Basico
20/03/2020 ,21:21:51

LEARN CSS GRID

A guide to learning CSS grid by @jonsuh

CSS Grid is a powerful tool that allows for two-dimensional layouts to be created on the web. This guide was created as a resource to help you better understand and learn Grid, and was organized in a way pense hacerlo lo más sentido cuando aprendi ellos I thought made the most sense when learning it.

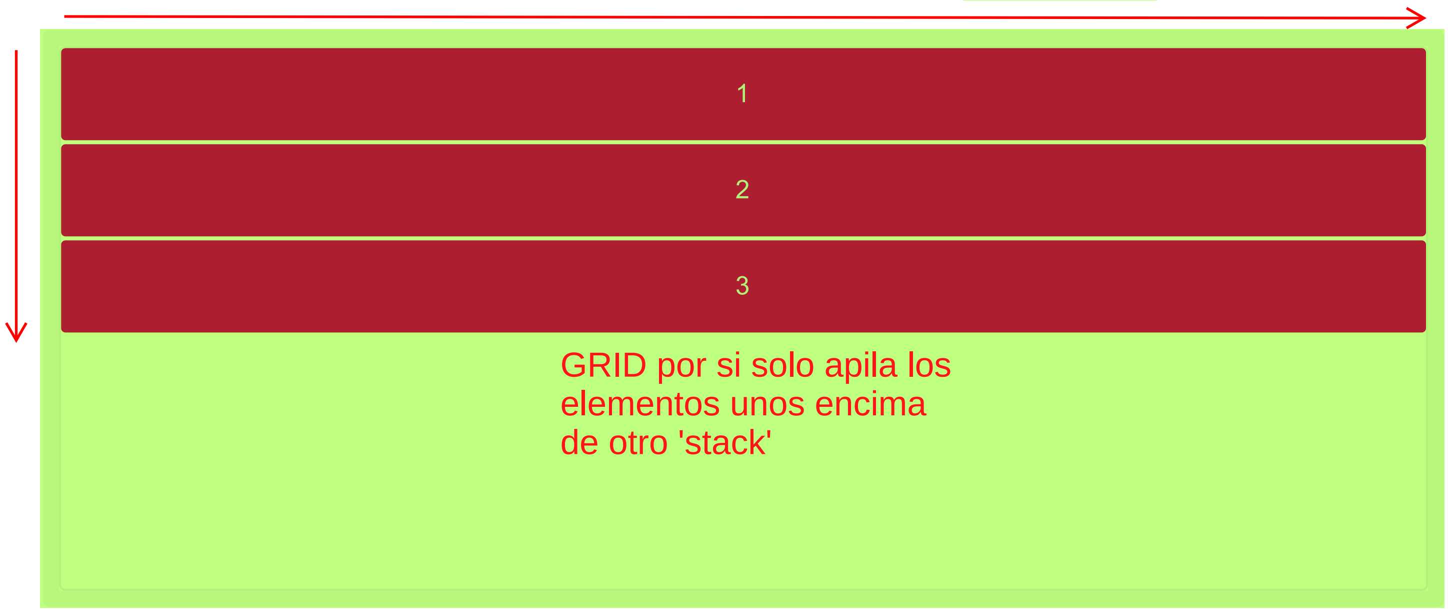
Table of Contents

Grid Container

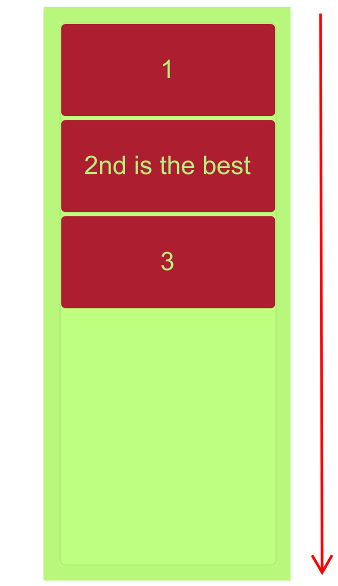
Crear un
Create a grid container by setting the `display` property con un valor de with a value of `grid` or `inline-grid`. Todo hijos directos de All direct children of grid containers se convierten en become grid items.

```
display: grid
```

estan colocados en filas por defecto y espaciados el maximo ancho de los
Grid items are placed in rows by default and span the full width of the grid container.


1
2
3
GRID por si solo apila los elementos unos encima de otro 'stack'

```
display: inline-grid
```



Se definen las estructuras mediante
las propiedades explicitas del GRID CONTAINER

Explicit Grid

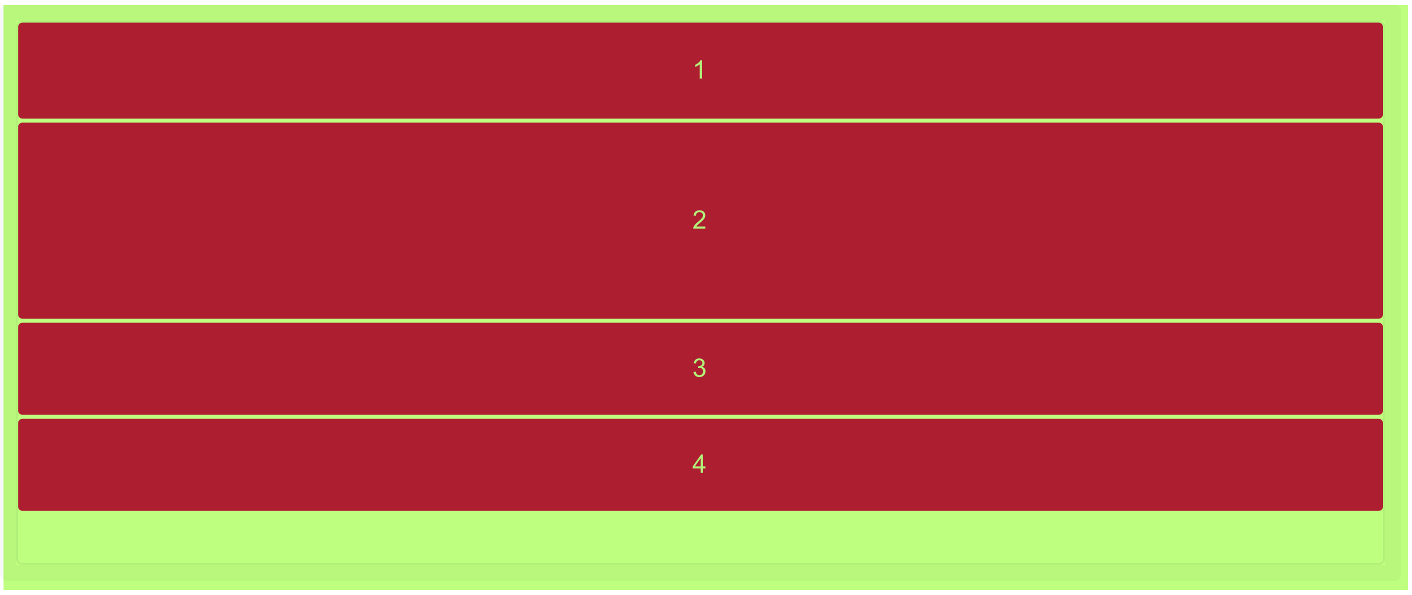
ajusta un grid para creando columnas y filas con el
Explicitly set a grid by creating columns and rows with the `grid-template-columns` and `grid-template-rows` properties.

```
grid-template-rows: 50px 100px
```

Una banda fila esta creada para cada valor especifico para
Tamaño valor de la banda puede estar algun
no negativo
valor longitud
A row track is created for each value specified for `grid-template-rows`. Track size values can be any non-negative, length value (`px`, `%`, `em`, etc.)

ha fijado el alto de
Items 1 and 2 have fixed heights of `50px` and `100px`.

Porque solo 2 filas bandas estan definidas , altura de elementos estan definidos por el contenido de cada
Because only 2 row tracks were defined, heights of items 3 and 4 are defined by the contents of each.



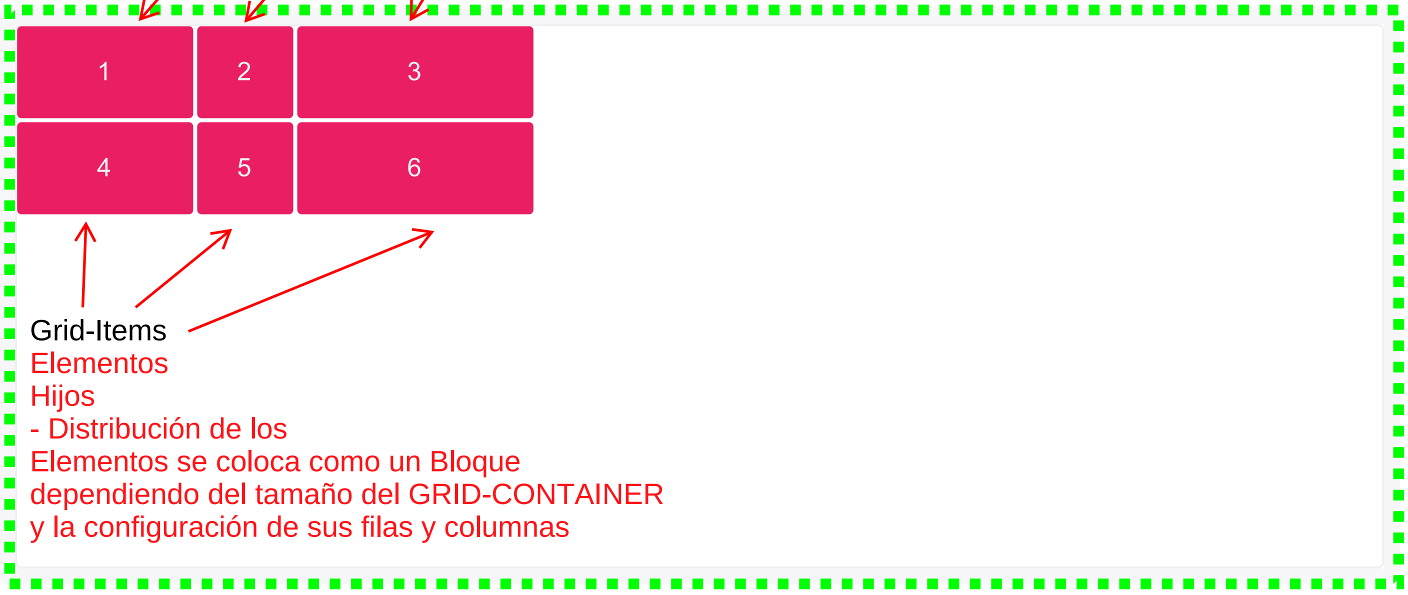
`grid-template-columns: 90px 50px 120px`

Como las filas una banda columna esta creada para cada valor especifico para
Like rows, a column track is created for each value specified for `grid-template-columns`.

Elemento 4,5 y 6 estaba colocado sobre una nueva fila banda porque solo 3 tamaño bandas columnas estaba definido y
Items 4, 5 and 6 were placed on a new row track because only 3 column track sizes were defined; and
porque ellos estaban colocados en bandas columnas 1 , 2 y 3 sus tamaños de columnas son equivalentes a element 1 , 2 y 3
because they were placed in column tracks 1, 2 and 3, their column sizes are equal to items 1, 2 and 3.

Element Grid 1 2 y 3 ha ajustado ancho de 90px 50px y 120px respectivamente
Grid items 1, 2 and 3 have fixed widths of `90px`, `50px` and `120px` respectively.

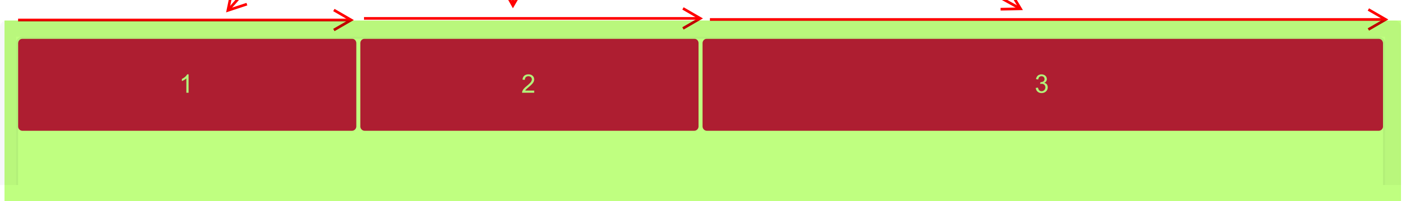
GRID-CONTENEDOR PRINCIPAL


1
2
3
4
5
6
Grid-Items
Elementos
Hijos
- Distribución de los
Elementos se coloca como un Bloque
dependiendo del tamaño del GRID-CONTAINER
y la configuración de sus filas y columnas

`grid-template-columns: 1fr 1fr 2fr`

ayuda crear flexibles bandas cuadriculas Ello representa una fraccion de el espacio disponible en el
The `fr` unit helps create flexible grid tracks. It represents a fraction of the available space in the grid container (works like Flexbox's unitless values).

In this example, items 1 and 2 take up the first two (of four) sections while item 3 takes up the last two.



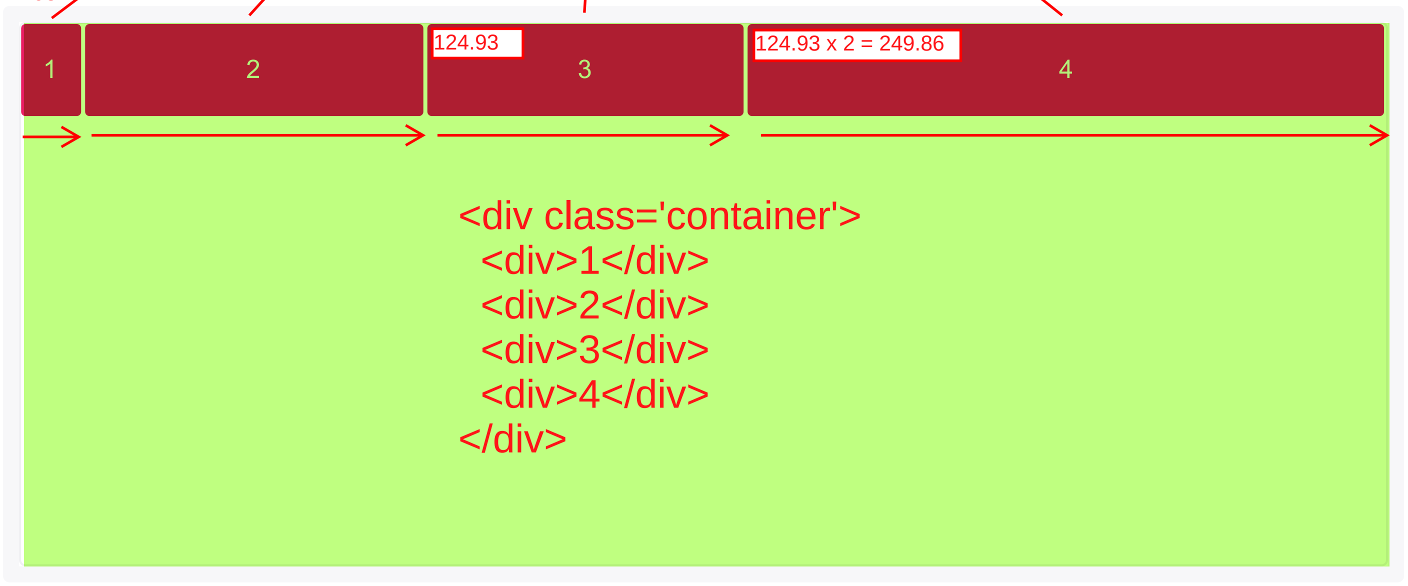
```
grid-template-columns: 3rem 25% 1fr 2fr
```

restante espacio cuando combinado con otros valores longitud
`fr` is calculated based on the remaining space when combined with other length values.

se restaria / desde el espacio disponible antes el tamaño de / es
In this example, `3rem` and `25%` would be subtracted from the available space before the size of `fr` is
calculado
calculated:

```
1fr = ((width of grid) - (3rem) - (25% of width of grid)) / 3
```

124.93 / 500px - 0.188 - 125px / 3



```
<div class='container'>
  <div>1</div>
  <div>2</div>
  <div>3</div>
  <div>4</div>
</div>
```

Minimum and Maximum Grid Track Sizes

El tamaño bandas puede ser definidas tener un / tamaño con la funcion
Tracks sizes can be defined to have a minimum and/or maximum size with the `minmax()` function.

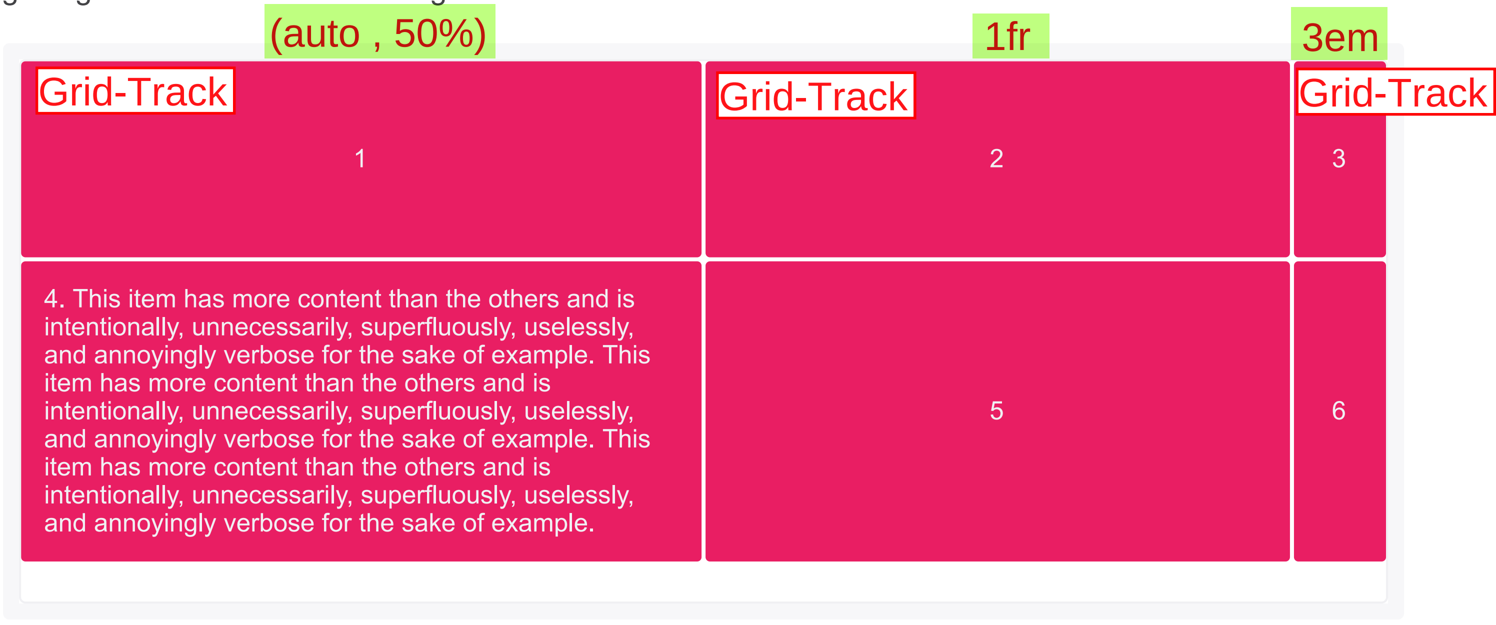
```
grid-template-rows:    minmax(100px, auto);
grid-template-columns: minmax(auto, 50%) 1fr 3em;
```

El / funcion acepta 2 argumentos / el 1º es el minimo tamaño de la franja/banda y el segundo el maximo tamaño
The `minmax()` function accepts 2 arguments: the first is the minimum size of the track and the second the
Junto a valores longitud / el valor puede tambien ser / el cual permite la banda/franjas crecer/estrecharse
maximum size. Alongside length values, the values can also be `auto`, which allows the track to

basado en el tamaño de el contenido
grow/stretch based on the size of the content.

En este ejemplo, la primera fila banda esta establecida para tener un minimo alto de 100px pero ello esta maximo tamaño de
In this example, the first row track is set to have a minimum height of `100px`, but its maximum size of
auto permitira la banda fila crecer ello el contenido es mayor que 100px
`auto` will allow the row track to grow it the content is taller than `100px`.

La primera columna de la banda tiene un minimo tamaño de 'auto' pero ello es maximo tamaño de prevenira ello desde
The first column track has a minimum size of `auto`, but its maximum size of `50%` will prevent it from
obteniendo no ancho como 50% de el grid container ancho
getting no wider than `50%` of the grid container width.

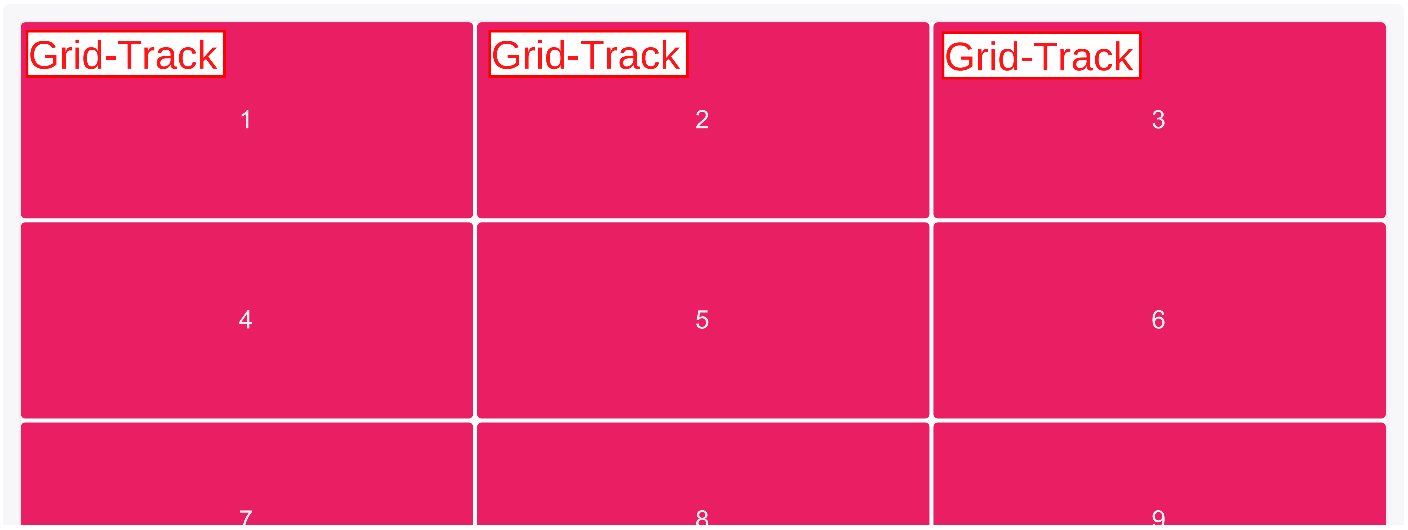

(auto , 50%)
1fr
3em
Grid-Track
Grid-Track
Grid-Track
1
2
3
4. This item has more content than the others and is intentionally, unnecessarily, superfluously, uselessly, and annoyingly verbose for the sake of example. This item has more content than the others and is intentionally, unnecessarily, superfluously, uselessly, and annoyingly verbose for the sake of example. This item has more content than the others and is intentionally, unnecessarily, superfluously, uselessly, and annoyingly verbose for the sake of example.
5
6

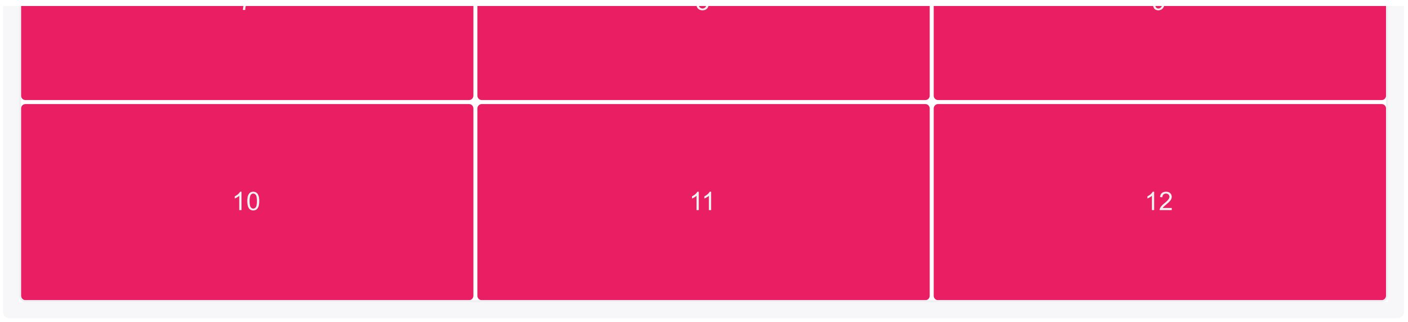
Repeating Grid Tracks

Define repetiendo grid banda/franja usando la Esto es util para los 'grid' con
Define repeating grid tracks using the `repeat()` notation. This is useful for grids with
elementos con tamaños equivalentes o algunos elementos
items with equal sizes or many items.

```
grid-template-rows:    repeat(4, 100px);
grid-template-columns: repeat(3, 1fr);
```

La notacion repeat() acepta 2 argumentos: 1º representa el numero de veces la definida banda/franja
The `repeat()` notation accepts 2 arguments: the first represents the number of times the defined tracks
deberia repetir y el segundo es la banda definida
should repeat, and the second is the track definition.





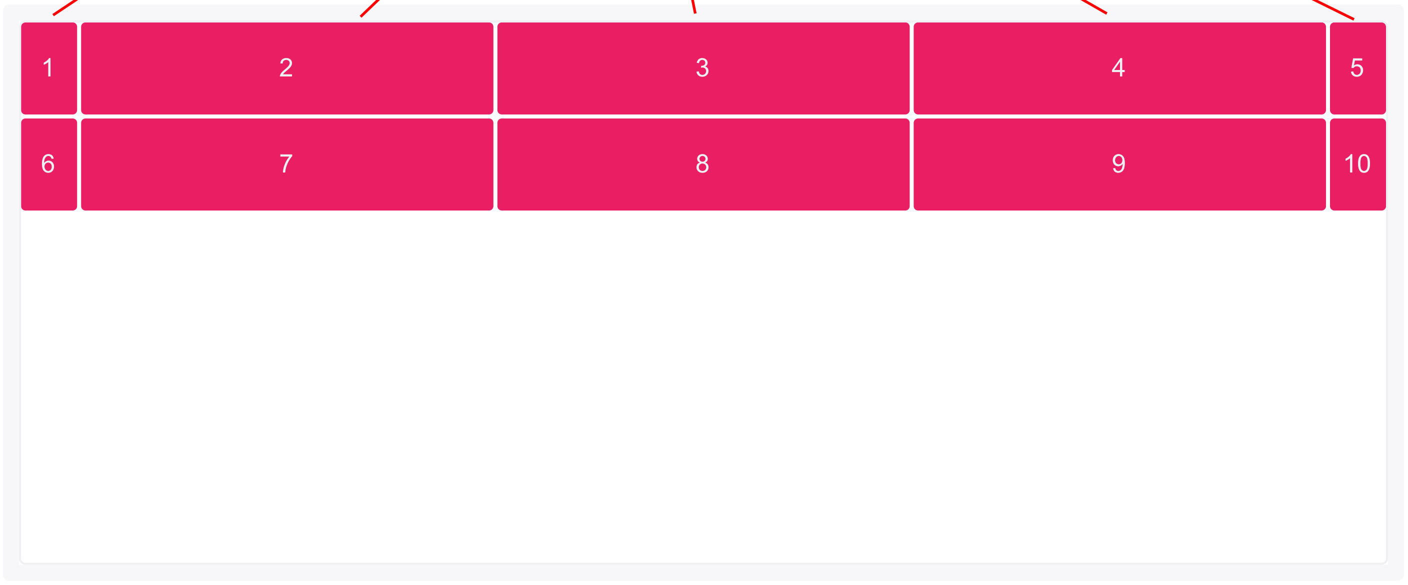
```
grid-template-columns: 30px repeat(3, 1fr) 30px
```

puede tambien ser usado dentro banda listada

`repeat()` can also be used within track listings.

ejemplo el 1° y la ultima columna banda tiene ancho de 30px y la 3 banda columna entre
creada por tiene ancho de cada

In this example, the first and last column tracks have widths of `30px`, and the 3 column tracks in between, created by `repeat()`, have widths of `1fr` each.



Grid Gaps (Gutters)

canaletas , cañerias

canaletas entre columnas
filas

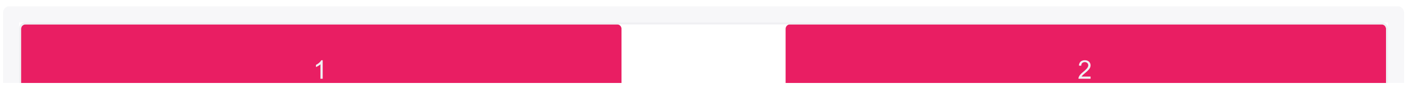
The `grid-column-gap` and `grid-row-gap` properties create gutters between columns and rows.

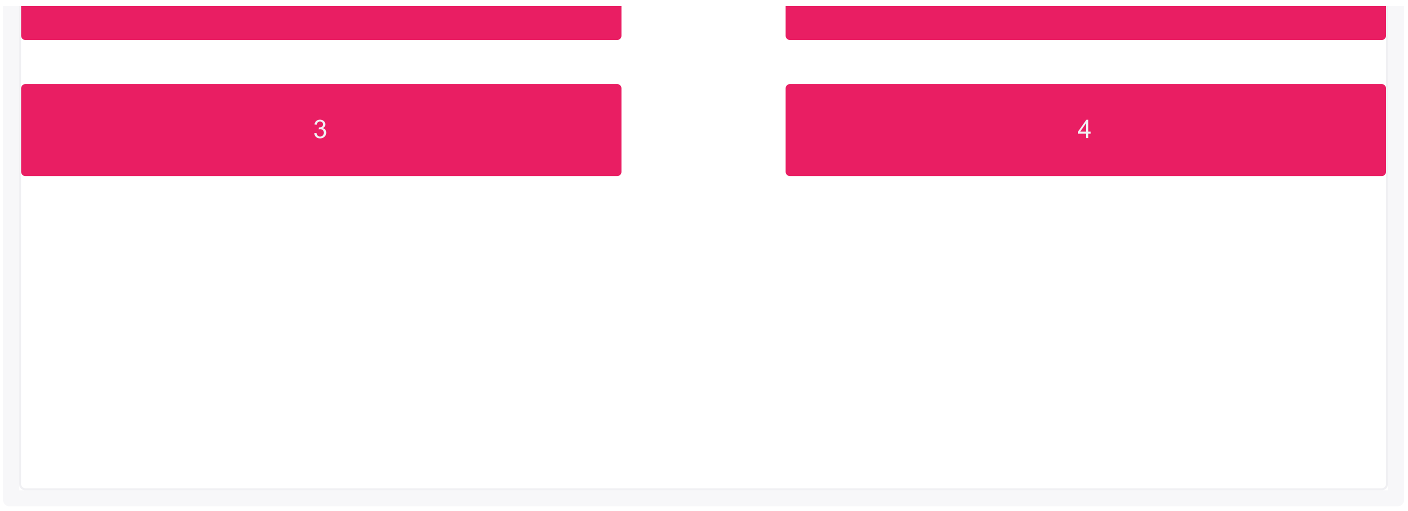
y no entre el borde de la grid container

Grid gaps are only created in between columns and rows, and not along the edge of the grid container 🙌 .

```
grid-row-gap:    20px;
grid-column-gap: 5rem;
```

Gap size values can be any non-negative, length value (`px`, `%`, `em`, etc.)

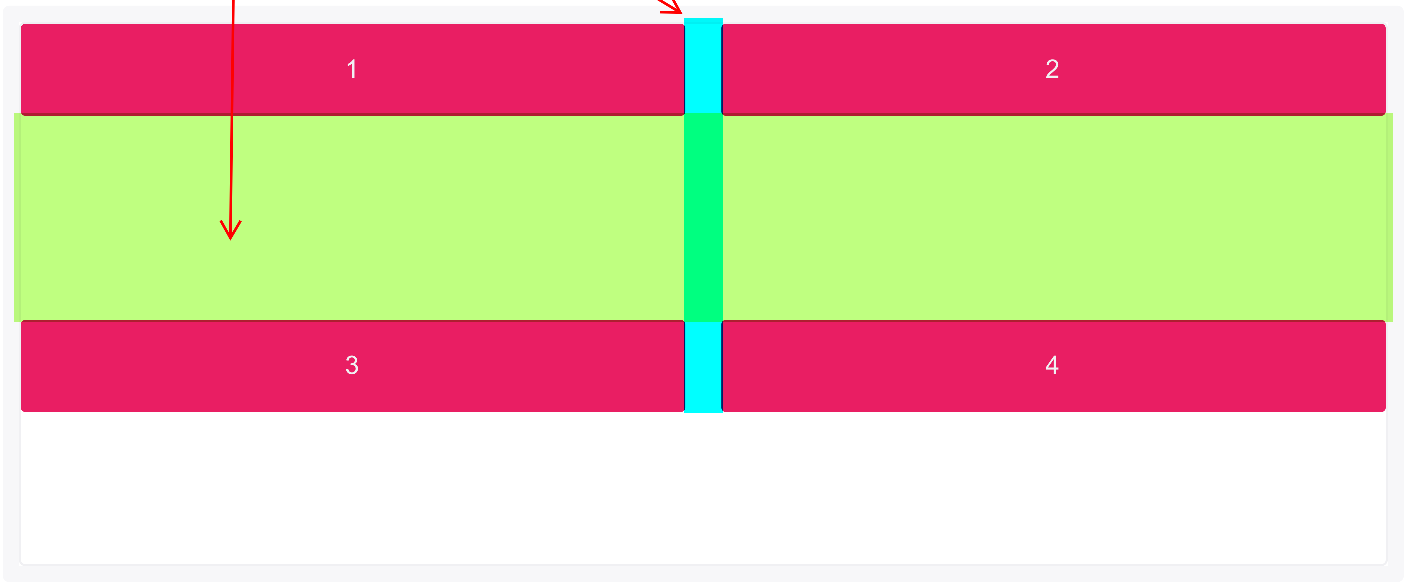




`grid-gap: 100px 1em`

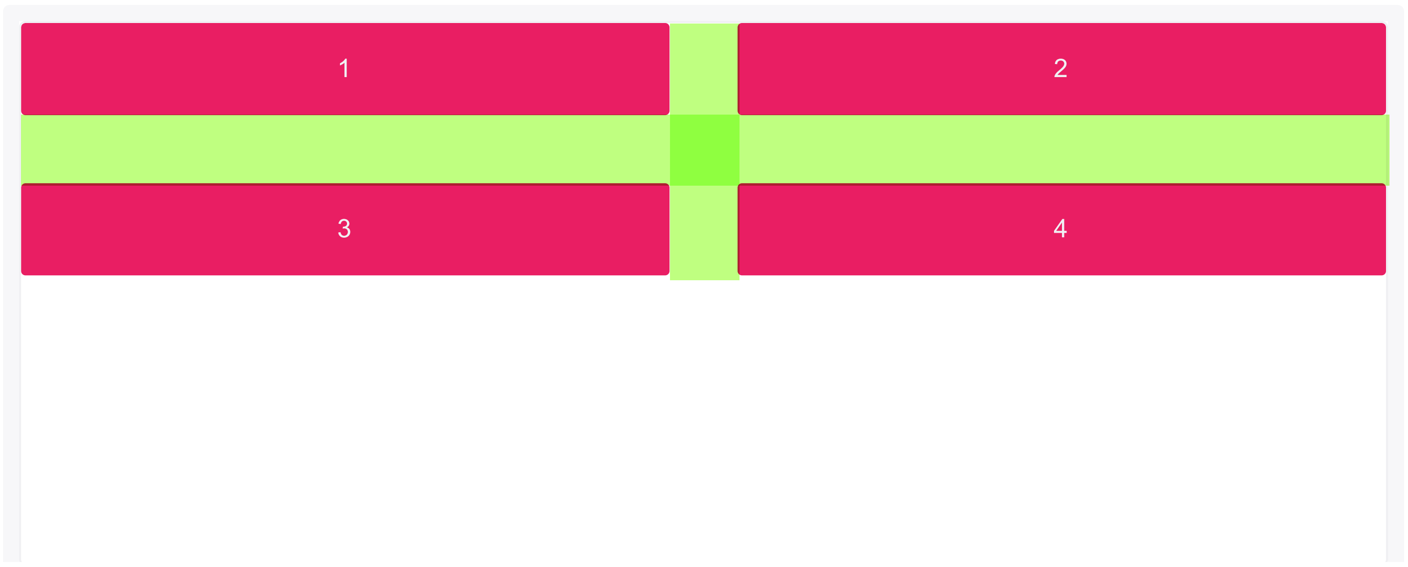
`grid-gap` is shorthand for `grid-row-gap` and `grid-column-gap`.

If two values are specified, the first represents `grid-row-gap` and the second `grid-column-gap`.



`grid-gap: 2rem`

One value sets equal row and column gaps.



ELEMENTOS HIJOS

Una vez creado el CONTENEDOR PADRE y la estructura principal estableciendo las medidas y las delimitaciones tenemos que posicionar los elementos dentro de la CUADRICULA de forma conjunta o individual (grid-template)

Positioning Items by Grid Line Numbers

son esencialmente lineas que representa el comienzo de , el final de o entre columnas

Grid lines are essentially lines that represent the start of, the end of, or between column

y bandas de filas

and row tracks.

Cada linea , comenzando desde el comienzo de la banda y en la direccion del grid , es numerado

Each line, starting from the start of the track and in the direction of the grid, is numbered

incrementalmente comenzado desde 1

incrementally starting from 1.

```
grid-row-start:    2;
grid-row-end:      3;
grid-column-start: 2;
grid-column-end:   3;
```

→ <div class='item-1'>1</div>

Esta resultado en estaba reposicionada por filas

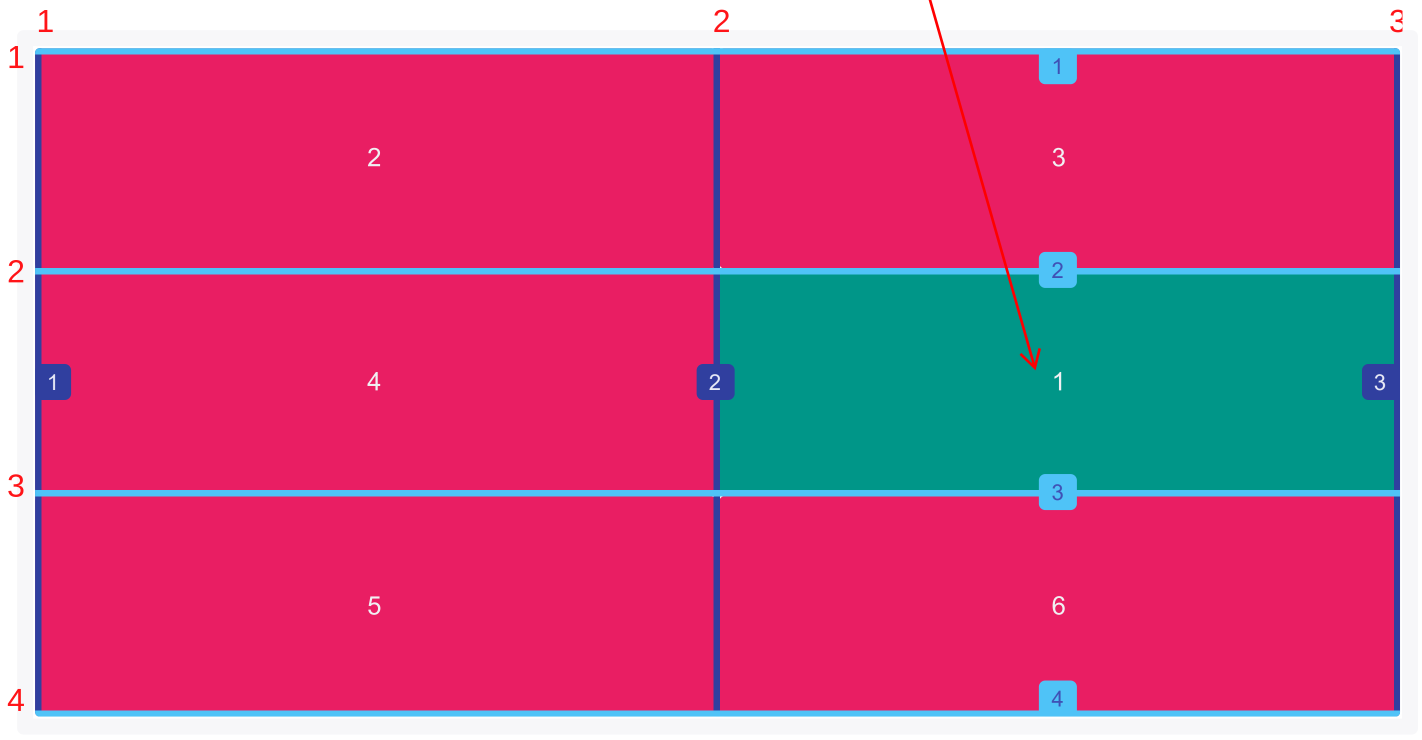
This 2-column by 3-row grid results in 3 column lines and 4 row lines. Item 1 was repositioned by row and

y columnas numero de lineas

column line numbers.

Si un elemento expande solo una fila o una columna no es necesario

If an item spans only one row or column, `grid-row/column-end` is not necessary.



```
grid-row:    2;
grid-column: 3 / 4;
```

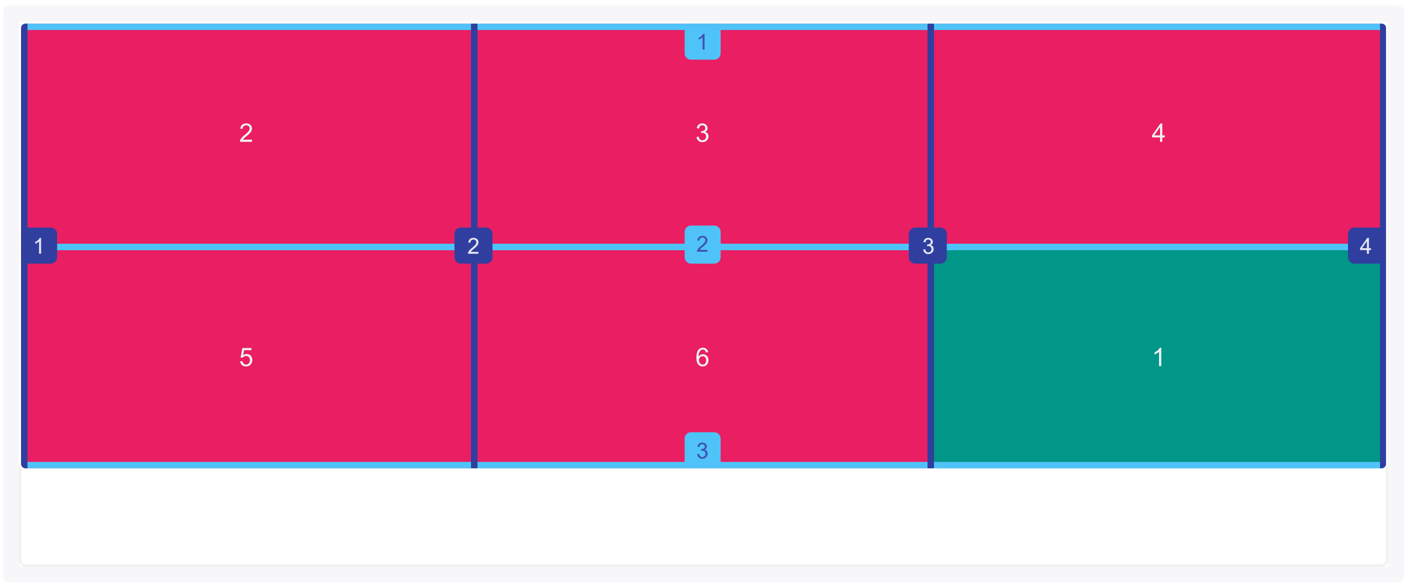
principio/final

`grid-row` is shorthand for `grid-row-start` and `grid-row-end`.

`grid-column` is shorthand for `grid-column-start` and `grid-column-end`.

Si un valor es proporcionado , ello especifica
If one value is provided, it specifies `grid-row/column-start`.

Si 2 valores estan especificados , el primer valor corresponde a y el segundo
If two values are specified, the first value corresponds to `grid-row/column-start` and the second
y debe de estar separado por una seguida barra
`grid-row/column-end`, and must be separated by a forward slash `/`.

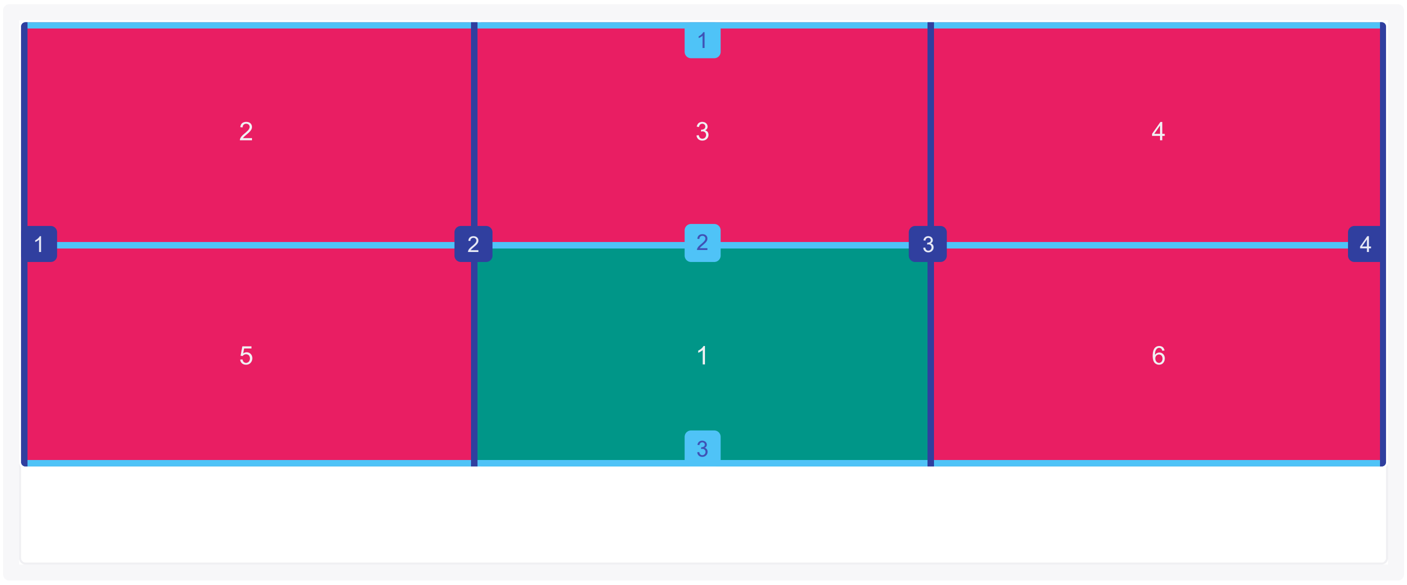


row start column start row end column end

```
grid-area: 2 / 2 / 3 / 3
```

`grid-area` is shorthand for `grid-row-start`, `grid-column-start`, `grid-row-end` and `grid-column-end`.

Si cuatro valores son especificados el primero corresponde a
If four values are specified, the first corresponds to `grid-row-start`, the second `grid-column-start`, the third `grid-row-end` and the fourth `grid-column-end`.

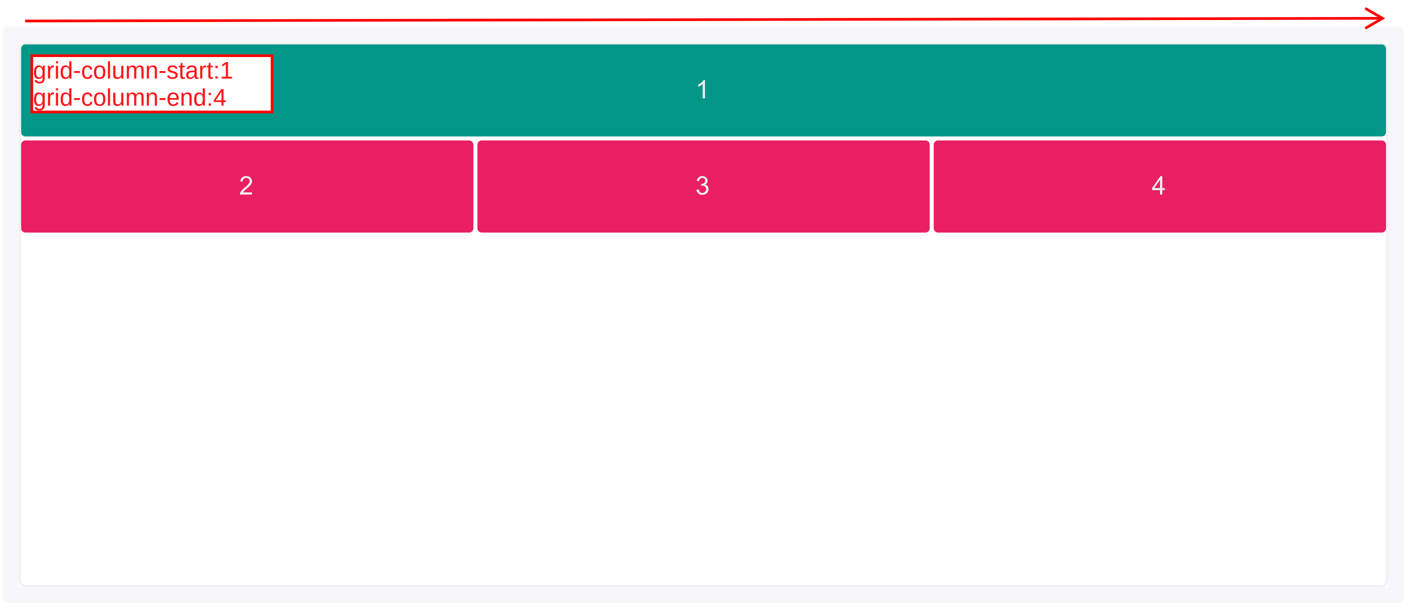


Espaciando Elementos cruzados filas y columnas
Spanning Items Across Rows and Columns

espacian solo una banda columna y fila por defecto pero puede espaciar multiples filas
Grid items span only one column and row track by default, but can span multiple row
y o bandas columnas usando la mismas propiedades a
and/or column tracks using the same properties to position them.

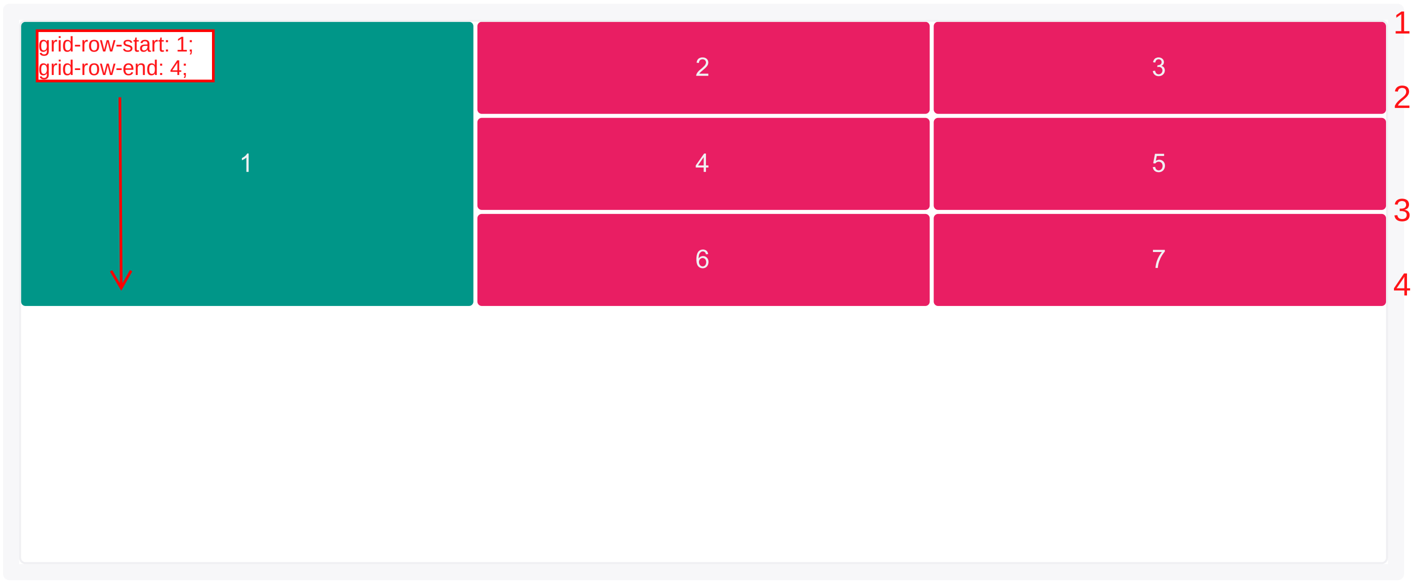
```
grid-column-start: 1;
grid-column-end:   4;
```

Establecer un grid item a espaciar mas como una banda columna por configuracion a un numero linea
Set a grid item to span more than one column track by setting `grid-column-end` to a column line
columna que esta mas que uno columna esperando desde
number that is more than one column away from `grid-column-start`.



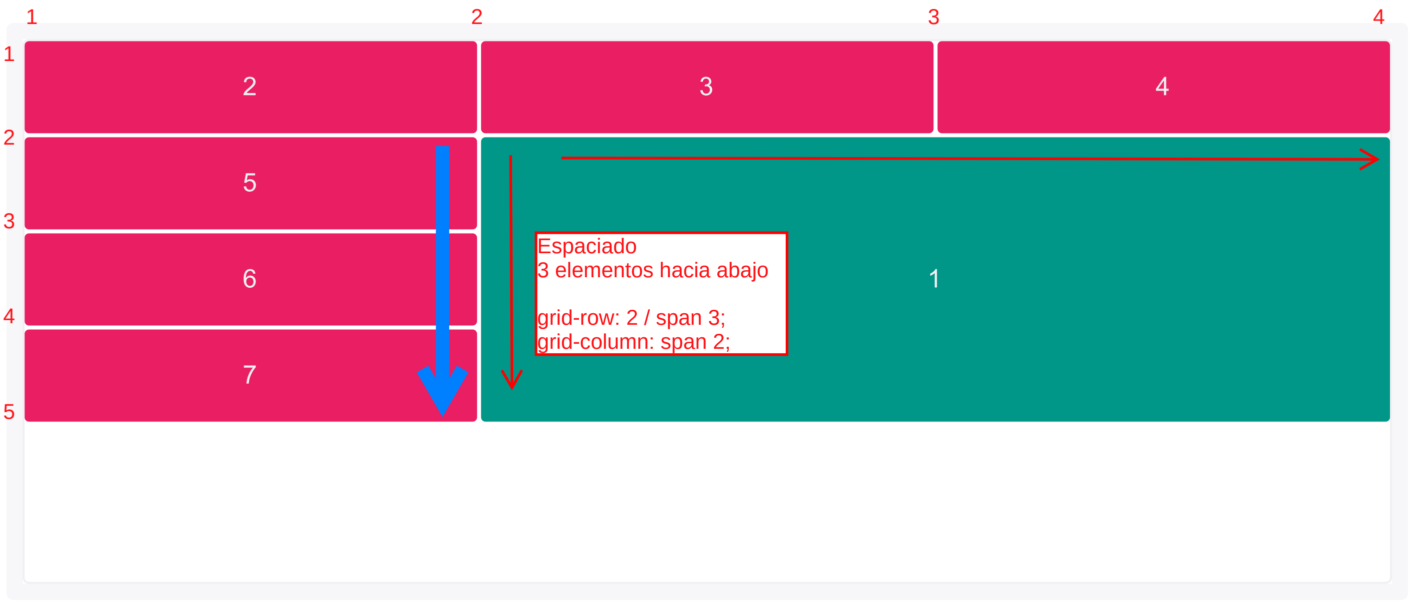
```
grid-row-start: 1;
grid-row-end:   4;
```

puede tambien espaciar cruces multiple bandas filas ajustando a mas que una banda de la fila espera
Grid items can also span across multiple row tracks by setting `grid-row-end` to more than one row track away.



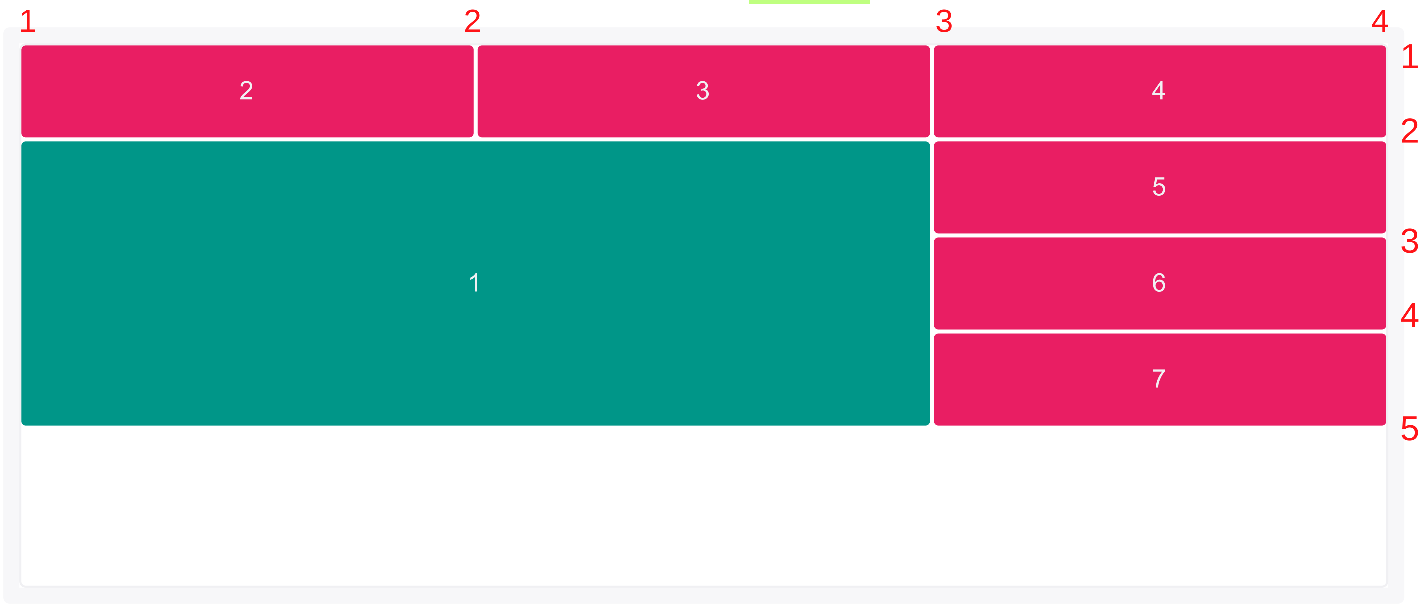
```
grid-row:    2 / 5;
grid-column: 2 / 4;
```

Abreviatura propiedades puede tambien ser usado a posicion y espaciado mas
Shorthand properties `grid-row` and `grid-column` can also be used to position and span grid items more
que una fila o columna
than one row or column.



```
grid-row:    2 / span 3;
grid-column: span 2;
```

The keyword span, followed by the # of columns or rows to span, can also be used.



Nombrando Grid Lineas
Naming Grid Lines

puede ser nombrado cuando definiendo el grid con el
Grid lines can be named when defining the grid with the `grid-template-rows` and `grid-template-columns` properties. Nombre lineas puedes entonce estar referenciadas posiconar
Line names can then be referenced to position grid items.

```
grid-template-rows:    [row-1-start] 1fr [row-2-start] 1fr [row-2-end];
grid-template-columns: [col-1-start] 1fr [col-2-start] 1fr [col-3-start] 1fr [col-3-end];
```

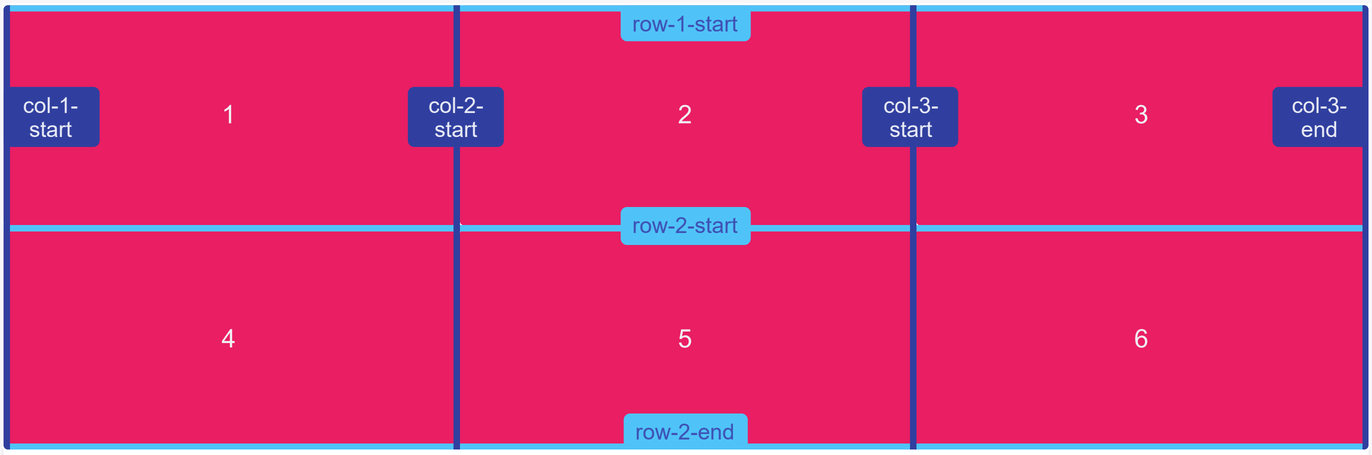
Assign names to grid lines when defining your grid with the `grid-template-rows` and `grid-template-columns` properties.

En linea nombres , evitar palabras clave que aparecen en la especificacion no causar confusión

In line names, avoid keywords that appear in the specification (e.g. `span`) to not cause confusion.

Asignando nombre de lineas deben estar envolvidas en brackets cuadrados y colocando relativo a los grid bandas

Assigned line names must be wrapped in square brackets `[name-of-line]` and placed relative to the grid tracks.



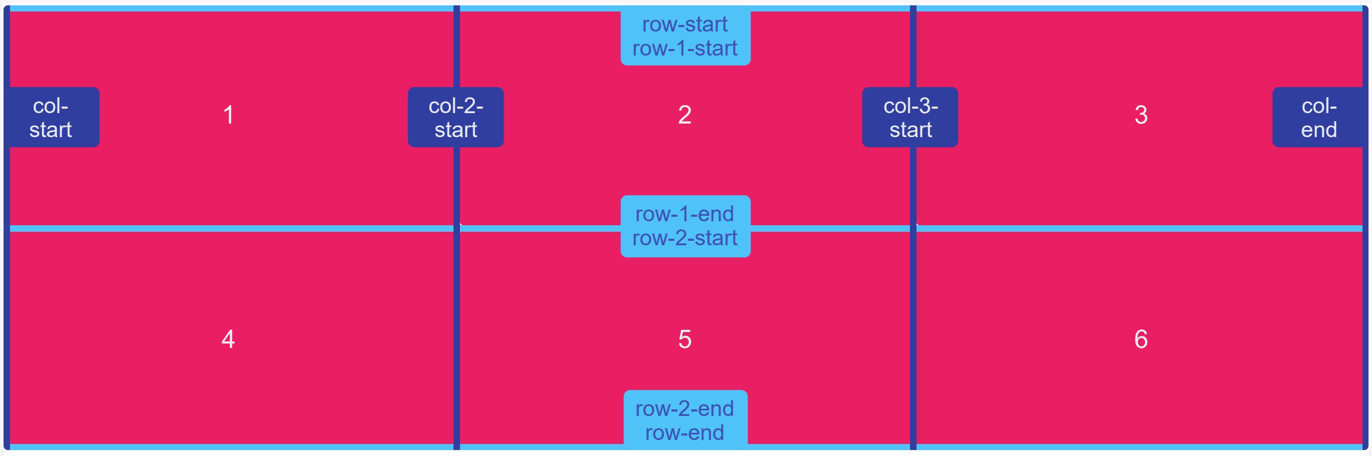
```
grid-template-rows:    [row-start row-1-start] 1fr [row-1-end row-2-start] 1fr [row-2-end row-end];
grid-template-columns: [col-start] 1fr [col-2-start] 1fr [col-3-start] 1fr [col-end];
```

Multiples nombres puedes estar asignados a para añadir nombres dentro cuadrados corchetes y separando cada con un espacio en blanco

Multiple names can be assigned to grid lines by adding names within square brackets and separating each with a whitespace.

Cada nombre linea puede entonces estar referenciado cuando

Each line name can then be referenced when positioning grid items by line names.



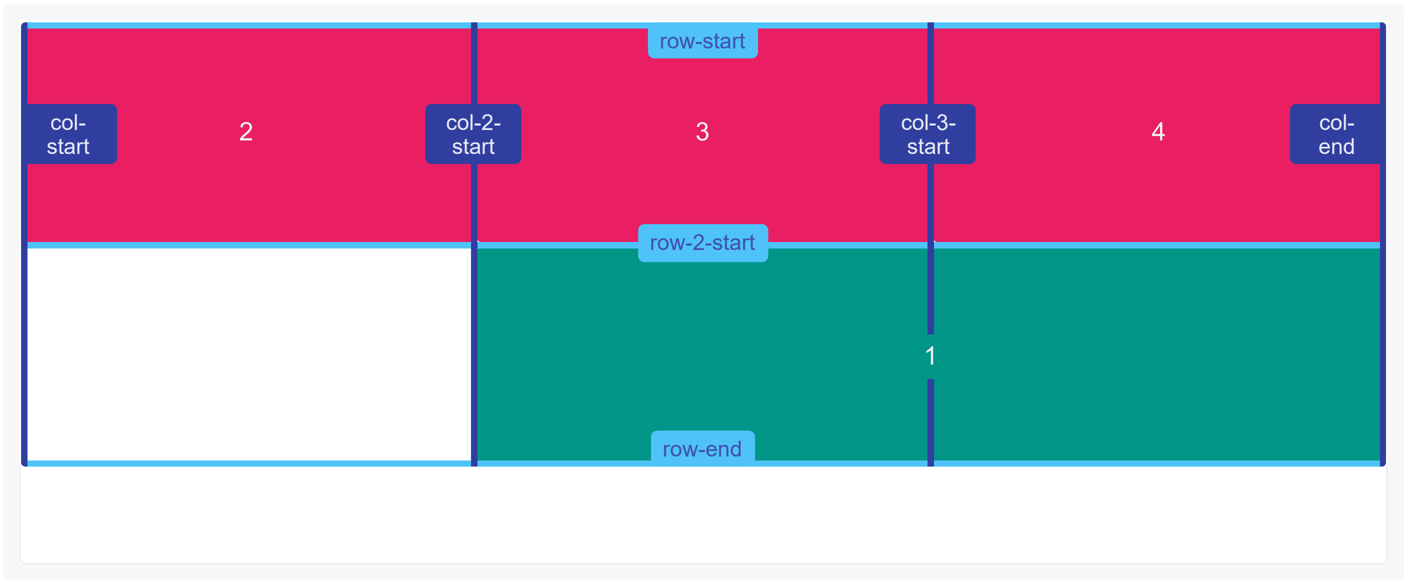
```
grid-template-rows:    [row-start row-1-start] 1fr [row-1-end row-2-start] 1fr [row-2-end row-end];
grid-template-columns: [col-start] 1fr [col-2-start] 1fr [col-3-start] 1fr [col-end];
```

Positioning Items by Line Names

With named grid lines, items can be positioned by line names and numbers.

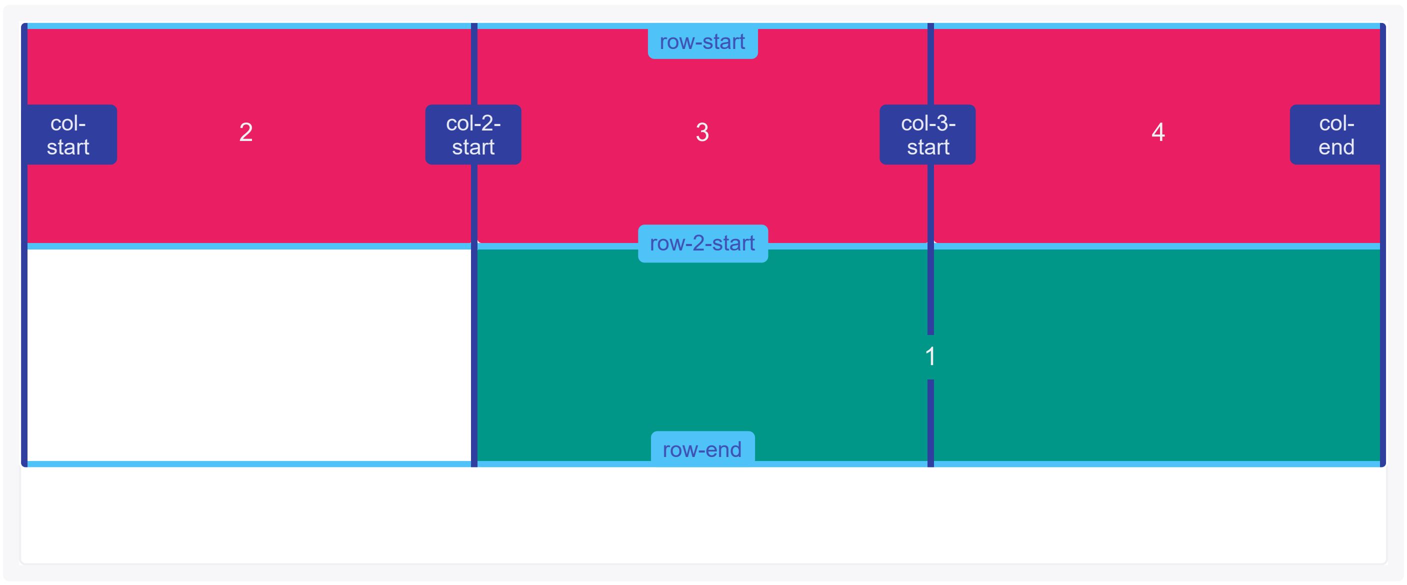
```
grid-row-start:    row-2-start;
grid-row-end:      row-end;
grid-column-start: col-2-start;
grid-column-end:   col-end;
```

Referenced line names should not be wrapped in square brackets.



```
grid-row:    row-2-start / row-end;
grid-column: col-2-start / col-end;
```

`grid-row` and `grid-column` shorthand properties also support the use of grid line names when positioning items.



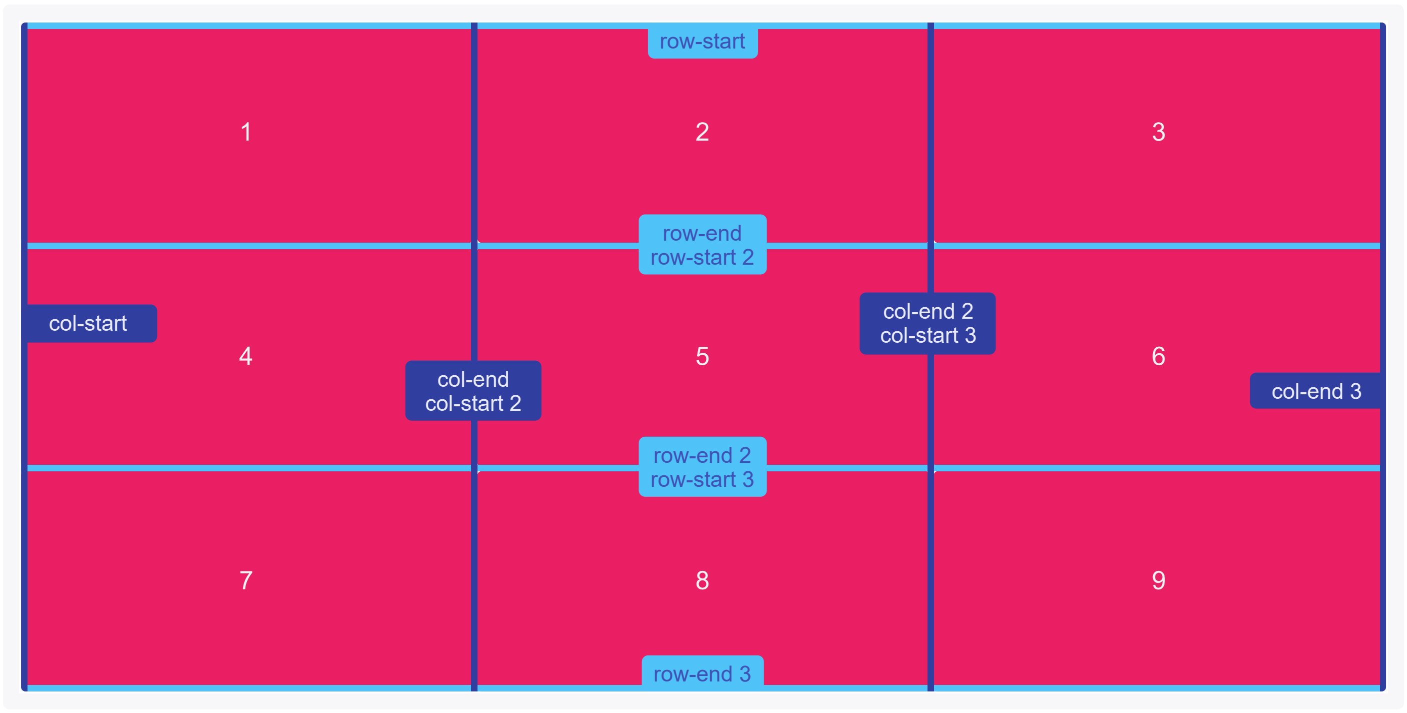
Naming and Positioning Items by Grid Lines with the Same Name

Lines can be assigned the same name with the `repeat()` function. This can save you time from having to name each line in track definitions.

```
grid-template-rows: repeat(3, [row-start] 1fr [row-end]);
grid-template-columns: repeat(3, [col-start] 1fr [col-end]);
```

Line name assignments can also be included within the `repeat()` function. This results in multiple grid lines with the same names.

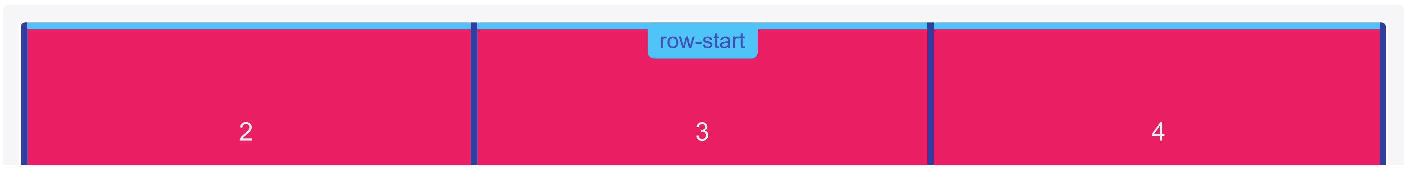
Lines with the same name are also assigned the a line's position/name's occurrence number, which allows it to be uniquely identified from another line with the same name.

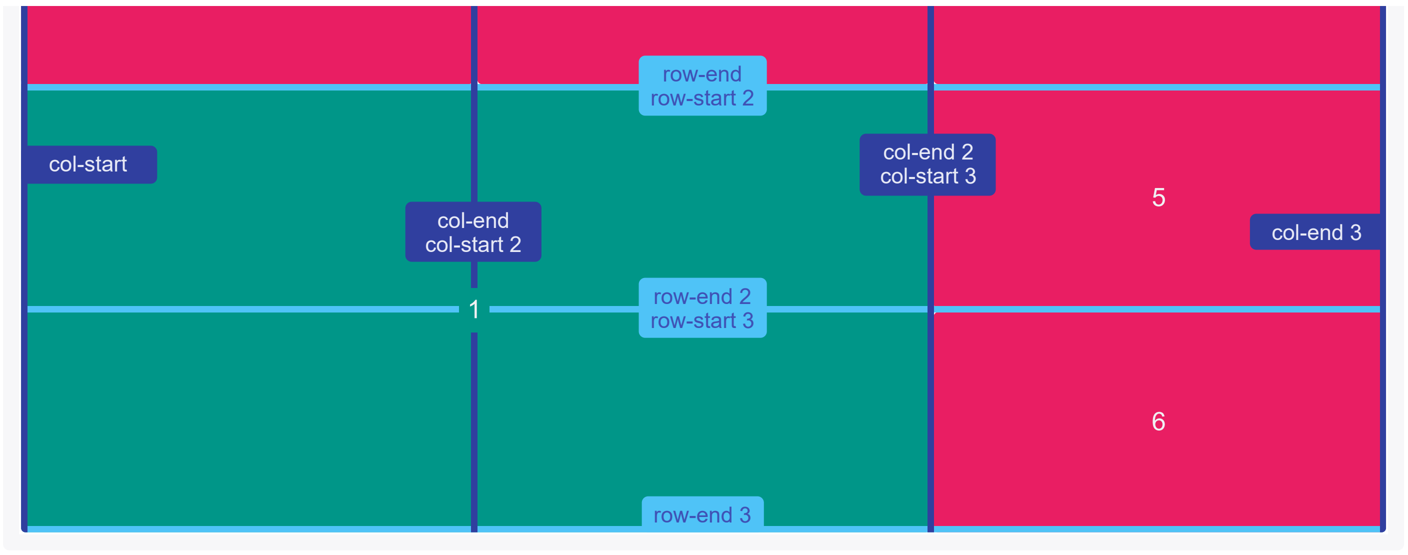


```
grid-row:    row-start 2 / row-end 3;
grid-column: col-start / col-start 3;
```

To position items by lines with the same name, reference the line's name and position/name's occurrence number—the name and number should be separated by a whitespace.

In this example, item 1's row position starts at the 2nd grid line named `row-start` and ends at the 3rd grid line named `row-end`; and its column position starts at the 1st grid line named `col-start` and ends at the 3rd grid line named `col-start`.





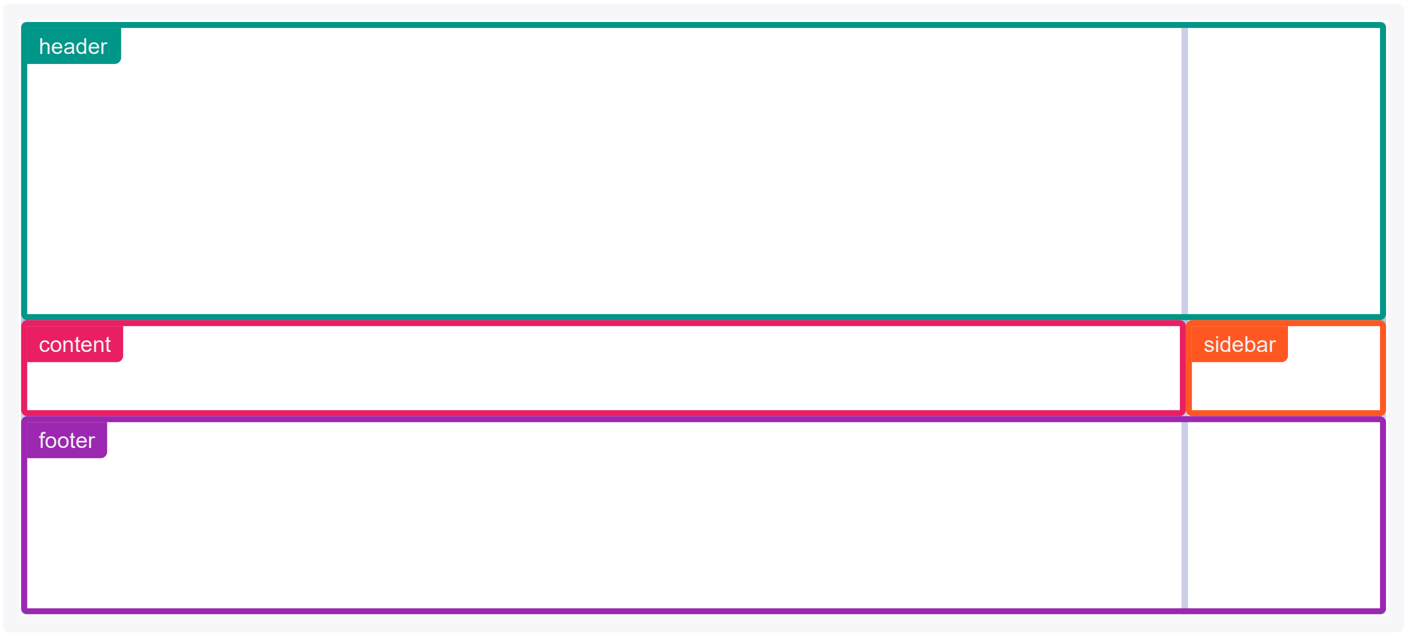
Naming and Positioning Items by Grid Areas

Como nombre puede tambien ser nombrado con la propiedad
Like grid line names, grid areas can also be named with the `grid-template-areas`
nombres puede entonces ser referenciado a posicion
property. Names can then be referenced to position grid items.

```
grid-template-areas:  "header header"
                      "content sidebar"
                      "footer footer";
grid-template-rows:    150px 1fr 100px;
grid-template-columns: 1fr 200px;
```

Establecer de nombres deberian estar envueltos en singular o doble consultas y cada nombre separado por un
Sets of names should be surrounded in single or double quotes, and each name separated by a
espacio en blanco
whitespace.

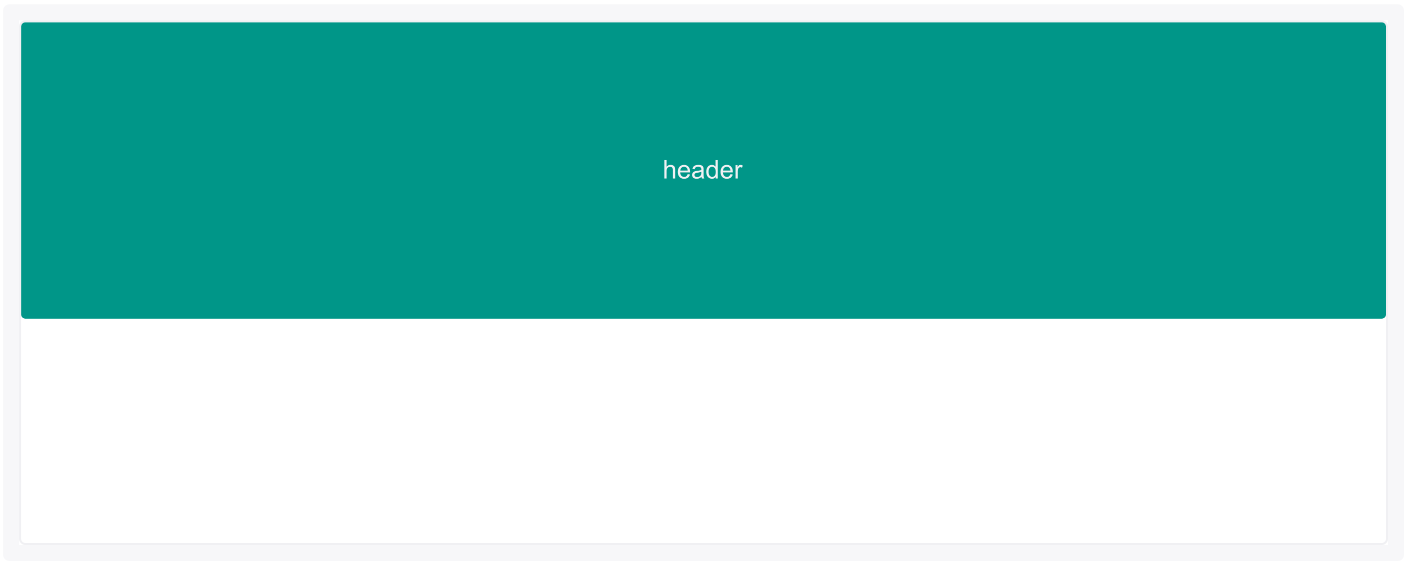
Cada establecido nombre define una fila y cada nombre define una columna
Each set of names defines a row, and each name defines a column.



```
grid-row-start:    header;
grid-row-end:      header;
grid-column-start: header;
grid-column-end:   header;
```

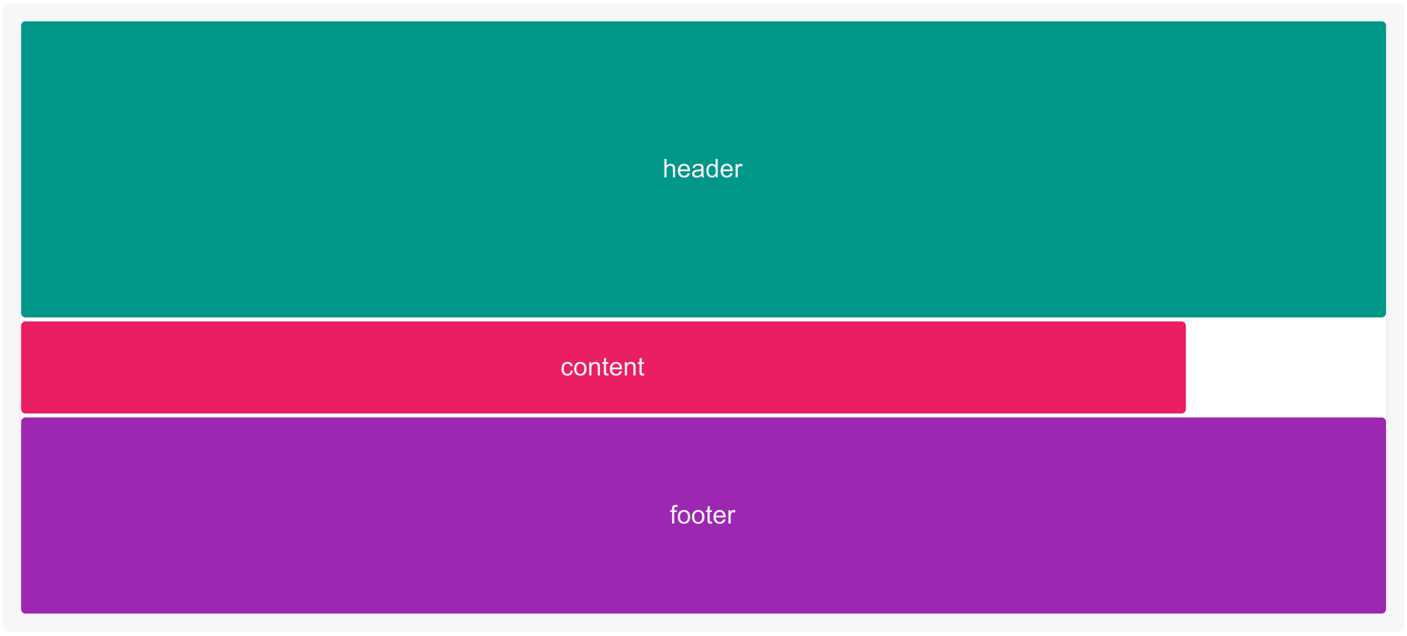
nombres puede ser referenciados por la misma propiedad a posicion grid items

Grid area names can be referenced by the same properties to position grid items: `grid-row-start`, `grid-row-end`, `grid-column-start`, and `grid-column-end`.



```
grid-row:    footer;
grid-column: footer;
```

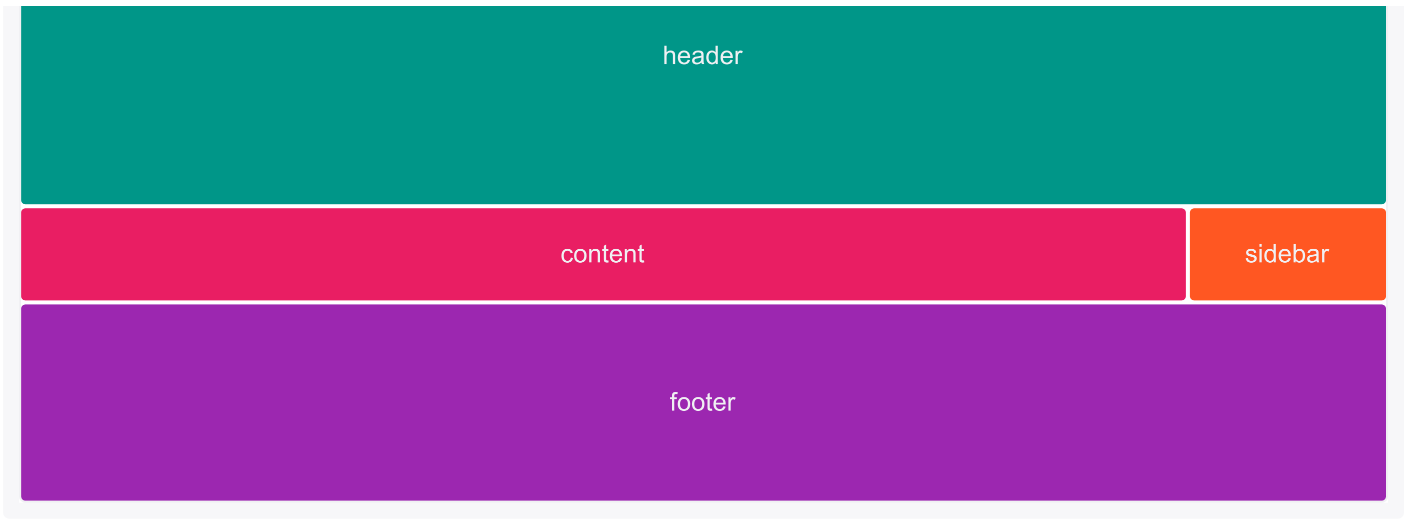
The `grid-row` and `grid-column` shorthand properties can also reference grid area names.



```
grid-area: sidebar;
```

propiedad puede tambien ser usada referenciar nombre grid area

The `grid-area` shorthand property can also be used to reference grid area names.



Implicit Grid

Un grid implicito es creado cuando un necesita posicionar elementos fuera de el explicito grid

An implicit grid is created when a grid needs to position items outside of the explicit grid

porque no hay bastante espacio para los elementos en el explicito definido bandas/franjas o tu decides

because there isn't enough space for items in the explicitly defined tracks or you decide to

posicionar alguna cosa fuera de el explicito grid aquellos elementos son cuando autocolocan en

position something outside of the explicit grid. Those items are then auto-placed in the

grid implicito

implicit grid.

El grid implicito puede ser definido usando el

The implicit grid can be defined using the `grid-auto-rows`, `grid-auto-columns`, and `grid-auto-flow` properties.

Solo crea explicitamente 1 fila

```
grid-template-rows:    70px;
grid-template-columns: repeat(2, 1fr);
grid-auto-rows:        140px;
```

En este ejemplo nosotros tenemos solo definido 1 fila banda/franja , por eso

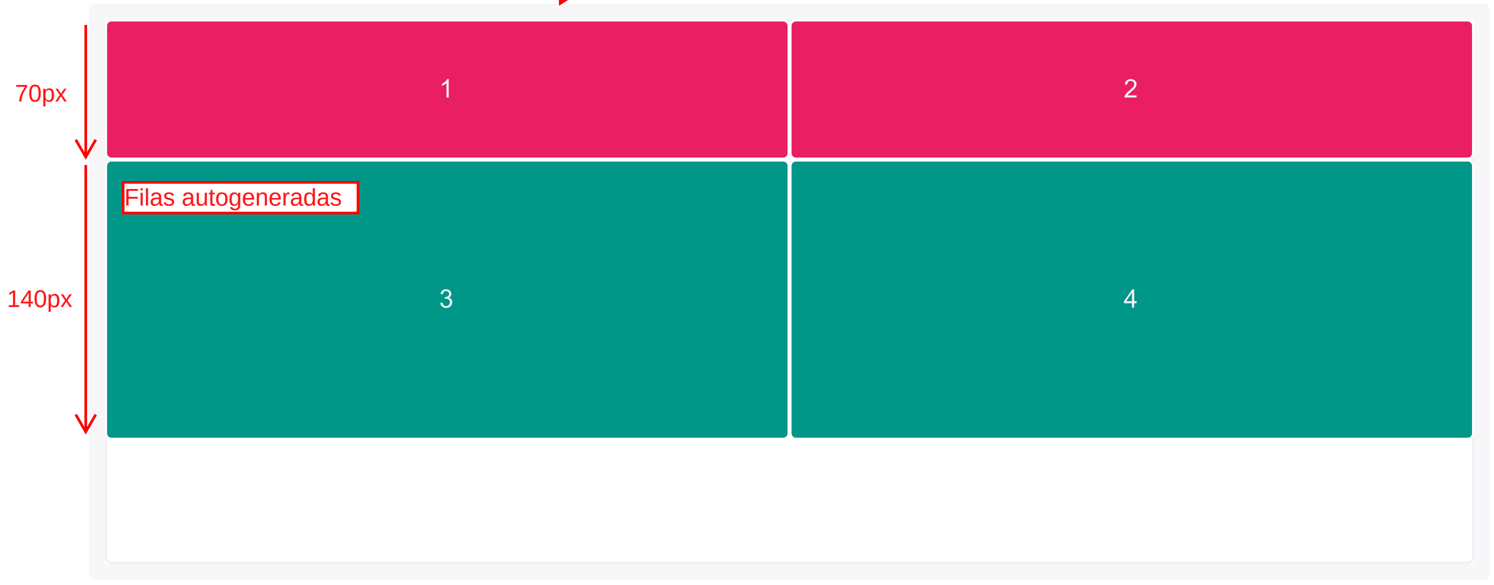
In this example we've only defined one row track, therefore grid items 1 and 2 are `70px` tall.

Una segunda banda fila fue auto-creada crear estancia para elementos 3 y 4 define la banda fila

A second row track was auto-created to make room for items 3 and 4. `grid-auto-rows` defines the row

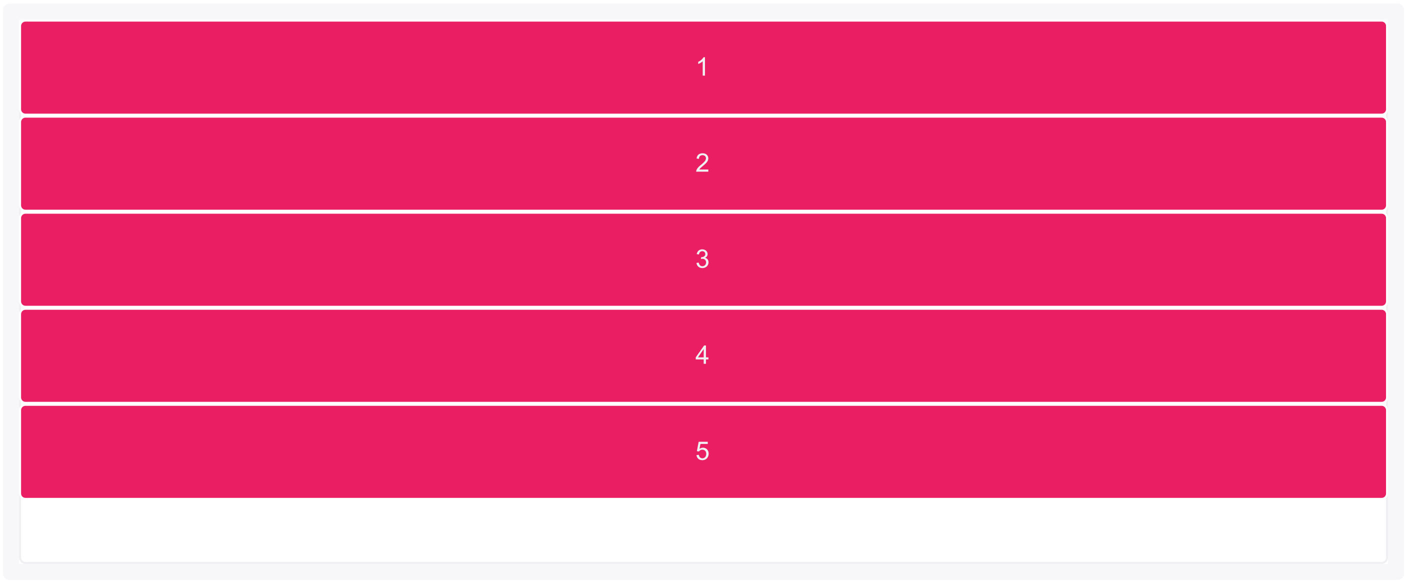
tamaño en la grid implicito , el cual es reflejado por los 140px de alto de elementos 3 y 4

track sizes in the implicit grid, which is reflected by the the `140px` heights of items 3 and 4.



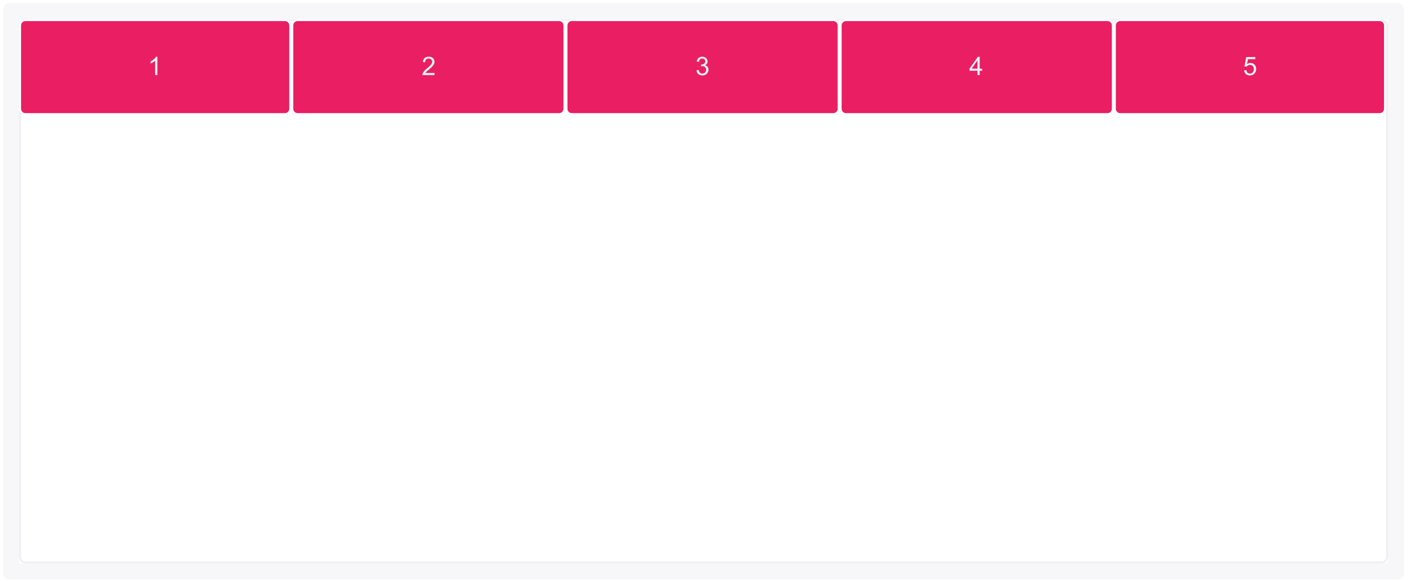
```
grid-auto-flow: row
```

Por defecto flujo (dirección) de una grid es fila
The default flow (direction) of a grid is `row`.



```
grid-auto-flow: column
```

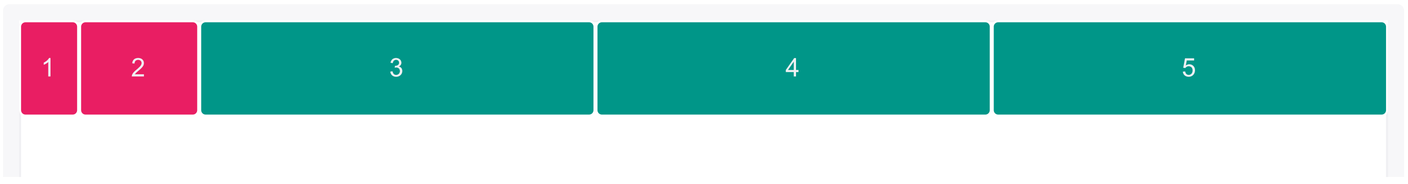
Un grid esta flujo puede ser cambiado a columna
A grid's flow can be changed to `column`.



```
                         column1  column2
grid-template-columns:   30px     60px;
grid-auto-flow:          column;
grid-auto-columns:       1fr;
```

nosotros tenemos solo definido el tamaño de los 1º 2 banda columnas ancho y elemento
In this example, we've only defined the sizes of the first two column tracks—item 1 is `30px` wide and item 2, `60px`.

Bandas columnas estan en el implicito crear estancias para elementos y tamaños bandas estan definidos por
Column tracks are auto-created in the implicit grid to make room for items 3, 4 and 5; and track sizes are defined by `grid-auto-columns`.



Implicitly Named Grid Areas

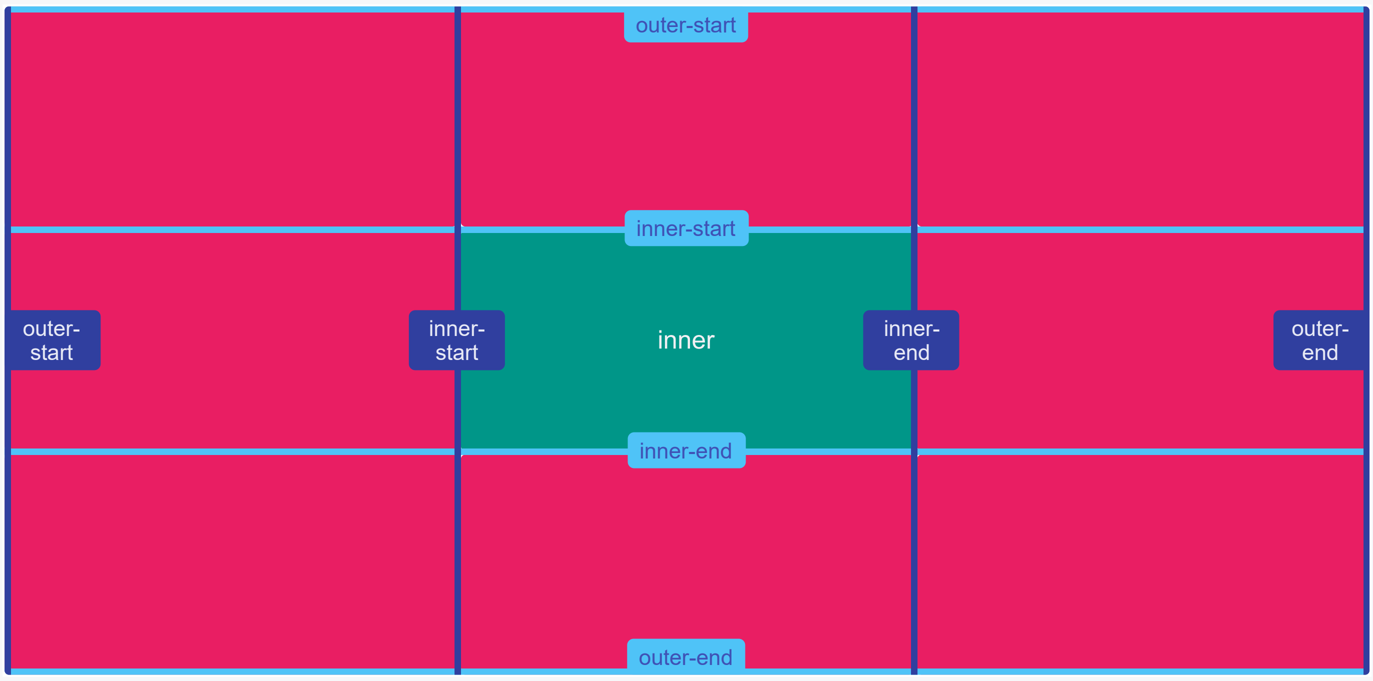
Grid lines can generally be named whatever you'd like, but assigning names ending in `-start` and `-end comes` with added benefits—they implicitly create named grid areas, which can be referenced for positioning.

```
grid-template-rows:    [outer-start] 1fr [inner-start] 1fr [inner-end] 1fr [out
grid-template-columns: [outer-start] 1fr [inner-start] 1fr [inner-end] 1fr [out
```

In this example, both rows and columns have `inner-start` and `inner-end` lines, which implicitly assigns the grid area's name as `inner`.

```
grid-area: inner
```

Grid items can then be positioned by the grid area name as opposed to line names.

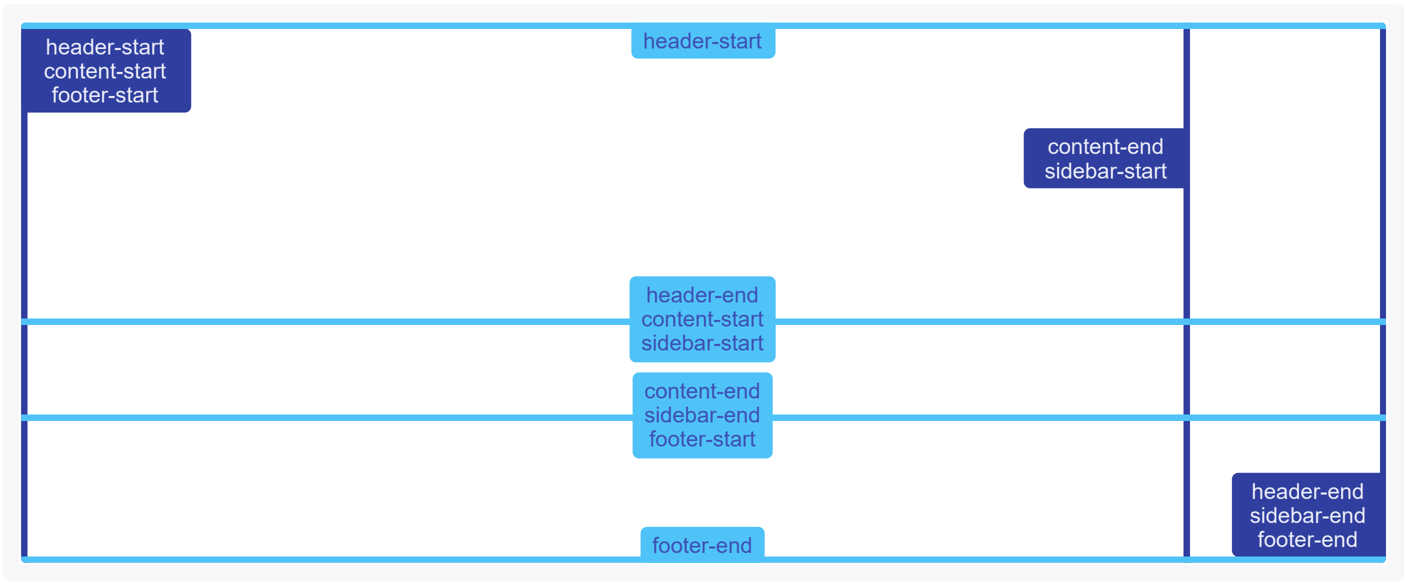


Implicitly Named Grid Lines

Implicitly named grid lines work in reverse to implicitly named grid areas—naming grid areas implicitly assigns names to grid lines.

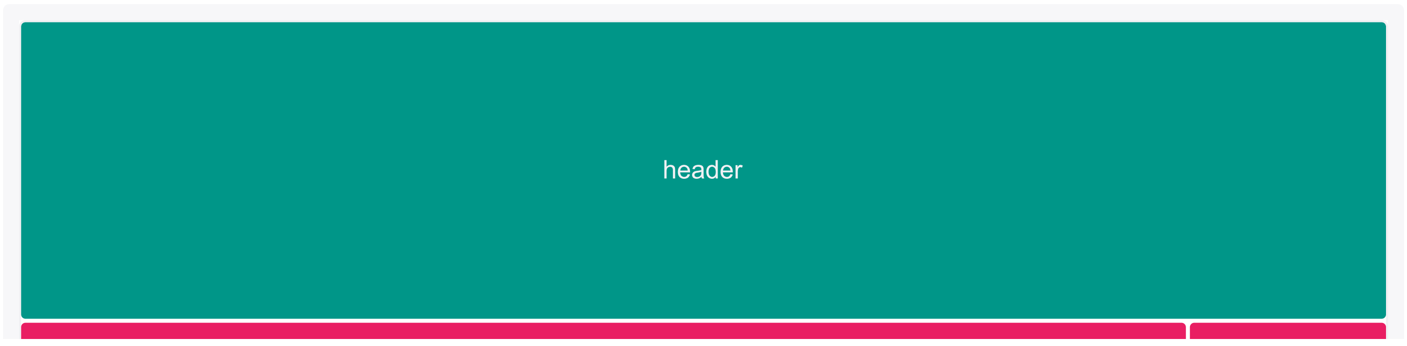
```
grid-template-areas:   "header header"
                       "content sidebar"
                       "footer footer";
grid-template-rows:    80px 1fr 40px;
grid-template-columns: 1fr 200px;
```

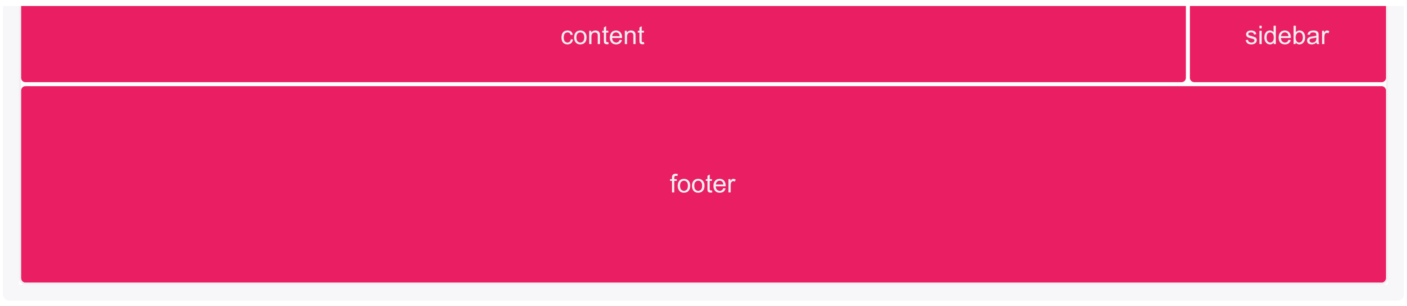
Named grid areas will implicitly name the grid lines along the edges of the area. Those grid lines will be named based on the area name and suffixed with `-start` or `-end`.



```
grid-row-start:    header-start;
grid-row-end:      content-start;
grid-column-start: footer-start;
grid-column-end:   sidebar-end;
```

In this example, the header was positioned using the implicit grid line names.

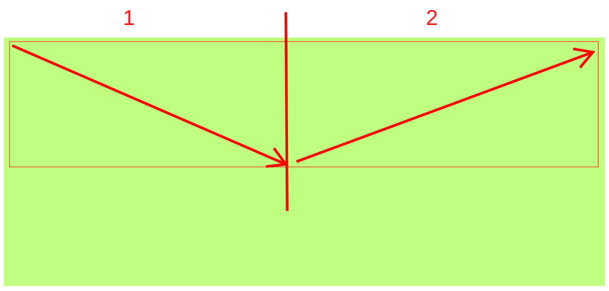




Superposición

Layering Grid Items

puede ser por capas/apliadas por correctamente posicionando ellos y asignando
Grid items can be layered/stacked by properly positioning them and assigning `z-index`
cuando necesario
when necessary.



```
.item-1,
.item-2 {
  grid-row-start:  1;
  grid-column-end: span 2;
}
```

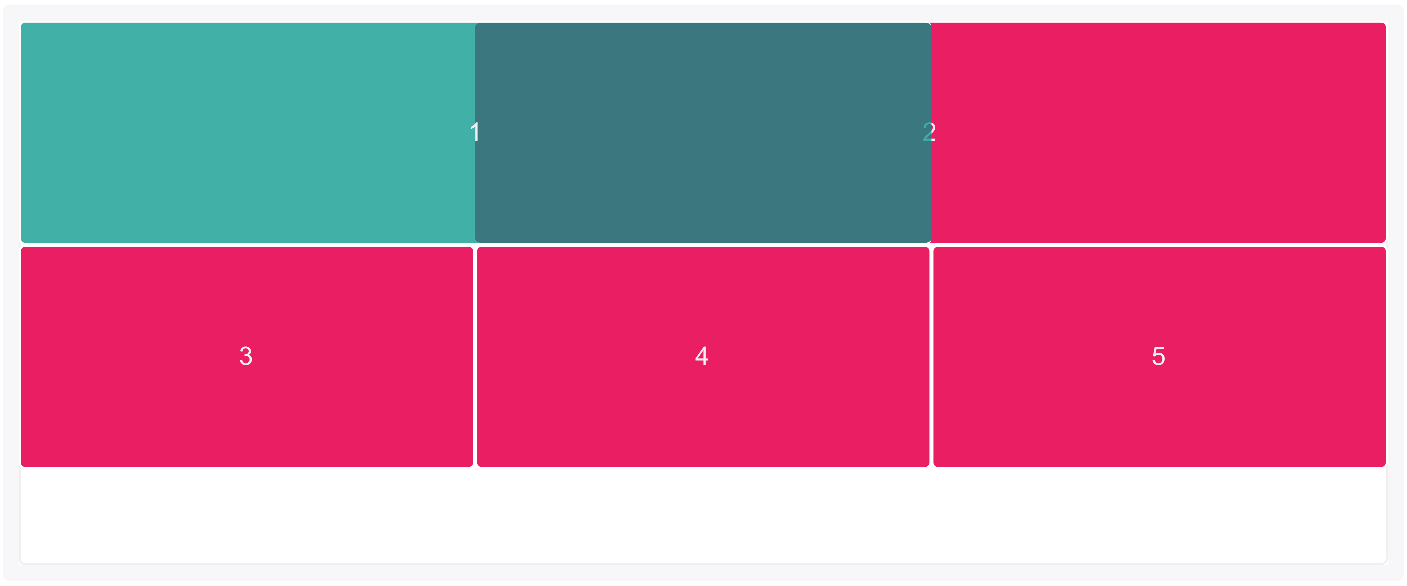
expande hasta el final

```
.item-1 { grid-column-start: 1; z-index: 1; }
.item-2 { grid-column-start: 2 }
```

En este ejemplo , elemento 1 y 2 estan posicionados a comenzar sobre la fila 1 y establecer el espaciado columna 2
In this example, items 1 and 2 are positioned to start on row line 1 and set to span 2 columns.

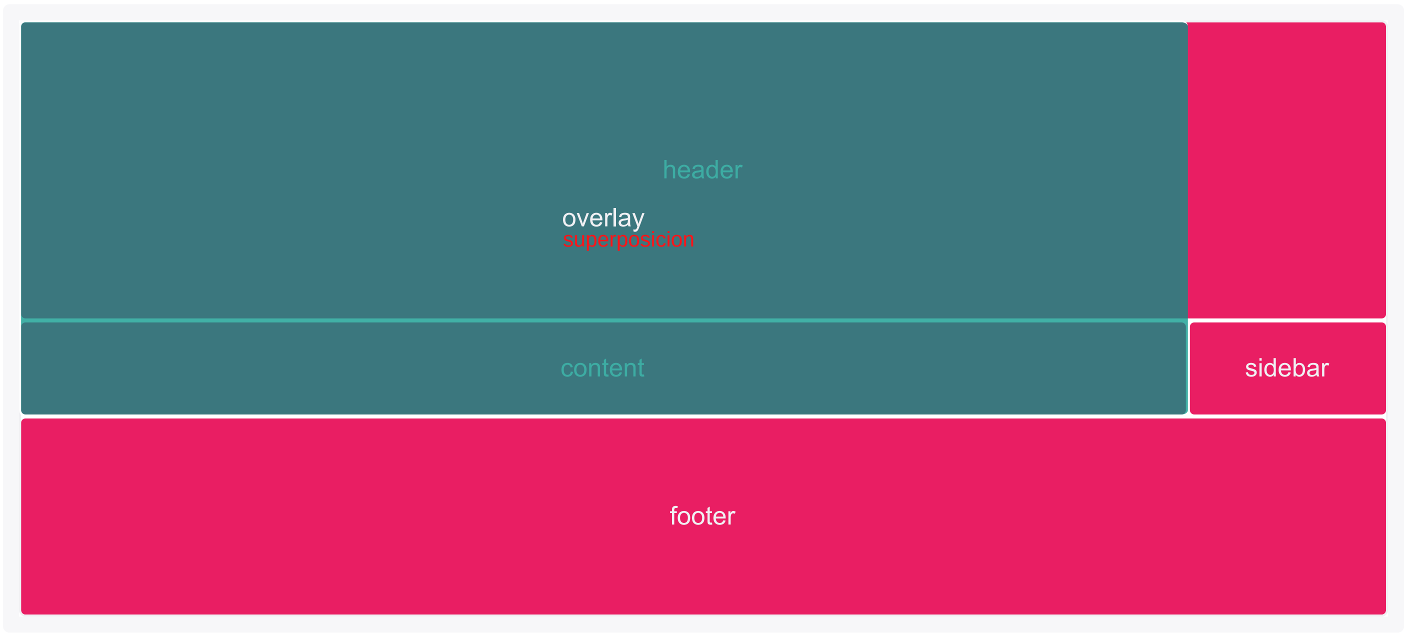
Ambos elementos estan posicionados para numero grid line , Elemento 1 esta establecido al comienzo sobre la linea columna 1 y elemento 2 sobre
Both items are positioned by grid line numbers. Item 1 is set to start at column line 1, and item 2 at
linea columna 2 , el cual resulta en ambos elementos superposicion en el centro banda columna
column line 2, which results in both items overlapping in the center column track.

Por defecto , el elemento 2 podria sentarse en lo alto del elemento 1 , sin embargo , nosotro hemos creado para asignando
By default, item 2 would sit on top of item 1; however, we've created stacking context by assigning `z-index: 1`
a elemento 1 , resultando ello sentar en lo alto de elemento 2
to item 1, resulting it to sit on top of item 2.



```
grid-row-start:    header-start;
grid-row-end:      content-end;
grid-column-start: content-start;
grid-column-end:   sidebar-start;
z-index: 1;
```

En este ejemplo esta posicionado y de capas en alto usando implicito nombre desde el
In this example, a grid item is positioned and layered on top using implicit grid line names from the
definido
defined `grid-template-areas`.



Alienando elementos grid 'elementos hijos' (Alineación Caja)

Aligning Grid Items (Box Alignment)

complementos CSS Grid permite elementos estar alineados a lo largo
CSS's Box Alignment Module complements CSS Grid to allow items to be aligned along
las filas de columnas ejes elementos hijos
the row of column axis.

alinean elementos a lo largo eje filas y
`justify-items` and `justify-self` align items along the row axis, and `align-items` and
alineando elementos a lo largo eje columna
`align-self` align items along the column axis.

estan aplicados a los y soportan los siguientes valores
`justify-items` and `align-items` are applied to the grid container and support the following values:

- `auto`
- `normal`
- `start`
- `end`
- `center`
- `stretch`
- `baseline`
- `first baseline`
- `last baseline`

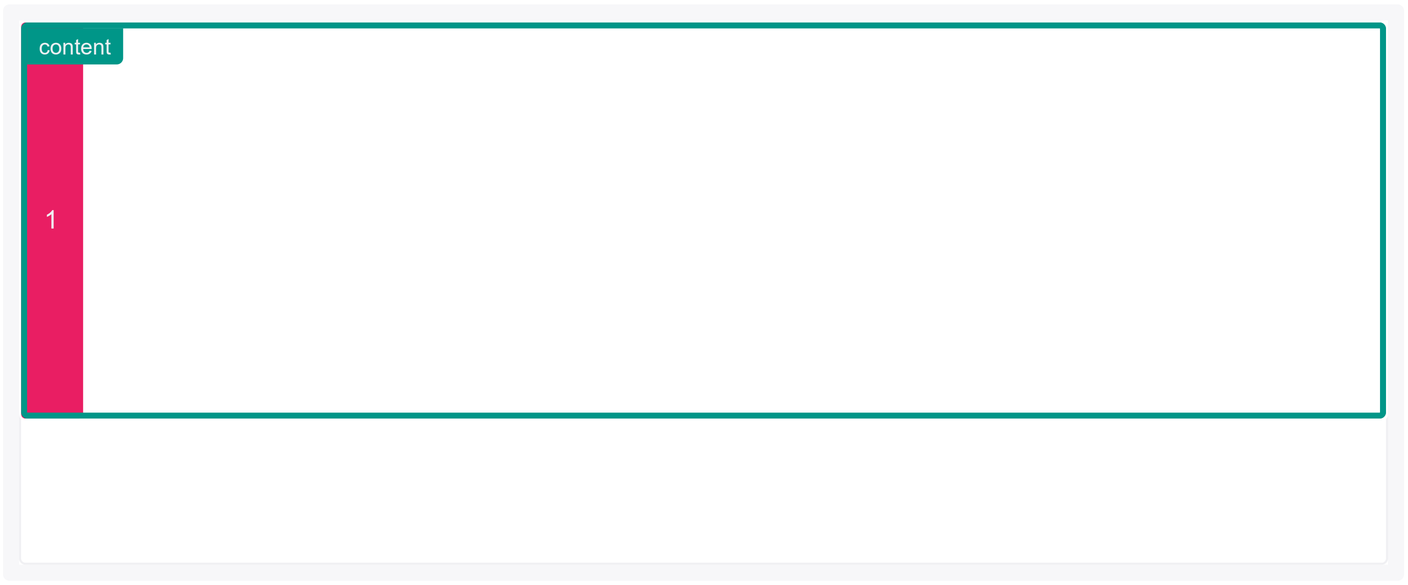
```
.grid {
  grid-template-rows: 80px 80px;
  grid-template-columns: 1fr 1fr;
  grid-template-areas: "content content"
                       "content content";
}
.item { grid-area: content }
```

Eje Horizontal : Contenedor

```
.grid {
  justify-items: start
}
```

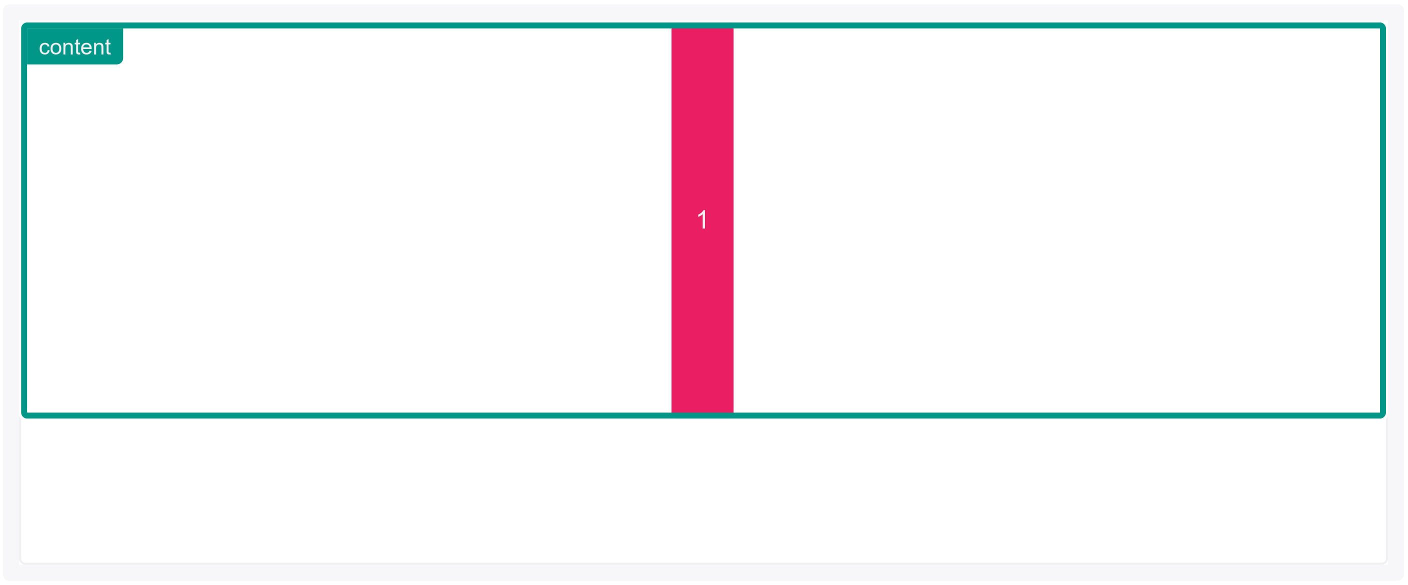
Elementos son posicionados sobre el comienzo de el eje fila (linea numero fila 1)

Items are positioned at the start of the row axis (row line number 1).



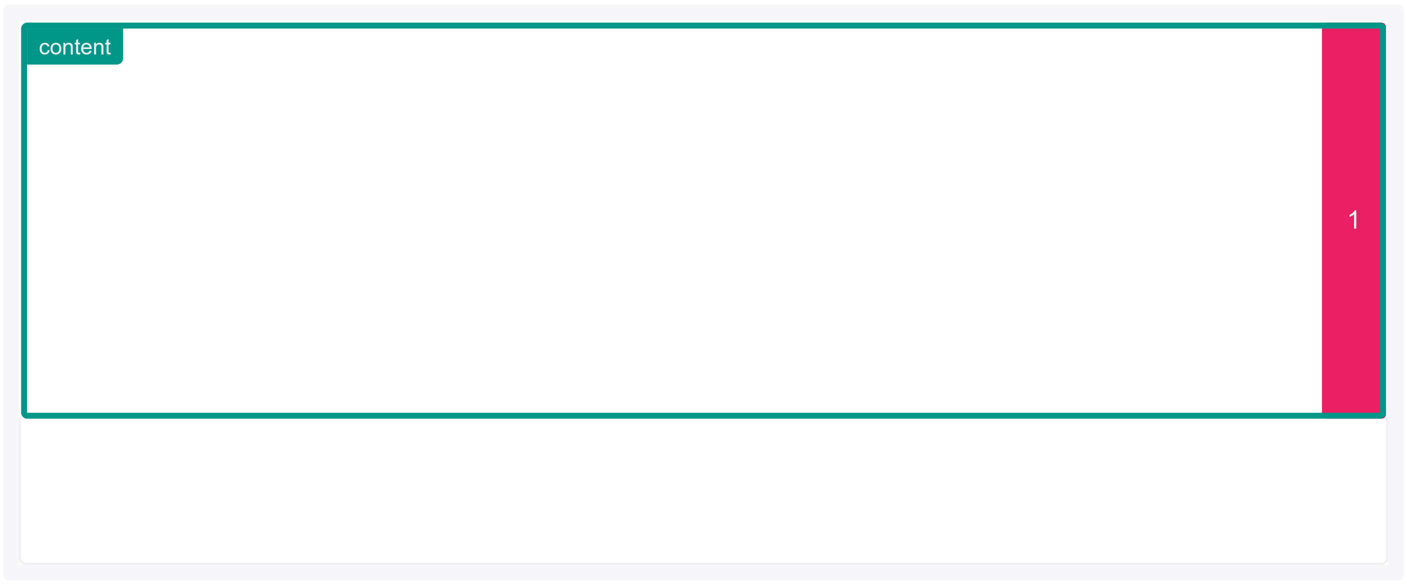
```
justify-items: center
```

Items are positioned at the center of the row axis.



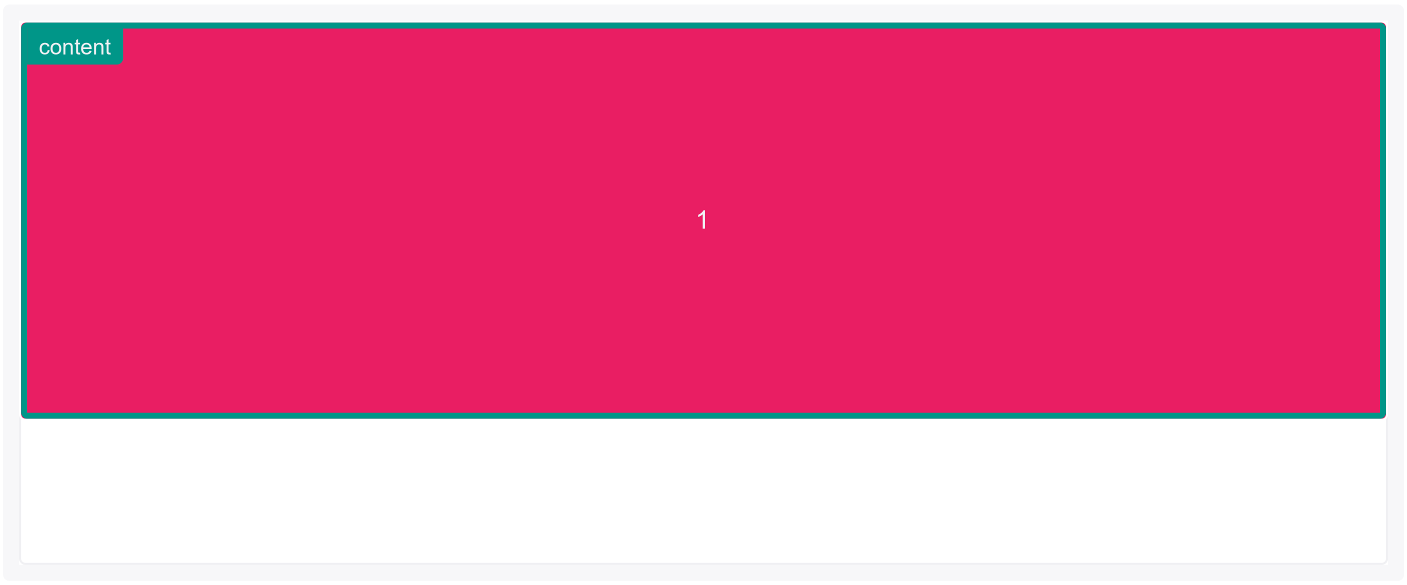
```
justify-items: end
```

Items are positioned at the end of the row axis.



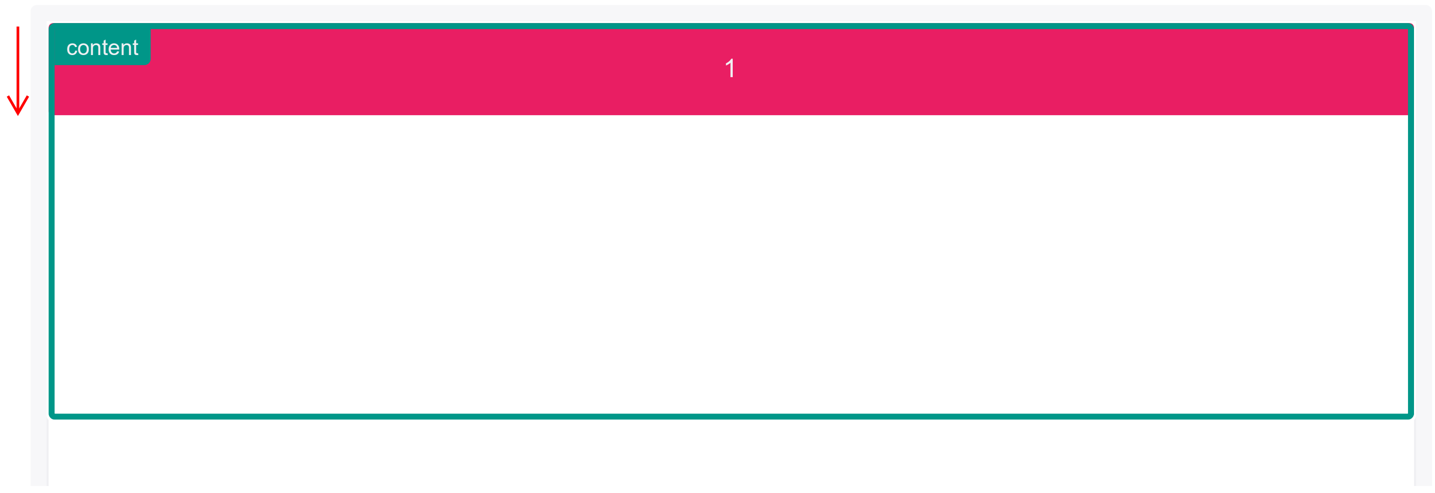
`justify-items: stretch`

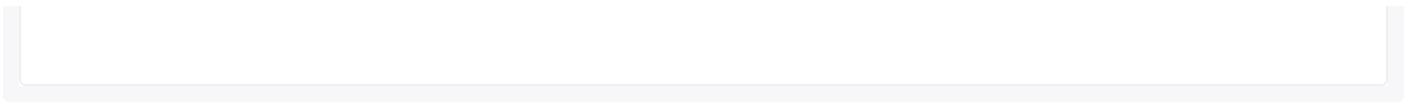
Items are stretched across the entire row axis. `stretch` is the default value.



`align-items: start`

Items are positioned at the start of the column axis (column line `1`).

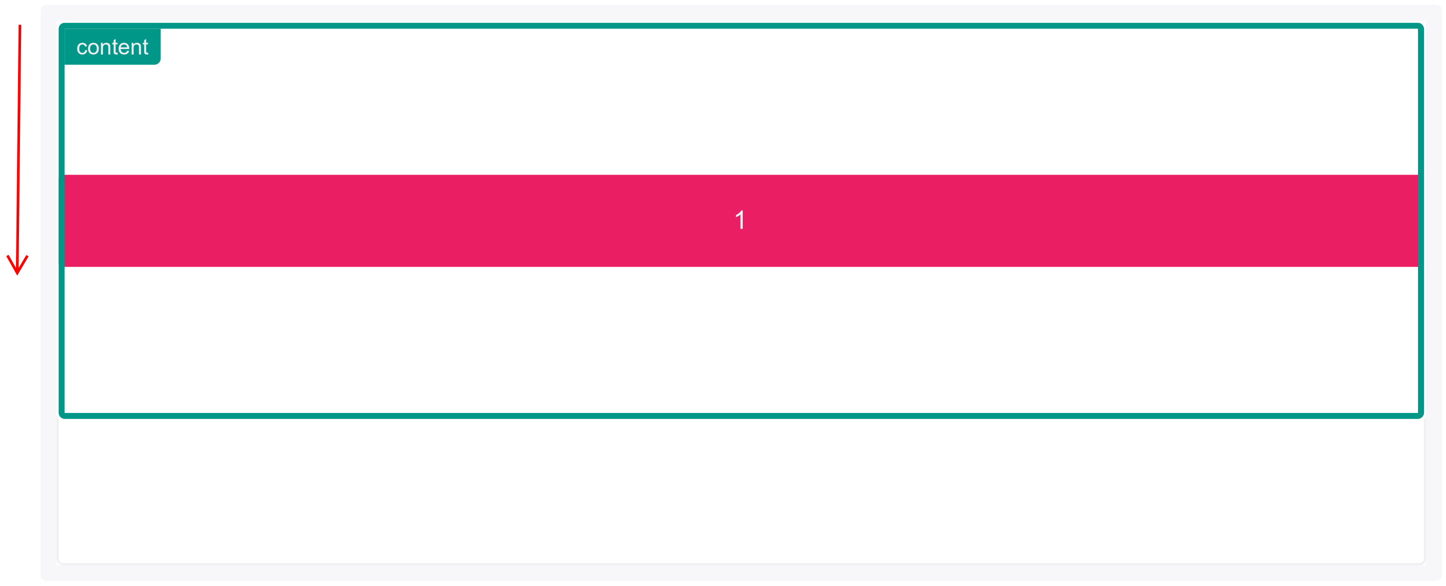




Eje Vertical

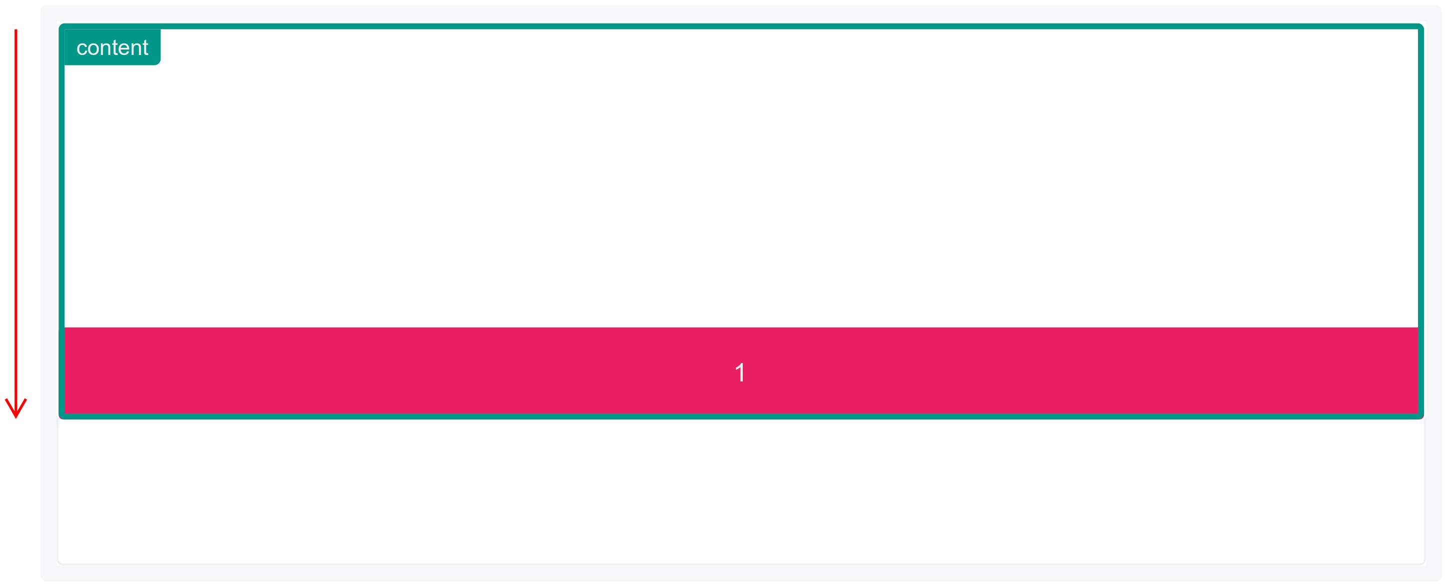
`align-items: center`

Elementos estan posicionados sobre el centro de el eje
Items are positioned at the center of the column axis.



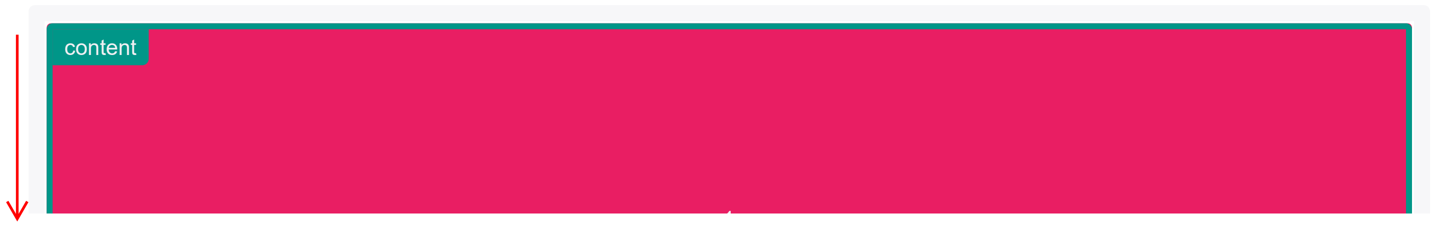
`align-items: end`

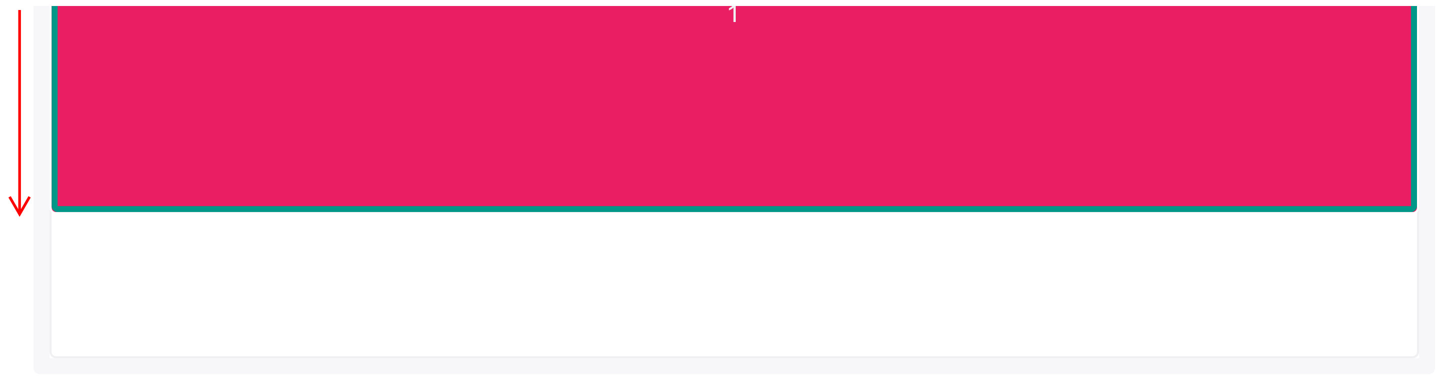
Elementos estan posicionados sobre el final de la columna eje
Items are positioned at the end of the column axis.



`align-items: stretch`

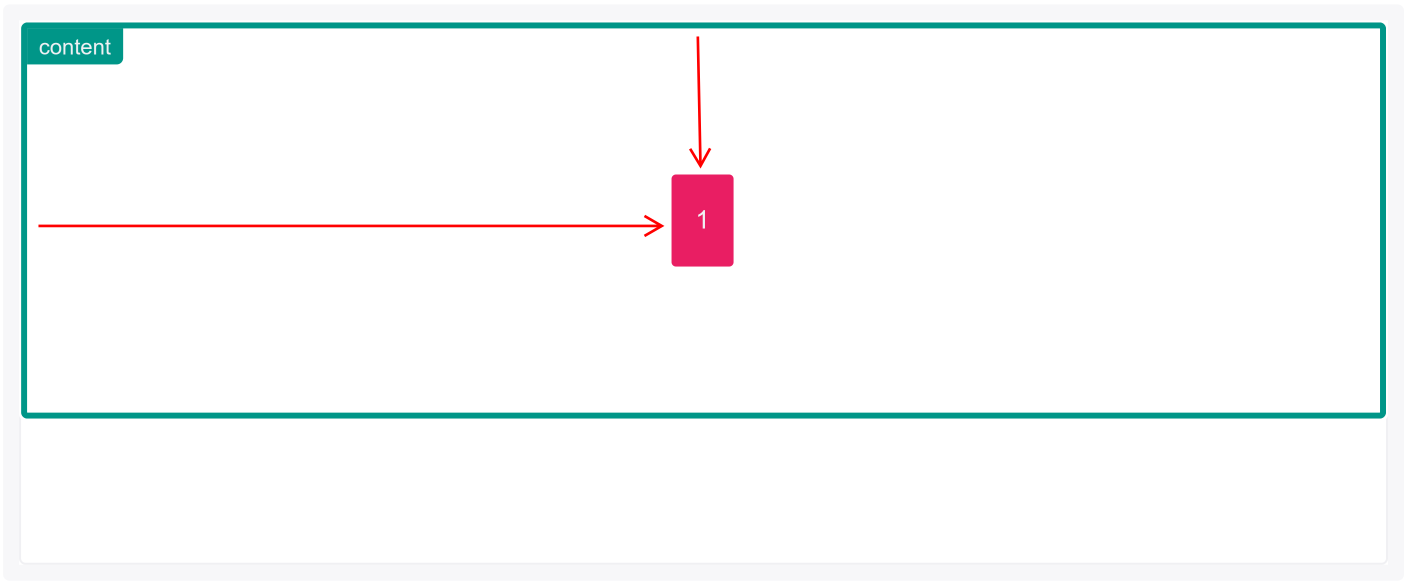
Elemento esta estirado cruzando la entera eje columna
Items are stretched across the entire column axis.





```
justify-items: center
align-items:   center
```

Elementos estan posicionados sobre el centro de los ejes de la fila y de la columna
Items are positioned at the center of the row and column axes.



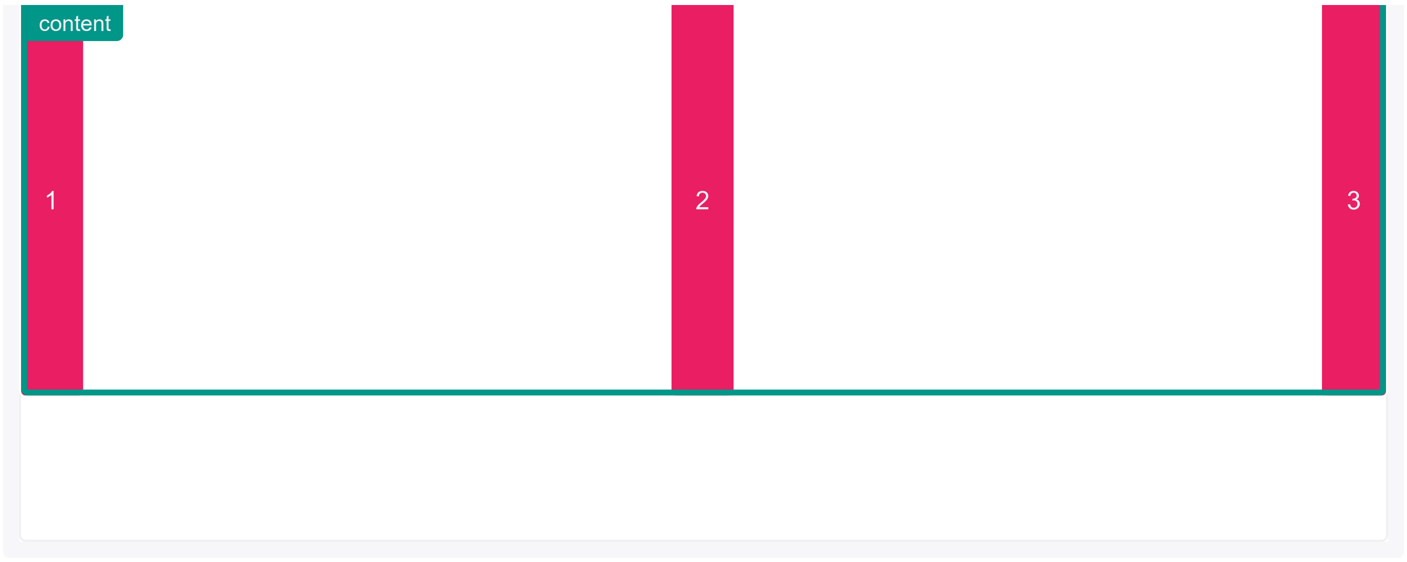
Elementos individuales puede ser alineado ellos mismo con el
Individual items can be self-aligned with the `align-self` and `justify-self` properties.
These properties support the following valuse:

- `auto`
- `normal`
- `start`
- `end`
- `center`
- `stretch`
- `baseline`
- `first baseline`
- `last baseline`

Eje horizontal

```
.item-1 { justify-self: start }
.item-2 { justify-self: center }
.item-3 { justify-self: end }
```

alinear elementos individuales a lo largo del eje fila
`justify-self` aligns individual items along the row axis.

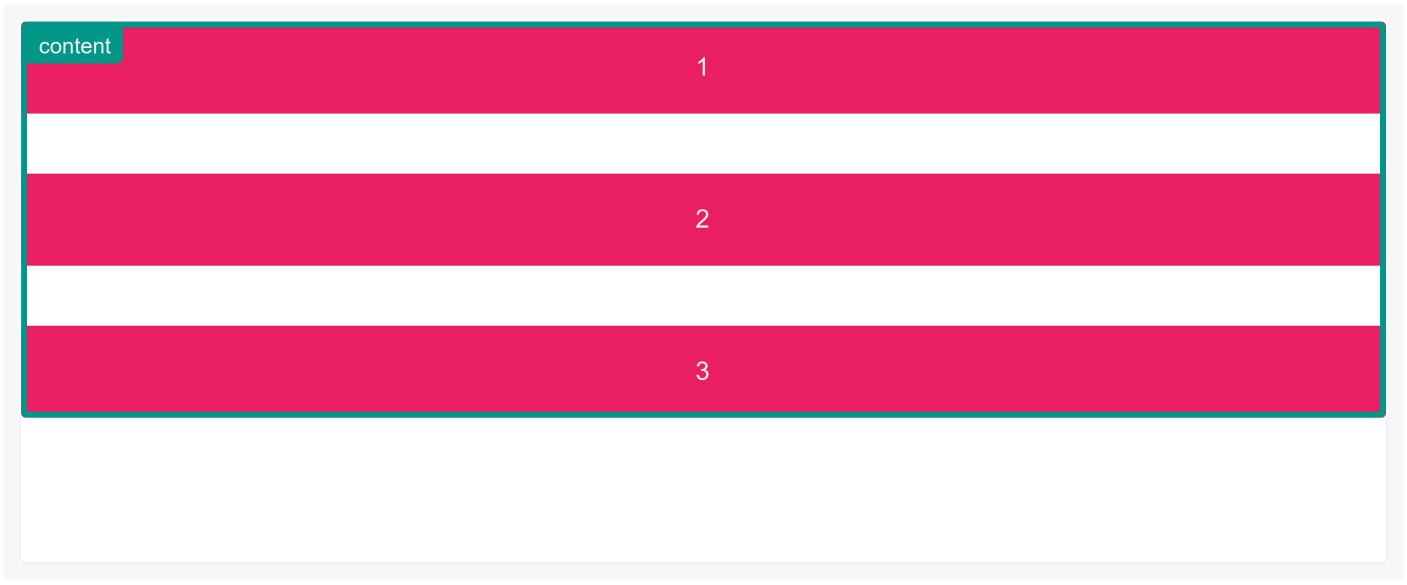


Eje vertical

```
.item-1 { align-self: start }
.item-2 { align-self: center }
.item-3 { align-self: end }
```

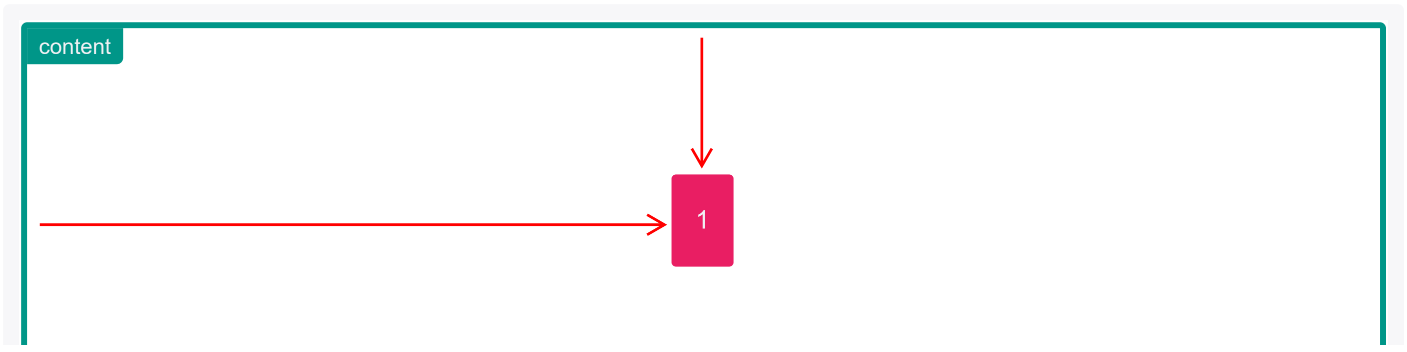
alinear elementos a lo largo del eje columna

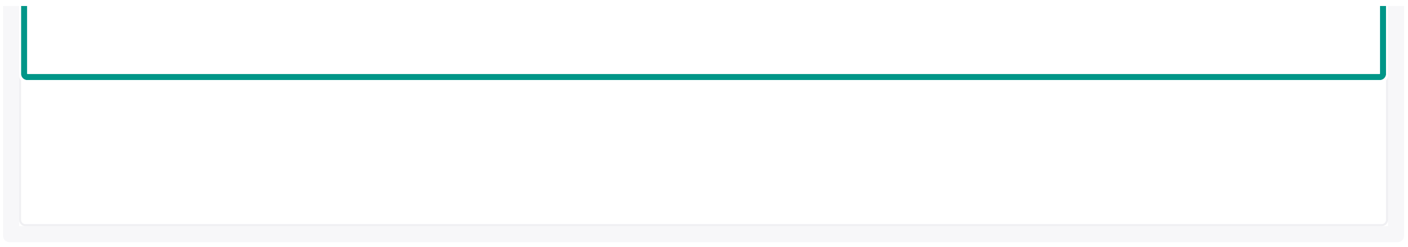
`align-self` aligns items along the column axis.



```
.item-1 {
  justify-self: center
  align-self:   center
}
```

Item 1 is positioned at the center of the row and column axes.





Alinear los grid 'bandas/franjas'

Aligning Grid Tracks

puede ser alineado relativo a el a lo largo del eje fila y columna

Grid tracks can be aligned relative to the grid container along the row and column axes.

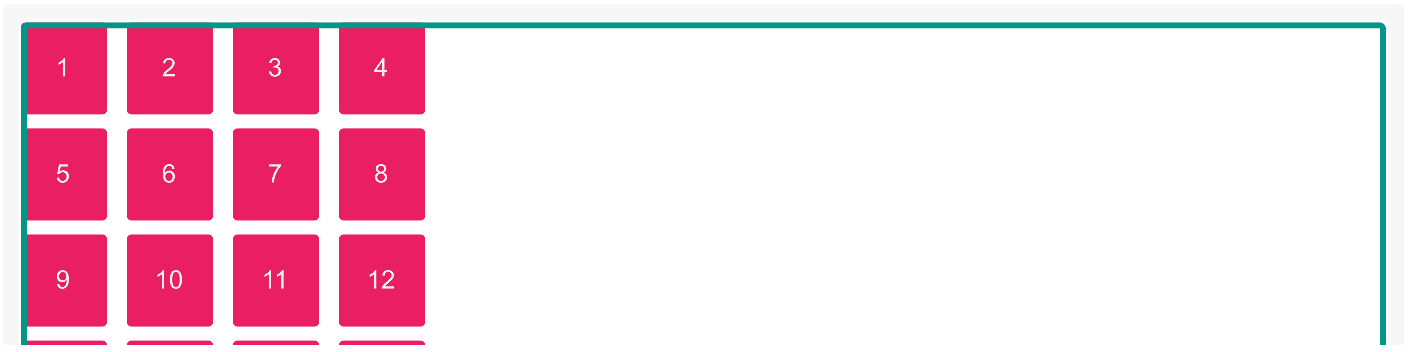
alinear bandas a lo largo el eje fila y a lo largo al eje columna

`align-content` aligns tracks along the row axis and `justify-content` along the column axis. They support the following properties:

- `normal`
- `start`
- `end`
- `center`
- `stretch`
- `space-around`
- `space-between`
- `space-evenly`
- `baseline`
- `first baseline`
- `last baseline`

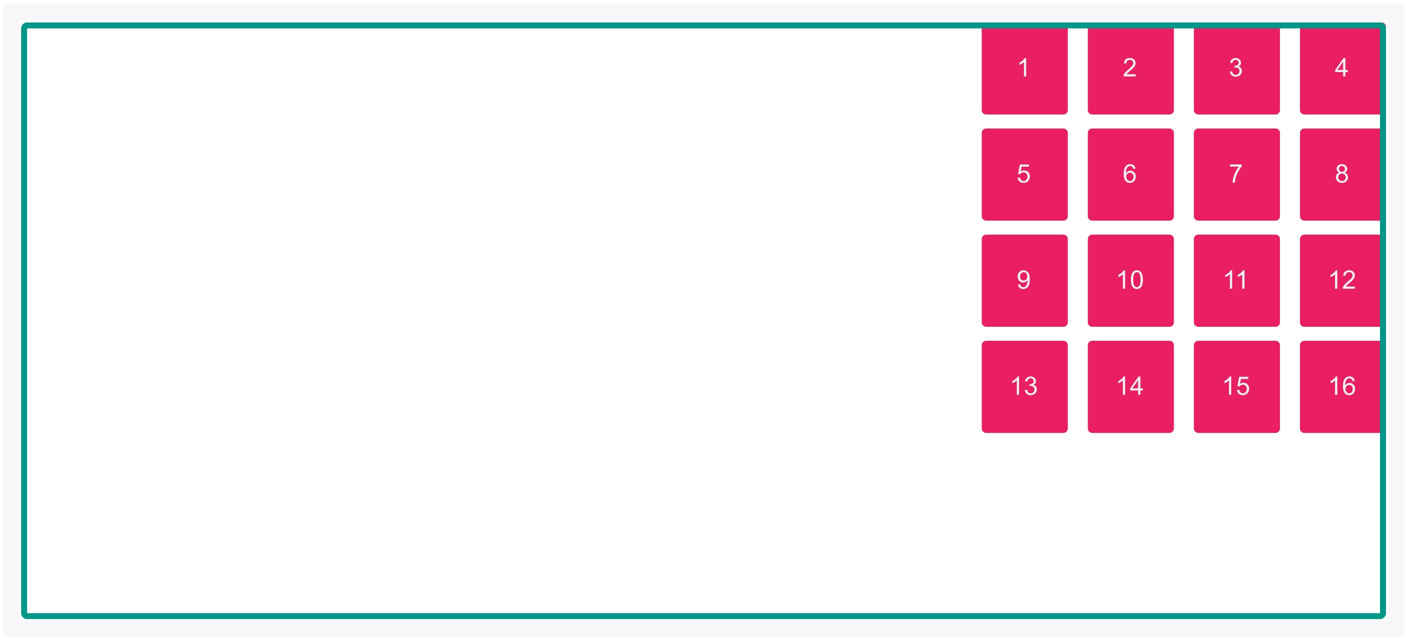
```
.grid {
  width: 100%;
  height: 300px;
  grid-template-columns: repeat(4, 45px);
  grid-template-rows: repeat(4, 45px);
  grid-gap: 0.5em;
  justify-content: start;
}
```

`start` aligns column tracks along and at the start of the row axis—it is the default value.



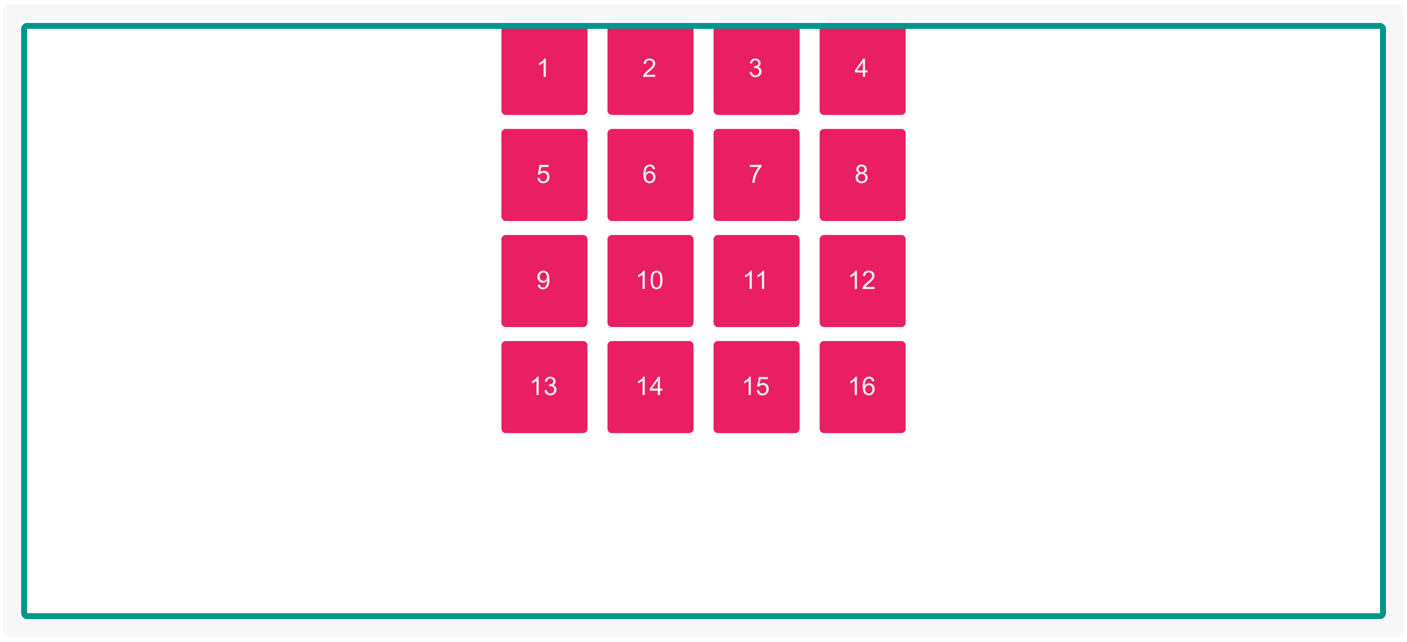
`justify-content: end;`

Columns are aligned at the end of the row axis.



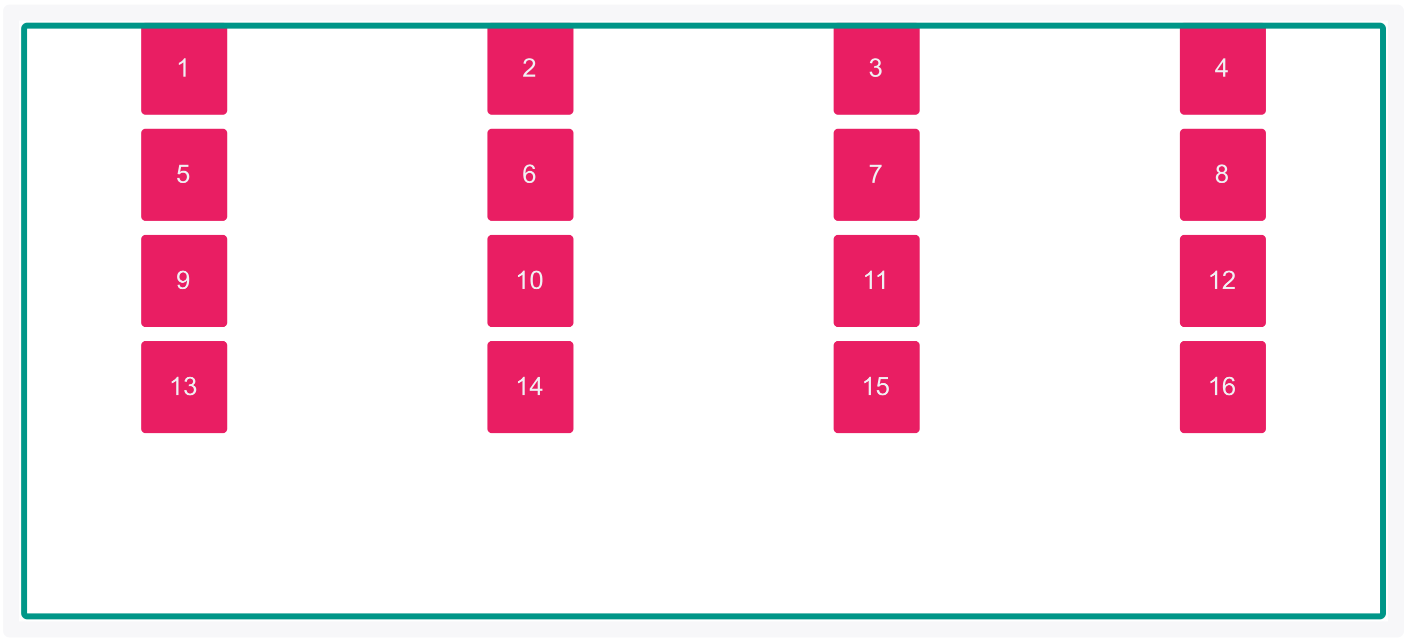
`justify-content: center;`

Columns are aligned at the center of the row axis.



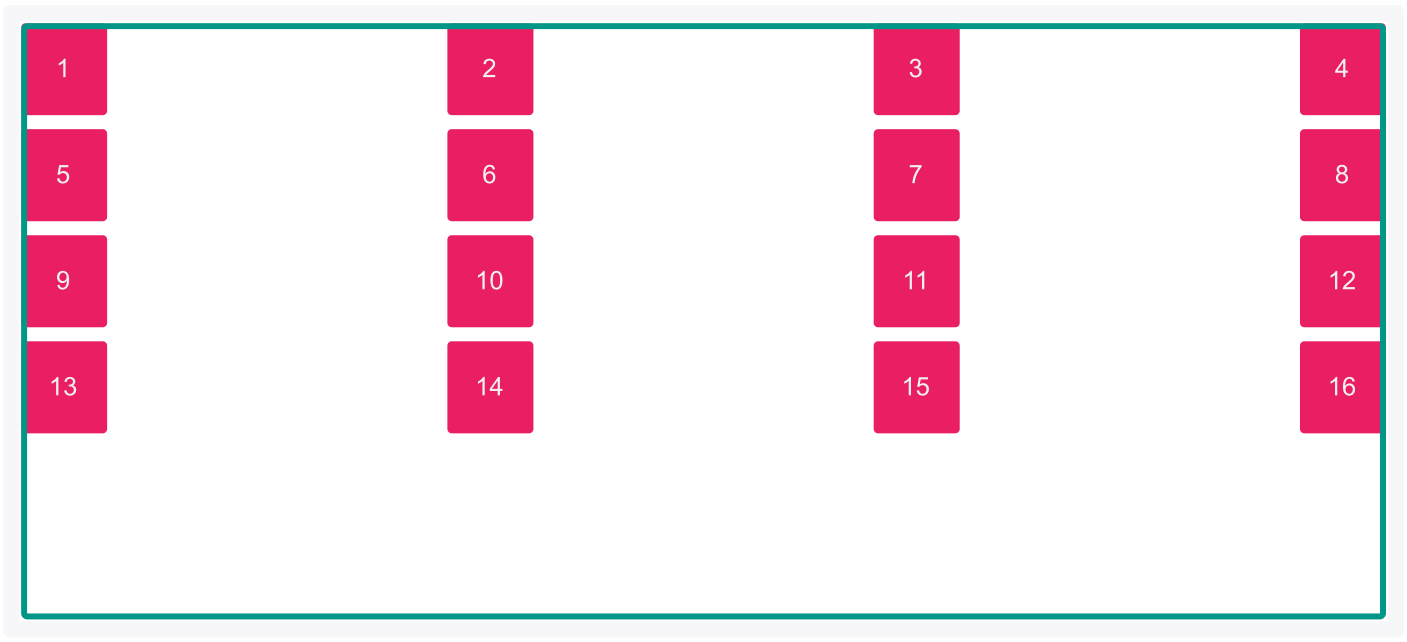
`justify-content: space-around;`

The remaining space of the grid container is distributed and applied to the start and end of each column track.



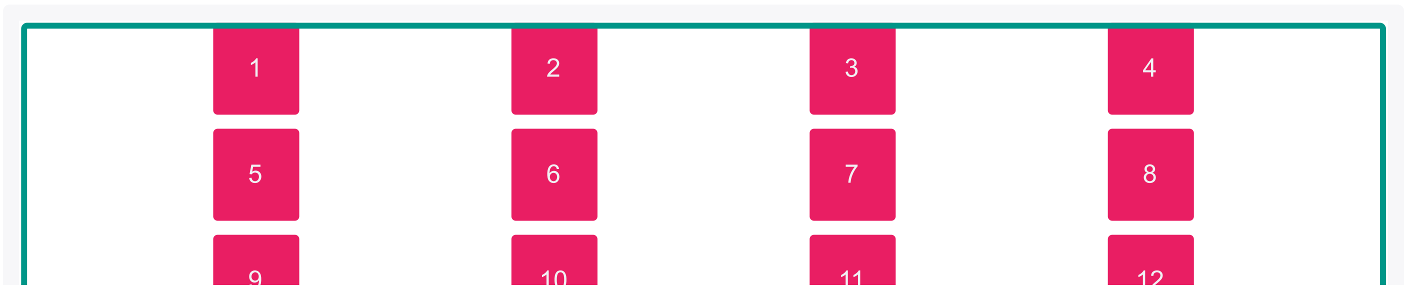
`justify-content: space-between;`

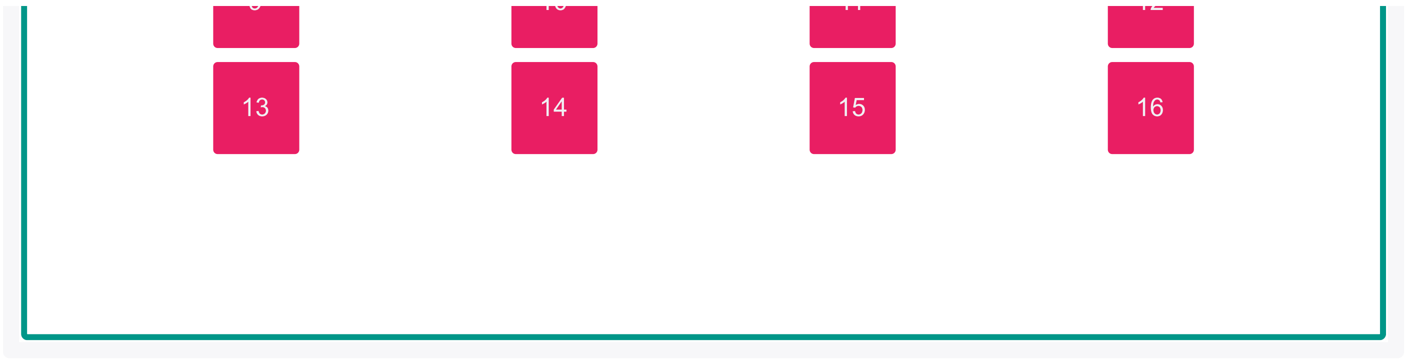
The remaining space is distributed between the column tracks.



`justify-content: space-evenly;`

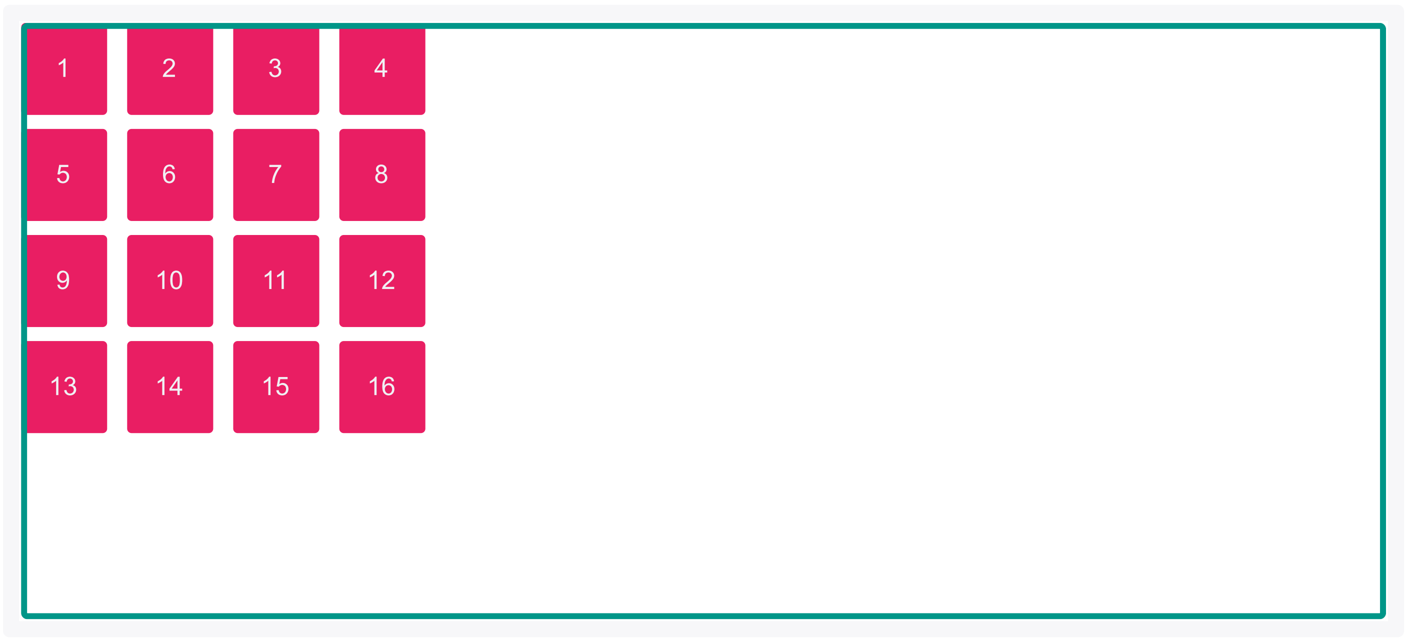
The remaining space is distributed where the space between the columns are equal to the space at the start and end of the row track.





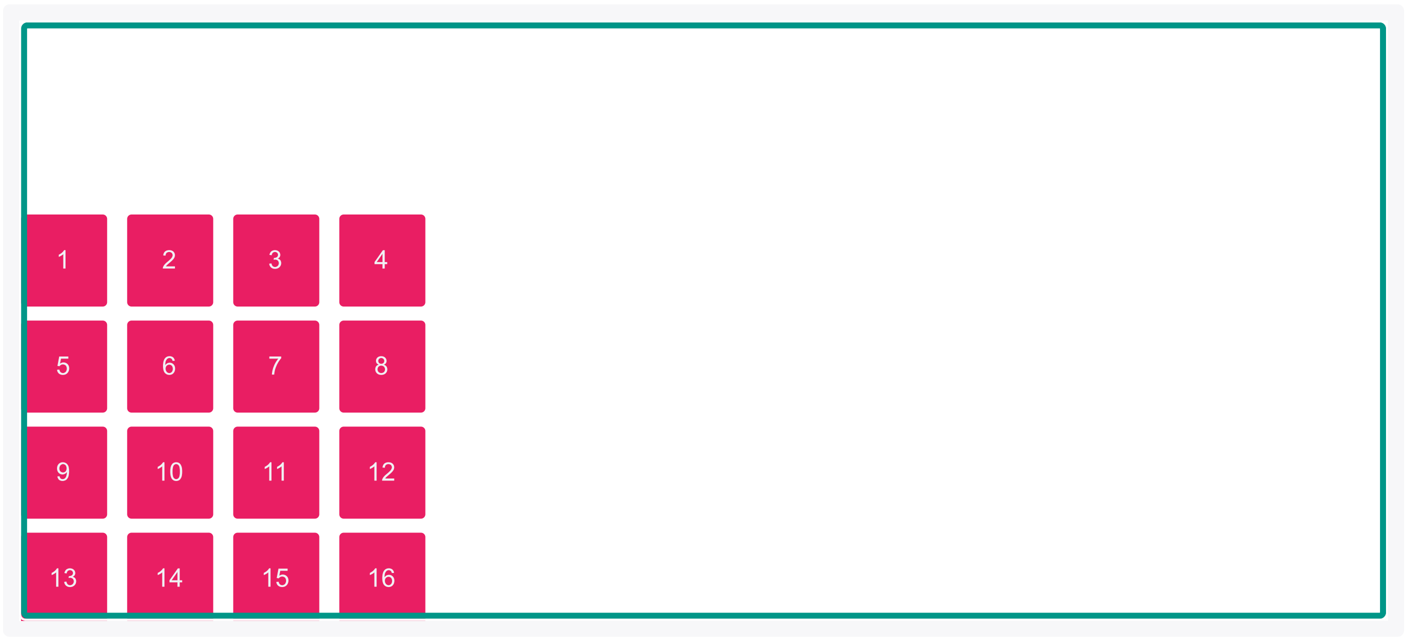
`align-content: start;`

`start` aligns rows at the start of the column axis and is the default value.



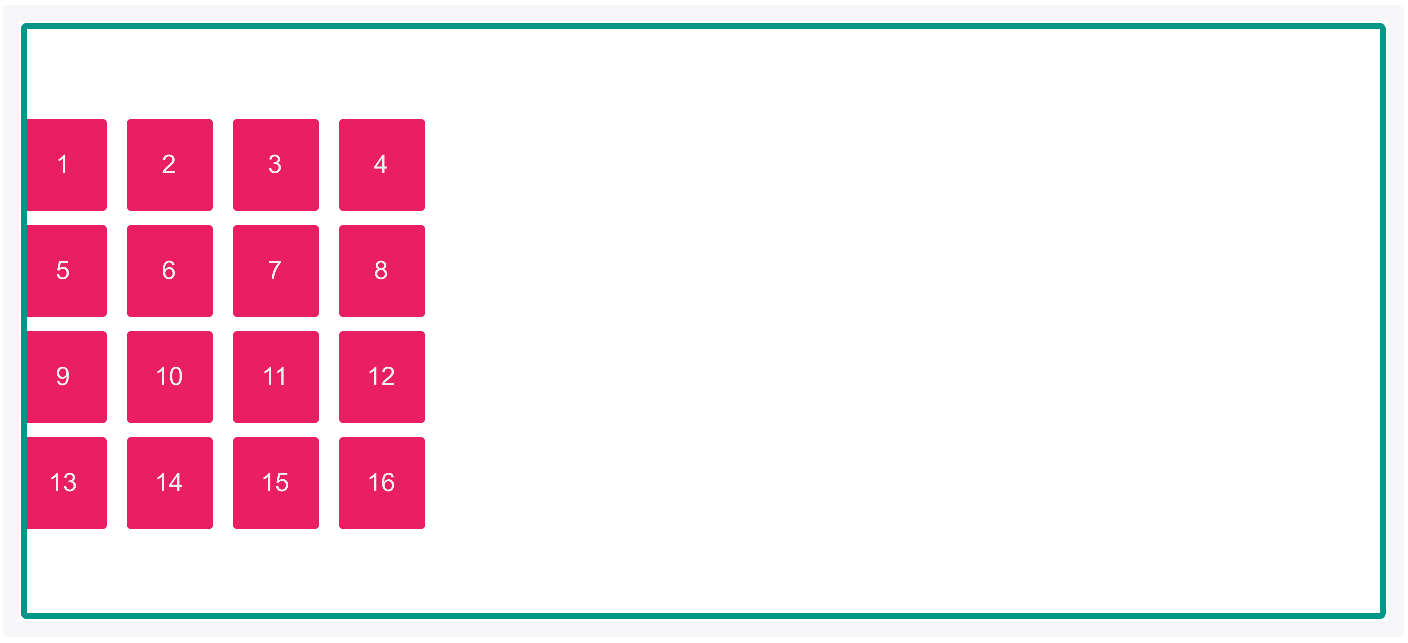
`align-content: end;`

Rows are aligned at the end of the column axis.



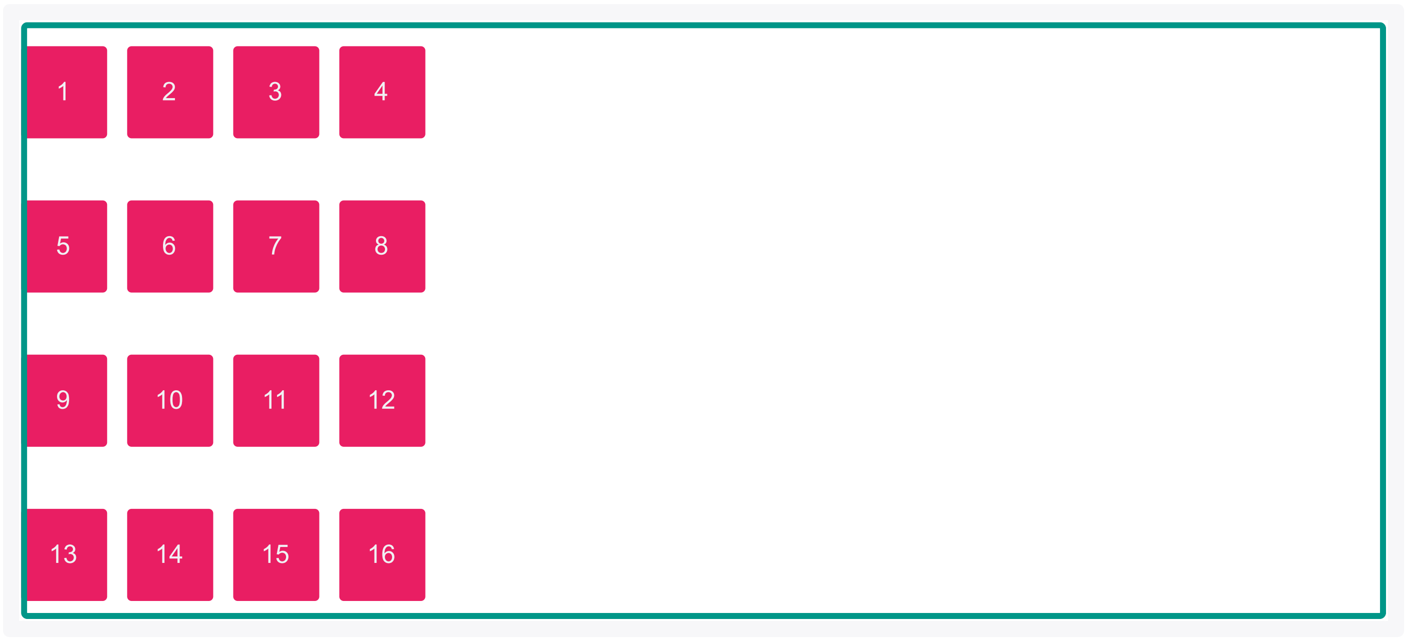
`align-content: center;`

Rows are aligned at the center of the column axis.



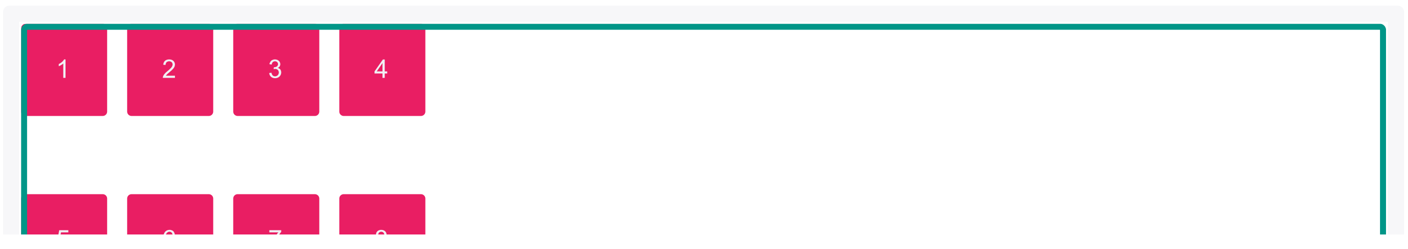
`align-content: space-around;`

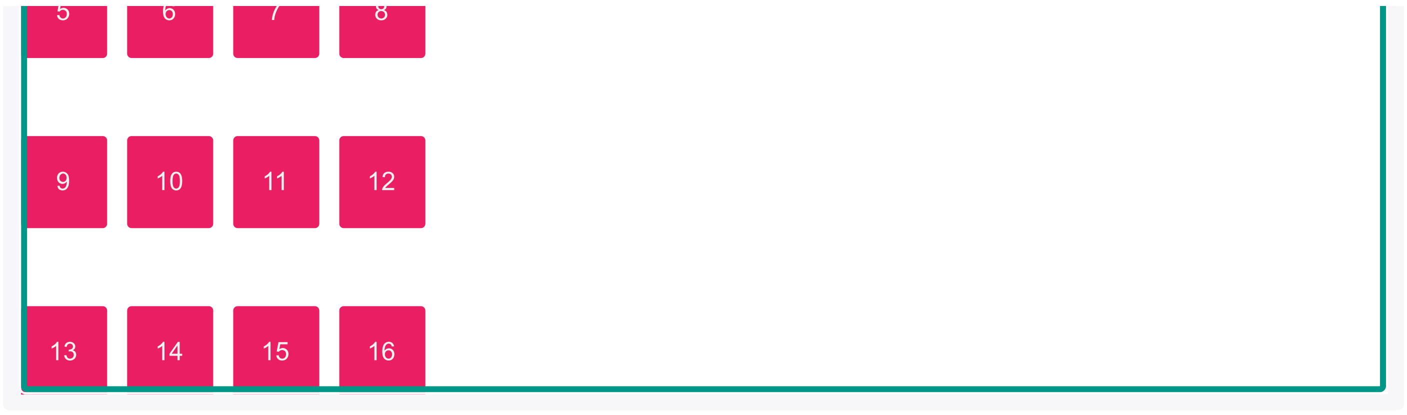
The remaining space of the grid container is distributed and applied to the start and end of each row track.



`align-content: space-between;`

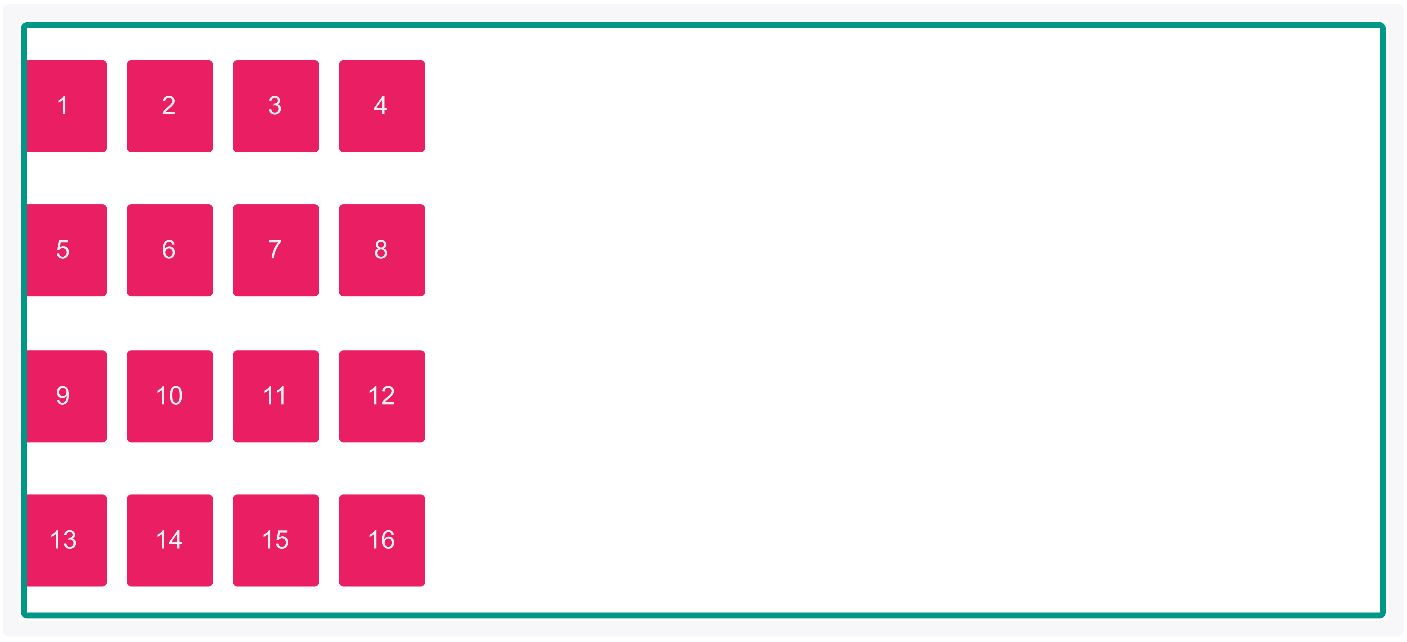
The remaining space is distributed between the row tracks.





`align-content: space-evenly;`

The remaining space is distributed where the space between the rows are equal to the space at the start and end of the column track.



This guide is designed to give you a fairly comprehensive overview of Grid; however, it doesn't pretend to be a complete technical documentation. Be sure to check out the specs of Mozilla Developer Network and W3C for an even deeper dive.

Here are some other fantastic resources on CSS Grid:

- Complete Guide to Grid on CSS Tricks
- Grid by Example by Rachel Andrew
- The CSS Workshop by Jen Simmons
- Grid Garden by Codepip
- Spring Into CSS Grid by Joni Trythall

I'm susceptible to making mistakes or being wrong. If you see a typo or a mistake, please reach out to me on Twitter or via email.

Huge thank you to Mozilla Developer Network and W3C for the CSS Grid resources; ladies Jen Simmons and Rachel Andrew, who are major contributors to Grid, and it wouldn’t be where it’s at without them; and my amazing company, Planning Center ♡, for allowing me the time to dive in and learn CSS Grid during Free Week.