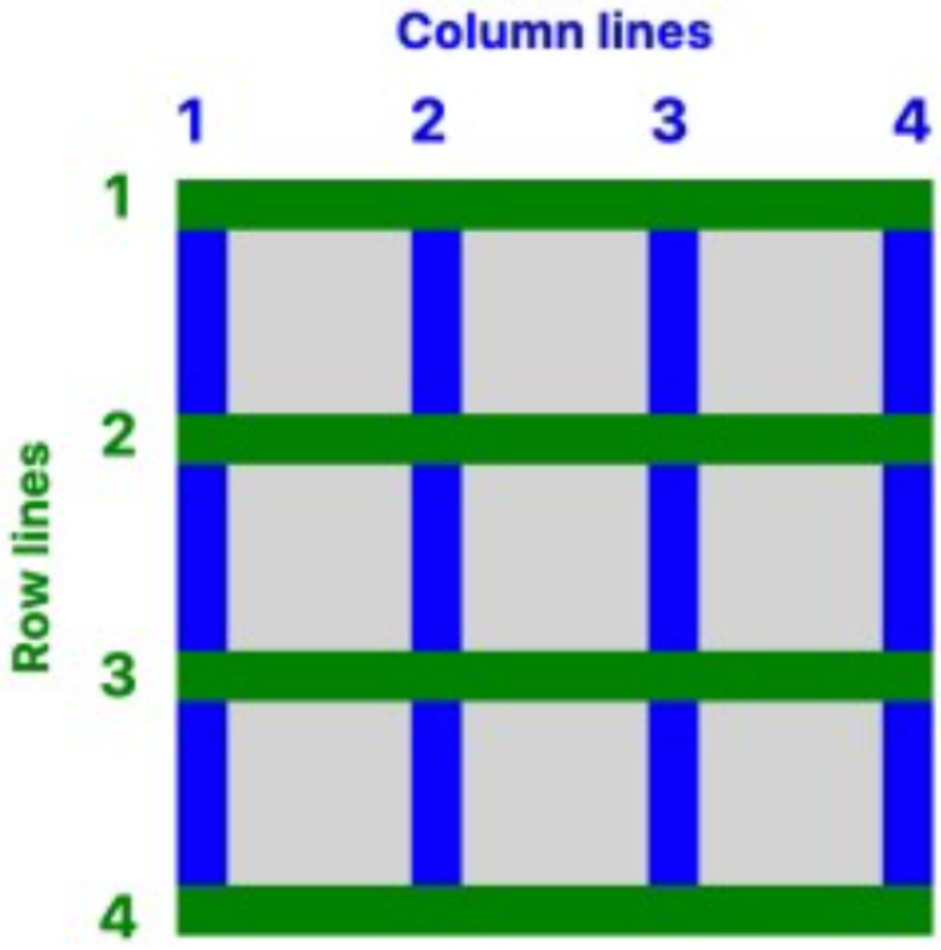
Empty Area

Update the CSS code for grid-template-areas of Problem 4 with the code snippet below and remove <div class=”sidebar”>Sidebar</div> from HTML code.

```
grid-template-areas:“header header header”
                    “.     content content”
                    “footer footer footer”;
```

Application of empty area CSS code

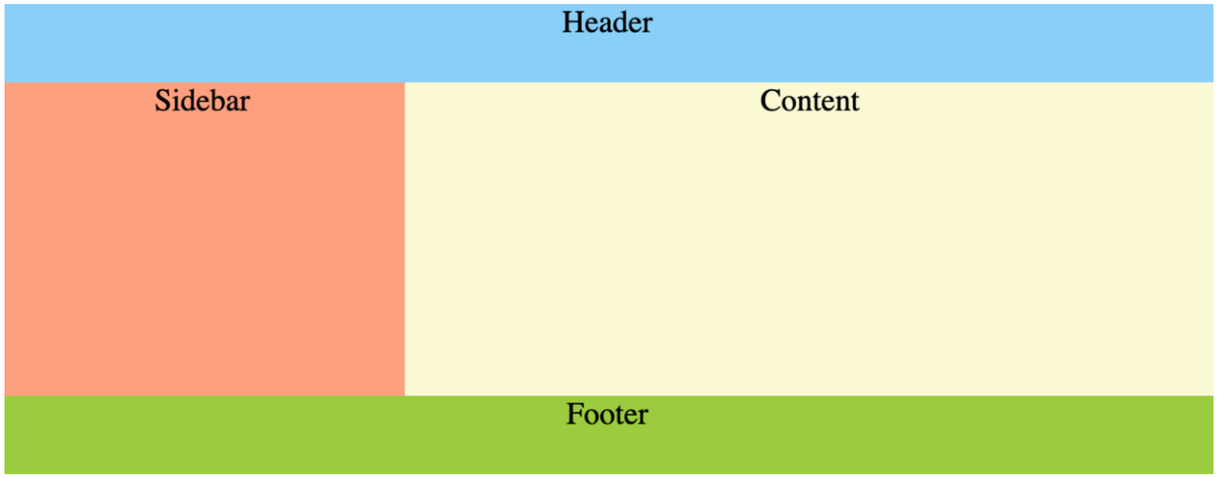
Lines in GRID



Lines in a Grid

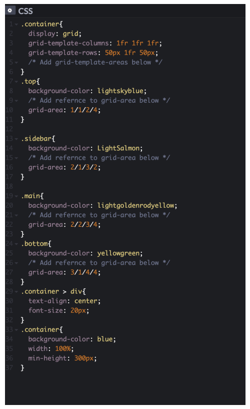
- Hypothetical horizontal and vertical lines that create the grid are referred to as lines.
- Lines start with count 1 for the top and left most lines.
- These lines can be used to refer to an area in the grid properties.

> *Problem 5: Let’s try the Problem 4 layout again, but this time using grid lines instead of naming the areas.*



Here, we are going to define grid areas without naming them in the container, or in other words, we are not going to use the grid-template-areas property in the container.

- So, instead of this, the grid-areas property with grid lines can be used to tell the component which grids it can acquire.



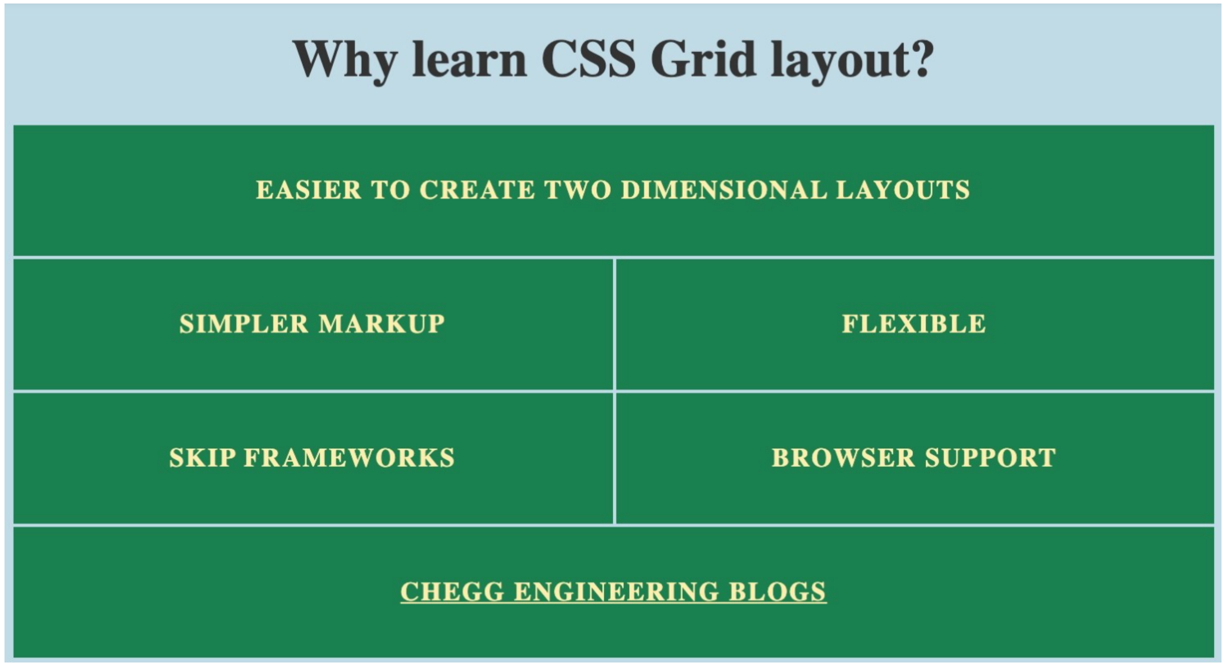
```css
.container{
  display: grid;
  grid-template-columns: 1fr 1fr 1fr;
  grid-template-rows: 50px 1fr 50px;
  /* Add grid-template-areas below */
}
.top{
  background-color: lightskyblue;
  /* Add refernce to grid-area below */
  grid-area: 1/1/2/4;
}

.sidebar{
  background-color: LightSalmon;
  /* Add refernce to grid-area below */
  grid-area: 2/1/3/2;
}

.main{
  background-color: lightgoldenrodyellow;
  /* Add refernce to grid-area below */
  grid-area: 2/2/3/4;
}
.bottom{
  background-color: yellowgreen;
  /* Add refernce to grid-area below */
  grid-area: 3/1/4/4;
}
.container > div{
  text-align: center;
  font-size: 20px;
}
.container{
  background-color: blue;
  width: 100%;
  min-height: 300px;
}
```

CSS Code using grid lines

Layout Project

As promised in the beginning, this is your hands-on project. Are you ready to give it a try? Recreate the layout below.

- **Share your completed Pen in our Cohort Slack.**


Why learn CSS Grid layout?

EASIER TO CREATE TWO DIMENSIONAL LAYOUTS

SIMPLER MARKUP

FLEXIBLE

SKIP FRAMEWORKS

BROWSER SUPPORT

CHEGG ENGINEERING BLOGS

CSS GRID: Layout Project

CREATED BY
Vicki Bealman