

Grid

CheatSheet in 2021



CSS Grid Cheat Sheet Illustrated in 2021 🏆

#css

#webdev

#beginners

#tutorial



Joy Shaheb

4. März • Updated on 27. Mai • 6 min read

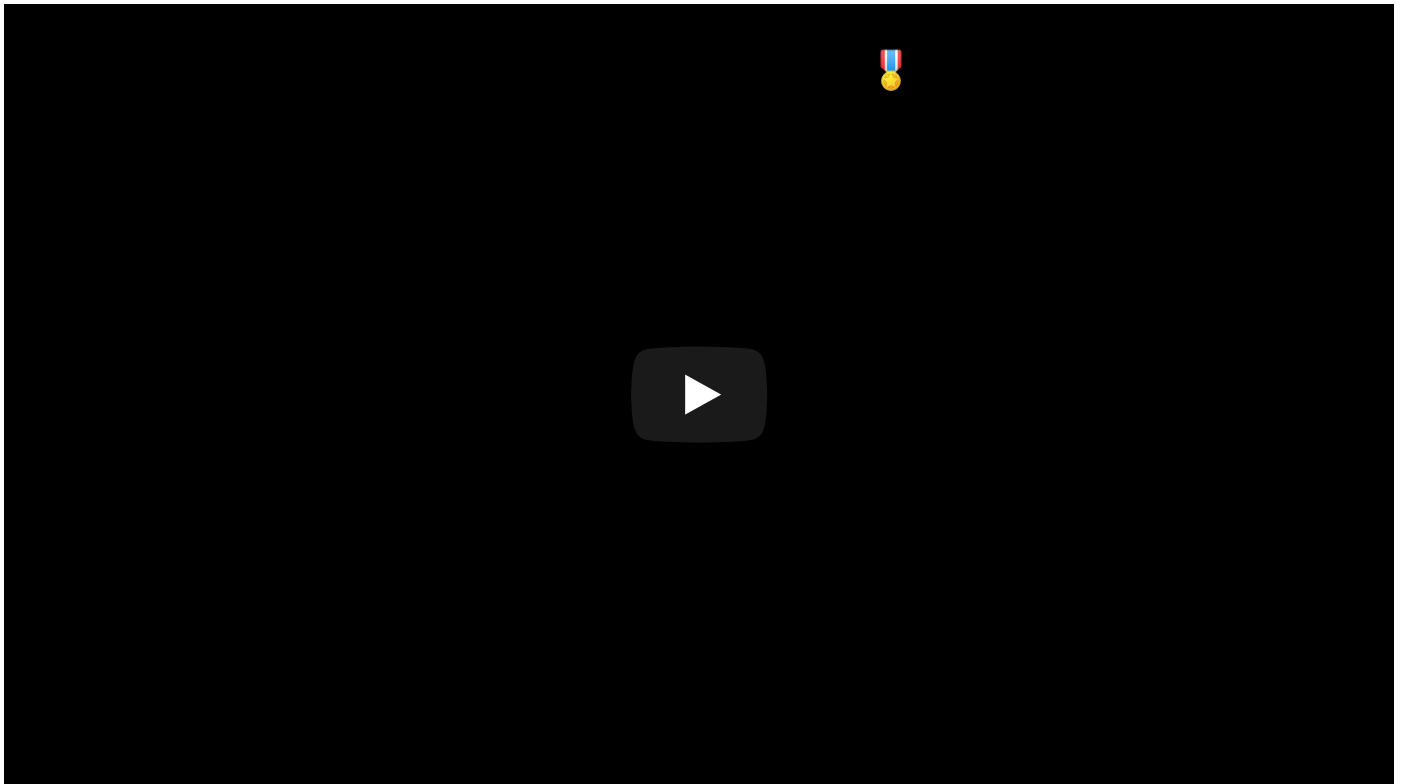
Today we're gonna learn **CSS Grid** basics so that you can make your own responsive sites. I'll explain how each of Grid's properties work along with a CheatSheet that covers everything you can do with Grid. Let's Go 🏆

Table of Contents :

- [Grid Architecture](#)
- Parent Properties
 - [grid-template-columns](#)
 - [grid-template-rows](#)
 - [grid-template-areas](#)
 - [column & row gap](#)
 - [justify-items](#) || [align-items](#)
 - [justify-content](#) || [align-content](#)
- Children Properties
 - [grid-column : start/end](#)
 - [grid-row : start/end](#)
 - [grid-area](#)
 - [justify-self](#) || [align-self](#)
- [Short Hands](#)

- [Conclusion](#)

You can watch this tutorial on YouTube as well if you like:



First, What is Grid?

What is Grid?



Grid is a blueprint for making websites.

The Grid model allows us to layout the content of our website. Not only that, it helps us create the structures needed for creating responsive websites for multiple devices.

... ..

Here's a simple demo which I created using Grid as the main blueprint.

Purple

Home Store Cart



Hey There 🖐️

Let's Sing Together

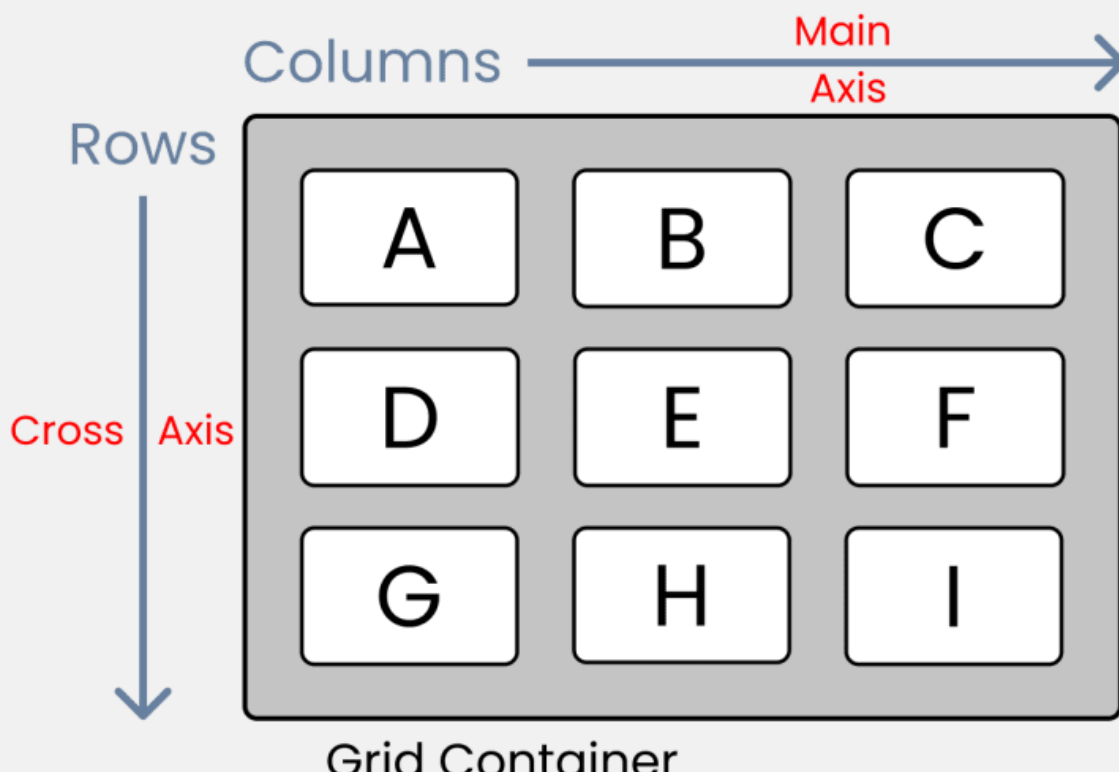
Log in

Sign up

Grid Architecture

So how does Grid architecture work? The Grid items [Contents] are distributed along the main axis and cross axis. Using various Grid properties, we manipulate the items to create our website layouts.

Grid Architecture



How to Set Up the Project

Let's Code Together



For this project, you need to know little bit of HTML, CSS, and how to work with VS code. Follow along with me as we complete the following tasks:

1. Create a folder named "Project-1" & Open VS Code
2. Create index.html and style.css files
3. Install Live Server and run it.

Or, you can just open Codepen and start coding.

At the end of this tutorial, you will be able to make accurate and beautiful website layouts.

And...we're all set! Let's start coding. 😊

Let's Start Coding !





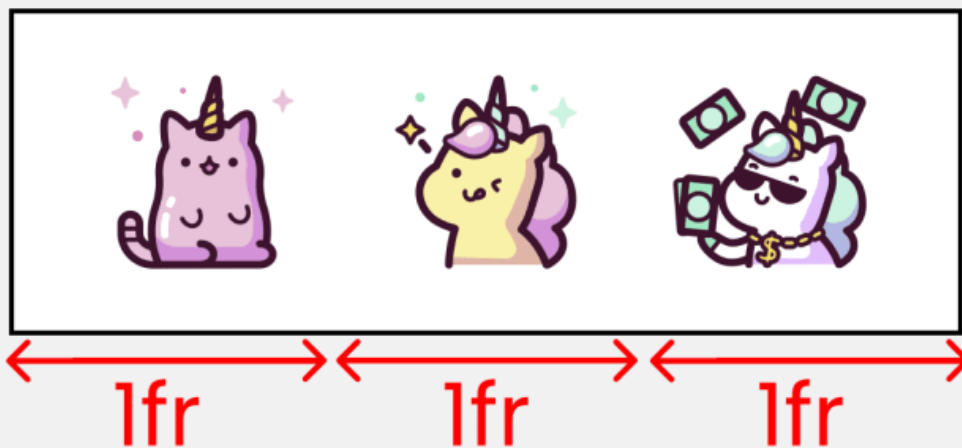
grid-template-columns

This is used to define the **Number & Width** of columns. You can either individually set the width of each column, or set an uniform width for all columns using "repeat()" function.

```
grid-template-columns : 200px auto 100px;
```



```
grid-template-columns : repeat(3, 1fr);
```



grid-template-rows

This is used to define the **Number & Height** of rows. You can either individually set the height of each row, or set an uniform height for all rows using "repeat()" function.

grid-template-rows : 200px auto 100px



```
grid-template-rows : repeat(3, 1fr);
```

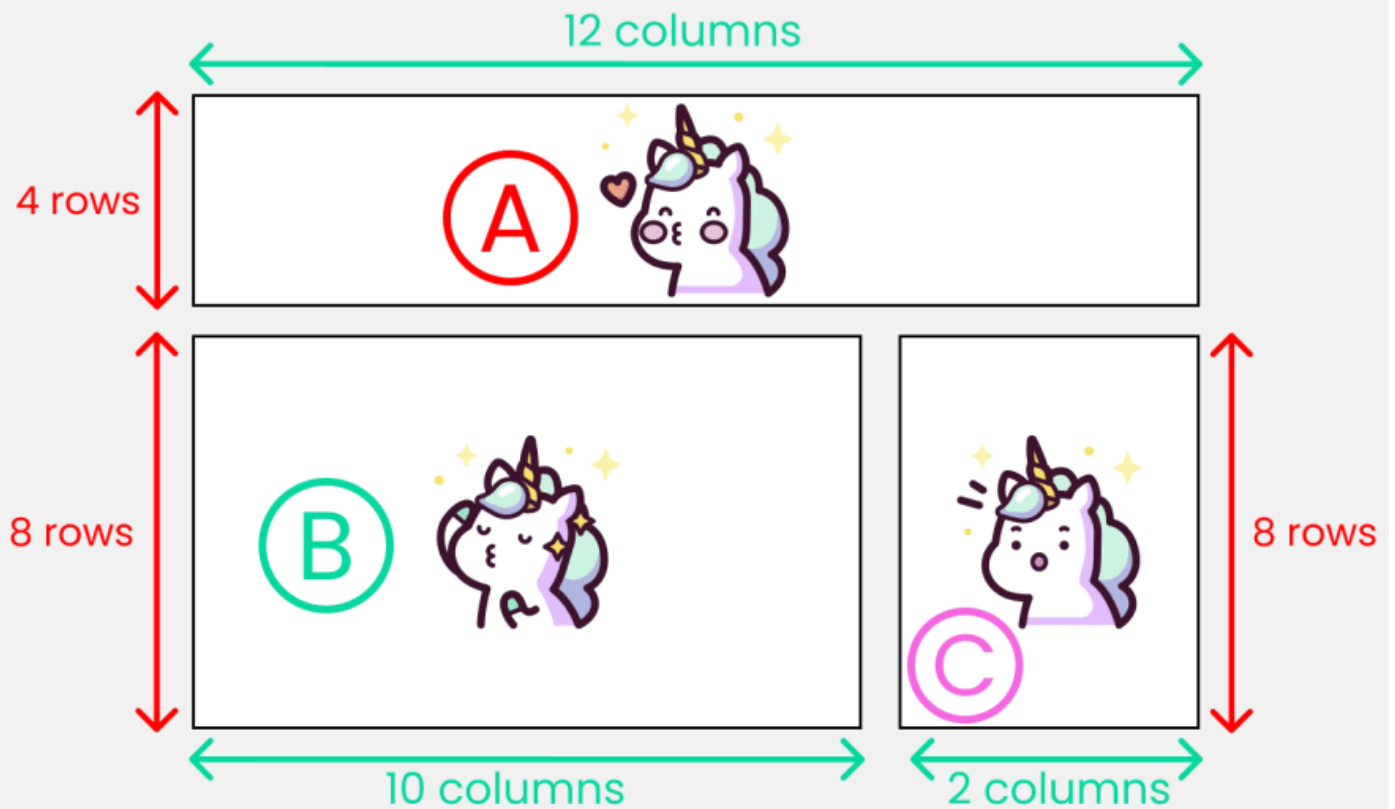


grid-template-areas

This is used to specify the amount of space a grid cell should carry in terms of column & row across the parent container. Life's Quite Easier With this as it allows us to see visually what we're doing.

grid-template-areas

12 row X 12 column Layout



Call it, the blueprint(template) of our layout 🙌

The Code

grid-template-areas :

"A A A A A A A A A A A A"

"B B B B B B B B B B B C C"

"B B B B B B B B B B B C C";

column-gap :

This property is used to place gap between **Columns** inside the grid 🙌

column-gap: 50px

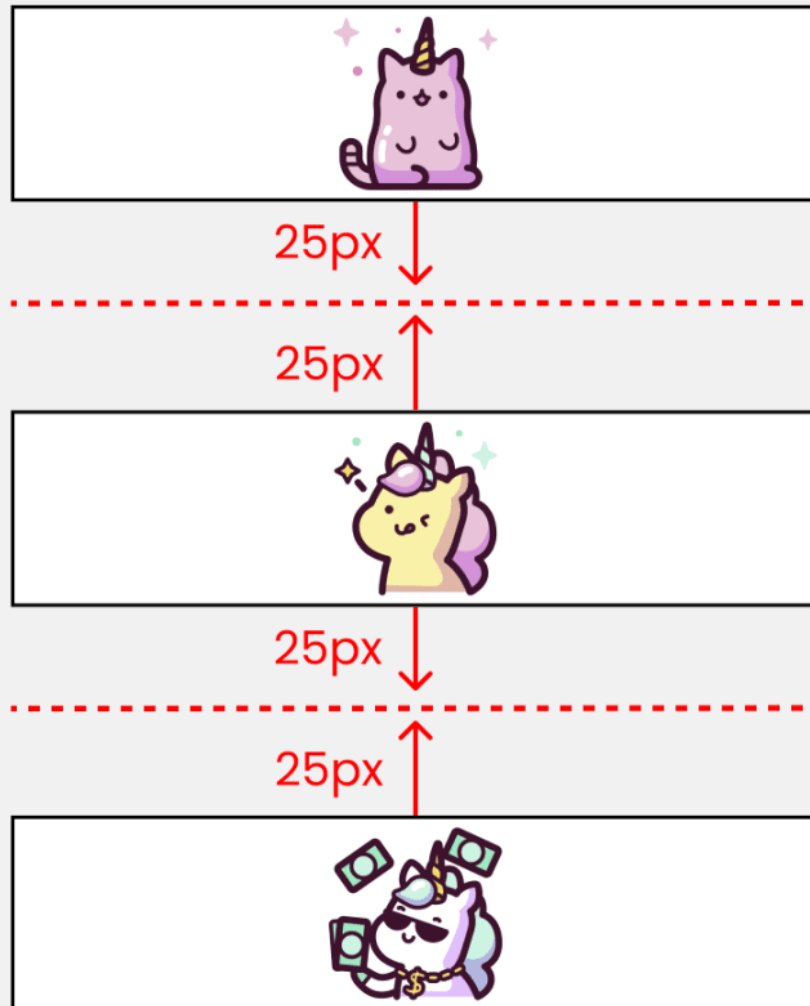


Red **Dotted** lines are called -> grid lines

row-gap :

This property is used to place gap between **Rows** inside the grid 👉

row-gap: 50px



Red **Dotted** lines are called -> grid lines

justify-items :

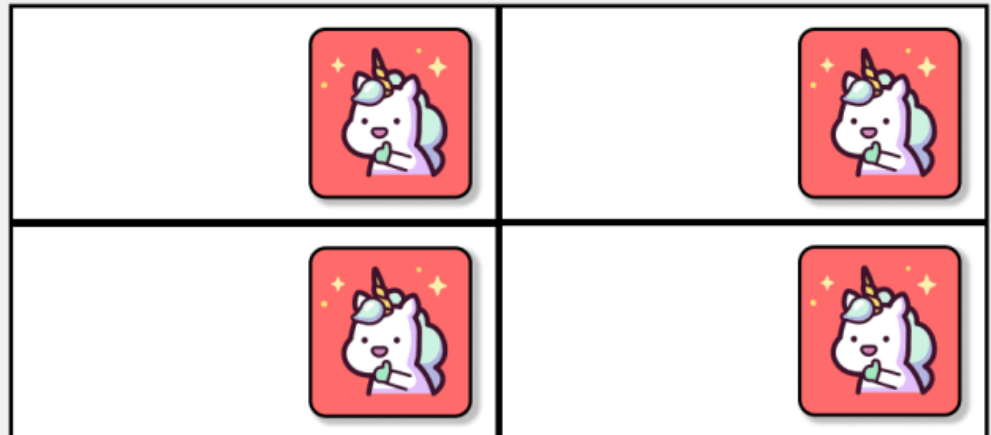
This is used to position grid-items (children) inside grid container along the **X-Axis[Main Axis]**. The 4 values are 🖱️

justify-items -> X Axis

Start



end



justify-items -> X Axis

center



stretch

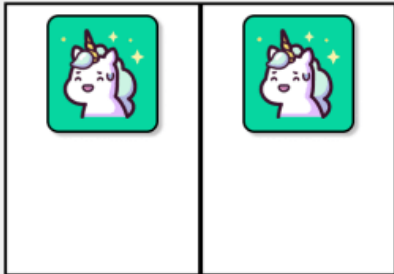


align-items :

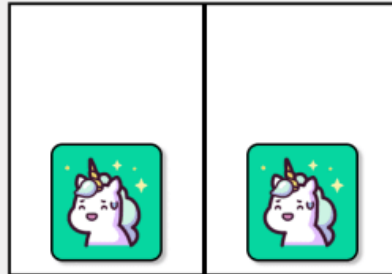
This is used to position grid-items (children) inside grid container along the Y-Axis[Cross Axis]. The 4 values are 👉

align-items -> Y Axis

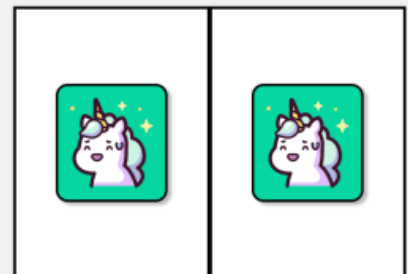
Start



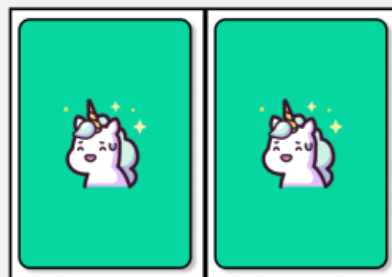
end



center



stretch

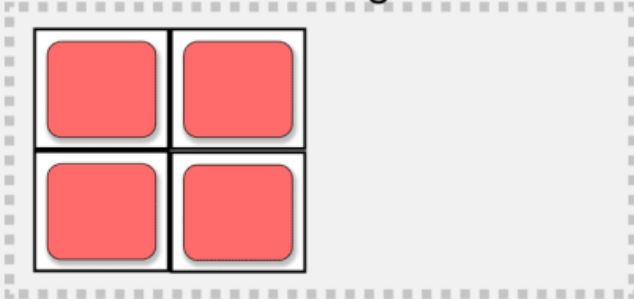


justify-content :

This is used to position our grid [Basically everything] inside grid container along the **X-Axis**[Main Axis]. The 7 values are 🙋

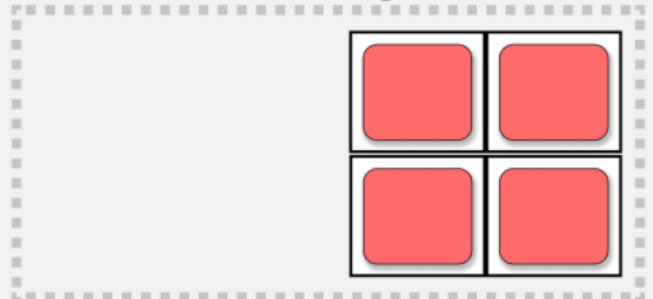
justify-content -> X Axis

grid container

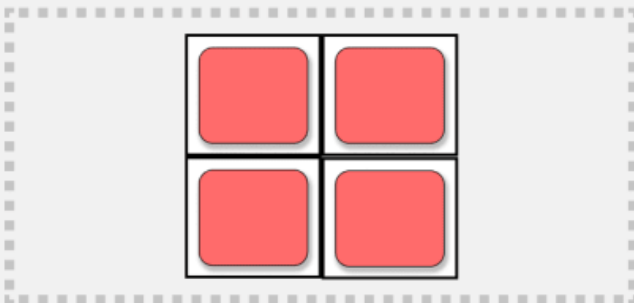


start

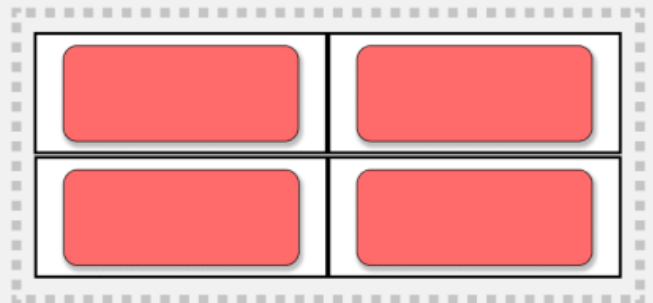
grid container



end



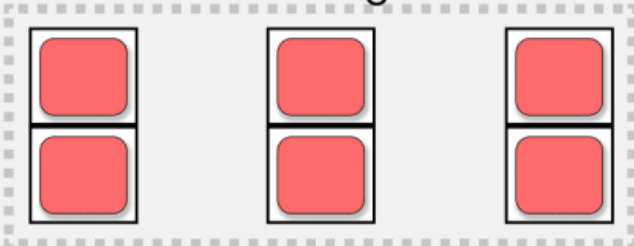
center



stretch

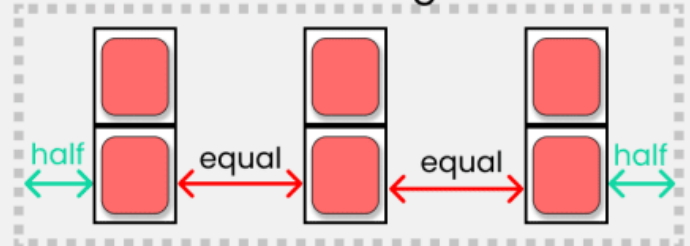
justify-content -> X Axis

grid container



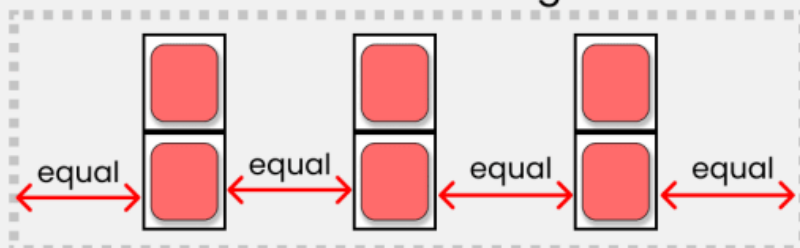
space-between

grid container



space-around

grid container



space-evenly

align-content :

This is used to position our grid [Basically everything] inside grid container along the **Y-Axis**[Cross Axis]. The 7 values are 🖱️

align-content -> Y Axis

grid container

start

grid container

end

grid container

center

grid container

stretch

align-content -> Y Axis

grid container

space-between

grid container

half
equal
equal
half

space-around

grid container

equal
equal
equal
equal

space-evenly

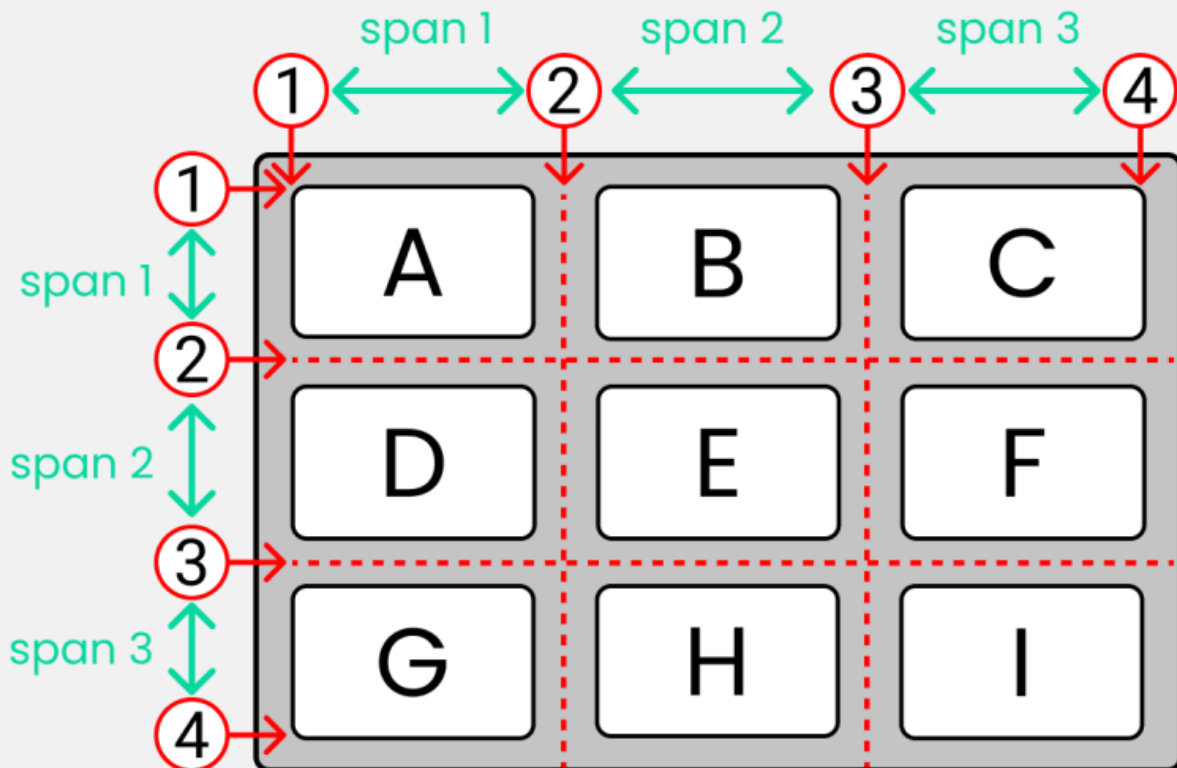
Children Properties

Grid **Children** Properties



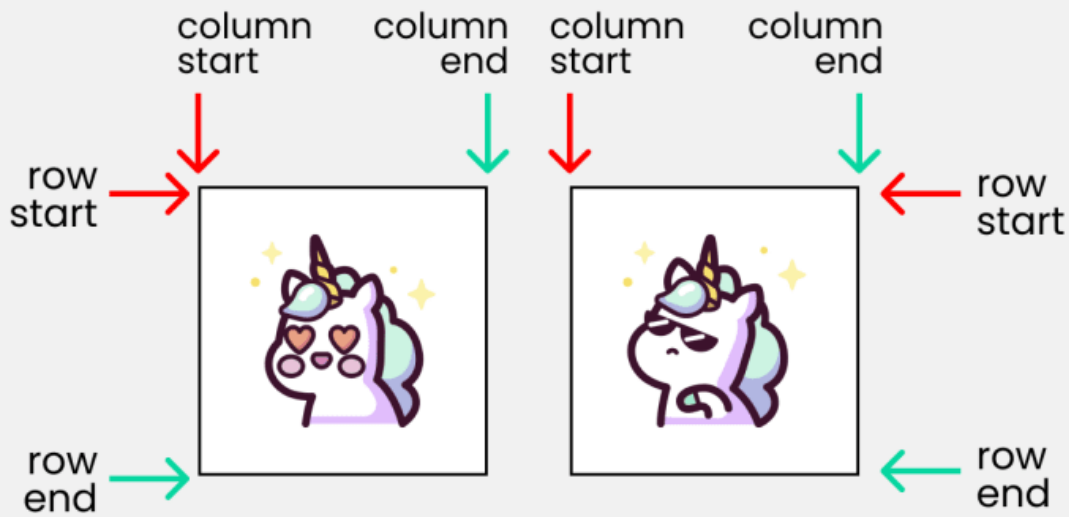
The Grid Scale

Grid **Scale**



3 X 3 layout

grid column & row -> **start** & **end** points



grid-column : start/end

THESE 2 properties are used to join multiple **COLUMNS** together. It is a shorthand of **grid-column-start** & **grid-column-end**. Skip to this topic on the video above if you're confused using the timestamps.

grid-row : start/end

THESE 2 properties are used to join multiple **ROWS** together. It is a shorthand of **grid-row-start** & **grid-row-end**.

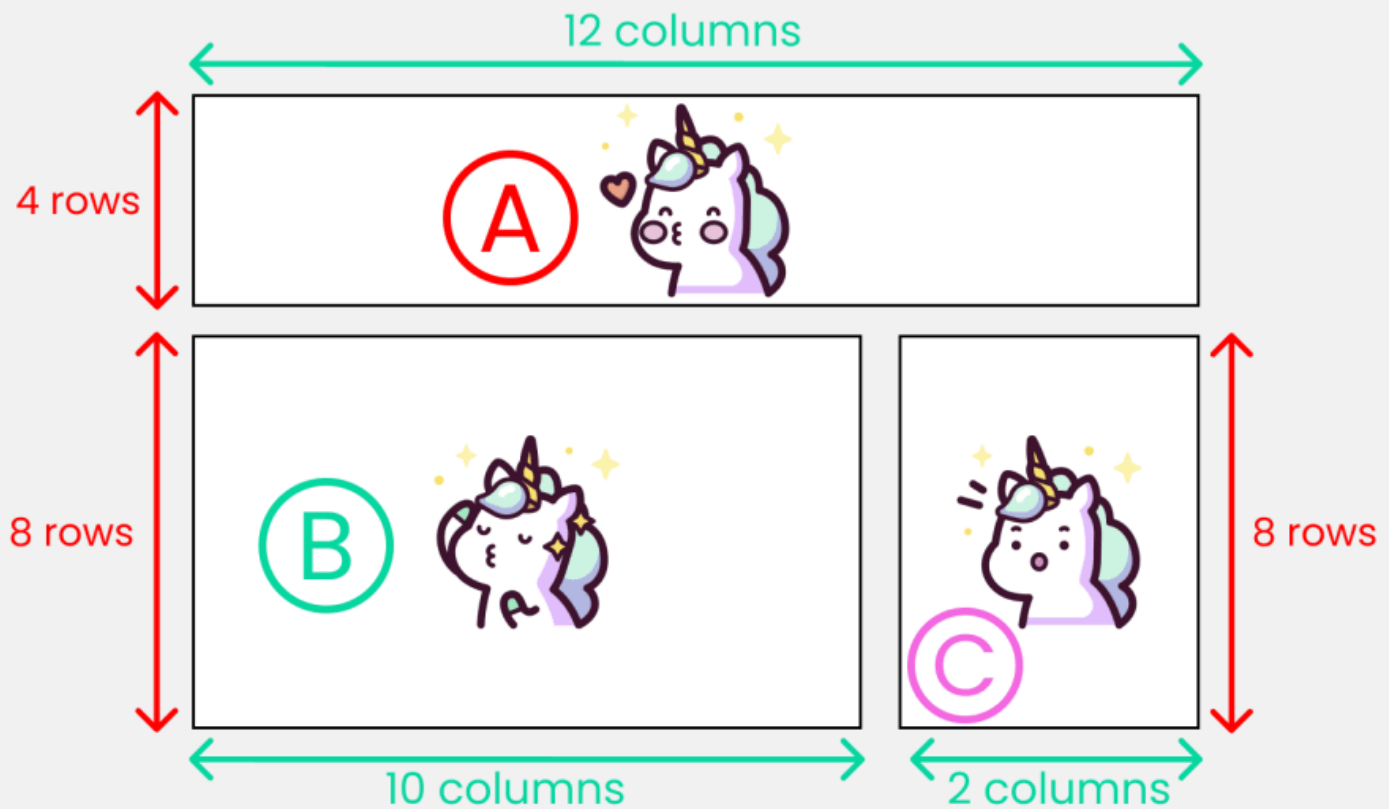
Skip to this topic on the video above if you're confused using the timestamps.

grid-area :

At first, we need to setup [grid-template-areas](#) 🙌 Once done, we have to specify the names used in parent class inside the children classes, like this 🙌

grid-areas

12 row X 12 column Layout



Specifying grid-template-areas inside parent container 🙋

The Parent class

```
.container{
```

```
  grid-template-areas :
```

```
    "A A A A  A A A A  A A A A"
```

```
    "B B B B  B B B B  B B C C"
```

```
    "B B B B  B B B B  B B C C";  
}
```

specifying the names used in parent container inside children classes with grid-areas



The Children classes

```
.image-1{  
    grid-area : A ;  
}
```

```
.image-2{  
    grid-area : B ;  
}
```

```
.image-3{  
    grid-area : C ;  
}
```

Justify-self :

This is used to position **1 individual** grid-item (children) inside grid container along the **X-Axis[Main Axis]**. The **4** values are

justify-self -> **X Axis**

start



end



center



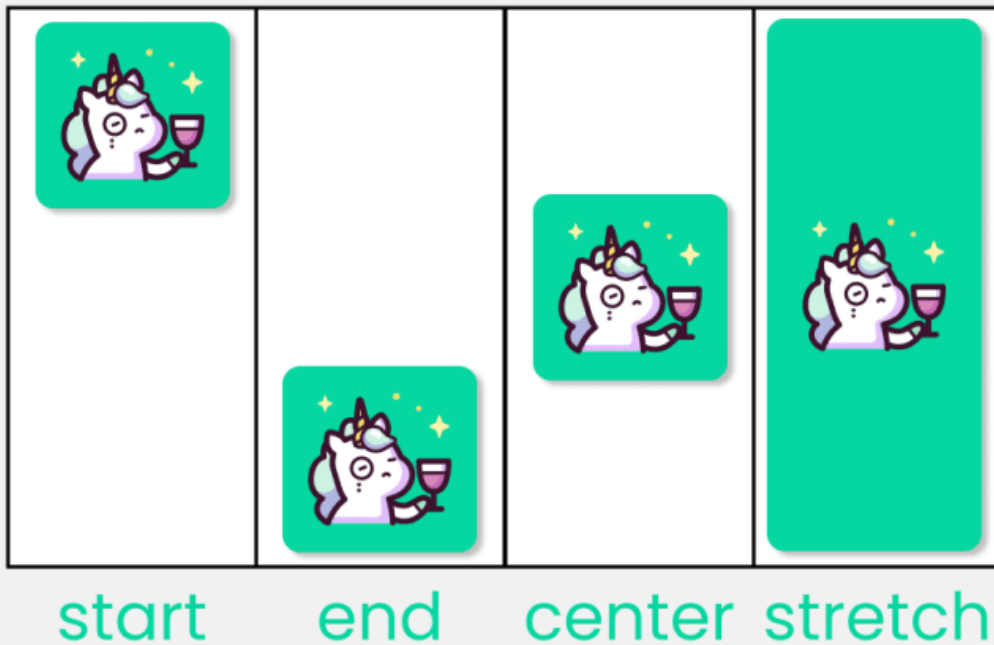
stretch



align-self :

This is used to position 1 **individual** grid-item (children) inside grid container along the **Y-Axis**[Cross Axis]. The 4 values are 🖱️

align-self -> Y Axis



Short Hands

Time to Investigate
Grid **Short Hands**



place-content :

place-content : align-content / justify-content



Y-axis



X-axis

place-items :

place-items : align-items / justify-items



Y-axis



X-axis

place-self :

place-self : align-self / justify-self



Y-axis



X-axis

grid-template :

grid-template : grid-template-rows / grid-template-columns

nan/grid-nan :

gap/ grid gap .

```
gap : row-gap column-gap;
```

Credits

[Unicorns](#)

Images Taken From  **flaticon**

Read Next :



Master CSS Flexbox 2021 🔥 - Build 5 Responsive Layouts 🏆 || CSS 2021

Joy Shaheb • Feb 4 • 5 min read

#css #tutorial #beginners #webdev



Acing CSS Grid Model in 2021 with 5 Exercises || CSS 2021 🔥

Joy Shaheb • Dec 22 '20 • 6 min read

#tutorial #beginners #webdev #css

Conclusion

Here's Your Medal For reading till the end ❤️

Suggestions & Criticisms Are Highly Appreciated ❤️

