!!Contenido Importante!!
07/02/2020 ,20:16:26

Completado
07/02/2020 ,20:16:31

CSS Grid Layout

C	S	S	
G	R	I	D

CODE VIEW

```
<style>
.container > div {
    background: #eee;
    padding: 1em;
    }
.container > div:nth-child(odd) {
    background: #ddd;
    }
.container {
    display: grid;
    grid-template-columns: 1fr 2fr 1fr;
    grid-auto-rows: minmax(120px, auto);
    grid-gap: 1em;
    }
</style>
<body>
<div class="container">
    <div>Grid item one.</div>
    <div>Grid item two.</div>
    <div>Grid item three.</div>
    <div>Grid item four.</div>
    <div>Grid item five.</div>
    <div>Grid item six.</div>
    <div>Grid item seven.</div>
    <div>Grid item eight.</div>
</div>
</body>
```

JUSTIFY AND ALIGN

GETTING STARTED

Starting CSS

← Add the CSS to the left. We are creating a grid container with three columns set up at 1fr 2fr 1fr.

Starting HTML

← Add the HTML to the left to create eight boxes (grid items) for our layout.

abajo es el cual obtendras en el navegador A continuación se muestra lo que obtienes en un navegador.

Below is what you get in a browser. So far, we have not done anything with justification or alignment.

Hasta ahora, no hemos hecho nada con justificación o alineación.

Grid item one. | Grid item two. | Grid item three.

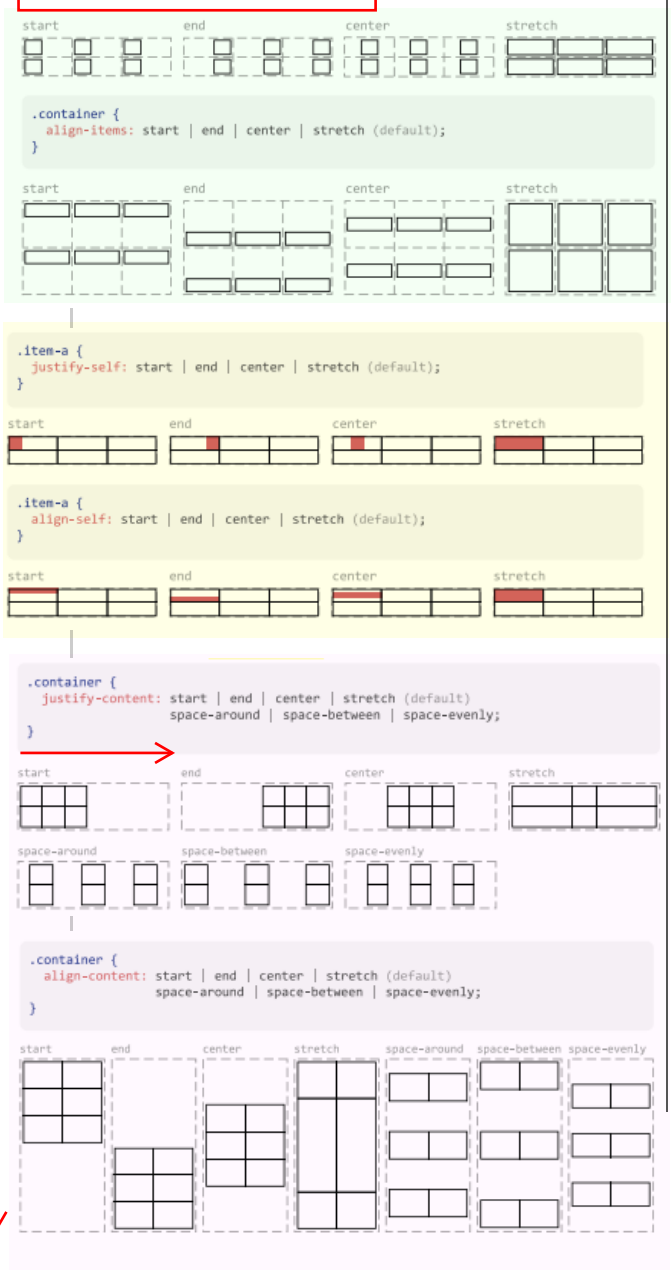
Grid item four. | Grid item five. | Grid item six.

Grid item seven. | Grid item eight.

Mucho antes de que apareciera CSS Grid, Flexbox nos dio nuevas y poderosas formas de alinear el contenido.
Al igual que Flexbox, CSS Grid le permite alinear elementos o alinear contenido dentro de elementos
. Hay solo seis propiedades: justify-items, align-items, justify-content, align-content, justify-self y align-self.

El término justificar pertenece a la alineación de fila (horizontal) y el término alinear corresponde a la alineación de columna (vertical).

justify-items: start , end , center , stretch



OVERVIEW

Mucho antes CSS GRID apareciera

Long before CSS Grid came out, Flexbox gave us powerful new ways to align content. Much like Flexbox, CSS Grid allows you to align elements or align content within elements. There are just six properties: justify-items, align-items, justify-content, align-content, justify-self, and align-self.

pertenece

The term justify pertains to row alignment (horizontal) and the term align pertains to column (vertical) alignment.

You can apply these properties in three ways. Alignment can be applied to:

Grid-items dentro de un grid-container

1. GRID ITEMS IN A GRID CONTAINER.
   - Rows (Horizontal): justify-items: start (start, center, end, stretch)
   - Columns (Vertical): align-items: start (start, center, end, stretch)

Contenido dentro de un grid-item

2. CONTENT WITHIN A GRID ITEM.
   - Rows (Horizontal): justify-self: start (start, center, end, stretch)
   - Columns (Vertical): align-self: start (start, center, end, stretch)

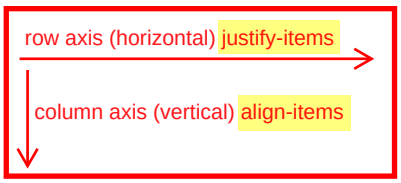
CUANDO GRID-ITEMS DENTRO DEL GRID-CONTENEDOR ES MENOR QUE EL ESPACIO USADO POR EL CONTENEDOR QUE LOS ALMACENAN

3. GRID ITEMS IN A GRID CONTAINER WHEN THE GRID IS SMALLER THAN THE SPACE USED BY THE CONTAINER.
   - Rows (Horizontal): justify-content: start (start, center, end, space-around, space-between, space-evenly)
   - Columns (Vertical): align-content: start (start, center, end, space-around, space-between, space-evenly)

1. ALIGNING ALL GRID ITEMS IN A GRID CONTAINER

a lo largo

A grid container aligns grid items along its row axis (horizontal) with justify-items. A grid container aligns grid items along its column axis (vertical) with align-items.



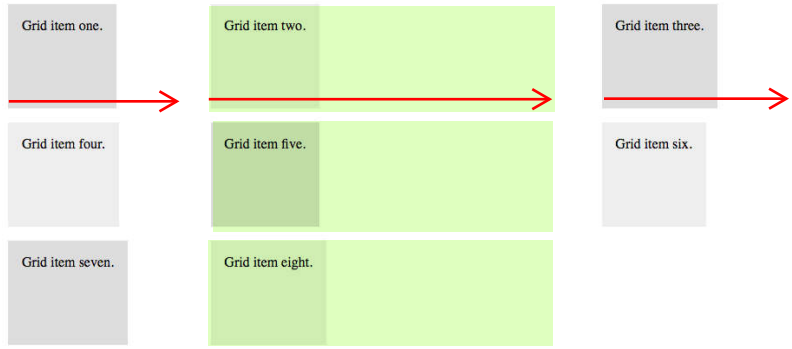
```
.container {
    display: grid;
    grid-template-columns: 1fr 2fr 1fr;
    grid-auto-rows: minmax(120px, auto);
    grid-gap: 1em;
    justify-items: start;
    }
<div class="container">
    <div>Grid item one.</div>
    <div>Grid item two.</div>
    <div>Grid item three.</div>
    <div>Grid item four.</div>
    <div>Grid item five.</div>
    <div>Grid item six.</div>
    <div>Grid item seven.</div>
    <div>Grid item eight.</div>
</div>
```

Row Alignment ← Add the CSS declaration justify-items: start to the grid container.

horizontal

Como la anchura del Grid-Item es menor que el ancho del Grid-Container se puede desplazar dentro de el

Below is what you get in a browser.



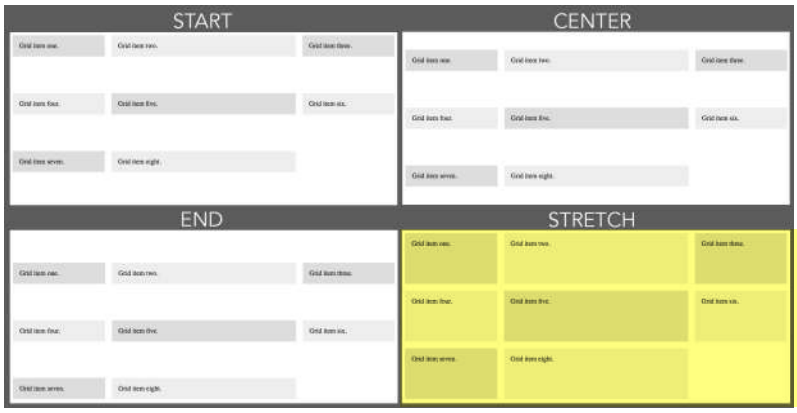
The property values could be any of the following: start | center | end | stretch, as shown below.

por defecto

```
.container {
    display: grid;
    grid-template-columns: 1fr 2fr 1fr;
    grid-auto-rows: minmax(120px, auto);
    grid-gap: 1em;
    justify-items: start;
    align-items: start;
    }
<div class="container">
    <div>Grid item one.</div>
    <div>Grid item two.</div>
    <div>Grid item three.</div>
    <div>Grid item four.</div>
    <div>Grid item five.</div>
    <div>Grid item five.</div>
    <div>Grid item six.</div>
    <div>Grid item seven.</div>
    <div>Grid item eight.</div>
</div>
```

Column Alignment ← Remove the declaration justify-items: start. Add the declaration align-items: start to the grid container.

The property values could be any of the following: start | center | end | stretch, as shown below.



2. ALIGNING CONTENT WITHIN A GRID ITEM

Habra momentos cuando desees alinear contenido dentro de un elemento de cuadricula determinado lograr

There will be times when you'll want to align content within a given grid item. In this example, we will use the justify-self property to achieve row alignment and the align-self property to achieve column alignment.

Habrá momentos en los que querrá alinear contenido dentro de un elemento grid-item.
En este ejemplo, utilizaremos la propiedad justify-self para lograr la alineación de filas y la propiedad align-self para lograr la alineación de columnas.

```
.container {
    display: grid;
    grid-template-columns: 1fr 2fr 1fr;
    grid-auto-rows: minmax(120px, auto);
    grid-gap: 1em;
    align-items: start
    }
.box1 {
    justify-self:start;
    }
.box2 {
    align-self:start;
    }
<div class="container">
    <div class="box1">Grid item one.</div>
    <div class="box2">Grid item two.</div>
    <div>Grid item three.</div>
    <div>Grid item four.</div>
    <div>Grid item five.</div>
    <div>Grid item six.</div>
    <div>Grid item seven.</div>
    <div>Grid item eight.</div>
</div>
```

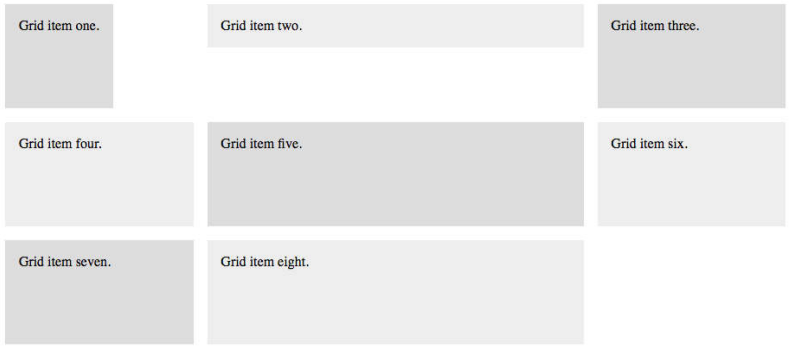
Row Alignment

← Remove the declaration align-items: start from the container. Add a class called box1 with a declaration of justify-self: start to handle row alignment. Then, add a class called box2 with a declaration of align-self: start to handle column alignment.

← Add the box1 and box2 classes to the first and second grid items, as shown to the left.

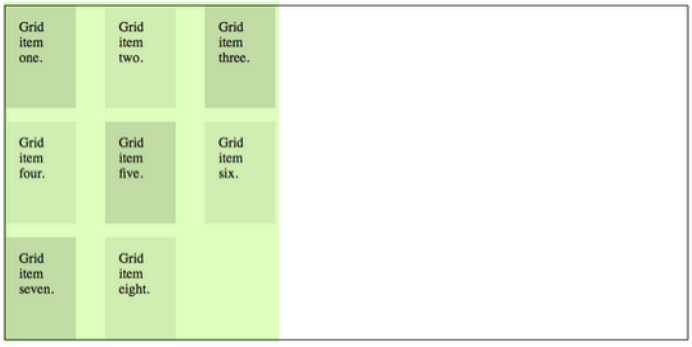
Abajo es lo que obtienes en un navegador Fijate que el Grid-Item

Below is what you get in a browser. Notice that grid item one has aligned its content to the left or starting position horizontally within the grid item. Also notice that the content in grid item two has aligned itself at the top or starting position vertically within the second grid item.



The property values could be any of the following: start | center | end | stretch, as shown below. Notice the effect of these changes on the first and second grid items.

Sometimes, you may have a situation where the items inside a grid container have fixed widths (such as using pixels) and the width of the items together on a row is less than the width of the container. In the example below, each of the grid items (shaded boxes) are set to a width of 100 pixels. It's a three column layout, so together the items take up only 300 pixels in width. Together, the items take up less space on a row than the width of the container (outlined in black).



A good example of the problem is seen in navigation when the items together need to be controlled for alignment.

The CSS properties here are called justify-content (for row/horizontal alignment) and align-content (for column/vertical alignment).

Possible values include start, center, end, space-around, space-between, or space-evenly.

3. ALIGNING ITEMS IN A GRID CONTAINER WHEN THE GRID IS SMALLER THAN THE SPACE USED BY THE CONTAINER

a veces

Sometimes, you may have a situation where the items inside a grid container have fixed widths (such as using pixels) and the

```
.container {
    display: grid;
    grid-template-columns: repeat(3,
    80px);
    grid-auto-rows: minmax(120px, auto);
    grid-gap: 1em;
    border: 1px solid black;
    }
<div class="container">
    <div>Grid item one.</div>
    <div>Grid item two.</div>
    <div>Grid item three.</div>
    <div>Grid item four.</div>
    <div>Grid item five.</div>
    <div>Grid item six.</div>
    <div>Grid item seven.</div>
    <div>Grid item eight.</div>
</div>
```

Starting CSS

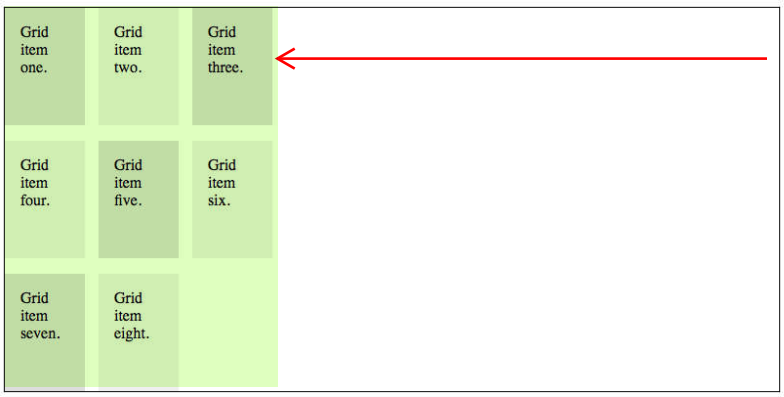
← Use the CSS to the left to create a three column layout. Each column will be 80 pixels (80px) wide, thus making them equal in width. Together, the grid items will take up only 240 pixels on a row. We are also adding a black border on the container to better see the relationship between the container and the grid items.

Starting HTML

← Add the HTML to the left to create eight boxes (grid items) for our layout.

Observe que con cada grid item

The result is shown below. Notice that with each grid item given a width of 80 pixels, it leaves all kinds of open space on the right side of the container.

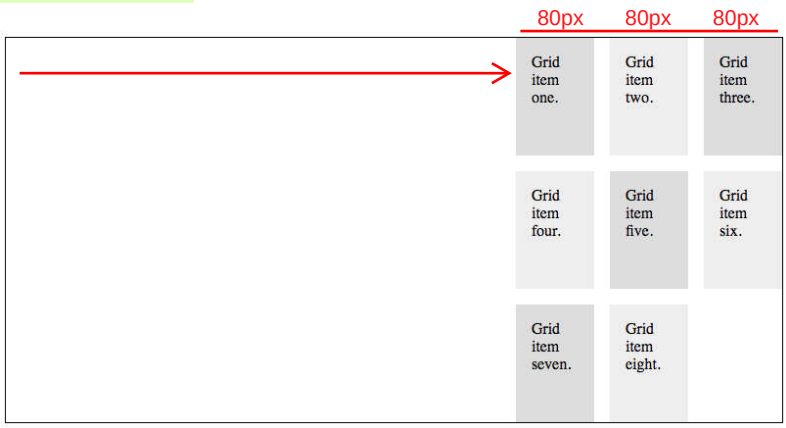


```
.container {
    display: grid;
    grid-template-columns: repeat(3, 80px);
    grid-auto-rows: minmax(120px, auto);
    grid-gap: 1em;
    border: 1px solid black;
    justify-content: end;
    }
<div class="container">
    <div>Grid item one.</div>
    <div>Grid item two.</div>
    <div>Grid item three.</div>
    <div>Grid item four.</div>
    <div>Grid item five.</div>
    <div>Grid item six.</div>
    <div>Grid item seven.</div>
    <div>Grid item eight.</div>
</div>
```

Row Alignment

← Add the declaration justify-content: end to change the alignment of the grid items.

The result is shown below. Notice how the grid is moved to the end of the row.



As noted, the justify-content property values could be any of the following: start | center | end | space-around | space-between | space-evenly. Notice the effect of using these values with the justify-content property.

```
.container > div {
    background: #eee;
    padding: 1em;
    }
.container {
    display: grid;
    grid-template-columns: repeat(4,
    100px);
    grid-auto-rows: minmax(100px, auto);
    grid-gap: 1em;
    border: 1px solid black;
    }
<div class="container">
    <div><a href="#">Home</a></div>
    <div><a href="#">About</a></div>
    <div><a href="#">Services</a></div>
    <div><a href="#">Location</a></div>
</div>
```

Aligning Navigation on a Row

surgir cuando necesitando
alineamiento elementos horizontal dentro un menu

A common alignment problem comes up when needing to horizontally align items within a menu/navigation area. As seen below, the default alignment gives no ability to control the menu's location nor does it allow us to control the spacing between items in the menu.

la alineación predeterminada
no permite controlar
la ubicación del menú ni
nos permite controlar
el espacio entre los
elementos del menú

Starting CSS

← Use the CSS to the left to create simple styles for a horizontal navigation area. We'll need four columns for the four grid items used to hold the menu items. We are setting each to a width of 100 pixels.

CUANDO CREAMOS GRID-CONTAINER tenemos :
- POR UN LADO LAS COLUMNAS
- POR OTRO LADO LOS ELEMENTOS (Grid-Items) dentro de las COLUMNAS

Starting HTML

← Create the HTML to the left to create the four grid items.

Observa abajo que el menu de elementos estan todos alineados a lo largo del lado izq

Notice below that the menu items are all aligned along the left side of the larger container.

de largo del contenedor


```
.container > div {
    background: #eee;
    padding: 1em;
    }
.container {
    display: grid;
    grid-template-columns: repeat(4,
    100px);
    grid-auto-rows: minmax(100px, auto);
    grid-gap: 1em;
    border: 1px solid black;
    justify-content: space-evenly;
    }
<div class="container">
    <div><a href="#">Home</a></div>
    <div><a href="#">About</a></div>
    <div><a href="#">Services</a></div>
    <div><a href="#">Location</a></div>
</div>
```

← Add the declaration justify-content: space-evenly to the container. This will put an even amount of space around each of the grid items in the container.

esto pondra una uniforme cantidad de espacio alrededor de cada grid items dentro del contenedor

Below shows how nicely justify-content: space-evenly can be used to control the horizontal alignment of navigation items.

```
.container{
    width: 100%;
    height: 800px;

    display: grid;
    grid-template-columns: repeat(4,100px);
    grid-template-rows: repeat(2,200px);
    grid-gap: 10px;

    justify-content: center;
    align-content: center;
    background-color: #ccc;
}
```

Column Alignment

There is rarely a time when you should need to use align-content. That's because we usually want content to start at the top of an element anyway. Still, there will be times when the height of a container grows beyond the height of a particular grid item or the grid container could be given an absolute height using pixels (i.e. height: 500px) and that height may be taller than the grid items it contains. The illustration below shows the effect of applying the values start, center, end, space-around, space-between, and space-evenly to the align-content property.

CSS Grid Layout
Steven D. Anderson, Ph.D.
James Madison University
School of Media Arts & Design
anderssd@jmu.edu