**CSS – Grid Layout**

เป็นมาตรฐานที่สร้างขึ้นมาเพื่อจัดการเลย์เอาท์สองมิติ มีไว้สำหรับกำหนดการแสดงผล หรือจัด layout ในรูปแบบสองมิติ  
และช่วยให้ผู้เขียนจัดองค์ประกอบให้เป็นคอลัมน์และแถวได้เช่นเดียวกับตาราง ด้วยการแบ่งพื้นที่ออกเป็นส่วน ๆ ซึ่งจะประกอบไปด้วย parent หรือเรียกกันว่า grid container และ children หรือที่เรียกอีกอย่างหนึ่งว่า grid item

A screenshot of a grid

Description automatically generated

1. สิ่งที่อยู่ใน Grid Container คือ Grid Item
2. เส้นแต่ละเส้นตาม แนวนอน คือ Grid Line Row
3. เส้นแต่ละเส้นตาม แนวตั้ง คือ Grid Line Column
4. Grid Line Row สองเส้นที่ตัดกันจนเกิดพื้นที่ คือ Grid Track Row หรือ Grid Row
5. Grid Line Column สองเส้นที่ตัดกันจนเกิดพื้นที่ คือ Grid Track Column หรือ Grid Column
6. พื้นที่ที่เกิดจากการตัดกันของ Grid Row และ Grid Column คือ Grid Cell
7. พื้นที่ที่เกิดจากการรวมของ Grid Cell ใกล้ ๆ คือ Grid Area
8. ช่องว่างภายนอกระหว่าง Grid Cell คือ Grid Gap

<!DOCTYPE html>

<html>

<head>

    <style>

        .grid-container {

            display: grid; /\* display property is set to grid or inline-grid. \*/

            grid-template-columns: auto auto auto;

            background-color: #2196F3;

            padding: 10px;

        }

        .grid-item {

            background-color: rgba(255, 255, 255, 0.8);

            border: 1px solid rgba(0, 0, 0, 0.8);

            padding: 20px;

            font-size: 30px;

            text-align: center;

        }

    </style>

</head>

<body>

    <h1>Grid Elements</h1>

    <div class="grid-container">

        <div class="grid-item">1</div>

        <div class="grid-item">2</div>

        <div class="grid-item">3</div>

        <div class="grid-item">4</div>

        <div class="grid-item">5</div>

        <div class="grid-item">6</div>

        <div class="grid-item">7</div>

        <div class="grid-item">8</div>

        <div class="grid-item">9</div>

    </div>

</body>

</html>

A screenshot of a computer

Description automatically generated

**grid-template-columns**

จะเป็นการกำหนดว่าตารางจะมีกี่ column และแต่ละ column ความกว้างเท่าไหร่

ซึ่งจำนวนของ column จะตามจำนวนค่าของความกว้างที่ใส่เข้าไป

ใส่ 1 ค่าก็จะมี 1 column : ใส่ 10 ค่า ก็จะมี 10 column

โดยความกว้างจะใช้หน่วยวัดขนาดใน CSS ได้ตามปกติ และสามารถใส่ค่า auto เพื่อให้ความกว้างของ Column เฉลี่ยกันเพื่อเติมช่องว่างที่เหลืออยู่ให้เต็มได้

ทดลอง ปรับ css grid ให้เป็น 5 Column

        .grid-container {

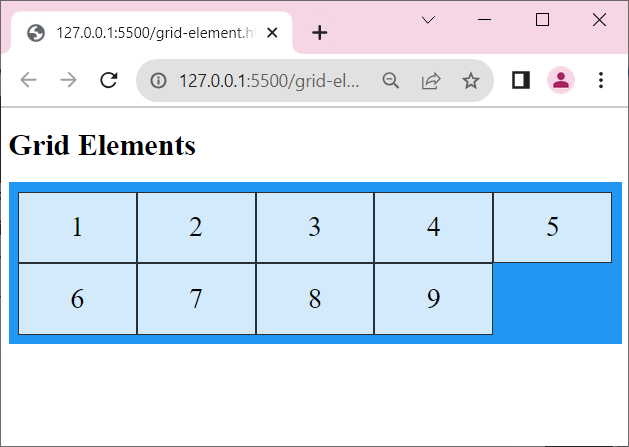
            display: grid;

            grid-template-columns: auto auto auto auto auto;

            background-color: #2196F3;

            padding: 10px;

        }



**grid-template-rows**

คล้ายกับ grid-template-columns แต่จะเป็นการกำหนดในส่วนของ row แทน

.grid-container {

            display: grid;

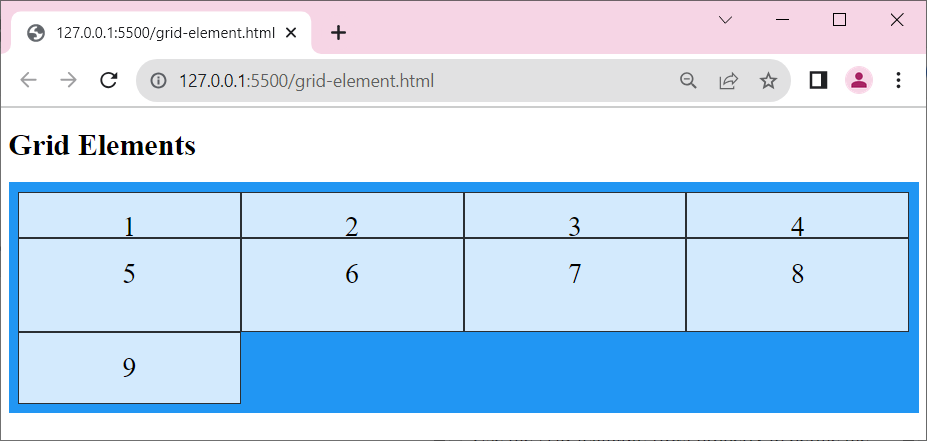
            grid-template-columns: auto auto auto auto;

            grid-template-rows: 50px 100px;

            background-color: #2196F3;

            padding: 10px;

      }



**Grid Gaps**

A screenshot of a grid

Description automatically generated

* column-gap
* row-gap
* gap

.grid-container {

            display: grid;

            grid-template-columns: auto auto  auto auto;

            grid-template-rows: 50px 100px;

            column-gap: 50px;

            row-gap: 50px;

            background-color: #2196F3;

            padding: 10px;

        }

A screenshot of a computer

Description automatically generated

.grid-container {

            display: grid;

            grid-template-columns: auto auto  auto auto;

            grid-template-rows: 50px 100px;

            /\* column-gap: 50px;

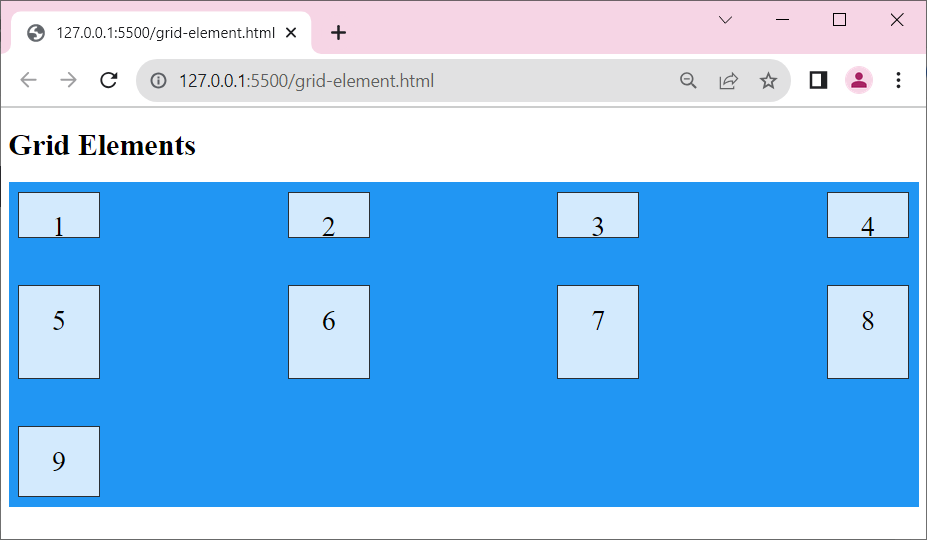
            row-gap: 50px; \*/

            gap: 50px 200px; /\* gap: 50px \*/

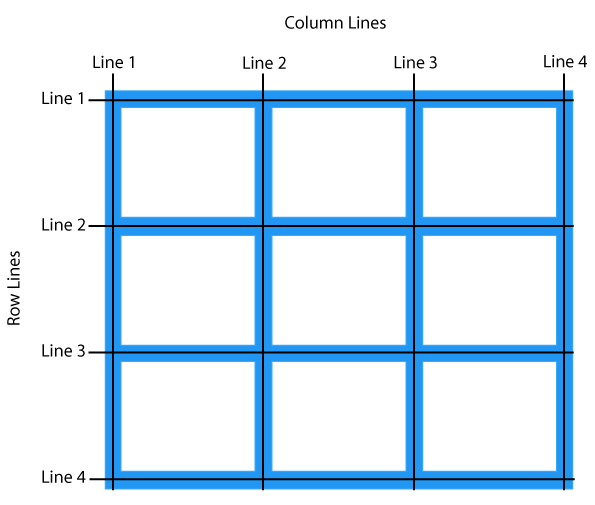
            background-color: #2196F3;

            padding: 10px;

        }



**Grid Lines**



***css***

.item1 {

            grid-column-start: 1;

            grid-column-end: 4;

      }

***html***

<div class="grid-item item1">1</div>

A screenshot of a computer

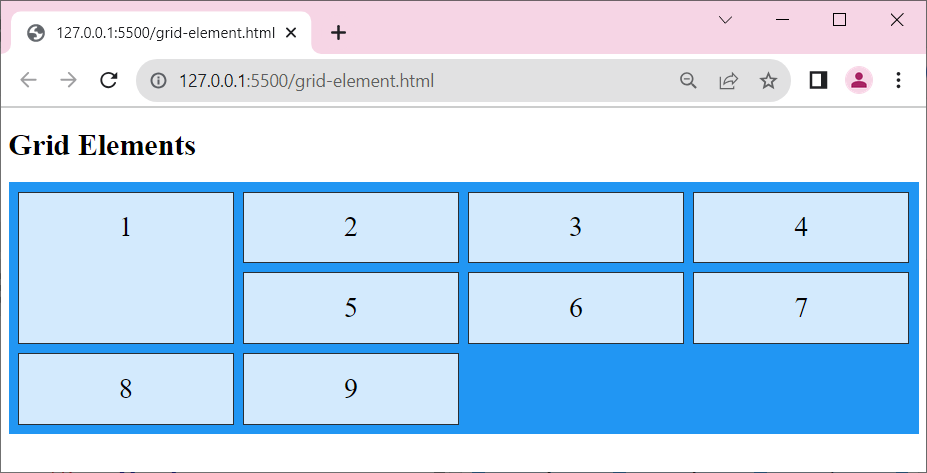
Description automatically generated

.item1 {

            grid-row-start: 1;

            grid-row-end: 3;

      }



All CSS Grid Properties

|  |  |
| --- | --- |
| **Property** | **Description** |
| [column-gap](https://www.w3schools.com/cssref/css3_pr_column-gap.asp) | Specifies the gap between the columns |
| [gap](https://www.w3schools.com/cssref/css3_pr_gap.asp) | A shorthand property for the *row-gap* and the *column-gap* properties |
| [grid](https://www.w3schools.com/cssref/pr_grid.asp) | A shorthand property for the *grid-template-rows, grid-template-columns, grid-template-areas, grid-auto-rows, grid-auto-columns*, and the *grid-auto-flow* properties |
| [grid-area](https://www.w3schools.com/cssref/pr_grid-area.asp) | Either specifies a name for the grid item, or this property is a shorthand property for the *grid-row-start*, *grid-column-start*, *grid-row-end*, and *grid-column-end* properties |
| [grid-auto-columns](https://www.w3schools.com/cssref/pr_grid-auto-columns.asp) | Specifies a default column size |
| [grid-auto-flow](https://www.w3schools.com/cssref/pr_grid-auto-flow.asp) | Specifies how auto-placed items are inserted in the grid |
| [grid-auto-rows](https://www.w3schools.com/cssref/pr_grid-auto-rows.asp) | Specifies a default row size |
| [grid-column](https://www.w3schools.com/cssref/pr_grid-column.asp) | A shorthand property for the *grid-column-start* and the *grid-column-end* properties |
| [grid-column-end](https://www.w3schools.com/cssref/pr_grid-column-end.asp) | Specifies where to end the grid item |
| [grid-column-gap](https://www.w3schools.com/cssref/pr_grid-column-gap.asp) | Specifies the size of the gap between columns |
| [grid-column-start](https://www.w3schools.com/cssref/pr_grid-column-start.asp) | Specifies where to start the grid item |
| [grid-gap](https://www.w3schools.com/cssref/pr_grid-gap.asp) | A shorthand property for the *grid-row-gap* and *grid-column-gap* properties |
| [grid-row](https://www.w3schools.com/cssref/pr_grid-row.asp) | A shorthand property for the *grid-row-start* and the *grid-row-end* properties |
| [grid-row-end](https://www.w3schools.com/cssref/pr_grid-row-end.asp) | Specifies where to end the grid item |
| [grid-row-gap](https://www.w3schools.com/cssref/pr_grid-row-gap.asp) | Specifies the size of the gap between rows |
| [grid-row-start](https://www.w3schools.com/cssref/pr_grid-row-start.asp) | Specifies where to start the grid item |
| [grid-template](https://www.w3schools.com/cssref/pr_grid-template.asp) | A shorthand property for the *grid-template-rows*, *grid-template-columns* and *grid-areas* properties |
| [grid-template-areas](https://www.w3schools.com/cssref/pr_grid-template-areas.asp) | Specifies how to display columns and rows, using named grid items |
| [grid-template-columns](https://www.w3schools.com/cssref/pr_grid-template-columns.asp) | Specifies the size of the columns, and how many columns in a grid layout |
| [grid-template-rows](https://www.w3schools.com/cssref/pr_grid-template-rows.asp) | Specifies the size of the rows in a grid layout |
| [row-gap](https://www.w3schools.com/cssref/css3_pr_row-gap.asp) | Specifies the gap between the grid rows |

## grid-template-columns Property

CSS

.grid-container {

  display: grid;

  grid-template-columns: auto auto auto auto;

  gap: 10px;

  background-color: #2196F3;

  padding: 10px;

}

.grid-container > div {

  background-color: rgba(255, 255, 255, 0.8);

  text-align: center;

  padding: 20px 0;

  font-size: 30px;

}

**HTML**

<div class="grid-container">

  <div>1</div>

  <div>2</div>

  <div>3</div>

  <div>4</div>

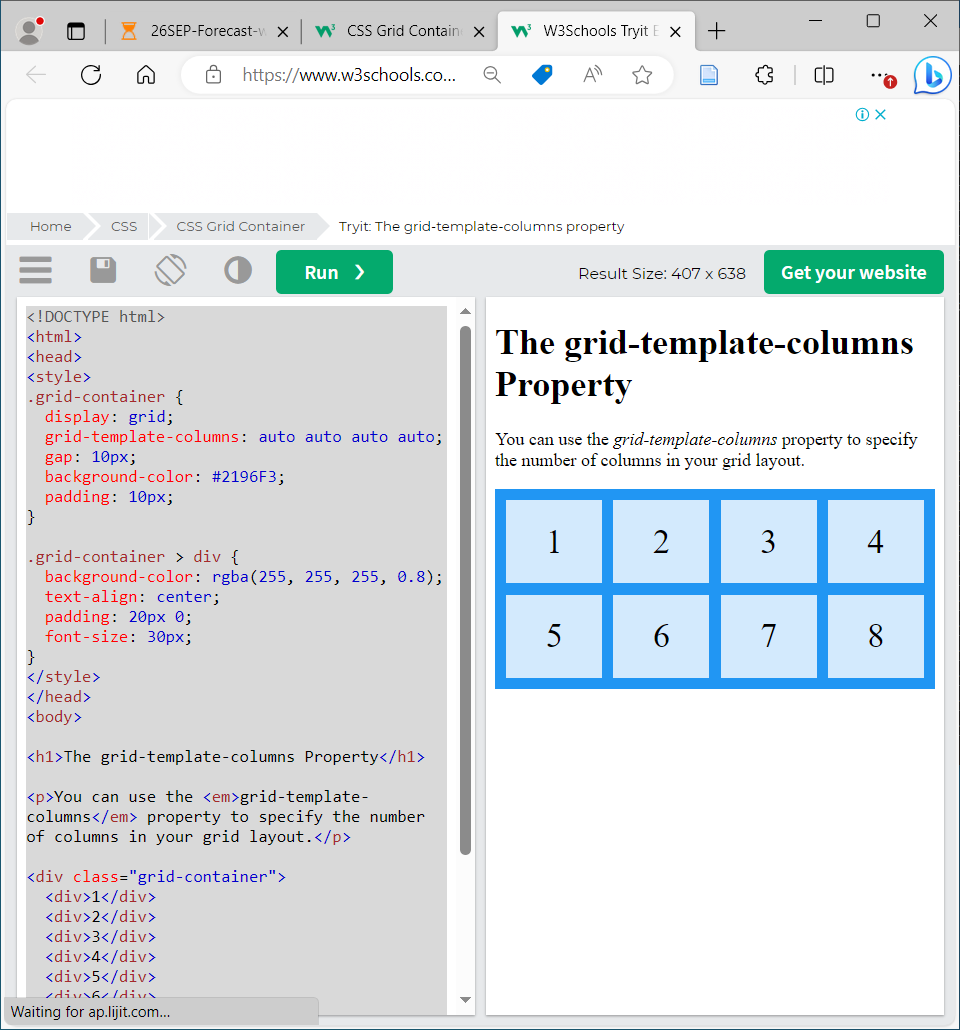
  <div>5</div>

  <div>6</div>

  <div>7</div>

  <div>8</div>

</div>



## grid-template-rows Property

**CSS**

.grid-container {

  display: grid;

  grid-template-columns: auto auto auto;

  grid-template-rows: 80px 200px;

  gap: 10px;

  background-color: #2196F3;

  padding: 10px;

}

.grid-container > div {

  background-color: rgba(255, 255, 255, 0.8);

  text-align: center;

  padding: 20px 0;

  font-size: 30px;

}

**HTML**

<div class="grid-container">

  <div>1</div>

  <div>2</div>

  <div>3</div>

  <div>4</div>

  <div>5</div>

  <div>6</div>

</div>

A screenshot of a computer

Description automatically generated

## justify-content Property

.grid-container {  
  display: grid;  
  justify-content: space-evenly;  
 justify-content: space-around;  
 justify-content: space-between;  
 justify-content: center;  
 justify-content: start;  
 justify-content: end;  
}

A screenshot of a computer

Description automatically generated

## align-content Property

.grid-container {  
  display: grid;  
  height: 400px;  
  align-content: center;  
 align-content: space-evenly;  
 align-content: space-around;  
 align-content: space-between;  
 align-content: start;  
 align-content: end;

}

A screenshot of a computer

Description automatically generated

## grid-column Property

**CSS**

.grid-container {

  display: grid;

  grid-template-columns: auto auto auto auto auto auto;

  gap: 10px;

  background-color: #2196F3;

  padding: 10px;

}

.grid-container > div {

  background-color: rgba(255, 255, 255, 0.8);

  text-align: center;

  padding: 20px 0;

Try again to modify CSS

.item1 {  
  grid-column: 1 / span 3;  
}

.item2 {  
  grid-column: 2 / span 3;  
}

  font-size: 30px;

}

.item1 {

  grid-column: 1 / 5;

}

**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 class="item5">5</div>

  <div class="item6">6</div>

  <div class="item7">7</div>

  <div class="item8">8</div>

  <div class="item9">9</div>

  <div class="item10">10</div>

  <div class="item11">11</div>

  <div class="item12">12</div>

  <div class="item13">13</div>

  <div class="item14">14</div>

  <div class="item15">15</div>

</div>

A screenshot of a computer

Description automatically generated

Item1 will start on column 1 and end before column 5

## grid-row Property

.grid-container {

  display: grid;

  grid-template-columns: auto auto auto auto auto auto;

  gap: 10px;

  background-color: #2196F3;

  padding: 10px;

}

.grid-container > div {

  background-color: rgba(255, 255, 255, 0.8);

  text-align: center;

  padding: 20px 0;

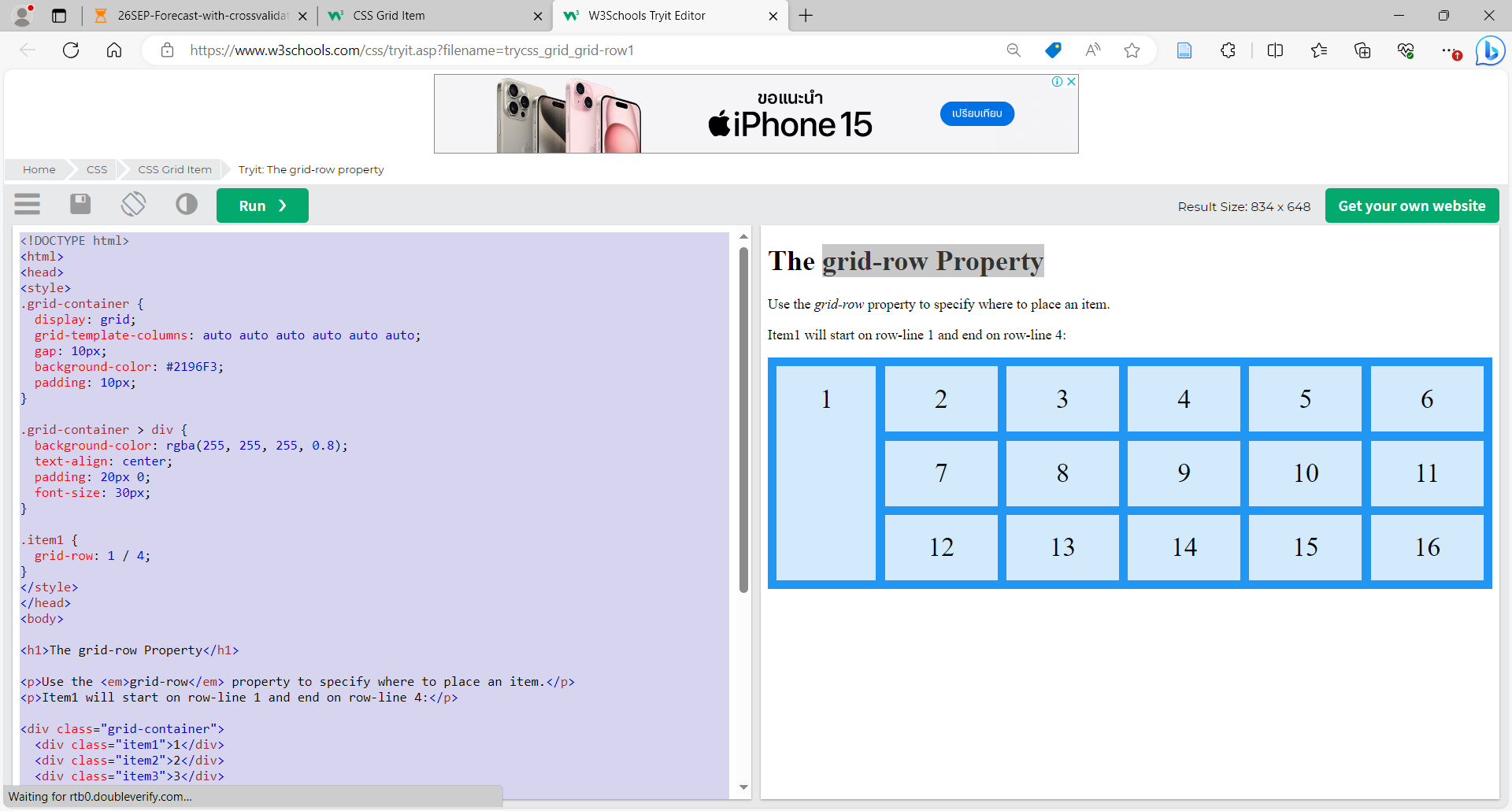
  font-size: 30px;

}

.item1 {

  grid-row: 1 / 4;

}



Try again to modify CSS

.item1 {  
  grid-row: 1 / span 2;  
}

grid-area Property

***CSS***

.grid-container {

            display: grid;

            grid-template-columns: auto auto auto auto auto auto;

            gap: 10px;

            background-color: #2196F3;

            padding: 10px;

        }

        .grid-container > div {

            background-color: rgba(255, 255, 255, 0.8);

            text-align: center;

            padding: 20px 0;

Try again to modify CSS

.item8 {  
  grid-area: 2 / 1 / span 2 / span 3;  
}

            font-size: 30px;

        }

        .item8 {

            grid-area: 1 / 2 / 5 / 6;

        }

***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 class="item5">5</div>

        <div class="item6">6</div>

        <div class="item7">7</div>

        <div class="item8">8</div>

        <div class="item9">9</div>

        <div class="item10">10</div>

        <div class="item11">11</div>

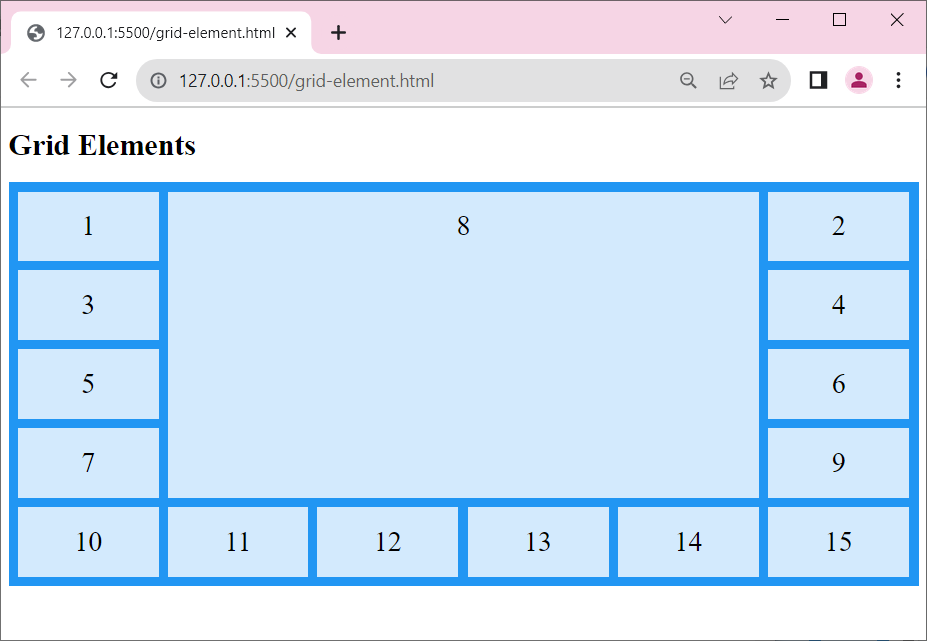
        <div class="item12">12</div>

        <div class="item13">13</div>

        <div class="item14">14</div>

        <div class="item15">15</div>

    </div>



## Naming Grid Items

**CSS**

.item1 {

  grid-area: myArea;

}

.grid-container {

  display: grid;

  grid-template-areas: 'myArea myArea myArea myArea myArea';

  gap: 10px;

  background-color: #2196F3;

  padding: 10px;

}

.grid-container > div {

  background-color: rgba(255, 255, 255, 0.8);

  text-align: center;

  padding: 20px 0;

  font-size: 30px;

}

**HTML**

<div class="grid-container">

  <div class="item1">1</div>

  <div class="item2">2</div>

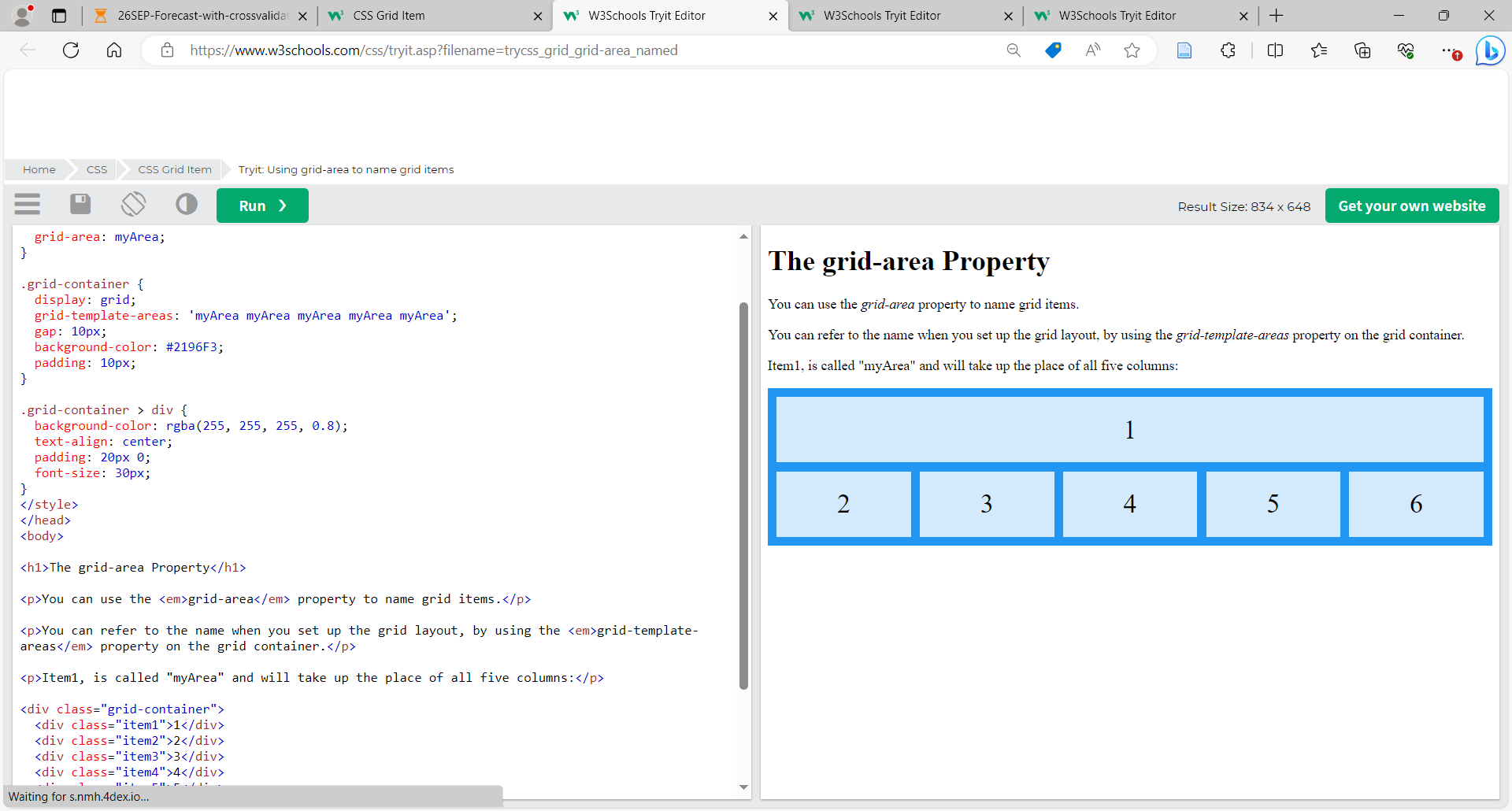
  <div class="item3">3</div>

  <div class="item4">4</div>

  <div class="item5">5</div>

  <div class="item6">6</div>

</div>



Try again to modify CSS

.item1 {  
  grid-area: myArea;  
}  
.grid-container {  
  grid-template-areas: 'myArea myArea . . .';  
}

And change for make item11 span two columns and two rows.

.grid-container {  
  grid-template-areas: 'myArea myArea . . .' 'myArea myArea . . .';  
}

## grid-area Property

<!DOCTYPE html>

<html>

<head>

<style>

    .item1 { grid-area: header; }

    .item2 { grid-area: menu; }

    .item3 { grid-area: main; }

    .item4 { grid-area: right; }

    .item5 { grid-area: footer; }

    .grid-container {

        display: grid;

        grid-template-areas:

            'header header header header header header'

            'menu main main main right right'

            'menu footer footer footer footer footer';

        gap: 10px;

        background-color: #2196F3;

        padding: 10px;

    }

    .grid-container > div {

        background-color: rgba(255, 255, 255, 0.8);

        text-align: center;

        padding: 20px 0;

        font-size: 30px;

    }

    </style>

</head>

<body>

    <h1>The grid-area Property</h1>

    <div class="grid-container">

        <div class="item1">Header</div>

        <div class="item2">Menu</div>

        <div class="item3">Main</div>

        <div class="item4">Right</div>

        <div class="item5">Footer</div>

    </div>

</body>

</html>

A screenshot of a computer

Description automatically generated

## Order of the Items

**CSS**

.grid-container {

  display: grid;

  grid-template-columns: auto auto auto;

  gap: 10px;

  background-color: #2196F3;

  padding: 10px;

}

.grid-container > div {

  background-color: rgba(255, 255, 255, 0.8);

  text-align: center;

  padding: 20px 0;

  font-size: 30px;

}

/\* grid-row-start / grid-column-start / grid-row-end / grid-column-end.\*/

.item1 { grid-area: 1 / 3 / 2 / 4; }

.item2 { grid-area: 2 / 3 / 3 / 4; }

.item3 { grid-area: 1 / 1 / 2 / 2; }

.item4 { grid-area: 1 / 2 / 2 / 3; }

.item5 { grid-area: 2 / 1 / 3 / 2; }

.item6 { grid-area: 2 / 2 / 3 / 3; }

**HTML**

<h1>Sort the Items</h1>

<div class="grid-container">

  <div class="item1">1</div>

  <div class="item2">2</div>

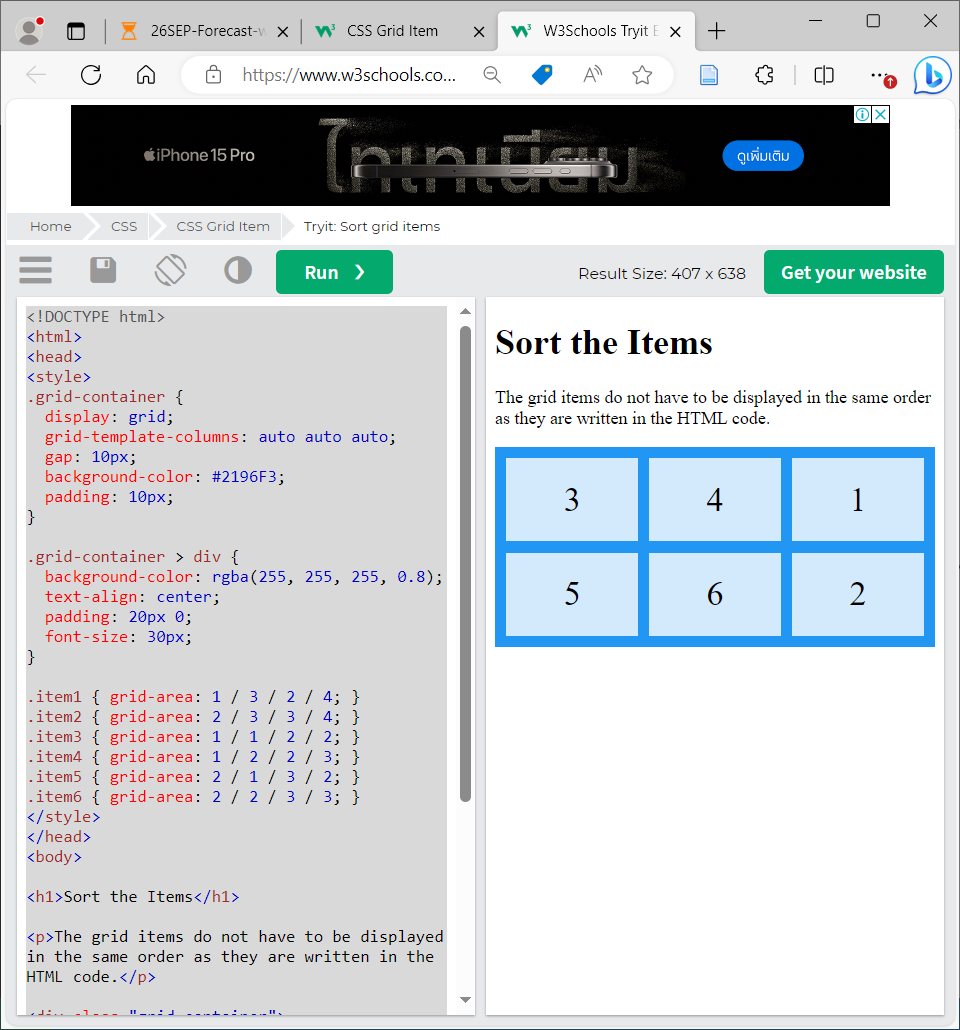
  <div class="item3">3</div>

  <div class="item4">4</div>

  <div class="item5">5</div>

  <div class="item6">6</div>

</div>



# **Re-arrange the Order on Small Devices**

**And Resize the window to 500 pixels see the effect.**

.grid-container {

  display: grid;

  grid-template-columns: auto auto auto;

  gap: 10px;

  background-color: #2196F3;

  padding: 10px;

}

.grid-container > div {

  background-color: rgba(255, 255, 255, 0.8);

  text-align: center;

  padding: 20px 0;

  font-size: 30px;

}

@media only screen and (max-width: 500px) {

  .item1 { grid-area: 1 / span 3 / 2 / 4; }

  .item2 { grid-area: 3 / 3 / 4 / 4; }

  .item3 { grid-area: 2 / 1 / 3 / 2; }

  .item4 { grid-area: 2 / 2 / span 2 / 3; }

  .item5 { grid-area: 3 / 1 / 4 / 2; }

  .item6 { grid-area: 2 / 3 / 3 / 4; }

}

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

## CSS Layout Templates

**equal** columns

**HTML**

<h2>CSS Template using Float</h2>

<div class="header">

  <h2>Header</h2>

</div>

<div class="row">

  <div class="column" style="background-color:#aaa;">Column</div>

  <div class="column" style="background-color:#bbb;">Column</div>

  <div class="column" style="background-color:#ccc;">Column</div>

</div>

<div class="footer">

  <p>Footer</p>

## A screenshot of a computer Description automatically generated

## 3 way to create CSS Template

## Float

## Flexbox

## Grid

## A screenshot of a computer Description automatically generated

## using Float

\* {

  box-sizing: border-box;

}

body {

  font-family: Arial, Helvetica, sans-serif;

}

/\* Style the header \*/

.header {

  background-color: #f1f1f1;

  padding: 30px;

  text-align: center;

  font-size: 35px;

}

/\* Create three equal columns that floats next to each other \*/

.column {

  float: left;

  width: 33.33%;

  padding: 10px;

  height: 300px; /\* Should be removed. Only for demonstration \*/

}

/\* Clear floats after the columns \*/

.row:after {

  content: "";

  display: table;

  clear: both;

}

/\* Style the footer \*/

.footer {

  background-color: #f1f1f1;

  padding: 10px;

  text-align: center;

}

/\* Responsive layout - makes the three columns stack on top of each other instead of next to each other \*/

@media (max-width: 600px) {

  .column {

    width: 100%;

  }

}

## using Flexbox

\* {

  box-sizing: border-box;

}

body {

  font-family: Arial, Helvetica, sans-serif;

}

/\* Style the header \*/

.header {

    background-color: #e68415;

  padding: 30px;

  text-align: center;

  font-size: 35px;

}

/\* Container for flexboxes \*/

.row {

  display: -webkit-flex;

  display: flex;

}

/\* Create three equal columns that sits next to each other \*/

.column {

  -webkit-flex: 1;

  -ms-flex: 1;

  flex: 1;

  padding: 10px;

  height: 300px; /\* Should be removed. Only for demonstration \*/

}

/\* Style the footer \*/

.footer {

    background-color: #e69620;

  padding: 10px;

  text-align: center;

}

/\* Responsive layout - makes the three columns stack on top of each other instead of next to each other \*/

@media (max-width: 600px) {

  .row {

    -webkit-flex-direction: column;

    flex-direction: column;

  }

}

## using Grid

**HTML**

<div class="grid-container">

  <div class="header">

    <h2>Header</h2>

  </div>

  <div class="left" style="background-color:#757070;">Column</div>

  <div class="middle" style="background-color:#d0f110;">Column</div>

  <div class="right" style="background-color:#117fda;">Column</div>

  <div class="footer">

  <p>Footer</p>

  </div>

</div>

**CSS**

\* {

  box-sizing: border-box;

}

body {

  font-family: Arial, Helvetica, sans-serif;

}

/\* Style the header \*/

.header {

  grid-area: header;  background-color: #f1f1f1;  padding: 30px;  text-align: center;

  font-size: 35px;

}

/\* The grid container \*/

.grid-container {

  display: grid;

  grid-template-areas:

    'header header header header header header'

    'left left middle middle right right'

    'footer footer footer footer footer footer';

  /\* grid-column-gap: 10px; - if you want gap between the columns \*/

}

.left,

.middle,

.right {

  padding: 10px;

  height: 300px; /\* Should be removed. Only for demonstration \*/

}

/\* Style the left column \*/

.left {

  grid-area: left;

}

/\* Style the middle column \*/

.middle {

  grid-area: middle;

}

/\* Style the right column \*/

.right {

  grid-area: right;

}

/\* Style the footer \*/

.footer {

  grid-area: footer;  background-color: #f1f1f1;  padding: 10px;  text-align: center;

}

/\* Responsive layout - makes the three columns stack on top of each other instead of next to each other \*/

@media (max-width: 600px) {

  .grid-container  {

    grid-template-areas:

      'header header header header header header'

      'left left left left left left'

      'middle middle middle middle middle middle'

      'right right right right right right'

      'footer footer footer footer footer footer';

  }

}

A screenshot of a computer

Description automatically generated

**unequal** columns

## using Float

**HTML**

<div class="header">

    <h2>Header</h2>

  </div>

    <div class="row">

    <div class="column side" style="background-color:#757070;">Column</div>

    <div class="column middle" style="background-color:#d0f110;">Column</div>

    <div class="column side" style="background-color:#117fda;">Column</div>

  </div>

    <div class="footer">

    <p>Footer</p>

  </div>

A screenshot of a computer

Description automatically generated

## 3 way to create CSS Template

## Float

## Flexbox

## Grid

A screenshot of a computer

Description automatically generated

## using Float

**CSS**

\* {

  box-sizing: border-box;

}

body {

  font-family: Arial, Helvetica, sans-serif;

}

/\* Style the header \*/

.header {

  background-color: pink;

  padding: 30px;

  text-align: center;

  font-size: 35px;

}

/\* Create three unequal columns that floats next to each other \*/

.column {

  float: left;

  padding: 10px;

  height: 300px; /\* Should be removed. Only for demonstration \*/

}

/\* Left and right column \*/

.column.side {

  width: 25%;

}

/\* Middle column \*/

.column.middle {

  width: 50%;

}

/\* Clear floats after the columns \*/

.row:after {

  content: "";

  display: table;

  clear: both;

}

/\* Style the footer \*/

.footer {

  background-color: orange;

  padding: 10px;

  text-align: center;

}

/\* Responsive layout - makes the three columns stack on top of each other instead of next to each other \*/

@media (max-width: 600px) {

  .column.side, .column.middle {

    width: 100%;

  }

}

## using Flexbox

\* {

  box-sizing: border-box;

}

body {

  font-family: Arial, Helvetica, sans-serif;

}

/\* Style the header \*/

.header {

  background-color: pink;

  padding: 30px;

  text-align: center;

  font-size: 35px;

}

/\* Container for flexboxes \*/

.row {

  display: -webkit-flex;

  display: flex;

}

/\* Create three unequal columns that floats next to each other \*/

.column {

  float: left;

  padding: 10px;

  height: 300px; /\* Should be removed. Only for demonstration \*/

}

/\* Left and right column \*/

.column.side {

  -webkit-flex: 1;

   -ms-flex: 1;

   flex: 1;

}

/\* Middle column \*/

.column.middle {

  -webkit-flex: 2;

  -ms-flex: 2;

  flex: 2;

}

/\* Style the footer \*/

.footer {

  background-color: orange;

  padding: 10px;

  text-align: center;

}

/\* Responsive layout - makes the three columns stack on top of each other instead of next to each other \*/

@media (max-width: 600px) {

  .row {

    -webkit-flex-direction: column;

    flex-direction: column;

  }

}

## using Grid

**HTML**

<div class="grid-container">

  <div class="header">

    <h2>Header</h2>

  </div>

  <div class="left" style="background-color:#757070;">Column</div>

  <div class="middle" style="background-color:#d0f110;">Column</div>

  <div class="right" style="background-color:#117fda;">Column</div>

  <div class="footer">

    <p>Footer</p>

  </div>

</div>

**CSS**

\* {

  box-sizing: border-box;

}

body {

  font-family: Arial, Helvetica, sans-serif;

}

/\* Style the header \*/

.header {

  grid-area: header;

  background-color: pink;

  padding: 30px;

  text-align: center;

  font-size: 35px;

}

/\* The grid container \*/

.grid-container {

  display: grid;

  grid-template-areas:

    'header header header header header header'

    'left middle middle middle middle right'

    'footer footer footer footer footer footer';

  /\* grid-column-gap: 10px; - if you want gap between the columns \*/

}

.left,

.middle,

.right {

  padding: 10px;

  height: 300px; /\* Should be removed. Only for demonstration \*/

}

/\* Style the left column \*/

.left {

  grid-area: left;

}

/\* Style the middle column \*/

.middle {

  grid-area: middle;

}

/\* Style the right column \*/

.right {

  grid-area: right;

}

/\* Style the footer \*/

.footer {

  grid-area: footer;

  background-color: orange;

  padding: 10px;

  text-align: center;

}

/\* Responsive layout - makes the three columns stack on top of each other instead of next to each other \*/

@media (max-width: 600px) {

  .grid-container  {

    grid-template-areas:

      'header header header header header header'

      'left left left left left left'

      'middle middle middle middle middle middle'

      'right right right right right right'

      'footer footer footer footer footer footer';

  }

}

A screenshot of a computer

Description automatically generated

## Topnav

A screenshot of a computer

Description automatically generated**HTML**

<div class="topnav">

  <a href="#">Link</a>

  <a href="#">Link</a>

  <a href="#">Link</a>

</div>

<div class="content">

  <h2>3 - CSS Template Topnav</h2>

</div>

<div class="footer">

  <p>Footer</p>

</div>

**CSS**

\* {

  box-sizing: border-box;

  font-family: Arial, Helvetica, sans-serif;

}

body {

  margin: 0;

  font-family: Arial, Helvetica, sans-serif;

}

/\* Style the top navigation bar \*/

.topnav {

  overflow: hidden;  background-color: blue;

}

/\* Style the topnav links \*/

.topnav a {

  float: left;  display: block;  color: white;  text-align: center;  padding: 14px 16px;

  text-decoration: none;

}

/\* Change color on hover \*/

.topnav a:hover {

  background-color: yellow;

  color: black;

}

/\* Style the content \*/

.content {

  background-color: lightblue;  padding: 10px;

  height: 200px; /\* Should be removed. Only for demonstration \*/

}

/\* Style the footer \*/

.footer {

  background-color: orange;  padding: 10px;

}

## Sidenav

**HTML**

<div class="sidenav">

  <a href="#">Link</a>

  <a href="#">Link</a>

  <a href="#">Link</a>

</div>

<div class="content">

  <h2>3 - CSS Template Sidenav</h2>

</div>

**CSS**

\* {

  box-sizing: border-box;

}

body {

  margin: 0;

  font-family: Arial, Helvetica, sans-serif;

}

/\* Style the side navigation \*/

.sidenav {

  height: 100%;

  width: 200px;

  position: fixed;

  z-index: 1;

  top: 0;

  left: 0;

  background-color: blue;

A screenshot of a computer

Description automatically generated  overflow-x: hidden;

}

/\* Side navigation links \*/

.sidenav a {

  color: white;

  padding: 16px;

  text-decoration: none;

  display: block;

}

/\* Change color on hover \*/

.sidenav a:hover {

  background-color: yellow;

  color: black;

}

/\* Style the content \*/

.content {

  margin-left: 200px;

  padding-left: 20px;

}