In bootstrap, the **grid system** is useful to build a quick web page layouts which are responsive to the different devices based on the screen sizes. In bootstrap 4, the grid system is built with mobile-first flexbox and it allows up to **12** columns across the page.

The bootstrap grid system will use a series of containers, rows, and columns to define the layout and to align the content appropriately based on the device. In grid system, you are allowed to add up to 12 columns and you can add as many rows as you like and the columns will re-arrange automatically based on the device screen size.

By using bootstrap grid system, we can create a responsive web page layout by defining the **12** columns individually or by grouping the columns together to create wider columns like as shown below.

Bootstrap Grid Classes

The bootstrap 4 has included the following 5 predefined grid classes to scale the content depending on the device or viewport size.

* .col-\*
* .col-sm-\*
* .col-md-\*
* .col-lg-\*
* .col-xl-\*

Here, the asterisk (**\***) is the span width of the column from **1** to **12**. Following table lists how the grid system classes will work across multiple devices.

| **Class** | **Device Type** | **Width** |
| --- | --- | --- |
| .col-\* | Extra Small | <576px |
| .col-sm-\* | Small | ≥576px |
| .col-md-\* | Medium | ≥768px |
| .col-lg-\* | Large | ≥992px |
| .col-xl-\* | Extra Large | ≥1200px |

Structure of Bootstrap Grid

To create a responsive web page layouts using bootstrap grid system, we need to use rows and columns within the [container](https://www.tutlane.com/tutorial/bootstrap/bootstrap-containers#divbtcntrcls) or [container-fluid](https://www.tutlane.com/tutorial/bootstrap/bootstrap-containers#divbtcntfldcls) like as shown below.

<div class="container">

    <div class="row">

        <div class="col-\*-\*"></div>

        <div class="col-\*-\*"></div>

        <div class="col-\*-\*"></div>

    </div>

</div>

If you observe the above code, we created a grid layout by specifying the rows and columns within .container. Here, the first star (\*) will indicate the responsiveness and that should be either sm, md, lg, or xl. The second star (\*) will indicate a number and that should be from **1** to **12** for each row.

Every time, while we create a grid layout we need to place the content within the columns (.col or .col-\*-\*) and those columns must be the children of rows (.row) and those rows must be placed inside of a container (.container or .container-fluid).

In the bootstrap grid system, we are allowed to add up to **12** columns within a row. In case, if we add more than **12** columns within a row, then those extra columns will wrap into a new line.

Bootstrap Grid Layout with Responsive Three Column

Following is the example of creating a responsive layout with three columns to support the devices whose screen width is more than 576px in bootstrap.

[Live Preview](https://www.tutlane.com/example/bootstrap/bootstrap-responsive-three-column-example)

<div class="container">

    <div class="row">

        <div class="col-sm-4">col-sm-4</div>

        <div class="col-sm-4">col-sm-4</div>

        <div class="col-sm-4">col-sm-4</div>

    </div>

    <div class="row">

        <div class="col-sm-2">col-sm-2</div>

        <div class="col-sm-4">col-sm-4</div>

        <div class="col-sm-6">col-sm-6</div>

    </div>

    <div class="row">

        <div class="col-sm-2">col-sm-2</div>

        <div class="col-sm-5">col-sm-5</div>

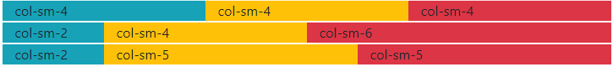
        <div class="col-sm-5">col-sm-5</div>

    </div>

</div>

The above bootstrap example will create a layout with three columns to support the devices whose screen width is more than **576px** in bootstrap. The defined three columns will automatically adjust on top of each other in mobile devices or screens that are less than **576px** wide.

When we execute the above bootstrap example, we will get the result like as shown below.



Bootstrap Grid Layout with Auto Columns

In bootstrap, we can create equal or auto width columns for all the devices such as extra small, small, medium, large, and extra-large by specifying only .col class without any column numbers.

The bootstrap grid system will automatically adjust the columns which defined with .col class based on the device screen width.

The following example will show how to create equal-width columns to support all the devices in bootstrap.

[Live Preview](https://www.tutlane.com/example/bootstrap/bootstrap-auto-columns-layout-example)

<div class="container">

    <div class="row">

        <div class="col">col 1</div>

        <div class="col">col 2</div>

        <div class="col">col 3</div>

    </div>

    <div class="row">

        <div class="col">col 1</div>

        <div class="col">col 2</div>

    </div>

    <div class="row">

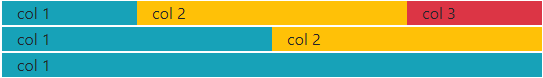
        <div class="col">col 1</div>

    </div>

</div>

If you observe the above example, we created columns with only .col class. So, the bootstrap grid system will adjust columns size equally based on the device screen size.

When we execute the above bootstrap example, we will get the result like as shown below.



If you observe the above result, the columns automatically adjusted equally based on the device screen size.

If required, in auto layout grid system also we can set the width for one column and the sibling columns will resize automatically.

Following is the example of auto layout grid system with equal and unequal columns in bootstrap.

[Live Preview](https://www.tutlane.com/example/bootstrap/bootstrap-grid-layout-with-equal-and-unequal-columns-examples)

<div class="container">

    <div class="row">

        <div class="col">col 1</div>

        <div class="col-sm-6">col 2</div>

        <div class="col">col 3</div>

    </div>

    <div class="row">

        <div class="col">col 1</div>

        <div class="col">col 2</div>

    </div>

    <div class="row">

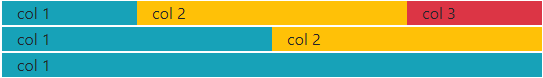
        <div class="col">col 1</div>

    </div>

</div>

If you observe the above example, we defined width for one column (.col-sm-6) in first row and other columns will adjust automatically.

When we execute the above bootstrap example, we will get the result like as shown below.



This is how we can set the width for one column and the sibling columns will resize automatically in the auto layout grid system.

Bootstrap Grid Layout with Variable Width Columns

In bootstrap, by using col-{breakpoint}-auto classes we can create variable-width columns to resize based on the width of their content.

Following is the example of creating variable width columns to resize based on the width of content in bootstrap.

[Live Preview](https://www.tutlane.com/example/bootstrap/bootstrap-grid-with-variable-width-columns-example)

<div class="container">

    <div class="row">

        <div class="col-sm-5">col 1</div>

        <div class="col-md-auto">Variable width column content</div>

        <div class="col-sm-3">col 2</div>

    </div>

    <div class="row">

        <div class="col">col 1</div>

        <div class="col">col 2</div>

        <div class="col-md-auto">Variable width column content</div>

    </div>

</div>

If you observe the above example, we created variable-width columns by specifying col-md-auto class and those columns will automatically adjusted based on the width of content.

When we execute the above bootstrap example, it will return the result like as shown below.

Bootstrap Grid with Variable Width Columns Example Result

If you observe the above result, the **gray** color part is empty due to the columns width adjusted based on content and defined size.

Bootstrap Grid Layout with Different Column Sizes

Till now, we created grid layout specific to the particular devices i.e. either extra small, small, large, or extra-large. In bootstrap, we can create more flexible layouts by combining the multiple column sizes to change the orientation of the columns based on the device size.

For example, the small screens need to have one column at each row, whereas the larger screens can contain more than one column at each row, so you can set smaller column size but multiple columns on each row. To understand what I mean we will see an example.

Following is the example of creating the bootstrap grid layout by mixing the different column sizes for device sizes.

[Live Preview](https://www.tutlane.com/tutorial/bootstrap/bootstrap-grid-system)

<div class="container">

    <div class="row ">

        <div class="col-lg-4 col-md-4 col-sm-6 col-xs-12">Content Area 1</div>

        <div class="col-lg-4 col-md-4 col-sm-6 col-xs-12">Content Area 2</div>

        <div class="col-lg-4 col-md-4 col-sm-6 col-xs-12">Content Area 3</div>

    </div>

</div>

The above example will show one row with **3** columns in large and medium screens, in small screens it will show one row with **2** columns and another one more row with another column in half screen. In extra small screens like smartphones we get to see **3** rows, each one will have one full screen column.

When we execute the above bootstrap example, we will get the result like as shown below.

Bootstrap Grid System with Mixed Columns Width Example Result

This is how we can create more flexible grid layouts by combining the multiple column sizes to change the orientation of the columns based on the device size in bootstrap.

Bootstrap Grid Layout Column Offsetting

In the bootstrap grid layout, we can move columns to the right by using .offset-\* grid classes. To offset a column you have to append offset-\* suffix to your existing column class. For example, if your column class is .col-md-4 you can offset the column with the class .offset-md-4. The number after offset symbolizes how many columns to add as margin.

Following is the example of offsetting the grid columns using .offset-\* classes in bootstrap.

[Live Preview](https://www.tutlane.com/example/bootstrap/bootstrap-grid-layout-offsetting-columns-example)

<div class="container">

    <div class="row">

        <div class="col-md-4">.col-md-4</div>

        <div class="col-md-4 offset-md-4">.col-md-4 .offset-md-4</div>

    </div>

    <div class="row">

        <div class="col-md-3 offset-md-3">.col-md-3 .offset-md-3</div>

        <div class="col-md-3 offset-md-3">.col-md-3 .offset-md-3</div>

    </div>

    <div class="row">

        <div class="col-md-6 offset-md-3">.col-md-6 .offset-md-3</div>

    </div>

</div>

If you observe the above example, we are increasing the left margin of a column by **4** & **3** columns by using offset-md-4 and offset-md-3 classes.

When we execute the above bootstrap example, we will get the result like as shown below.



Bootstrap Grid Layout Column Reordering

In the bootstrap grid system, we can reorder the appearance of grid columns by using .order-\* classes without changing their actual order in grid layout.

In the grid layout, the order of grid columns will vary based on the defined order numbers. For example, the grid columns with lower-order (e.g., .col, .order-1) or without order classes will appear first and the columns with higher-order numbers will appear after the lower order number columns. The grid system will support ordering the columns from **1** through **12** across all five grid tiers.

Following is the example of changing the appearance of grid columns by specifying the order number in bootstrap.

[Live Preview](https://www.tutlane.com/example/bootstrap/bootstrap-grid-layout-reorder-columns-example)

<div class="container">

    <div class="row">

        <div class="col order-10">First</div>

        <div class="col order-4">Second</div>

        <div class="col">Third</div>

    </div>

</div>

If you observe the above example, we are changing the order of columns by specifying the order numbers (order-10, order-4) and for third column we didn’t specified any order class.

When we execute the above bootstrap example, we will get the result like as shown below.

Bootstrap Grid Layout with Column Reordering Example Result

If you observe the above result, the appearance of grid columns changed based on the order number of columns. Here, the third column appeared first because we didn’t specify any order class so it will consider as lowest.

We can also change the order of grid layout columns by using .order-first and .order-last classes like as shown below.

<div class="container">

    <div class="row">

        <div class="col order-last">First</div>

        <div class="col order-first">Second</div>

        <div class="col">Third</div>

    </div>

</div>

If you observe the above example, we are trying to change the columns appearance by using .order-first and .order-last classes.

When we execute the above bootstrap example, we will get the result like as shown below.

Bootstrap Change Grid Columns Order Example Result

If you observe the above result, the second column appeared first because of assigning the .order-first class and the first column moved to last due to .order-last class.

Bootstrap Grid Nesting Columns

Nesting columns means that you can add columns inside another column. You can insert up to **12** columns inside another column without necessarily using all **12** columns. To do this you need to add a new row (.row) and set of .col-md-\* columns within an existing .col-md-\* column.

Following is the example of nesting the grid columns within another column in the bootstrap grid layout.

[Live Preview](https://www.tutlane.com/example/bootstrap/bootstrap-grid-layout-nesting-columns-example)

<div class="container">

    <div class="row">

        <div class="col-sm-9">First Col

            <div class="row">

                <div class="col-xs-8 col-sm-6">

                    Col 1 inside first column

                </div>

                <div class="col-xs-4 col-sm-6">

                    Col 2 inside first column

                </div>

            </div>

        </div>

        <div class="col">Second Col</div>

    </div>

</div>

If you observe the above example, we created nested grid columns by adding a new row (.row) and set of columns within an existing .col-sm-9 column.

When we execute the above bootstrap example, we will get the result like as shown below.

Bootstrap grid layout with column nesting example result

 This is how we can create nested columns in the bootstrap grid layout based on our requirements.

Bootstrap Grid Column Wrapping

Bootstrap is designed for **12** columns per row. If you use more than **12** columns within a row, then the extra columns will wrap onto a new line as one unit.

Following is the example of creating a bootstrap grid layout with more than **12** columns per row to achieve column wrapping.

[Live Preview](https://www.tutlane.com/example/bootstrap/bootstrap-grid-column-wrapping-example)

<div class="container">

    <div class="row">

        <div class="col-sm-9">First Column</div>

        <div class="col-sm-4">Second Column</div>

        <div class="col-sm-8">Third Column</div>

    </div>

</div>

If you observe the above example, the first column which is **9** columns width and the second is **4 (9 + 4 = 13)**, so the columns cannot be in one row and the extra columns will wrap into a new line.

When we execute the above example, we will get the result like as shown below.

Bootstrap grid column wrapping example result

This is how the column wrapping will happen in the bootstrap grid system when you use more than 12 columns within a row.