Bootstrap Grid

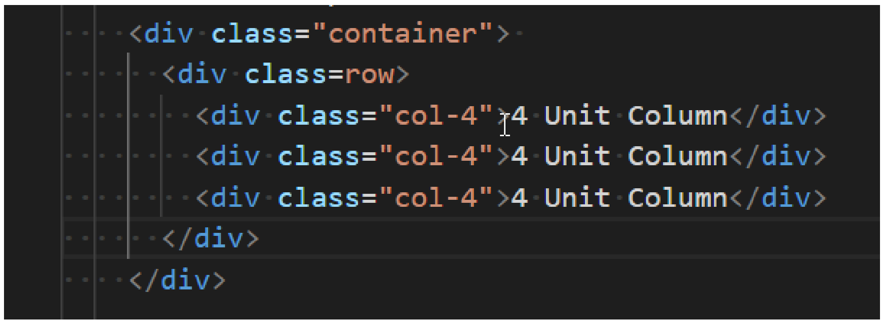
Lesson Time: 60 Minutes

In CSS, we learned how to use CSS Flexbox to start creating responsive layouts and positioning our elements on the page. Bootstrap has an alternative system called Bootstrap Grid. This lesson covers the differences.

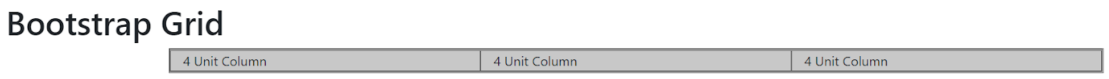
1. To get started using Bootstrap Grid, give the parent element the BS class **container**
2. Inside of the container parent, we'll create elements for our horizontal rows with the BS class **row**
3. Inside of our rows, we'll create our cells with the BS class **col.** Each bootstrap row is made up of **12 columns**.
    a. We have many options on our col class.
    b. We can use a suffix of -xs, -sm, -md, and -lg to apply the column layout to different screen sides.
        i. -xs = extra small device screen (575 px)
        ii. -sm = small device screen (767 px)
        iii. -md = medium device screen (991 px)
        iv. -lg = large device screen (1199 px)
    c. We can specify the number of columns to take up with col-*-* where the first dash is screen size (xs, sm, md, lg) and the second is the number of columns.

Example 1: Columns

This bootstrap grid is a single row, with 3 columns. Each column takes takes up 4 units of the 12 unit grid, for a nice 3 column layout.



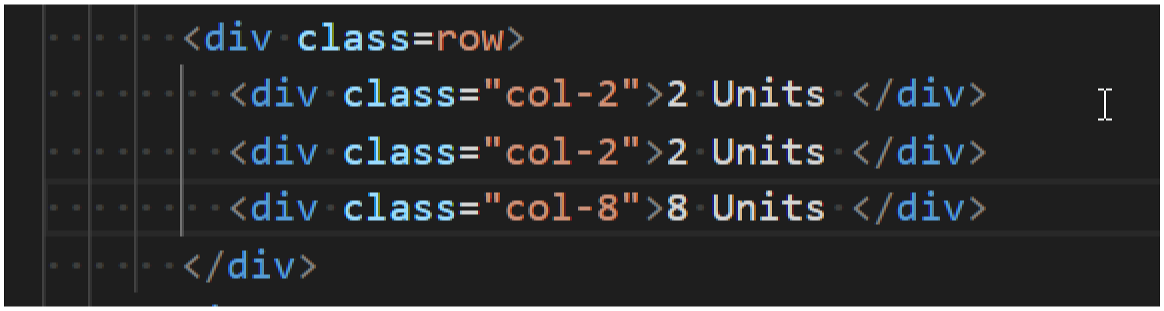
```
<div class="container">
  <div class=row>
    <div class="col-4">4 Unit Column</div>
    <div class="col-4">4 Unit Column</div>
    <div class="col-4">4 Unit Column</div>
  </div>
</div>
```

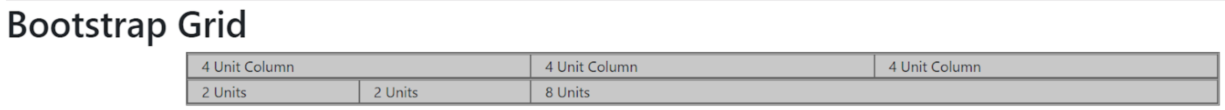

Bootstrap Grid

4 Unit Column	4 Unit Column	4 Unit Column

Example 2: Columns

This row is made up of 3 columns of uneven sizes.

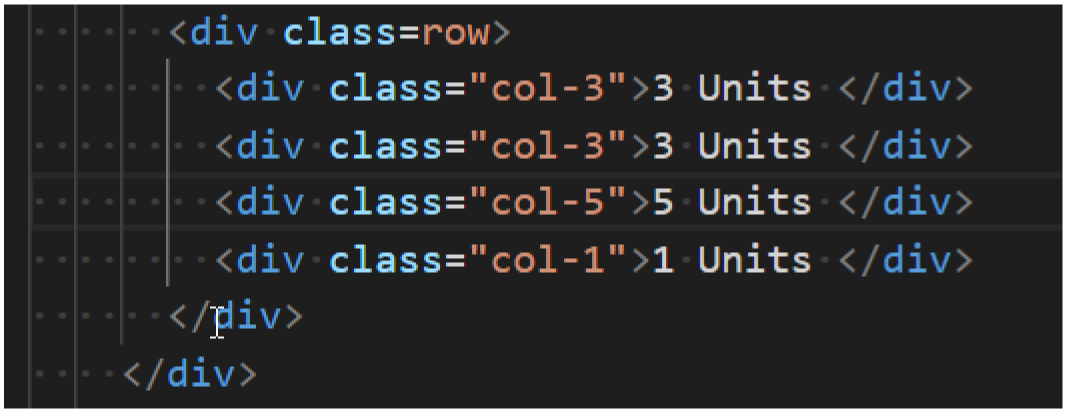

```
<div class=row>
  <div class="col-2">2 Units </div>
  <div class="col-2">2 Units </div>
  <div class="col-8">8 Units </div>
</div>
```

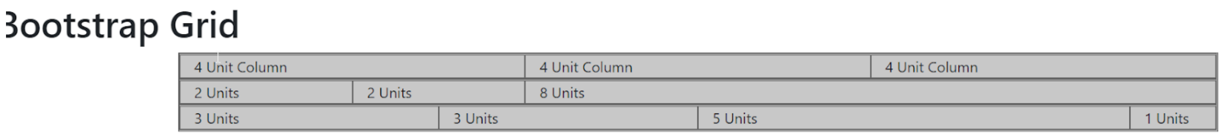

Bootstrap Grid

4 Unit Column		4 Unit Column	4 Unit Column
2 Units	2 Units	8 Units	

Example 3: Columns

This row is made up of 4 columns of uneven sizes.

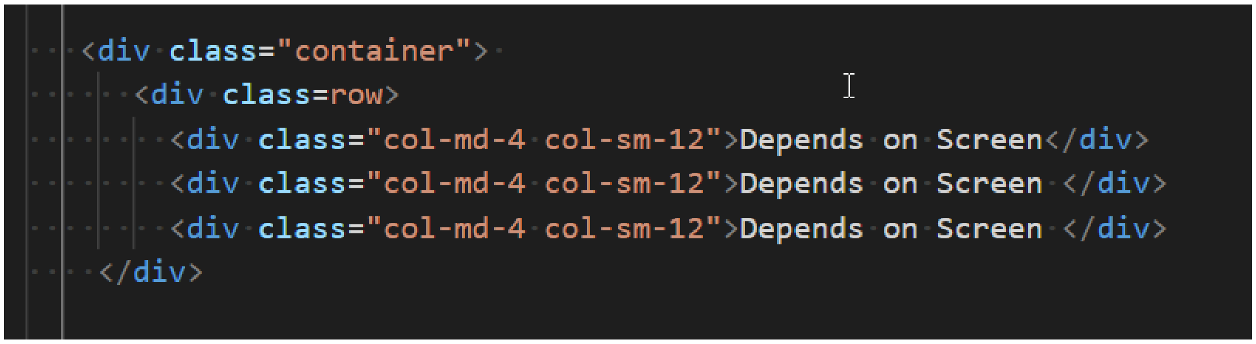

```
<div class=row>
  <div class="col-3">3 Units </div>
  <div class="col-3">3 Units </div>
  <div class="col-5">5 Units </div>
  <div class="col-1">1 Units </div>
</div>
</div>
```


Bootstrap Grid

4 Unit Column			4 Unit Column	4 Unit Column	
2 Units	2 Units		8 Units		
3 Units		3 Units	5 Units		1 Units

Example 4: Media Sizes

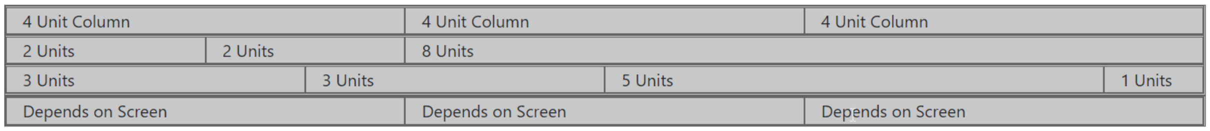
Now, let’s add the screen size restrictions . This code says on a medium size screen, these columns are 4 units wide, but on a small screen, make them 12 units wide instead. This is a common way to create responsive layout--on smaller screens we can stack elements.



```
   <div class="container">
      <div class=row>
        <div class="col-md-4 col-sm-12">Depends on Screen</div>
        <div class="col-md-4 col-sm-12">Depends on Screen </div>
        <div class="col-md-4 col-sm-12">Depends on Screen </div>
    </div>
```

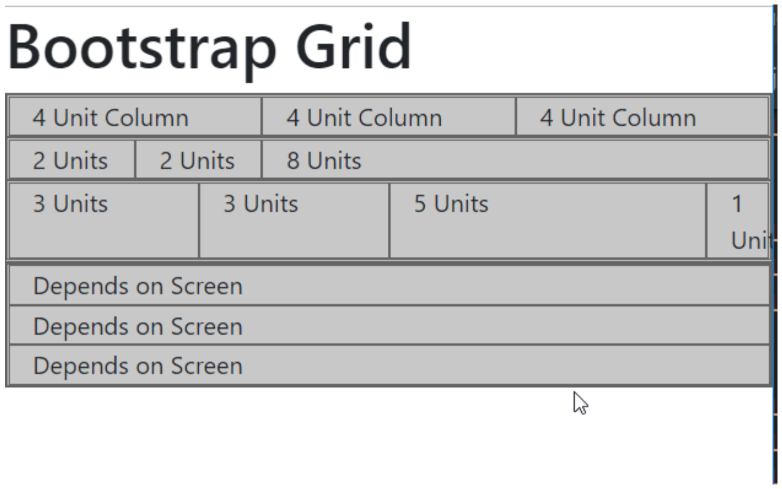
Results:

On a screen with over 991 pixels:



4 Unit Column | 4 Unit Column | 4 Unit Column
2 Units | 2 Units | 8 Units
3 Units | 3 Units | 5 Units | 1 Units
Depends on Screen | Depends on Screen | Depends on Screen

On a screen with less than 767 pixels (mobile)



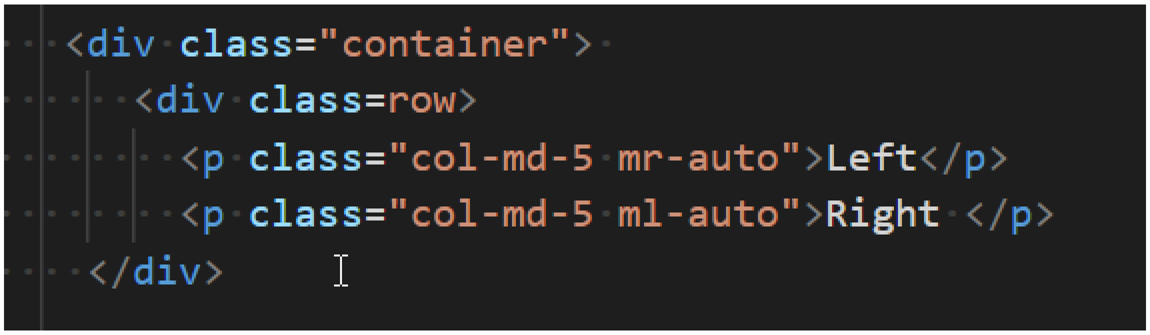
Bootstrap Grid

4 Unit Column | 4 Unit Column | 4 Unit Column
2 Units | 2 Units | 8 Units
3 Units | 3 Units | 5 Units | 1 Uni
Depends on Screen
Depends on Screen
Depends on Screen

Example 5: Margins
In this example, we are going to create two panels of content on the left and the right with open space between them by using bootstrap’s auto margins.

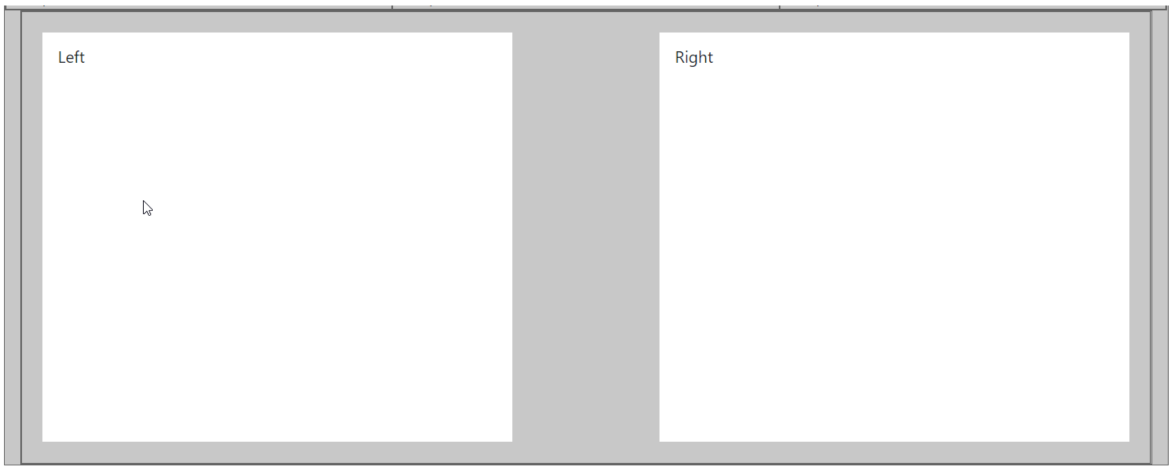
ml-auto strands for margin left auto. Setting the ml-auto pushes the element to the RIGHT.

mr-auto stands for margin right auto. Setting the mr-auto pushes the element to the LEFT.



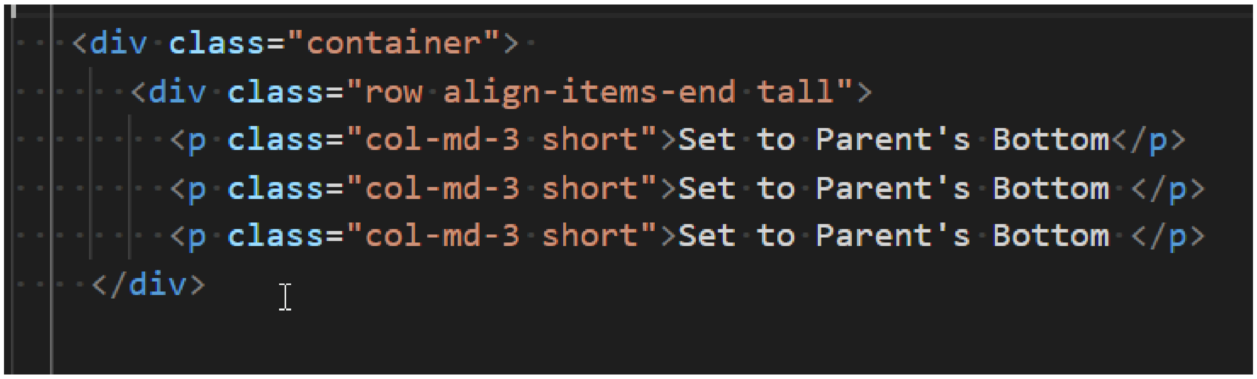
```
<div class="container">
    <div class=row>
      <p class="col-md-5 mr-auto">Left</p>
      <p class="col-md-5 ml-auto">Right </p>
  </div>
```

Result:



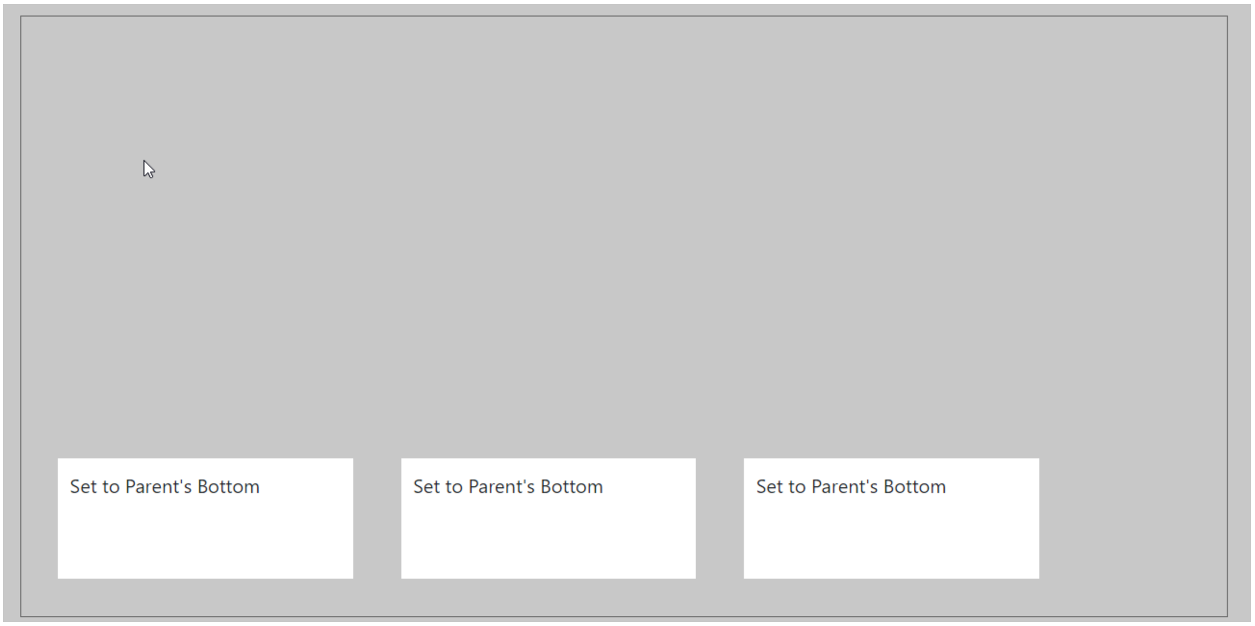
Example 6: Align-Items (Vertical Alignment)

We can use **align-items-start**, **align-items-center**, and **align-items-end** to set vertical alignment for all the child elements of a parent. Be aware that in this example, we are using two non-bootstrap custom CSS classes called .tall and .short to set the height of the elements in pixels.



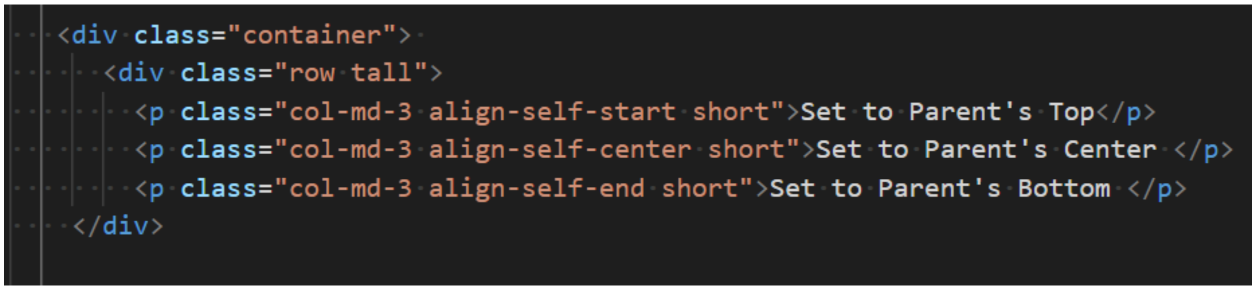
```html
   <div class="container">
      <div class="row align-items-end tall">
         <p class="col-md-3 short">Set to Parent's Bottom</p>
         <p class="col-md-3 short">Set to Parent's Bottom </p>
         <p class="col-md-3 short">Set to Parent's Bottom </p>
    </div>
```

Results



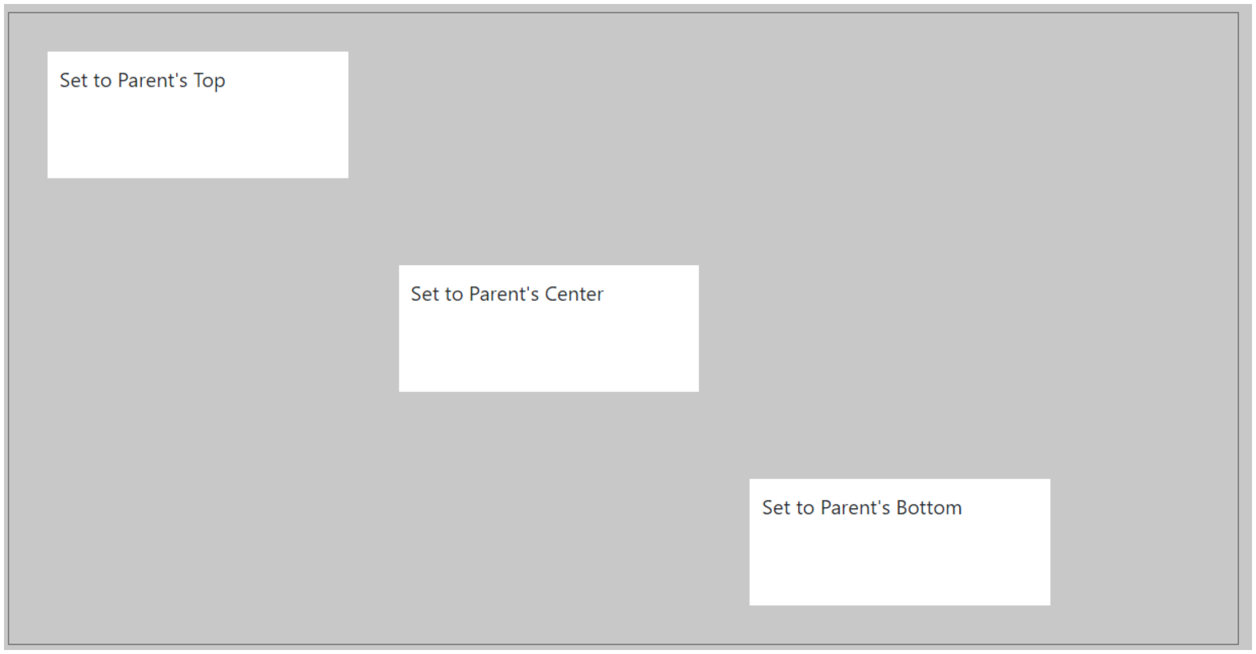
Example 7: Align-Items (Vertical Alignment)

We can also use align-self-start, align-self-center, and align-self-end to align child items relative to their parent.



```
<div class="container">
    <div class="row tall">
        <p class="col-md-3 align-self-start short">Set to Parent's Top</p>
        <p class="col-md-3 align-self-center short">Set to Parent's Center </p>
        <p class="col-md-3 align-self-end short">Set to Parent's Bottom </p>
</div>
```

Results



Hopefully, you see that bootstrap gives us the easiest way we’ve seen yet to work with layout, and it can be worth using just for the bootstrap grid layout alone!

Key Terms	
Lesson Files	12_2grid.html
Additional Resources	https://getbootstrap.com/docs/4.3/layout/grid/ https://www.sitepoint.com/bootstrap-grid-mastering-flexbox/
Further Learning	