Gym Landing Docs

Diagrama

Descripción generada automáticamente

Bootstrap Doc for me

Tema: Layout

* Con este primer tema vamos a lograr generar como un primer diseño, como los cuadros que yo hacia donde iba a poner donde iba x contenido y así pero con Bootstrap.

Subtema -> Breakpoints:

Los breakpoint son como anchos customizables los cuales determinan como nuestro diseño responsivo se comporta en equipos diferentes o diferentes viewports sizes in Bootstrap.

En pocas palabras es así:

Si la pantalla es 400X400px haga X cosa con el diseño.

Si es menor a ese tamaño, haga esto otro.

Y así, eso es lo que hacen los breakpoints, con haga X me refiero a por ejemplo usarlos cuando nuestros layout se debe adaptar a un particular viewport or device size.

Normalmente Bootstrap lo que intenta hacer es que un diseño pueda funcionar con el mínimo de detalles aplicando el smallest breakpoint.

Estos son los available breakpoints:

Imagen que contiene Interfaz de usuario gráfica

Descripción generada automáticamente

En otras palabras: Digamos que tenemos X container, y queremos usar el breakpoint que se llama small (sm), una vez la pantalla sea >= 576px, este container modificara su width, de igual manera con los otros, si este container le asignamos el breakpoint extra extra large (xxl) >= 1400px, este container modificara su ancho una vez se supere la marca de 1400px, en caso de que sea menor < el width del container será del 100%.

Aquí lo podemos ver

Tabla

Descripción generada automáticamente

Subtema -> Grid System:

Aquí nos menciona que Bootstrap al parecer usa como un sistema propio de Grid, no es como el grid tradicional que yo uso si no como que usa div con el valor de flexbox, en vez de usar un grid normal.

¿Y cómo usamos este grid? Bueno el sistema de grid de Bootstrap usa como una serie de contenedores, rows y columnas para diseñar y alinear el contenido, estos valores están construidos con flexbox.

Aquí hay un ejemplo de como luciría:

    <div class="container text-center">

      <div class="row">

        <div class="col">

          Column

        </div>

        <div class="col">

          Column

        </div>

        <div class="col">

          Column

        </div>

      </div>

    </div>

Como se puede ver prácticamente se aplica la lógica antes mencionada, primero vemos el contenedor, luego las rows y por ultimo las columnas, haciendo una serie como tal.

Y algo así es como funciona el grid de Bootstrap.

El ejemplo de arriba crea 3 columnas con el mismo ancho across all devices and viewports usando clases de grid predefinidas y todo esto se centra usa el div parent .container.

¿Bueno, y cómo funciona este tipo de grid?

* Este grid de Bootstrap soporta lo siguiente -> six responsive breakpoints. Esto es prácticamente aplicando la lógica explicada antes, estos breakpoints se le pueden añadir a las columnas del grid (e.g. .col-sm-4 also applies to sm, md, lg, xl, and xxl). Esto significa que si por ejemplo, x columna tiene el valor de sm, cada vez que la pantalla sea menor a 576px esta columna va a ocupar el 100% del width, desplazando las otras columnas hacia abajo en caso de que haya mas.
* Containers center and horizontally pad your content. Bueno aqui nos da como un consejo de usar .container o .container-fluid, yo en este caso uso container fluid porque quiere que mi container ocupe el 100% de la pantalla en cualquier dispositivo.
* Rows are wrappers for columns. Cada columna tiene como un padding horizontal (called gutter) para controlar el espacio entre ellas, mas que el espacio, es mas para controlar el espacio entre EL CONTENIDO de ellas. Luce algo así el espacio horizontal:



* Columns are incredible flexible. Hay mas de 12 plantillas de columnas por row, lo cual nos permite crear diferentes combinaciones de elementos that span any number of colums. Las clases de las columnas indican el numero de plantillas de columnas to span (e.g., col-4 spans four). Los anchos son seteados en porcentajes así siempre podemos tener el mismo tamaño relativo.
* Gutters are also responsive and customizable. Gutter classes are available across all breakpoints with all the same sizes as our margin and padding spacing. Bueno, esto lo que quire decir es que le podemos meter mas gutter a nuestras columnas, usando .gy-\* para y en el axis y gx para x en el axis. Con g-0 se remueve el gutter.

Subtema -> Grid options:

Aqui nos menciona lo ya mencionado, de que podemos usar los breakpoints también en las columnas, solo 6 (los ya mencionados). Aquí hay una tabla donde lo podemos visualizar:

Texto, Correo electrónico

Descripción generada automáticamente

Subtema -> Auto-layout columns

Aquí al parecer lo que nos quiere ensenar es como usar breakpoint-specific column classes para hacer un easy column sizing sin necesidad de usar una clase tan especifica como col-sm-6

* Equal-width – Aquí nos da um ejemplo de dos grid que aplican para cualquier dispositivo o viewport, from xs to xxl. Prácticamente si anadimos cualquier cantidad de unit-less classes para cada breakpoint que necesitemos, toda las columnas van a mantener el mismo width.
* Setting one column width – Auto-layout for flexbox grid columns, tambien significa que nosotros podemos setearle el width a una columna y hacer que las otras columnas se resize automáticamente alrededor de esta que hemos modificado.

Un ejemplo es algo así

<div class="container text-center">

      <div class="row">

        <div class="col">

          1 of 3

        </div>

        <div class="col-6">

          2 of 3 (wider)

        </div>

        <div class="col">

          3 of 3

        </div>

      </div>

</div>

Es como hacer esto col-n (n numero), donde n numero determina si se le mete mas width a esa columna.

* Variable width content

Use col-{breakpoint}-auto classes to size columns base don te natural width of their content.

Algo asi:

class=”col-md-auto”

Subtema -> Responsive classes

El grid de Bootstrap incluye como seis tiers de clases predefinidas para construir complex responsive layouts.

Customize the size of your columns on extra small, small, medium, large, or extra-large devices however you see fit.

* All breakpoint – Para grids que queremos que sean iguales desde el dispositivo mas pequeño al mas grande, usemos class=”col-\*” classes. Solamente es reemplazar el \* con el numero especifico para cuando necesitamos un particularly sized columna, en otro feel free to stick to

class=”col”

* Stacked to horizontal – luego dice que podemos a crear como un sistema basic de grid el cual empiece todo stackeado y luego se convierta horizontal at te small break point (sm), esto usando un set de class=”col-sm-\*” algo así luciría:

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

* Mix and match – Aquí pues nos indica de usar un mix de lo aprendido, algo así: Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

  Descripción generada automáticamente
* Row columns – Esto lo que hace es setear el numero de columnas que queremos que salga en una row, y luciría algo así:

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Esto lo que hace es que pone limita dos columnas antes de hacer un salto de línea. Si ponemos col-4 estas 4 columnas se verían así:

Forma

Descripción generada automáticamente con confianza baja

Subtema -> Nesting

Aqui este tema a lo único que quiere llegar a explicar es como hacer el nesting del contenido en un grid, que se hace por ejemplo, si ya tenemos un grid, lo único que debemos de hacer es añadir una nueva row y un set de col-sm-\* en una columna ya existente.

Nested rows should include a set of columns that add up to 12 or fewer (it is not required that you use all 12 available columns).

Luciria algo asi:

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Subtema -> Columns

aquí nos enseñan como usos extras que podemos usar en las columnas.

How they work?

* Columns build on the grid’s flexbox architecture. Flexbox means que tenemos opciones para cambiar columnas individuales y modificar grupo de columnas at the row level. Prácticamente nosotros elegimos como las columnas se hacen mas pequeñas, crecen o cambian de alguna manera.
* CUANDO TRABAJAMOS CON GRID LAYOUTS, TODO EL CONTENIDO VA EN LAS COLUMNAS.

Prácticamente la jerarquía iría de la siguiente manera – container -> row -> column -> content, en ocasiones muy raras puede que combinemos column y content.

* Bootstrap includes predefined classes for creating fast, responsive layouts. Con 6 breakpoints y una docena de columnas en cada tier de grid, tenemos prácticamente una docena de clases para crear cualquier diseño.

Propiedas que podemos usar en columnas

Alignment:

Use flexbox alignment utilities to vertically and horizontally align columns.

Esto es lo de toda la vida que ya sabíamos, lo de flexbox de justify content, justify self, align ítems o align self, nada mas que funciona un toque diferente, ver aquí:

<https://getbootstrap.com/docs/5.3/layout/columns/>

Subtema -> Gutters

Esto lo que hace es generar un gap en el contenido de las columnas, el cual es crado por un padding horizontal, lo que se hace es hacer un padding left y right en cada columna y así como que da la idea de que el contenido esta separado.

El gutter empieza en 1.5 rem (24px) de wide. El gutter se puede aplicar tanto vertical (y) como horizontalmente (x).

* Horizontal gutters, .gx-\*, este de acá se usa para setear el gap entre el contenido horizontalmente (en x), si el gutter es muy grande, el parent container puede ser que sea necesario updatearlo to avoid overflow. Luce algo así:

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

* Vertical gutters, .gy-\* esta clase de acá se usa para hacer control de el gap del columna content en vertical (y). Luce algo así:

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

* Horizontal y vertical gutters – Usando solamente .g-\* se aplica el gutter a los dos axis, tanto x como y.
* Row columns gutters – Gutter classes Tambien puede ser anadida a las columnas de los rows. En este ejemplo se hace: ESTO LA VERDAD NO LO ENTIENDO XD.Interfaz de usuario gráfica

  Descripción generada automáticamente