ARTÍCULO

CONCEPTOS FUNDAMENTALES EN BOOTSTRAP 4:
RESPONSIVE DESIGN, CONTAINER & GRID.

BOOTSTRAP & RESPONSIVE DESIGN

La primera definición de Bootstrap, extraída de la página oficial, dice que "*Bootstrap is an open source toolkit for developing with HTML, CSS, and JS. Quickly prototype your ideas or build your entire app with our Sass variables and mixins, responsive grid system, extensive prebuilt components, and powerful plugins built on jQuery*", i. e., Bootstrap es un framework gratis, mantenido por Twitter pero liberado como open-source, especializado en front-end para ayudar a hacer del desarrollo web más rápido y fácil.

Actualmente se encuentra en la versión 4, que ya da soporte en la mayoría de los navegadores de uso común, excepto para Internet Explorer 9 y menores. Para poder utilizar Bootstrap en este navegador, se debe usar la versión 3 de Bootstrap.

En el sitio web oficial o directamente en el paquete de descarga o instalación, Bootstrap ya incorpora no sólo plantillas base para modificar y no tener que desarrollar desde cero el código, sino también plantillas de CSS -las famosas Sass (Syntactically Awseome Style Sheets)- ya pre-desarrolladas para tipografías, formularios, tablas, paginaciones, modales, imágenes, carruseles, plugins de JS y mucho más.

Pero la principal característica de Bootstrap, reside en permitir crear muy fácilmente webs con ***diseño responsivo*** o "Responsive Web Designs". El diseño responsivo consiste básicamente en crear sitios webs que se ajusten inteligente y automáticamente para que se vean bien en cualquier dispositivo o resolución gráfica. Para ello usa clases con funcionalidades específicas, como la clase `.container` y un sistema de layout llamado "grid system".

Bootstrap también se basa en la filosofía "mobile first", la otra faceta del diseño responsivo, con estilos que son parte del núcleo del framework, pero en este trabajo, sólo nos aproximaremos al diseño responsivo como tal.

LA CLASE .CONTAINER

La clase `.container` provee los fundamentos para el layout general de una página y sus bloques de contenido.

Bootstrap provee específicamente la clase `.container` como un "contenedor" para envolver elementos y dar cabida al sistema de grilla o "grid system". Se utiliza para alinear

de forma pareja los margenes izquierdo y derechos de la página. La clase `.container` puede ser fija (“fixed”) o fluida (“fluid”).

Fixed Container

Un contenedor fijo es un `.container` de ancho (“width”) responsivo fijo. Mientras uno reajusta la resolución de la ventana del navegador, el ancho del contenedor se mantiene intacto hasta que pasa un punto de quiebre en la resolución (especificado por el programador en el código desarrollado), momento en el cual el contenedor se reajustará a la nueva dimensión de ancho según lo especificado:

```
<div class="container">
  <!-- Content here -->
</div>
```

A continuación, se especifican las dimensiones (en pixels, i.e., px) para los puntos de quiebre (“breakpoints”) según resoluciones de acuerdo a los tamaños de diversos dispositivos:

Breakpoints según resoluciones por dispositivo

ELEMENTO	TAMAÑO	DISPOSITIVO
COL (xs)	< 575	Por ej. celulares antiguos
SM	576 - 767	Por ej. smartphones
MD	768 - 991	Por ej. tablets
LG	992 - 1199	Por ej. notebooks
XG	1200 <	Por ej. monitores grandes/tv’s

Fluid Container

Un `.container-fluid` (contenedor fluido) extiende completamente el ancho del “viewport” o ventana gráfica *(“Una ventana gráfica es una región de visualización de polígonos en gráficos de computadora”* [extraído de Wikipedia,] i. e., el área visible dentro de la ventana del navegador. Es la etiqueta que mejor representa la web en movilidad, ya que nos permite indicar cómo se verá un proyecto web en los dispositivos móviles. Le dice al dispositivo qué área de pantalla está disponible al renderizar un documento, el nivel de escalado y el zoom que debe mostrar inicialmente. Hoy en día es un estándar que ya está soportado por la mayoría de dispositivos):

```
<div class="container-fluid">
  <!-- Content here -->
</div>
```

El contenedor se expandirá y contraerá de modo fluido mientras uno reajusta el tamaño de la ventana del navegador.

Esto contrasta con el ancho del contenedor fijo `(.container-fixed)` que aparenta dar "saltos" hacia el nuevo tamaño una vez que pasa el punto de quiebre.

Si bien se pueden crear páginas con Bootstrap sin utilizar la clase `.container`, son un requerimiento cuando se usa el sistema de grillas ("grid system").

GRID SYSTEM (o el sistema de grilla)

Este Sistema es usado para crear layouts responsivos. El mismo representa la forma en que los elementos se alinean en la página, según diferentes resoluciones. Es un conocimiento principal para trabajar con Bootstrap y entender cómo funciona es de vital importancia.

La grilla ("grid") consiste en containers, filas ("rows") y columnas ("columns"). El container es la raíz del sistema y, como vimos, se usa para controlar el ancho del layout, contrarrestando los márgenes negativos de la fila.

Filas en Bootstrap 4

Las filas de Bootstrap 4 son rodajas horizontales de la pantalla. Sólo se utilizan como envoltorios para las columnas. El único propósito de una fila es contener una o más columnas. Se utiliza con la clase `.row`:

```
<div class="row">
  ...
</div>
```

Algunos de los puntos más importantes para recordar cuando se usan filas:

- **Sólo se usan para contener columnas.** Si se coloca otro elemento dentro de `.row` junto con las columnas no se obtendrán los resultados esperados.
- **Se deben colocar en containers.** Si no se hace esto, se obtendría un scroll horizontal en la página. Esto sucede ya que las filas tienen un margen negativo a izquierda y

derecha de unos 15px. El `.container` tiene un padding de 15px, contrarrestando dichos márgenes.

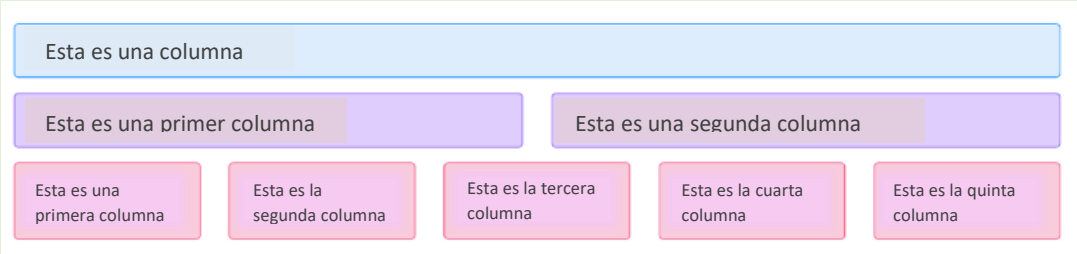
- **Las columnas deben ser hijas de las filas.** De otra manera, no se alinearían. Las filas y columnas se crearon para trabajar en conjunto bajo esta jerarquía.

Columnas en Bootstrap 4

Las columnas crean divisiones horizontales a través del viewport.

Si uno coloca una sola columna en la fila, esta ocupará todo el ancho. Si se colocan dos columnas, cada una ocupará un 50% del ancho. Tres columnas, cada una un 33% y así hasta llegar a dividir la pantalla en un máximo de doce (12) columnas:

```
<div class="container">
  <div class="row">
    <div class="col">
      ...
    </div>
  </div>
  <div class="row">
    <div class="col">
      ...
    </div>
    <div class="col">
      ...
    </div>
  </div>
  <div class="row">
    <div class="col">
      ...
    </div>
    <div class="col">
      ...
    </div>
    <div class="col">
      ...
    </div>
    <div class="col">
      ...
    </div>
    <div class="col">
      ...
    </div>
  </div>
</div>
```

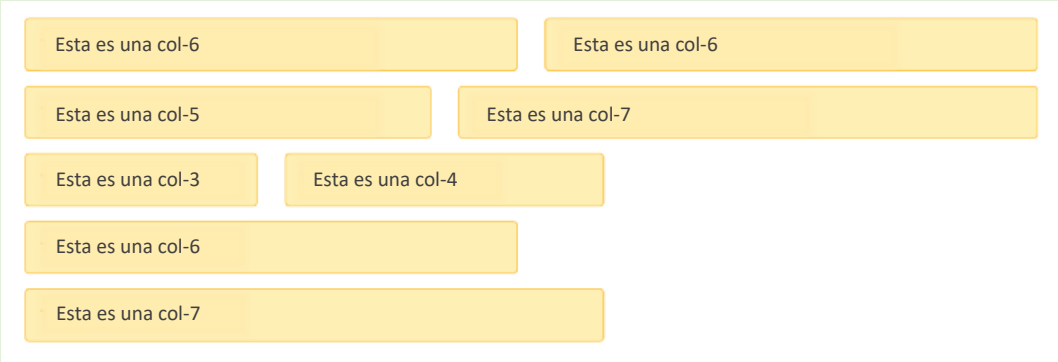


Definir tamaños para las columnas

El uso de la clase `.col` define el ancho de la columna de forma dinámica. Eso significa que, dependiendo de la cantidad de columnas en una fila, el ancho de una columna será el ancho del container dividido en el número de columnas. Pero hay otra manera de definir las columnas, usando clases y definiendo su tamaño.

Como se mencionó previamente, la grid de Bootstrap 4 consiste en 12 columnas. Se puede elegir cualquier tamaño entre 1 y 12 para las columnas. Si se quisiera hacer 4 columnas iguales, se debería usar `.col-3` para cada una, ya que 4 * 3 columnas c/u = 12. Para hacer columnas de diversos tamaños, sólo debemos jugar con esta pequeña fórmula. Por ejemplo:

```
<div class="row">
  <div class="col-6">
    ...
  </div>
  <div class="col-6">
    ...
  </div>
</div>
<div class="row">
  <div class="col-5">
    ...
  </div>
  <div class="col-7">
    ...
  </div>
</div>
<div class="row">
  <div class="col-3">
    ...
  </div>
  <div class="col-4">
    ...
  </div>
</div>
<div class="row">
  <div class="col-6">
    ...
  </div>
  <div class="col-7">
    ...
  </div>
</div>
```



Si la suma de `.col` en la fila no suma 12, puede no llegar a completar los vacíos o moverse a otra línea, desconfigurando el layout deseado.

Definir puntos de quiebre ("breakpoints") para las columnas

Para poder tener diferentes layout en diferentes pantallas se deben usar los puntos de quiebre o "breakpoints". La correcta utilización de los mismos es uno de los aspectos esenciales en el diseño responsivo.

Un "breakpoint" en Bootstrap 4 es una variable que refiere a una resolución de pantalla. Cuando se especifica un "breakpoint" para una clase, se está ordenando a esa clase ***que esté activa sólo para resoluciones que son al menos tan grandes*** como el número que representa dicho "breakpoint".

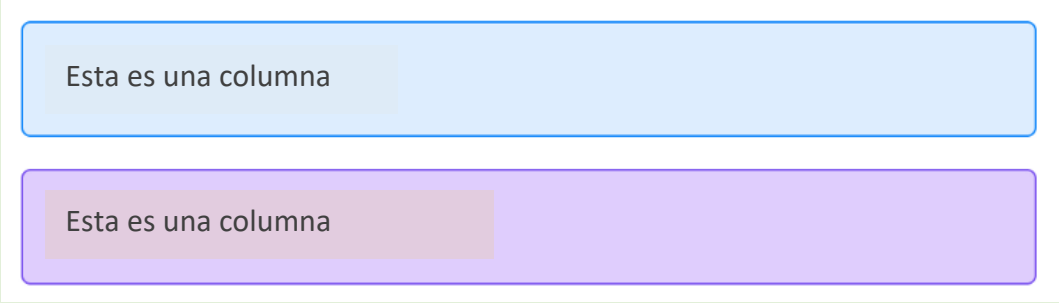
Se pueden repasar los "breakpoints" en la tabla que se encuentra al comienzo del documento.

La clase por defecto es la clase `.col-[breakpoint]`. Al usar esta clase, se está definiendo el comportamiento para las columnas sólo cuando se muestren en dispositivos que tienen una resolución al menos del tamaño del "breakpoint" definido. Hasta llegar al punto de quiebre especificado, las columnas se alinearán de forma vertical por defecto. Más allá del "breakpoint" lo harán horizontalmente gracias a la clase.

Cabe aclarar que, dado que la clase xs es el "breakpoint" por defecto, el infijo –xs que se usaba en Bootstrap 3 ya no se usa en la versión 4. Por lo cual, en lugar de usar `.col-xs-6`, sólo debemos usar `.col-6`.

Por ejemplo: si se quisiera mostrar dos columnas seguidas de forma vertical en pantallas pequeñas y en la misma línea para pantallas grandes, se debería usar el "breakpoint" `.col-lg` y verificar la apariencia de las columnas en diferentes pantallas. Para resoluciones menores a 992px, como en dispositivos móviles y tablets (breakpoint lg), las columnas se mostrarán verticalmente:

```
<div class="row">
 <div class="col-lg">
   ...
 </div>
 <div class="col-lg">
    ...
 </div>
</div>
```



Para dispositivos que tienen una resolución igual o mayor (tales como notebooks o televisores) al "breakpoint" ((≥992px) las columnas se mostrarán en la misma fila:

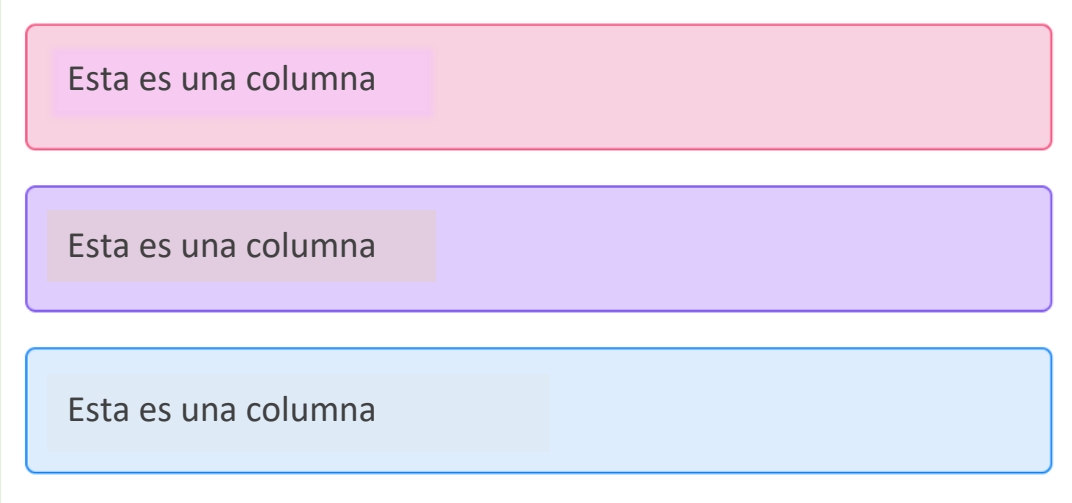
Definir tamaños para los puntos de quiebre ("breakpoints") de las columnas

Se pueden combinar los tamaños y "breakpoints" y usarlos en una única clase `.col-[breakpoint]-[tamaño]`.

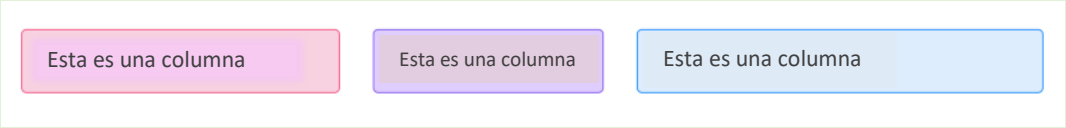
Por ejemplo, si se quiere tres columnas de diferentes tamaños para alinear en una fila, empezando con la resolución para notebooks:

```
<div class="row">
  <div class="col-lg-4">
    ...
  </div>
  <div class="col-lg-3">
    ...
  </div>
  <div class="col-lg-5">
    ...
  </div>
</div>
```

Resultado para resoluciones menores a 992px:



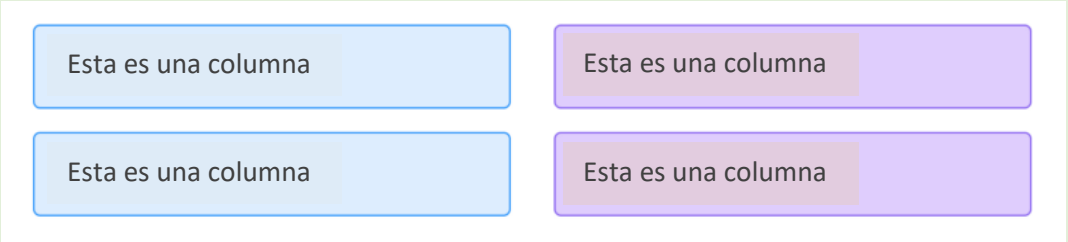
Resultado para pantallas ≥992px:



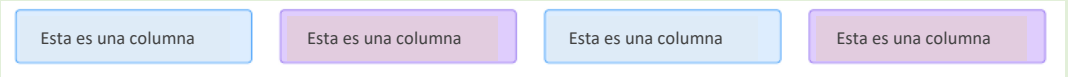
De igual manera si quisiéramos mostrar diferentes disposiciones para cada tipo de dispositivo, sólo debemos agregar múltiples clases para una única columna. Esto describirá el comportamiento para cada resolución.

```
<div class="row">
  <div class="col-sm-6 col-lg-3">
    ...
  </div>
  <div class="col-sm-6 col-lg-3">
    ...
  </div>
  <div class="col-sm-6 col-lg-3">
    ...
  </div>
  <div class="col-sm-6 col-lg-3">
    ...
  </div>
</div>
```

El resultado se mostrará de la siguiente manera en tablets:



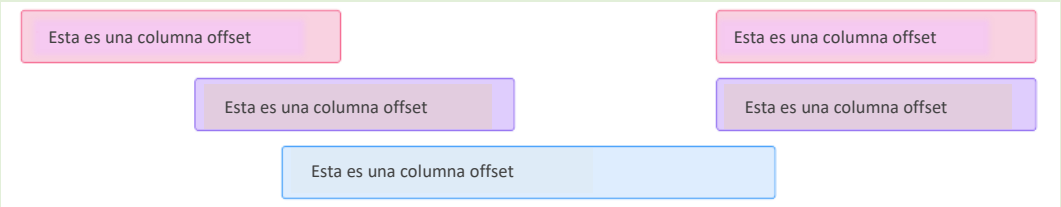
Y de la siguiente forma en notebooks y mayores resoluciones:



Columnas Offset

Si se desea dejar espacios entre columnas se dispone de la clase `.offset-[breakpoint]-[tamaño]` usada en conjunto con `.col-[breakpoint]-[tamaño]`. Esto se representaría como dejar una columna vacía ***antes*** de la siguiente columna. Se puede usar esta clase en cualquier fila. Por ejemplo:

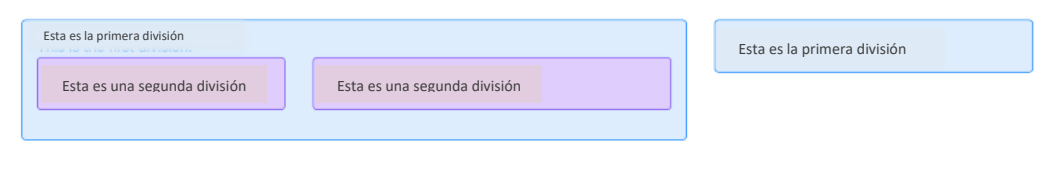
```
<div class="row">
  <div class="col-md-4">
     ...
  </div>
  <div class="col-md-4 offset-md-4">
     ...
  </div>
</div>
<div class="row">
  <div class="col-md-4 offset-md-2">
     ...
  </div>
  <div class="col-md-4 offset-md-2">
     ...
  </div>
</div>
<div class="row">
  <div class="col-md-6 offset-md-3">
     ...
  </div>
</div>
```



Anidar columnas

También se puede agregar una fila dentro de una columna. La fila en cuestión tendrá el ancho de su columna padre y será dividida en 12 columnas más chicas que también podrán ser referenciadas a través de la clase `.col-[breakpoint]`. Por ejemplo:

```
<div class="row">
    <div class="col-md-8">
        ...
        <div class="row">
            <div class="col-md-5">
                ...
            </div>
            <div class="col-md-7">
                ...
            </div>
        </div>
      </div>
    </div>
    <div class="col-md-4">
       ...
    </div>
</div>
```



De esta manera, se puede organizar la información en muchos y diferentes niveles. Las columnas son una forma sencilla de organizar y gestionar el espacio en la página.

Antes de terminar, conviene resaltar algunos puntos claves del diseño responsivo al usar la “grid” de Bootstrap 4:

- Las columnas se apilan de forma vertical (y se expanden al ancho completo) en la resolución de pantallas más pequeñas *a menos que* se use una clase `.col-[breakpoint]` específica al momento de codificar, lo cual previene el apilamiento vertical.
- Las clases más pequeñas de la “grid” también aplican en pantallas más grandes a menos que se sobrescriban de manera específica para ancho de pantallas más grandes. Por ejemplo, el código `<div class=”col-md-6"></div>` cumple la misma función que si escribiésemos `<div class=”col-md-6 col-lg-6"></div>`. Por lo tanto, sólo se necesita usar la clase para el ancho más pequeño para el cual se desea dar soporte.
- Las filas tienen la característica intrínseca de ser `display:flex`, por lo tanto, todas las columnas de la misma fila tienen la misma altura. Se deben usar los comandos

de `auto-margin` o Flexbox (`align-items` y `justify-content`) para alinear horizontal o verticalmente.

PÁGINA DESARROLLADA CON BOOTSTRAP

En el siguiente enlace, bootstrap4.tp.programacion3.utn, podrán encontrar una carpeta que contiene una página desarrollada con Bootstrap 4. Sólo deben descargar la carpeta haciendo click derecho y dando click en la opción "Descargar". Se descargará un comprimido .zip; al descomprimirlo, tendremos una carpeta con todos los recursos para ejecutar una página web desarrollada con Bootstrap. Sólo debemos ejecutar el archivo index.html en el navegador. Podrán también apropiarse del código (abriendo el mismo archivo en su editor de código preferido) y ¡probar uds mismos con Bootstrap 4!