CSS Grid

{desafío}
latam_

Activen las cámaras los que puedan y pasemos asistencia

Aplicar la propiedad grid para crear layouts responsivos.

- Unidad 1: Flexbox.
- Unidad 2: Grid.
- Unidad 3: Media Queries.
- Unidad 4: Animaciones con CSS.

Te encuentras aquí

{desafío} latam_

- Objetivo de la sesión
- Conceptualización
- Ejercitación
- Reflexión

Inicio

{desafío}
latam_

Objetivos

/*Crear una galería de imágenes y un layout con grid. */

/* Modificar los gaps de la grilla de un proyecto. */

/* Utilizar fracciones como unidad de medida para definir el tamaño de los ítems de una grilla. */

/* Diferenciar cuándo ocupar flexbox o grid.*/

Desarrollo

{desafío}
latam_

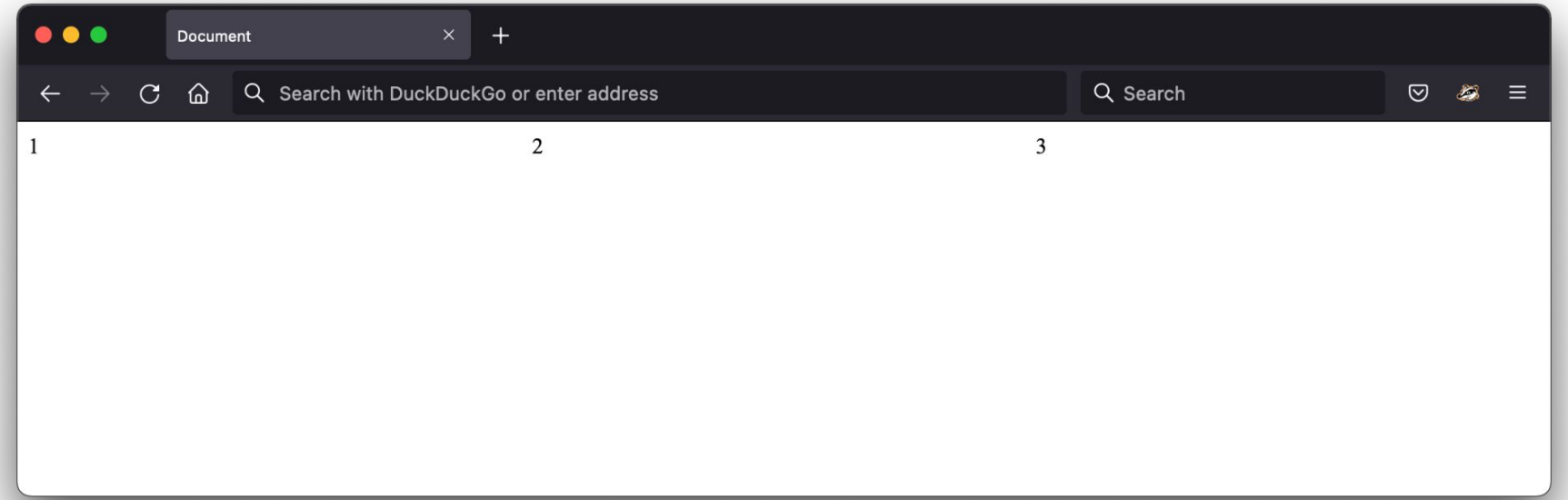
CSS Grid

¿Qué es Grid?

CSS grid es una herramienta que nos permite crear layouts complejos de forma sencilla.

CSS Grid

Construyendo un layout de 3 columnas



{desafío} latam_

CSS Grid

Grid-container y Grid-item

```html
<div class="grid-container">
  <div class="grid-item">
    1
  </div>
  <div class="grid-item">
    2
  </div>
  <div class="grid-item">
    3
  </div>
</div>
```

```css
.grid-container{
  display: grid;
  grid-template-columns: auto auto auto;
}
.grid-item{
}
```

3 columnas de tamaño automático

{desafío} latam_

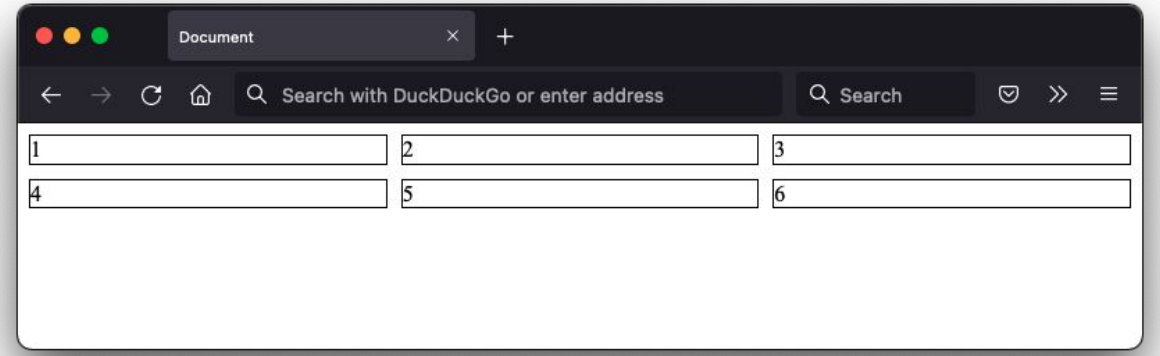
Ejercicio

- Modifica el CSS anterior para tener solo 2 columnas.
- ¿Qué sucede con el elemento sobrante?

Ejercicio
¡Manos al teclado!

{desafío} latam_

CSS Grid
Gaps



- En esta ocasión dejaremos 3 columnas.
- Debemos tener al menos 6 elementos dentro del grid-container.
- Agregaremos border a los elementos.
- Agregaremos gap: 10px al grid-container.

CSS Grid

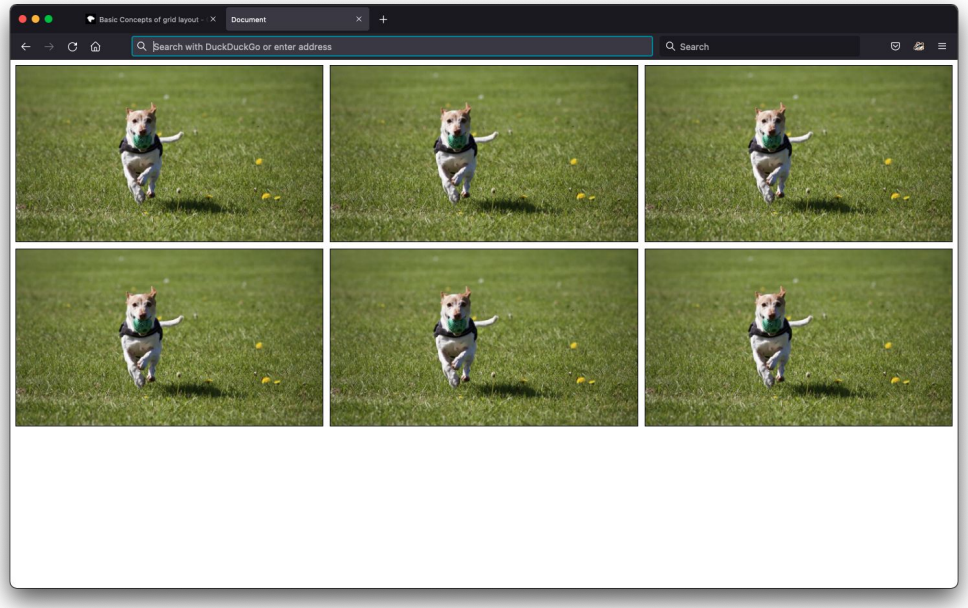
Gaps

La propiedad gap está compuesta de otras dos propiedades:

- column-gap y row-gap, las que podemos modificar por separado si lo necesitamos.

CSS Grid

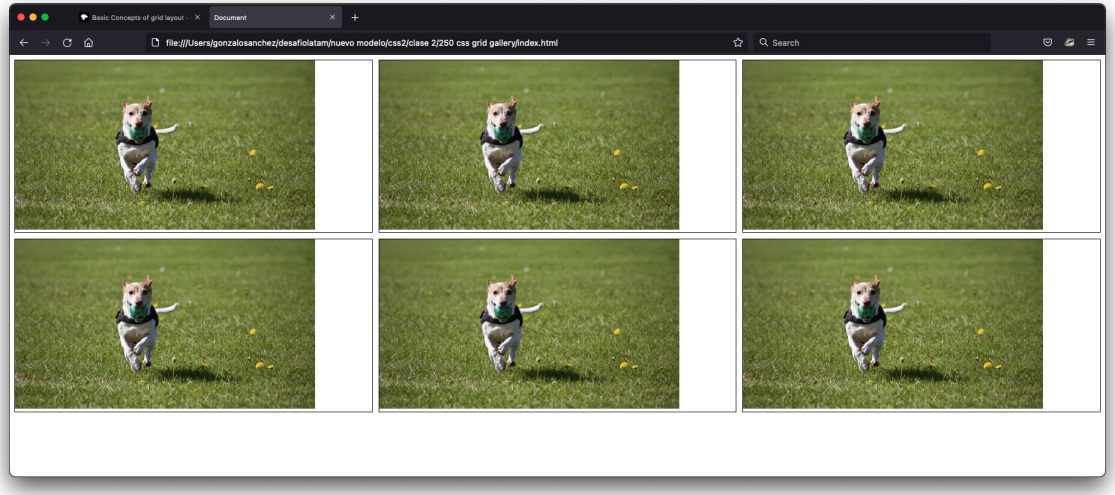
Utilizando las columnas para crear una galería de imágenes



Utilizando el grid-container con lo aprendido anteriormente, podemos crear una galería de imágenes.

CSS Grid

Ajustando el tamaño de la imagen



Dentro de la galería, puede quedar una separación entre las imágenes dependiendo del tamaño del viewport, esto, en caso de que no ajustemos el tamaño de la imagen.

CSS Grid

Utilizando las columnas para crear una galería de imágenes

```html
<div class="grid-container">
      <div class="grid-item">
        <img src="https://placedog.net/500" alt="">
      </div>
      <div class="grid-item">
        <img src="https://placedog.net/500" alt="">
      </div>
      <div class="grid-item">
        <img src="https://placedog.net/500" alt="">
      </div>
      <div class="grid-item">
        <img src="https://placedog.net/500" alt="">
      </div>
      <div class="grid-item">
        <img src="https://placedog.net/500" alt="">
      </div>
      <div class="grid-item">
        <img src="https://placedog.net/500" alt="">
      </div>
      <div class="grid-item">
        <img src="https://placedog.net/500" alt="">
      </div>
    </div>
```

```css
.grid-container {
  display: grid;
  grid-template-columns: auto auto auto;
  gap: 10px;
}
.grid-item {
  border: solid 1px black;
}

.grid-item img{
  height: 100%;
  width: 100%;
}
```

{desafío} latam_

Ejercicio

- Implementa una galería de imágenes utilizando Grid layout.
- Modifica el gap del grid para obtener el resultado visual que desees.
- Modifica el ancho y alto de la imagen para obtener el resultado visual que desees.

Ejercicio
¡Manos al teclado!

Objetivos

/*Crear una galería de imágenes y un layout con grid. */

/* Modificar los gaps de la grilla de un proyecto. */

/* Utilizar fracciones como unidad de medida para definir el tamaño de los ítems de una grilla. */

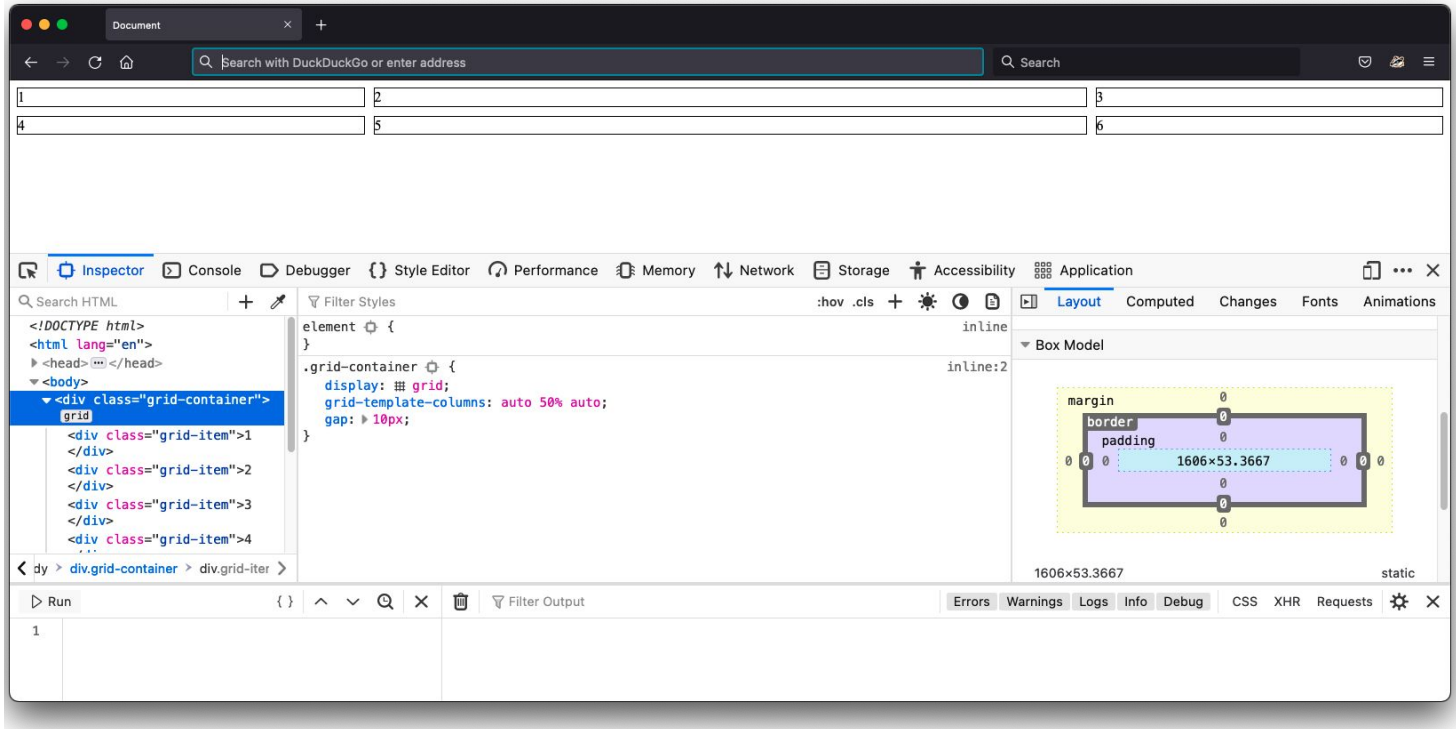
/* Diferenciar cuándo ocupar flexbox o grid.*/

{desafío} latam_

CSS Grid

Proporciones

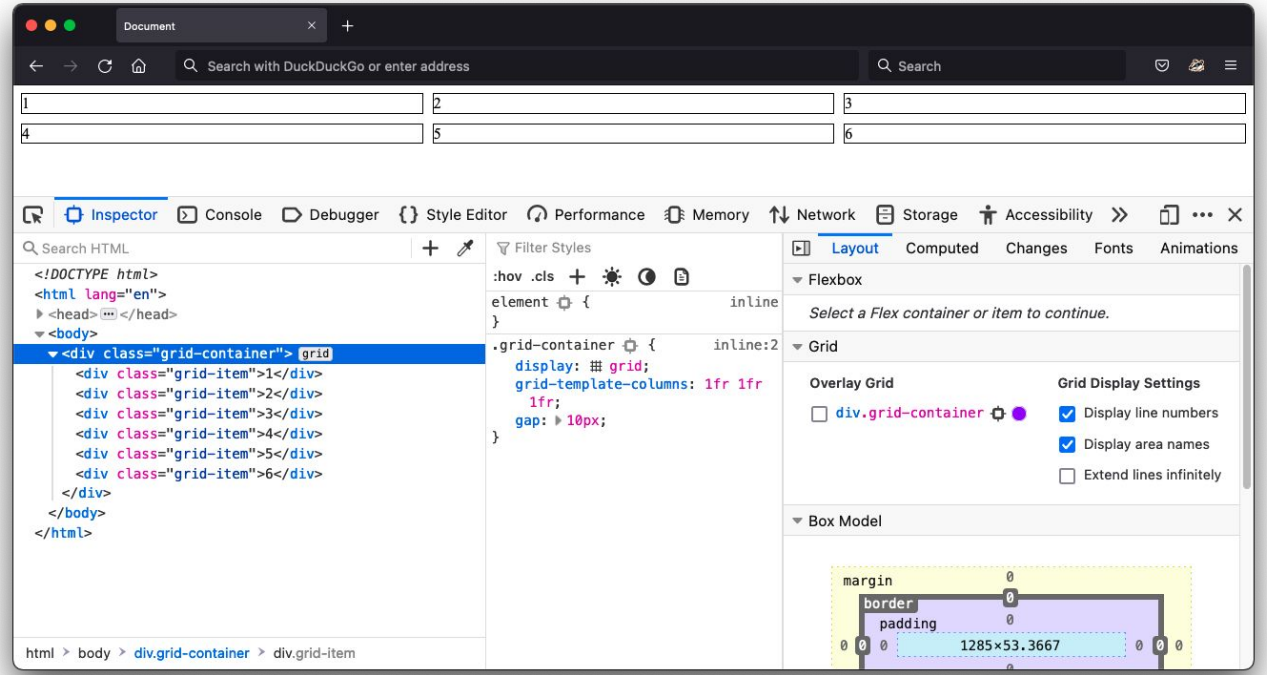
Hasta el momento, hemos dejado todas las columnas en tamaño automático, pero podemos especificar el tamaño de una o todas ocupando un ancho (width), dando a la columna 2 un 50% de ancho.



{desafío} latam_

CSS Grid

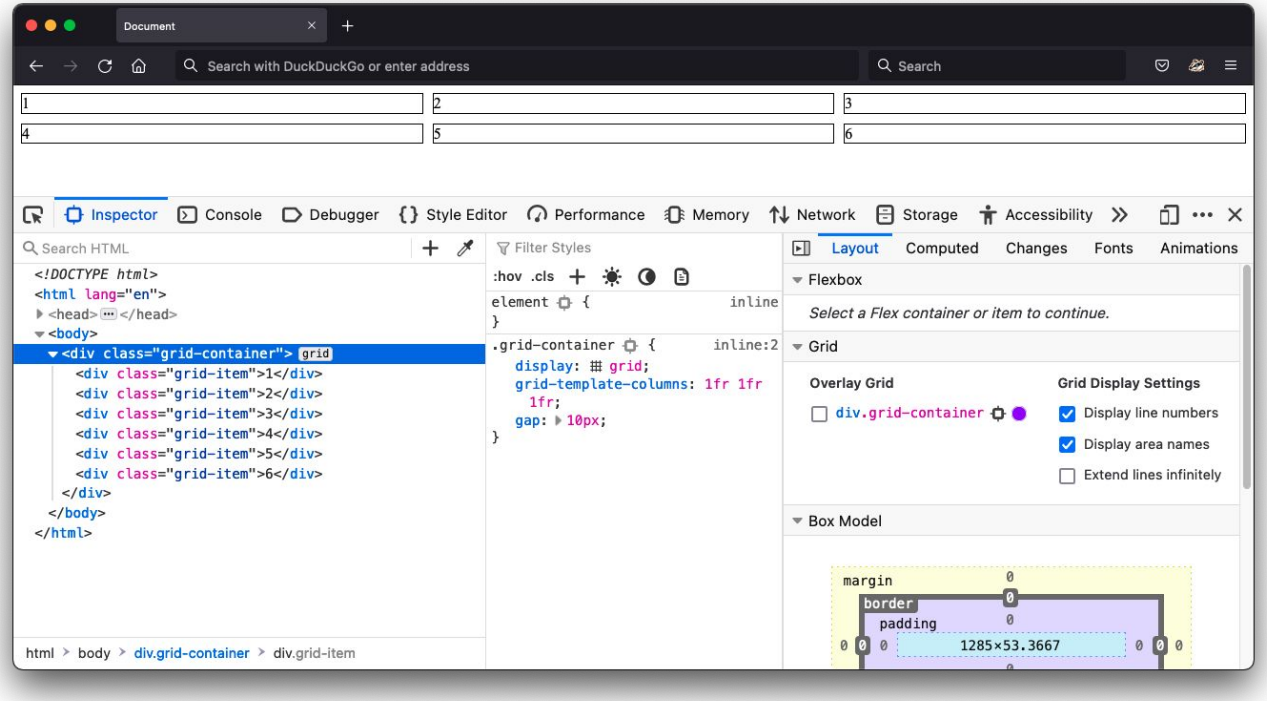
Proporciones (fr)



- Otra forma de asignar tamaños es con la unidad fr (**fr**action).
- Esta unidad divide el espacio libre en fracciones.
- Partamos cambiando el valor de grid-template-column por 1fr 1fr 1fr.

CSS Grid

Proporciones (fr)

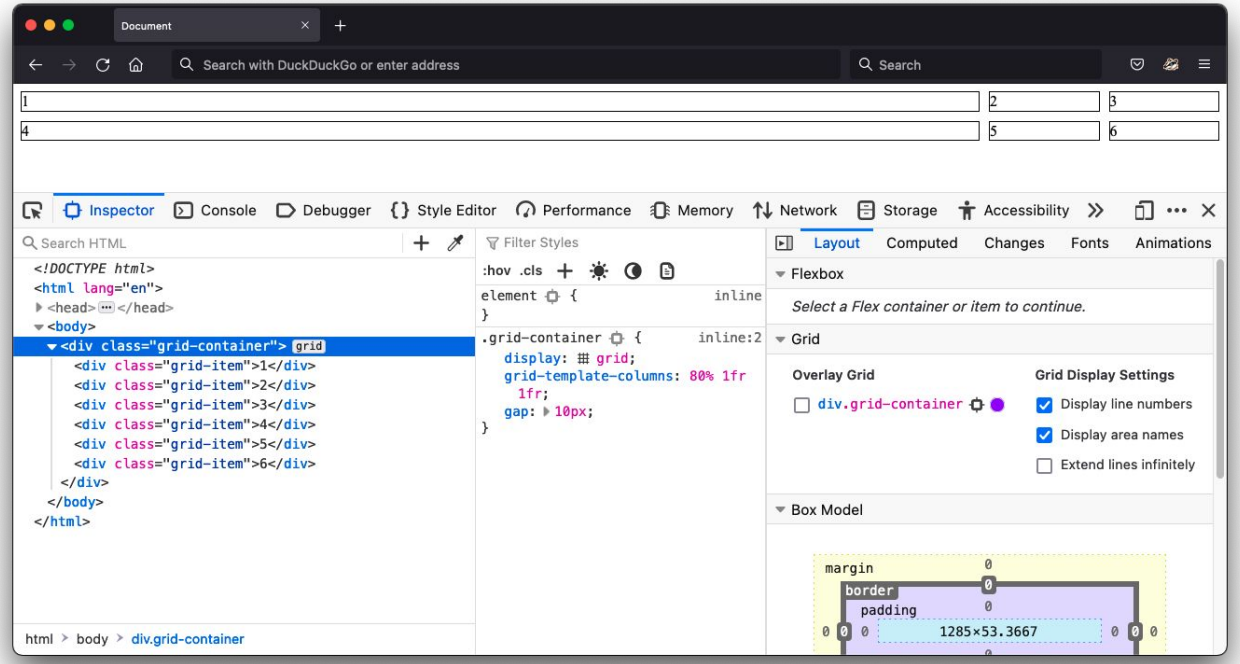


- Probemos cambiar cualquiera de las columnas por 2fr o por 3fr (o cualquier otro valor).

{desafío} latam_

CSS Grid

Combinando fracciones con otras unidades



Al combinar fracciones con otras unidades, tenemos que tener en cuenta que las fracciones utilizan el espacio que va quedando libre.

{desafío} latam_

Ejercicio

- Modifica la galería anterior, especificando al menos una de las columnas en porcentajes y el resto en fracciones.

Ejercicio
¡Manos al teclado!

Objetivos

/*Crear una galería de imágenes y un layout con grid. */ ✅

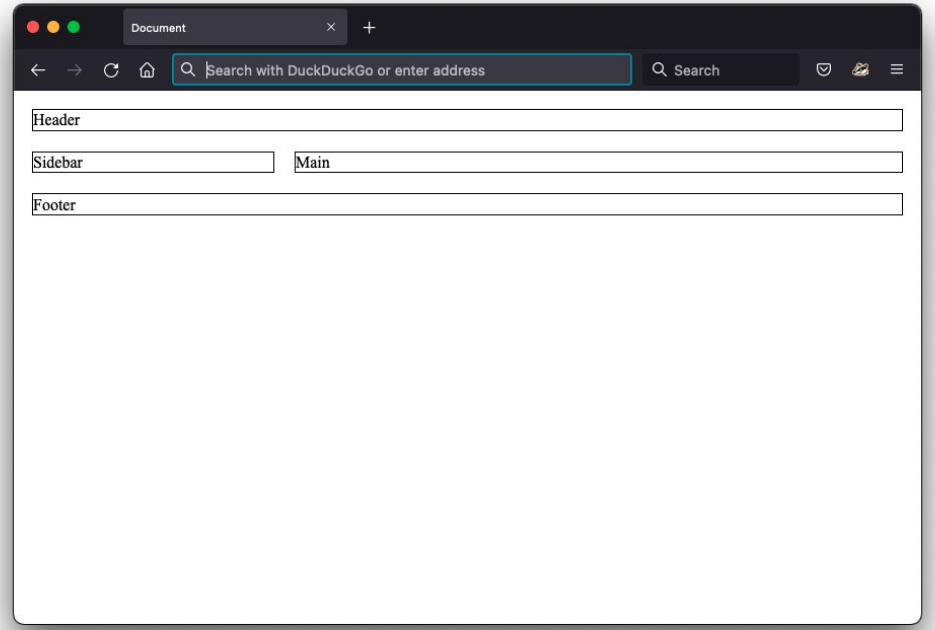
/* Modificar los gaps de la grilla de un proyecto. */ ✅

/* Utilizar fracciones como unidad de medida para definir el tamaño de los ítems de una grilla. */ ✅

/* Diferenciar cuándo ocupar flexbox o grid.*/

Layouts con Grid

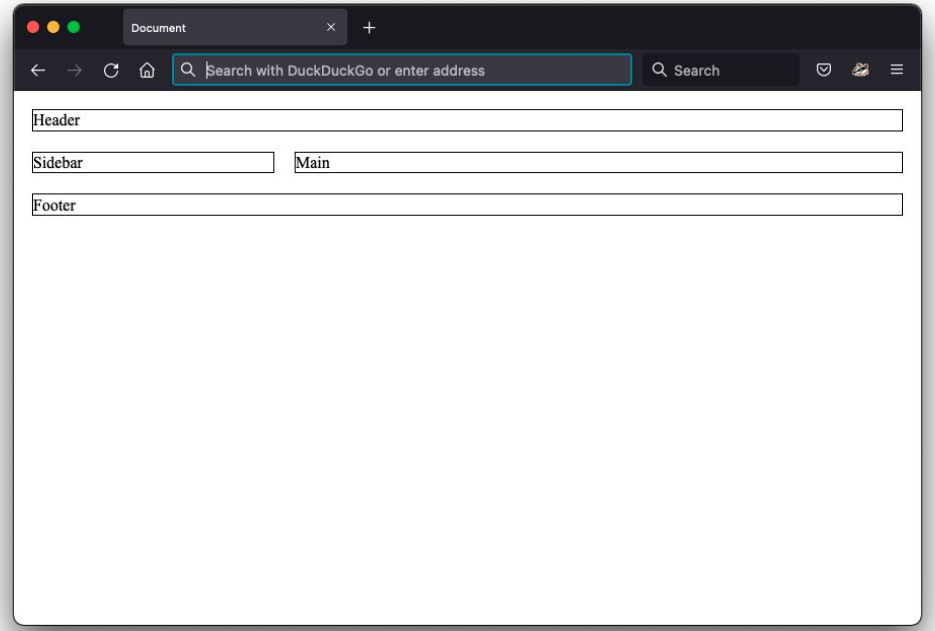
Creando layouts con CSS Grid



- Una de las mejores virtudes de CSS grid, es la capacidad de crear layouts.

Layouts con Grid

Grid area



- Para crear un layout, necesitamos tener diversos div (u otros elementos semánticos) con nombre.

Layouts con Grid

Grid area

```css
.header {
  grid-area: header
}

.main{
  grid-area: main
}

.footer{
  grid-area: footer
}
.grid-container{
  display: grid;
  grid-template-areas:
    'header'
    'main'
    'footer'
}
.box {
  border: solid 1px black;
  margin: 10px;
}
```

Le asignamos un nombre a cada sección del layout con grid-area, luego componemos un layout utilizando estos nombres dentro del contenedor grid.

```html
<div class="grid-container">
  <div class="header box"> Header </div>
  <div class="main box"> Main </div>
  <div class="footer box"> Footer </div>
</div>
```

Layouts con Grid

Un layout con múltiples columnas

```css
.header {
  grid-area: header
}

.main{
  grid-area: main
}

.sidebar{
  grid-area: sidebar
}

.footer{
  grid-area: footer
}
.grid-container{
  display: grid;
  grid-template-areas:
    'header header'
    'sidebar main'
    'footer footer'
}
```

La clave está en grid-template-area, que nos permite “dibujar” con los nombres de las áreas.

```html
<div class="grid-container">
  <div class="header box"> Header </div>
  <div class="sidebar box"> Sidebar </div>
  <div class="main box"> Main </div>
  <div class="footer box"> Footer </div>
</div>
```

{desafío} latam_

Layouts con Grid

Un layout con múltiples columnas

Probemos agregando una columna más en grid-template área, utilizando header, main y footer (uno por línea).

```css
.header {
  grid-area: header
}

.main{
  grid-area: main
}

.sidebar{
  grid-area: sidebar
}

.footer{
  grid-area: footer
}
.grid-container{
  display: grid;
  grid-template-areas:
    'header header'
    'sidebar main'
    'footer footer'
}
```

```html
<div class="grid-container">
  <div class="header box"> Header </div>
  <div class="sidebar box"> Sidebar </div>
  <div class="main box"> Main </div>
  <div class="footer box"> Footer </div>
</div>
```

Ejercicio

- Construir un layout utilizando grid-template-areas.
- El layout debe tener un menú arriba, una sección principal, un sidebar a la derecha y un footer abajo.

Ejercicio
¡Manos al teclado!

Flexbox vs Grid

¿Cuándo usar cada uno?

Ambas son herramientas poderosas, pero cada una tiene sus fortalezas específicas:

Flexbox: Layout Unidimensional

- Trabaja en una dirección (fila O columna)
- Ideal para distribución de elementos en línea
- Perfecto para componentes pequeños

Grid: Layout Bidimensional

- Trabaja en dos direcciones (filas Y columnas)
- Ideal para layouts complejos con áreas definidas
- Perfecto para estructuras de página completa

¿Cuándo usar Flexbox?

Usa Flexbox cuando necesites:

- **Alinear elementos** en una barra de navegación
- **Distribuir botones** en un formulario
- **Centrar contenido** verticalmente
- **Crear tarjetas** que se adapten al contenido
- **Organizar elementos** en una sola dirección

```
.navbar {
        display: flex;
        justify-content: space-between;
        align-items: center;
}
```

¿Cuándo usar Grid?

Usa Grid cuando necesites:

- **Crear layouts** de página completa (header, sidebar, main, footer)
- **Galerías de imágenes** con control preciso
- **Dashboards** con múltiples secciones
- **Layouts complejos** que requieren filas Y columnas
- **Posicionamiento exacto** de elementos

```
.page-layout {
        display: grid;
        grid-template-areas: "header header" "sidebar main" "footer footer";
}
```

Guía de decisión rápida

Pregúntate:	Flexbox	Grid
¿Es un layout simple en una dirección?	✅	❌
¿Necesito control sobre filas Y columnas?	❌	✅
¿Es un componente pequeño?	✅	❌
¿Es un layout de página completa?	❌	✅
¿Necesito que los elementos se adapten al contenido?	✅	❌
¿Necesito posicionamiento exacto?	❌	✅

Regla de oro:

Flexbox: Para componentes y layouts simples

Grid: Para layouts complejos y estructuras de página

Objetivos

/*Crear una galería de imágenes y un layout con grid. */ ✅

/* Modificar los gaps de la grilla de un proyecto. */ ✅

/* Utilizar fracciones como unidad de medida para definir el tamaño de los ítems de una grilla. */ ✅

/* Diferenciar cuándo ocupar flexbox o grid.*/ ✅

Cierre

{desafío}
latam_

Reflexionemos

¿Existe algún concepto que no hayas comprendido?

{desafío} latam_

¿Qué sigue?

- Revisar la guía que trabajarán de forma autónoma.
- Revisar en conjunto el desafío.

{desafío} latam_

Academia de talentos digitales