Twitter Bootstrap Exercise 1 - Introduction

Introduction

The goal of this exercise is to download and setup Bootstrap. Then, we are going to go through some of the basic concepts.

The current version of Bootstrap is 4.5

As stated on the official website, “Bootstrap is the most popular HTML, CSS, and **JS framework** for developing responsive, mobile first projects on the web”. Basically, Bootstrap is a front-end framework for faster and easier web development.

Getting started

You can either download Bootstrap from <http://getbootstrap.com/> or link directly to the Bootstrap files. I would recommend that you **link** to these files.

You can use any HTML editor/plugin (e.g. Emmet) you wish to create your code as there are many to choose from. Alternatively, you can even use notepad++ for example.

You can use the following link to get the code for your index.html page.

<https://getbootstrap.com/docs/4.5/getting-started/introduction/>

The code looks like this:

<!doctype html>

<html lang="en">

<head>

<!-- Required meta tags -->

<meta charset="utf-8">

<meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

<!-- Bootstrap CSS -->

<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@4.5.3/dist/css/bootstrap.min.css" integrity="sha384-TX8t27EcRE3e/ihU7zmQxVncDAy5uIKz4rEkgIXeMed4M0jlfIDPvg6uqKI2xXr2" crossorigin="anonymous">

<title>Hello, world!</title>

</head>

<body>

<h1>Hello, world!</h1>

<!-- Optional JavaScript; choose one of the two! -->

<!-- Option 1: jQuery and Bootstrap Bundle (includes Popper) -->

<script src="https://code.jquery.com/jquery-3.5.1.slim.min.js" integrity="sha384-DfXdz2htPH0lsSSs5nCTpuj/zy4C+OGpamoFVy38MVBnE+IbbVYUew+OrCXaRkfj" crossorigin="anonymous"></script>

<script src="https://cdn.jsdelivr.net/npm/bootstrap@4.5.3/dist/js/bootstrap.bundle.min.js" integrity="sha384-ho+j7jyWK8fNQe+A12Hb8AhRq26LrZ/JpcUGGOn+Y7RsweNrtN/tE3MoK7ZeZDyx" crossorigin="anonymous"></script>

<!-- Option 2: jQuery, Popper.js, and Bootstrap JS

<script src="https://code.jquery.com/jquery-3.5.1.slim.min.js" integrity="sha384-DfXdz2htPH0lsSSs5nCTpuj/zy4C+OGpamoFVy38MVBnE+IbbVYUew+OrCXaRkfj" crossorigin="anonymous"></script>

<script src="https://cdn.jsdelivr.net/npm/popper.js@1.16.1/dist/umd/popper.min.js" integrity="sha384-9/reFTGAW83EW2RDu2S0VKaIzap3H66lZH81PoYlFhbGU+6BZp6G7niu735Sk7lN" crossorigin="anonymous"></script>

<script src="https://cdn.jsdelivr.net/npm/bootstrap@4.5.3/dist/js/bootstrap.min.js" integrity="sha384-w1Q4orYjBQndcko6MimVbzY0tgp4pWB4lZ7lr30WKz0vr/aWKhXdBNmNb5D92v7s" crossorigin="anonymous"></script>

-->

</body>

</html>

One of the goals of Bootstrap is to build your web pages based on a Mobile-first style. The idea here is that we get the design right for a smaller screen first since the market trend and demand is there and then through our fluid design, our pages can adapt to larger user screens if needs be. Mobile-first styles are part of the core framework.

Reviewing the code above, to ensure proper rendering and touch zooming, this line was added:

<meta name="viewport" content="width=device-width, initial-scale=1">

content="width=device-width sets the width of the page to follow the screen-width of the device the user is using

initial-scale=1 sets the initial zoom level when the page is first loaded

We typically use “containers” to wrap the contents of our web pages. Bootstrap requires you to use one of the following containers:

1. The .container class provides a responsive fixed width container
2. The .container-fluid class provides a full width container, spanning the entire width of the viewport

Note: Containers are not nestable (you cannot put a container inside another container).

Change your HTML to see the difference when using both containers.

Grid System

Bootstrap uses Grids to help you layout your content. You can use a maximum of 12 columns across your page. Again, Bootstrap is a responsive design and therefore the columns will re-arrange automatically depending on what device the user is viewing the content on (screen size).

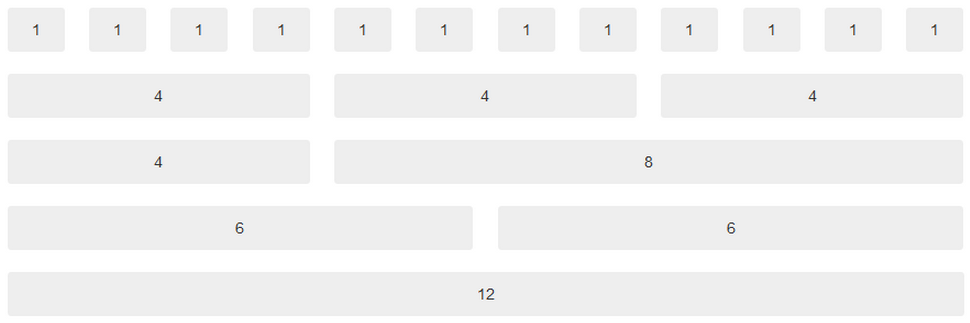


Figure 1: Bootstrap uses a 12 column grid system

There are four classes that you can use:

1. xs (for phones)
2. sm (for tablets)
3. md (for desktops)
4. lg (for larger desktops)

The classes above can be combined to create more dynamic and flexible layouts.

Let us now implement this system in our index.html file. The basic template we use is below where we must create a new row (<div class="row">), then add the number of columns (remembering not to go beyond 12) and then use the appropriate classes (class="col-\*-\*") and just mentioned above.

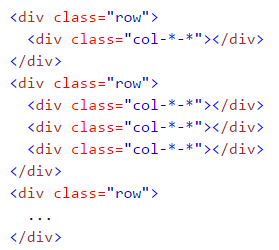


Figure 2: Basic grid template

Use the following HTML code:



Figure 3: Implementing the grid template

You will see the following screen when you load the page. Therefore, let us add some styles to enhance this display and message.



Figure 4: Result of loading the page in a browser

Add the following styles:

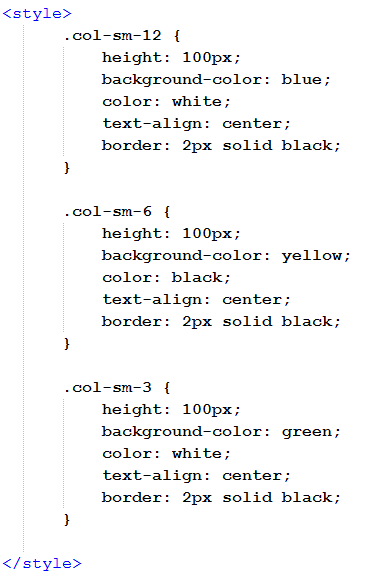


Figure 5: Styles to add

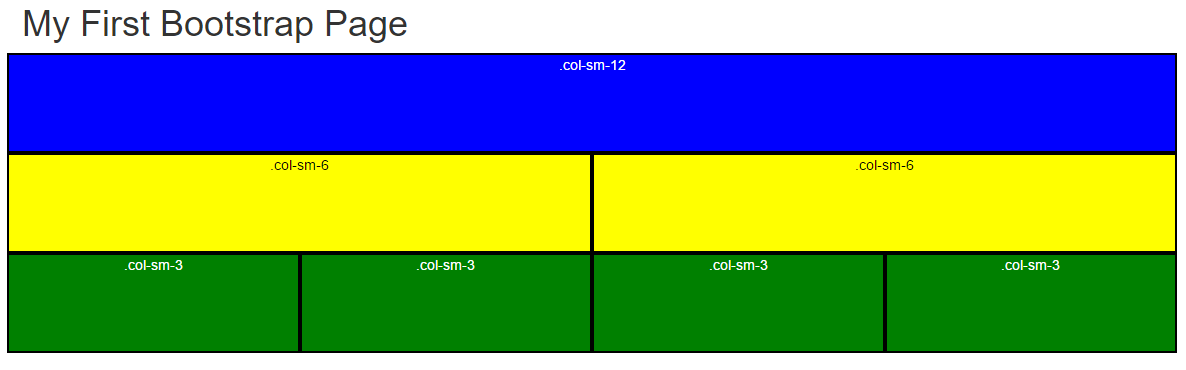


Figure 6: Result of loading the page with styles

As you can see the maximum number of columns per row is 12 and therefore you can decide the breakdown of the columns as per your design. Try to reduce the size of the browser and you will see the responsive design in action. As the screen size becomes smaller than the class style “SM” then the rows become stacked.

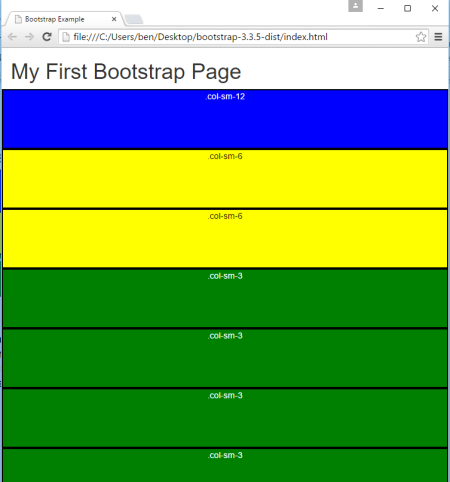


Figure 7: Responsive design adapting to smaller screen size

Let us now use some other class types to illustrate this point of responsive design even further.

Modify your styles to just have the following:

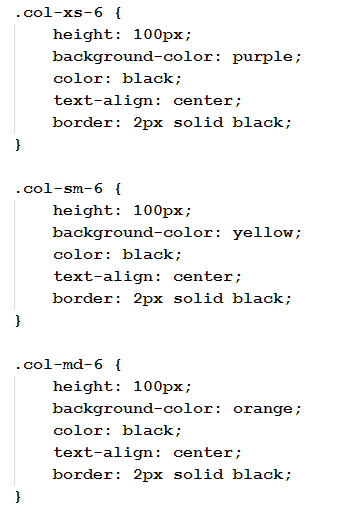


Figure 8: Updated styles

Modify your HTML to have the following:

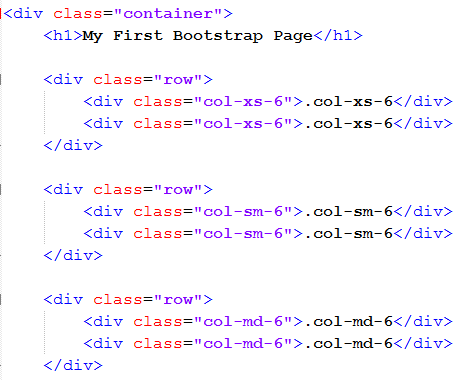


Figure 9: Updated HTML

When you view it in your browser (full screen) you will see the following, which shows everything laid out like specified in your code.

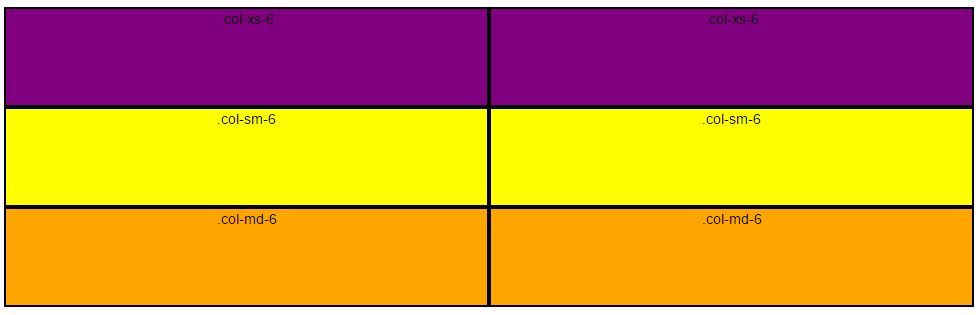


Figure 10: Initial result when viewing on a large screen

Notice the effect of reducing your browser screen width, which shows that when your page is viewed on a small device such as a tablet, the “MD” class is forced to stack its columns.

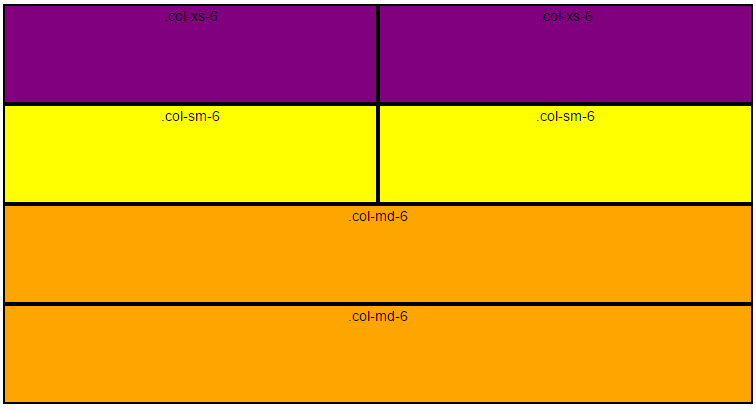


Figure 11: Result when viewing on a small screen

Finally, notice the effect of reducing your browser screen width even further, which would resemble a mobile phone. The class “SM” like the “MD” class before is forced to stack its columns. It is worth mentioning that no matter how small you make the screen from this point, the screen will still retain its shape as “XS” is the smallest.

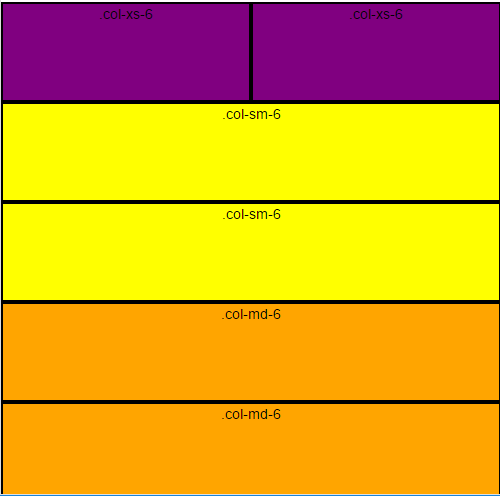


Figure 12: Result when viewing on the smallest screen (phone)

Ordering classes - Changing the order of columns dynamically

Bootstrap gives you the ability to change the order of the columns based on the device that is viewing your page. To demonstrate this setup the following styles and HTML

|  |  |
| --- | --- |
|  |  |

Figure 13: Styles and HTML setup for ordering example

When you load the page you should see the following:

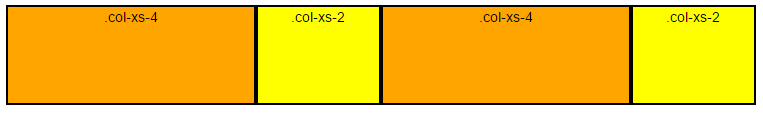
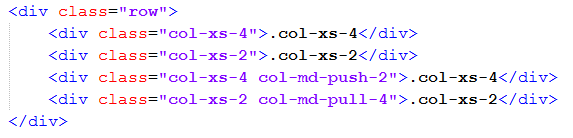


Figure 14: Result of displaying page in browser

The goal now is to swap (push and pull) the third and fourth columns when viewing the page on a larger device such as a desktop. Add the following push and pull pieces of code.



When you increase the screen width you should see the following change. Take note, before Bootstrap version 3.3.2 this only worked on large and medium size screens for push and pull (i.e. if you try to change the above code to col-**xs**-push-2 or col-**sm**-push-2, there will be no effect).

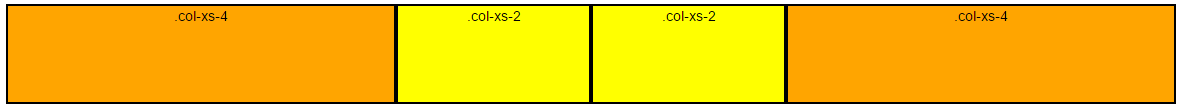


Figure 15: Result of using push and pull when page is viewed on a desktop

Offset classes

By using the Offset classes in Bootstrap, you have the ability to move your columns depending what the column is offset by. Modify your HTML as below. The offset only works on medium and large screens. Make sure you have a col-sm-4 style.

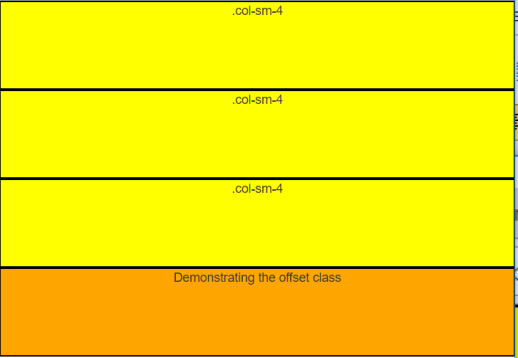
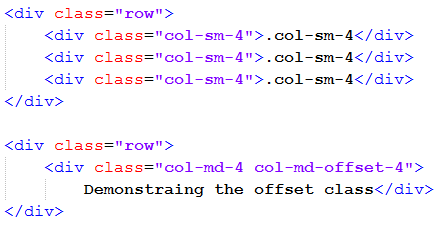


Figure 16: HTML and result in browser when the screen is **small**

When you increase your screen width (i.e. view the web page on a medium/desktop browser) you will then see the offset taking effect as shown below. Change the offset to 8 and see what happens. Change the offset to 9 and see what happens.

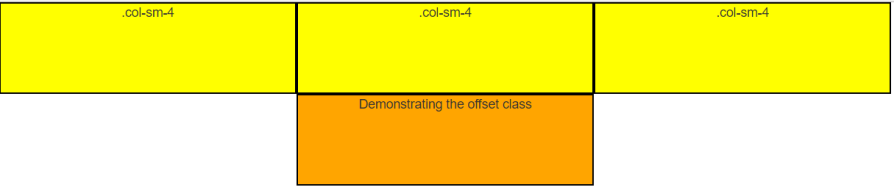
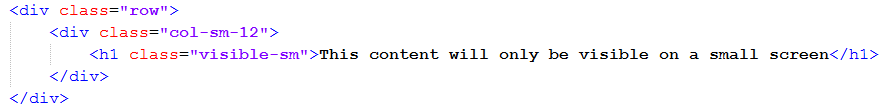


Figure 16: Offset taking effect on a **medium** screen and above

Responsive Utility classes

These classes give you the ability to hide and show content on your screen depending on what device you are using. There are two specific classes, .visible and .hidden.

Edit your HTML file to contain the following row. The idea here is that the text with the H1 heading will only appear when the screen size is small (sm). Try it out and see how the content is visible and hidden depending on your screen size. Ensure you have a .col-sm-4 style.



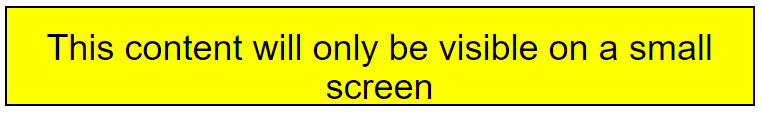


Figure 17: HTML and result in browser for a small screen

As another example, modify your HTML code to the following. Load the file in your browser and notice how the text changes depending on your screen size (browser width).

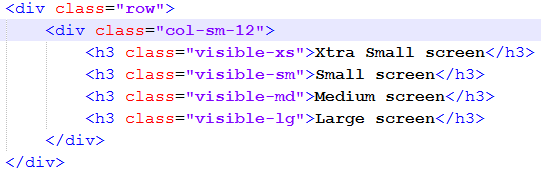


Figure 18: A different H3 for each screen size

The opposite effect happens for the .hidden class obviously. Modify your HTML as below and run the file in your browser to see the effect.

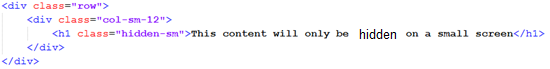


Figure 19: The hidden class

All these effects can be useful to custom your screen design, layout and content depending on the users device.