También puedes incluir Popper y nuestro JS por separado.

```
<script src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.11.6/dist/umd/popper.min.js"
integrity="sha384-oBqDVmMz9ATKxIep9tiCxS/Z9fNfEXiDAYTujMAeBAsjFuCZSmKbSSUnQlmh/jp3"
crossorigin="anonymous"></script>

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/js/bootstrap.min.js"
integrity="sha384-mQ93GR66B00ZXjt0YO5KlohRA5SY2XofN4zfuZxLkoj1gXtW8ANNCe9d5Y3eG5eD"
crossorigin="anonymous"></script>
```

¿Qué es un contenedor?

Un contenedor es el "padre" de todos los elementos de nuestra página web. Es una etiqueta que como regla general, va a contener todas las otras etiquetas del contenido de nuestra página.

¿Qué tipos de contenedores nos ofrece BootStrap?

BootStrap 4 nos ofrece dos tipos de contenedores:

- container : Ocupará diferentes anchos dependiendo del tamaño de la pantalla. Puede ocupar todas la pantalla o dejar unos márgenes a izquierda a derecha aunque, en este caso, siempre estará centrado.

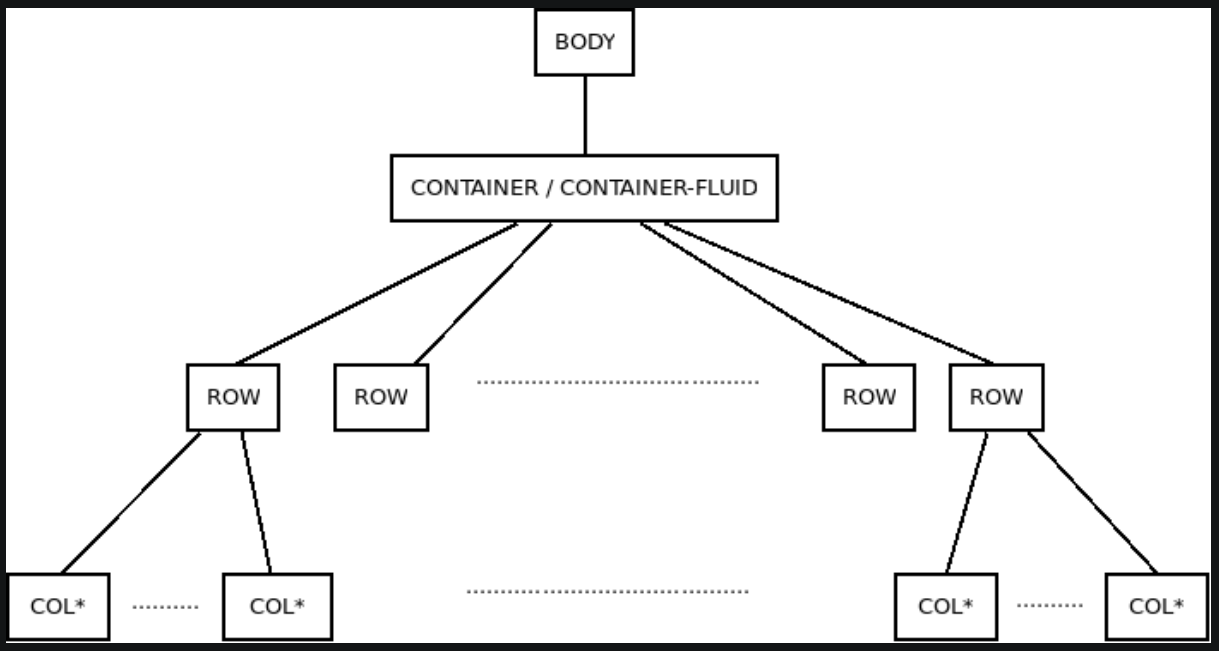
```
<body>

<div class="container">

</div>

</body>
```

- container-fluid: Ocupará todo el ancho (100%) de lo que podemos ver en el navegador.

```
<body>

  <div class="container-fluid">

  </div>

</body>
```

IMPORTANTE: Aunque los contenedores se pueden anidar unos dentro de otros son pocas las páginas que son tan complejas para necesitar contenedores anidados.

Como ya hemos hablado en el capítulo anterior es muy difícil que nos encontremos con páginas cuyos layouts sean tan complejos como para necesitar más de un contenedor. Así que, si nos ponemos de acuerdo en eso, en que sólo vamos a necesitar un contenedor, podemos fijar una estructura general que va a permanecer fijas para todos nuestros layouts:



Y dentro de cada fila de cada row, en principio podemos tener hasta 12 columnas, aunque ya veremos posteriormente que dado a que BootStrap 4 utiliza contenedores flex podemos usar las propiedades de estos contenedores flex para poner más columnas aunque, en realidad, es difícil que tengamos que utilizar más.

Todas estas columnas están pegadas unas a otras y tiene siempre un padding a izquierda y a derecha de 15px del que hablaremos posteriormente.

Otra cosa importante es que no tenemos que olvidarnos de utilizar *class="row"* ya que si nuestra estructura de columnas no está dentro de un contenedor con esa clase perderemos todas las propiedades de los contenedores flex, que son los sustentan la maquetación en BootStrap 4.

¿Qué es un breakpoint?

En el mundo del diseño responsivo un breakpoint es un tamaño de pantalla (en pixels) donde se produce un cambio en la disposición de los elementos que conforman nuestra página web.

Existen multitud de tipos de dispositivos, multitud de pantallas, pantallas que giran etc. Por lo tanto la elección de los distintos breakpoints no es algo trivial. Sin embargo, tenemos la suerte de que los muchachos de Twitter, usando su

experiencia y los datos que sus millones de usuarios les proporcionan, has decidido que la elección más adecuada son 4 breakpoints:

- En 576px
- En 768px
- En 992px
- En 1200px

Teniendo en cuenta estos 4 breakpoints podemos diferenciar entre 5 tipos de pantallas

- Pantallas extra pequeñas: Entre 0 y 576px
- Pantallas pequeñas: Entre 576px y 768px
- Pantallas medianas: Entre 768px y 992px
- Pantallas grandes: Entre 993px y 1200px
- Pantallas extra-grandes: A partir de 1200px

Juntando todo nos queda la siguiente tabla:

	Extra pequeñas	**Pequeñas**	**Medianas**	**Grandes**	**Extra Grandes**
Anchura máxima	576px	768px	992px	1200px	> 1200px
Prefijo de la clase	.col-	.col-sm-	.col-md-	.col-lg-	.col-xl-
Número de columnas	12				
Anchura del Gutter	30px (1px a cada lado de cada columna)				
Anidar	Sí				
Reordenar columnas	Sí				

De especial importancia son los prefijos CSS de las clases .col-* que son los que nos van a permitir establecer la anchura de los diferentes elementos que compongan nuestro layout. En definitiva son esas clases las que me proporciona BootStrap 4 para poder maquetar.

En el próximo apartado veremos cómo hacerlo.

En apartados anteriores vimos cuál era la estructura general de una página con BootStrap 4,vimos también cuáles eran los breakpoint establecidos y que, en principio, cada *row* puede contener hasta 12 columnas.

En este apartado vamos a ver como nosotros podemos establecer el tamaño de cada uno de los componentes de dicha *row* o filas y que además, podemos establecer distintos tamaños dependiendo del tamaño de nuestra pantallas.

Si queremos dar tamaño a cada de los componentes usaremos las clases vistas previamente, de esta manera :

- col-X: hará que ese contendor X/12 columnas en pantallas extra pequeñas.

```
<!-- col-6 ocupará justo la mitad en pantallas extra pequeñas -->
 <div class="col-6">
 </div>
```

- col-sm-X: hará que ese contendor X/12 columnas en pantallas pequeñas.

```
 <!-- col-sm-4 ocupará un tercio de la pantalla en pantallas pequeñas
-->
  <div class="col-sm-4">
  </div>
```

Y así sucesivamente para *col-md-X* para pantallas medianas, *col-lg-X* para pantalla grandes y *col-xl-X* para pantallas extragrandes.

Una cosa importante que hay que tener en cuenta es que si establecemos un tamaño para un cierto tamaño de pantallas ese tamaño se va a mantener para pantallas de mayor tamaño pero para pantallas menores ese div ocupará toda la pantalla, es decir si tenemos:

```
  <div class="col-md-6">
  </div>
```

Este div ocupará la mitad de la pantalla para pantallas de 768px (recordar los breakpoints) o superiores pero para tamaño menores ocupará todo el ancho de la fila que lo contiene.

Podemos también fijar a la vez anchos diferentes para distintos tipos de tamaños de pantallas. Para ello añadiremos varias clases a un mismo elemento, por ejemplo:

```
<div class="col-sm-6 col-md-4 col-xl-3">
</div>
```

En este caso ese div:

- Ocupará toda la fila para pantallas extrapequeñas (<576px)
- Ocupará la mitad de la fila para pantallas en 576px y 768px
- Ocupará un tercio de la fila para pantallas entre 768px y 992px
- Ocupará un cuarto para pantallas mayores de 992px

También es importante recordar que si, en un *row* nos pasamos de 12 columnas ese elemento que provoca que nos pasemos de las 12 columnas *"pasará"* debajo de los elementos de la fila actual ocupando el tamaño establecido.

Así por ejemplo si tenemos la siguiente estructura:

```
<div class="row">
    <div class="col-md-6">
    </div>
    <div class="col-md-4">
    </div>
    <div class="col-md-4">
    </div>
</div>
```

El tercer div irá abajo ya que 6+4+4 = 14 que es mayor que 12.

TAMAÑO AUTO
Es una de las novedades de BootStrap 4 que deriva directamente del uso de contenedor flex.

Usando elementos en los que no especifiquemos el número de columnas, el espacio que haya en la fila se va a distribuir de manera uniforme.

- col : Para todo tipo de pantallas
- col-sm : De pantallas pequeñas en adelante (>=576px)
- col-md : De pantallas medianas en adelante (>=768px)
- col-lg : De pantallas grande en adelante (>=992px)
- col-xl : Para pantallas de 1200px en adelante

Veamos el primer ejemplo:

```
<div class="row">
    <div class="col red">
    </div>
    <div class="col black">
    </div>
</div>
```

En este caso en cualquier tipo de pantallas esos dos elementos se repartirán a igual manera el ancho del contenedor principal.

Debemos de tener cuidado porque en caso de que lo que contengan esas partes ocupe más de la mitad no se respetará esa proporción. Veamos este caso con un ejemplo:

```
<div class="row">
    <div class="col red">
      <h1>PRIMERA PARTE</h1>
    </div>
    <div class="col black">
      <h1>SEGUNDA PARTE</h1>
    </div>
</div>
```

Si queremos que es reparto equitativo suceda para determinado de pantallas:

```
<div class="row">
    <div class="col-md yellow"></div>
    <div class="col-md pink"></div>
    <div class="col-md yellow"></div>
    <div class="col-md pink"></div>
    <div class="col-md yellow"></div>
</div>
```

En este caso todo se divide en 5 partes para pantallas menores de 768px pero en cuanto la pantalla es más pequeña que eso, todos los elementos pasarán a ocupar todo el centro de la pantalla.

Este funcionamiento abre la puerta a *rows* (filas) que tengan más de 12 columnas, por ejemplo una fila con 14 componentes:

```
<div class="row">
  <div class="col-md blue"></div>
  <div class="col-md pink"></div>
  <div class="col-md blue"></div>
  <div class="col-md pink"></div>
  <div class="col-md blue"></div>
  <div class="col-md pink"></div>
```

```
  <div class="col-md blue"></div>
  <div class="col-md pink"></div>
  <div class="col-md blue"></div>
  <div class="col-md pink"></div>
  <div class="col-md blue"></div>
  <div class="col-md pink"></div>
  <div class="col-md blue"></div>
  <div class="col-md pink"></div>
</div>
```

BootStrap 4 nos permite también ajustar el tamaño de las columnas al contenido de las mismas usando *col-{breakpoint}-auto*. Así por ejemplo:

```
`html <div class="row"> <div class="col-md-auto">Anchura ajustada</div> <div class="col-md-auto">Al contenido</div> <div class="col-md-auto">De las celdas que contiene la columna</div> </div>
```

IMPORTANTE: Debemos tener mucho cuidado cuando mezclemos tamaños automáticos junto tamaños, definidos por el usuario porque cuanto no quepan las columnas (y puede que no sumen 12) pasarán a la siguiente fila y se repartirán el espacio de la siguiente fila.

SALTO DE LÍNEA:
Si mientras estamos construyendo nuestro grid y queremos que haya un salto de línea podemos forzarlo de la siguiente manera:

```
  <div class="w-100"></div>
```

Con esa estructura aunque no hayamos ocupado las 12 columnas de nuestro grid podremos pasar a la siguiente fila. De esta manera:

```
<div class="row">

  <!-- Ocupamos 5/12 de la fila -->
  <div class="col-md-3"></div>
  <div class="col-md-2"></div>

  <!-- Forzamos el salto de línea -->
  <div class="w-100"></div>

  <!-- Ocupamos otra vez 5/12 pero en una nueva línea -->
  <div class="col-md-3"></div>
  <div class="col-md-2"></div>


</div>
```

Si nos damos cuenta esto mismo se podría haber conseguido utilizando varias *rows* (filas). De esta manera

```
<div class="row">

  <!-- Ocupamos 5/12 de la fila -->
  <div class="col-md-3"></div>
  <div class="col-md-2"></div>

</div>

<div class="row">

</div>

<div class="row">

  <!-- Ocupamos otra vez 5/12 pero en una nueva línea -->
  <div class="col-md-3"></div>
  <div class="col-md-2"></div>

</div>
```

De manera selectiva podemos establecer que ese cambio de línea se produzca en un determinado breakpoint de los establecidos por BootStrap 4.

FILAS SIN GUTTER (PADDING ENTRE ELEMENTOS):

Por defecto BootStrap 4 introduce separación entre el contenido de las columnas mediante *paddings* y márgenes a derecha e izquierda. Es lo que se denomina *gutter*

Si ése no es el efecto que queremos en nuestra maquetación podemos evitarlo añadiendo al elemento que tenga la clase *row* la clase *no-gutters*.

Así de esta manera las columnas o elementos de la fila no tendrán ese padding con la posibilidad de que el contenido de la misma queden pegados.

Un ejemplo sería:

```
<div class="container">
  <div class="row">
    <div class="col-md-6"><h3>Elemento con Gutter</h3></div>
    <div class="col-md-6"><h3>Elemento con Gutter</h3></div>
  </div>

  <div class="row no-gutters">
    <div class="col-md-6"><h3>Elemento sin gutter</h3></div>
    <div class="col-md-6"><h3>Segundo Elemento</h3></div>
  </div>
</div>
```

Tal y como hemos visto en apartados anteriores podemos dar diferentes tamaños a los distintos componentes o columnas de una *row* o fila.
Adicionalmente podemos establecer no sólo el tamaño de la columna si no también la posición a partir de la cuál se va a posicionar.

Para conseguir eso debemos añadir la clase *offset-X* donde X es el número de columnas a la izquierda que desplazaremos el elemento que queremos posicionar.

Este desplazamiento puede ser también establecido de manera diferente para cada tamaño de pantalla:

- offset-sm-X : Para desplazamientos para pantallas entre 576px y 768px.
- offset-md-X : Para desplazamientos para pantallas entre 768px y 992px.
- offset-lg-X : Para desplazamientos para pantallas entre 992 y 1200px.
- offset-xl-X : Para desplazamientos para pantallas mayores de 1200px.

Por ejemplo:

```
<div class="container-fluid">
  <div class="row">
    <div class="col-sm-4 offset-md-1 red"></div>
    <div class="col-sm-4 offset-sm-4 offset-md-2 blue"></div>
  </div>
</div>
```

En este ejemplo para pantallas mayores de 768px el primer div está desplazado una columna a la izquierda y ambos bloques se separan dos columnas mediante el uso de un desplazamiento de dos columnas en el segundo bloque.

Además para pantallas entre 576px y 768px no hay desplazamiento para el primer bloque y hay un desplazamiento de 4 columnas para el segundo bloque consiguiendo, de esta forma que ese bloque quede totalmente pegado a la derecha.

IMPORTANTE: En esta nueva versión de BooStrap 4 hay un cambio importante en el nombre de las clases para desplazar los elementos de la fila. En versiones anteriores esas clases de denominaban *col-yy-offset-X* donde yy son las siglas del tamaño de la pantalla.

Como ya hemos descrito en apartados anteriores, BootStrap 4 utiliza contenedores flex. Precisamente utilizando las propiedades de estos elementos podremos fijar la alineación de los componentes de las *rows* o filas.

Podremos establecer:

- La alineación vertical de los elementos de la fila.
- La alineación horizontal de los elementos de la fila.

Tanto para establecer la alineación horizontal como la vertical deberemos añadir una nueva clase al contenedor *row* o fila.

Alineación vertical

Si queremos alinear los distintos elementos que componen una fila de manera vertical podremos hacerlo añadiendo a la *row* fila una de estas tres clases:

- align-items-start : Los elementos de la fila se alinearán verticalemente con el borde superior de la fila.

```
 <div class="row align-items-start">
</div>
```

- align-items-center : Los elementos de la fila se alinearán centrados verticalmente.

```
 <div class="row align-items-center">
</div>
```

- align-items-end : Los elementos de la fila se alinearán verticalemtne con el borde inferior de la fila.

```
 <div class="row align-items-end">
</div>
```

Alineación horizontal

Para fijar la alineación horizontal de los elementos en la *row* (fila) dispondremos de las siguientes clases:

- justify-content-start : Los elementos empezarán en la parte izquierda y ocuparán el tamaño horizontal establecido. Es la opción por defecto.

```
 <div class="row justify-content-start">
</div>
```

- justify-content-center : Los elementos de la fila se centrarán horizontalmente.
- justify-content-end : Los elementos de la fila se alinearán al final (la derecha normalmente) de la fila.
- justify-content-around : El espacio restante se distribuirá de manera equitativa alrededor de los elementos empezando y acabando con espacios libres.

- justify-content-between : El espacio restante se distribuirá de manera equitativa entre los distintos elementos estando uno de ellos totalmente a la derecha y otro totalmente a la izquierda.

Con todas estas posibilidades y lo visto anteriormente es más que suficiente para casi cualquier Layout pero existen más posibilidades que podremos descubrir en la documentación oficial.

BootStrap 4 añade clases que nos van a permitir modificar los márgenes y los paddings de las columnas. Según la documentación oficial son las *Utilidades de espaciado*.

Márgenes entre columnas

Para establecer los márgenes entre las columnas usaremos la clas *m{lado}-{tamaño}* o si queremos distinguir según los distintos tamaños de pantalla:

- m{lado}-sm-{tamaño} Para pantallas entre 576px y 768px.
- m{lado}-md-{tamaño} Para pantallas entre 768px y 992px.
- m{lado}-lg-{tamaño} Para pantallas entre 992px y 1200px.
- m{lado}-xl-{tamaño} Para pantallas de más de 1200px.

Pudiendo ser lado:

- t para margen superior (top).
- b para margen inferior (bottom).
- l para margen izquierdo (left).
- r para margen derecho (right).
- x para los márgenes superior e inferior.
- y para los márgenes izquierdo y derecho.
- En blanco si es para todos los lados.

Y pudiendo ser tamaño de tamaño:

- 0 : No hay margen
- 1 : 0.25rem
- 2 : 0.25rem
- 3 : 1rem
- 4 : 1.25rem
- 5 : 3rem
- auto : Para clases que establecen una margen auto

Centrado horizontal

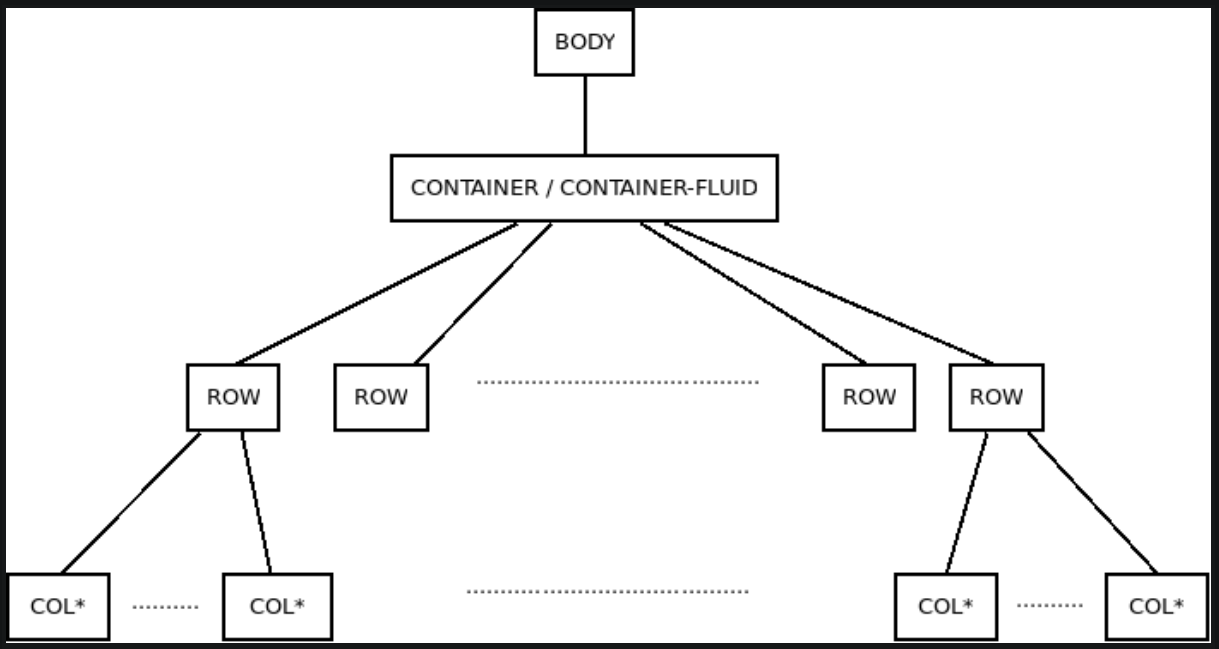
Relacionado con el tema de los márgenes hay que destacar que BooStrap 4 introduce una nueva clase *mx-auto* que permite centrar horizontalmente un elemento dentro de su contenedor siempre que tenga una

Paddings

La notación de las clases para establecer los paddings en las columnas es muy similar a la notación de las clases para establecer márgenes entre las columnas. Simplemente tenemos que cambiar la *m* por la *p*

ANIDANDO FILAS:

Hasta ahora estamos estado maquetando conforme a la estructura que hemos definido en apartados anteriores.



Esta estructura establece una jerarquía *contenedor -> row -> col*, es decir, todas las filas tienen elementos que única y exclusivamente pertenecen a dicha fila.

Sin embargo, los layout reales son más complejos que eso y para conseguir esas estructuras complejas vamos a tener que anidar filas (*row*) dentro de filas. Debemos de tener en cuenta que cada una de esas filas hijas podrá tener, de igual manera, hasta 12 columnas.

En la siguiente imagen tenemos un ejemplo de ese tipo de layouts con las distintas filas señaladas.