Bootstrap 4 Documentation

Bootstrap is a free front-end framework for faster and easier web development. It includes HTML and CSS based design templates for typography, forms, buttons, tables, navigation, modals, image carousels and many other, as well as optional JavaScript plugins and gives you the ability to easily create responsive designs.

Bootstrap 4 uses HTML elements and CSS properties that require the HTML5 doctype. Always include the HTML5 doctype at the beginning of the page, along with the lang attribute and the correct character set:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="utf-8">
</head>
</html>
```

Bootstrap 4 is designed to be responsive to mobile devices. Mobile-first styles are part of the core framework. To ensure proper rendering and touch zooming, add the following <meta> tag inside the <head> element:

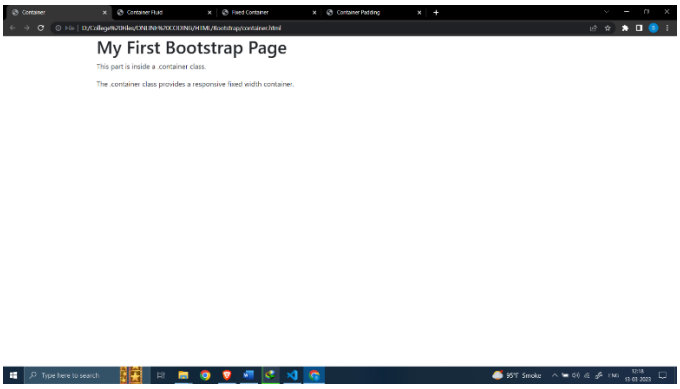
```
<meta name="viewport" content="width=device-width, initial-scale=1">
```

The width=device-width part sets the width of the page to follow the screen-width of the device (which will vary depending on the device). The initial-scale=1 part sets the initial zoom level when the page is first loaded by the browser.

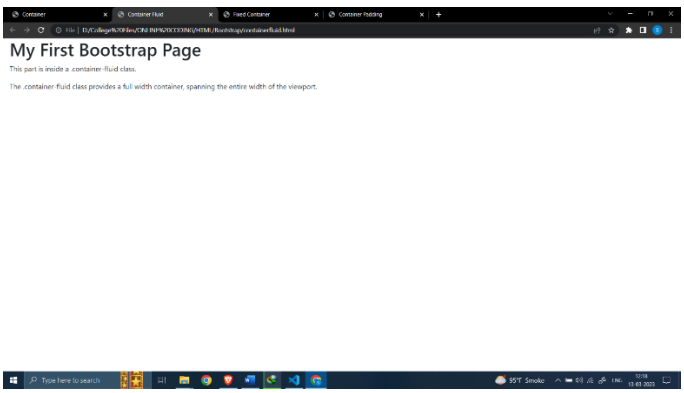
Container

Bootstrap 4 also requires a containing element to wrap site contents. There are two container classes to choose from:

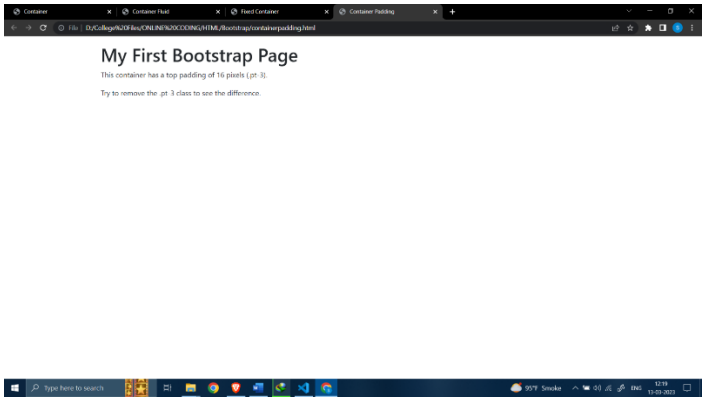
1. The **.container** class provides a responsive fixed width container



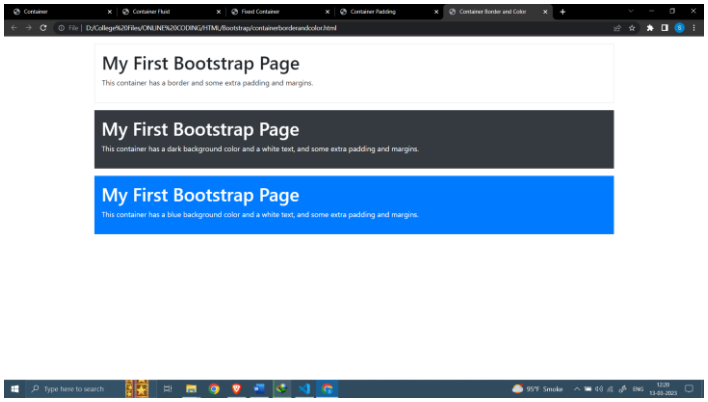
2. The **.container-fluid** class provides a full width container, spanning the entire width of the viewport



By default, containers have 15px left and right padding, with no top or bottom padding. Therefore, we often use spacing utilities, such as extra padding and margins to make them look even better. For example, .pt-3 means "add a top padding of 16px":

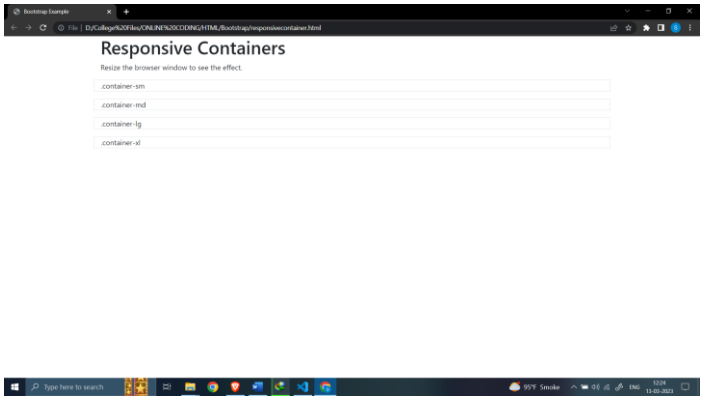


Other utilities, such as borders and colors, are also often used together with containers:



You can also use the .container-sm|md|lg|xl classes to create responsive containers. The max-width of the container will change on different screen sizes/viewports:

Class	extra small (less than 576px)	small (greater than 576px)	medium (greater than 768px)	large (greater than 992px)	extra large (greater than 1200px)
.container-sm	100%	540px	720px	960px	1140px
.container-md	100%	100%	720px	960px	1140px
.container-lg	100%	100%	100%	960px	1140px
.container-xl	100%	100%	100%	100%	1140px



Grids

Bootstrap's grid system is built with flexbox and allows up to 12 columns across the page. If you do not want to use all 12 columns individually, you can group the columns together to create wider columns. The grid system is responsive, and the columns will re-arrange automatically depending on the screen size. Make sure that the sum adds up to 12 or fewer (it is not required that you use all 12 available columns).

The Bootstrap 4 grid system has five classes:

.col	(extra small devices - screen width less than 576px)
.col-sm	(small devices - screen width equal to or greater than 576px)
.col-md	(medium devices - screen width equal to or greater than 768px)
.col-lg	(large devices - screen width equal to or greater than 992px)
.col-xl	(xlarge devices - screen width equal to or greater than 1200px)

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

The following is a basic structure of a Bootstrap 4 grid:

- Control the column width, and how they should appear on different devices
- create a row ( ). Then, add the desired number of columns (tags with appropriate **.col-*-*** classes). The first star (*) represents the responsiveness: sm, md, lg or xl, while the second star represents a number, which should add up to 12 for each row.

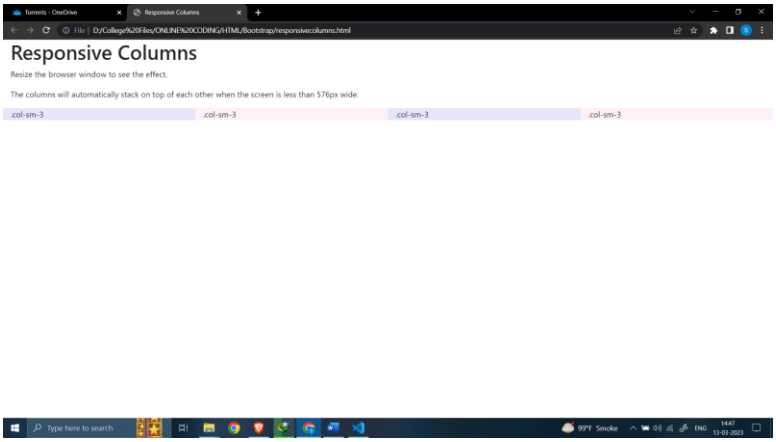
```
<div class="row">
      <div class="col-*-*"></div>
      <div class="col-*-*"></div>
</div>
<div class="row">
      <div class="col-*-*"></div>
      <div class="col-*-*"></div>
      <div class="col-*-*"></div>
</div>
```

- Or let Bootstrap automatically handle the layout
- instead of adding a number to each col, let bootstrap handle the layout, to create equal width columns: two "col" elements = 50% width to each col. three cols = 33.33% width to each col. four cols = 25% width, etc. You can also use .col-sm|md|lg|xl to make the columns responsive.

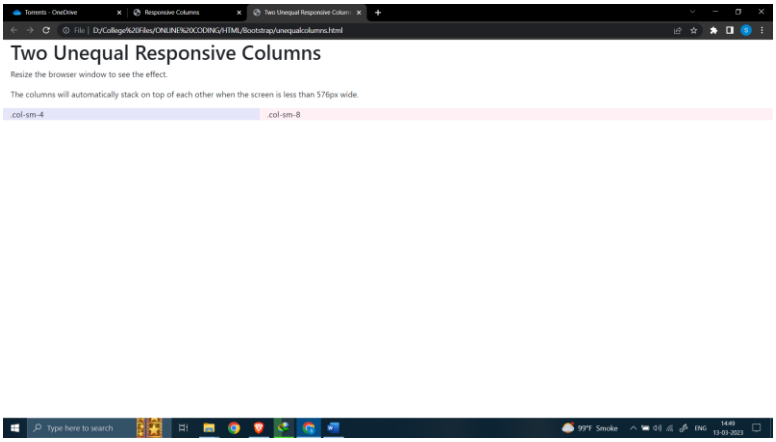
```
<div class="row">
      <div class="col"></div>
      <div class="col"></div>
      <div class="col"></div>
</div>
```

The following example shows how to create three equal-width columns, on all devices and screen widths:

The following example shows how to create four equal-width columns starting at tablets and scaling to extra large desktops. On mobile phones or screens that are less than 576px wide, the columns will automatically stack on top of each other:



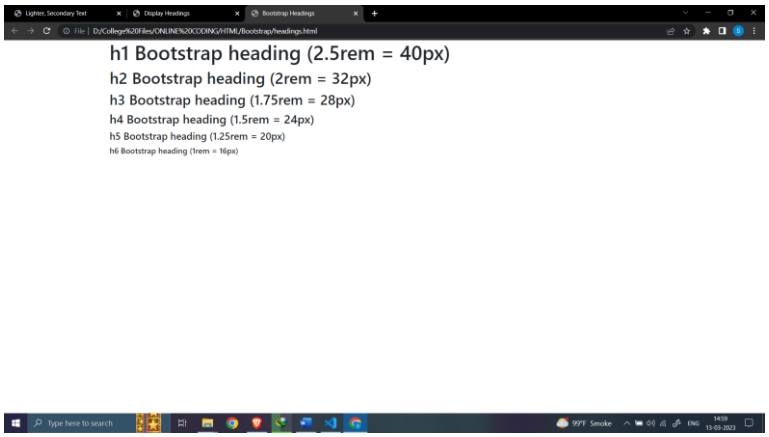
The following example shows how to get two various-width columns starting at tablets and scaling to large extra desktops:



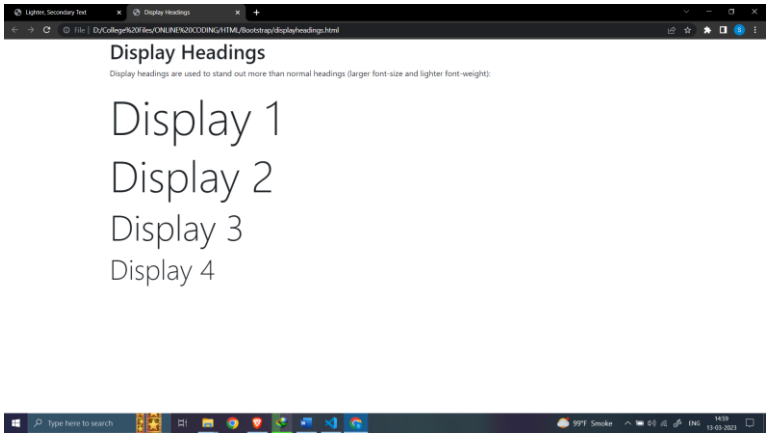
Typography

Bootstrap 4 uses a default font-size of 16px, and its line-height is 1.5. The default font-family is "Helvetica Neue", Helvetica, Arial, sans-serif. In addition, all <p> elements have margin-top: 0 and margin-bottom: 1rem (16px by default).

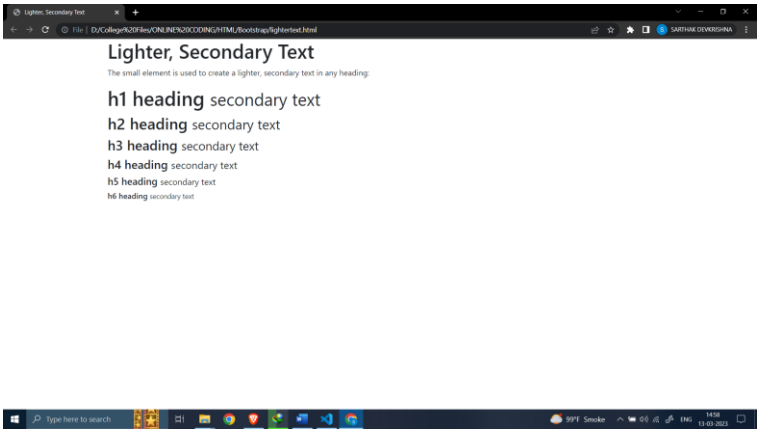
Bootstrap 4 styles HTML headings (<h1> to <h6>) with a bolder font-weight and an increased font-size:



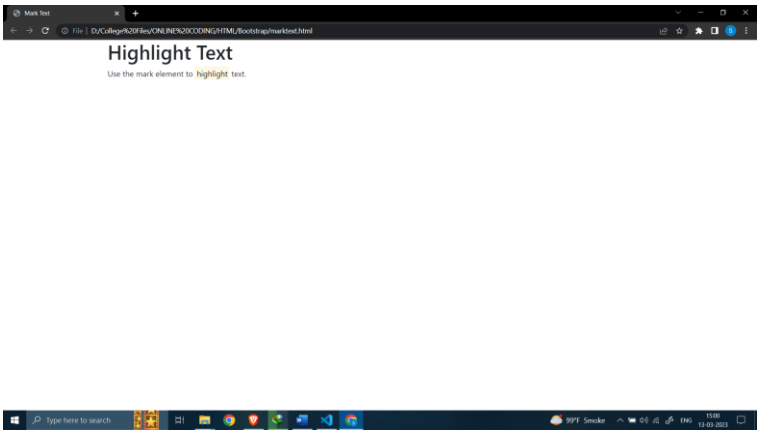
Display headings are used to stand out more than normal headings (larger font-size and lighter font-weight), and there are four classes to choose from: **.display-1**, **.display-2**, **.display-3**, **.display-4**:



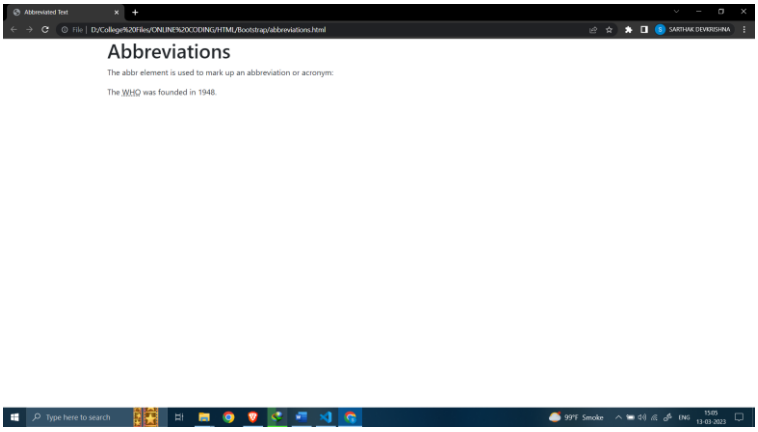
In Bootstrap 4 the HTML <small> element is used to create a lighter, secondary text in any heading:



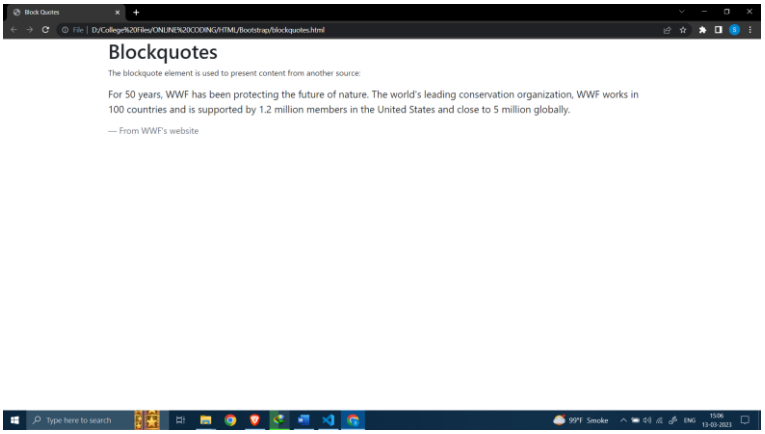
Bootstrap 4 will style the HTML <mark> element with a yellow background color and some padding:



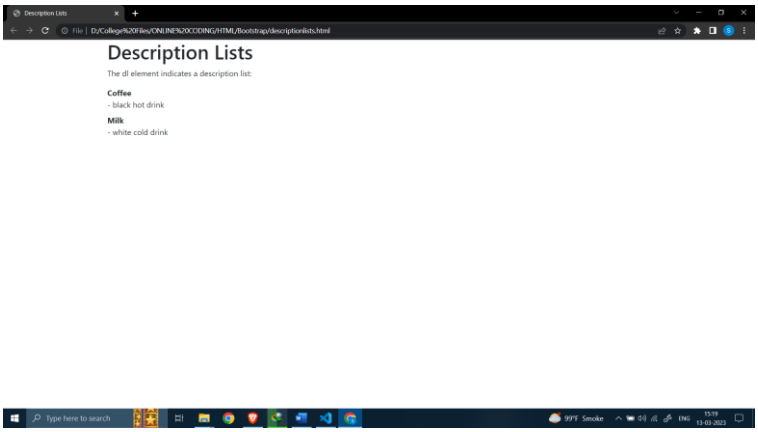
Bootstrap 4 will style the HTML <abbr> element with a dotted border bottom:



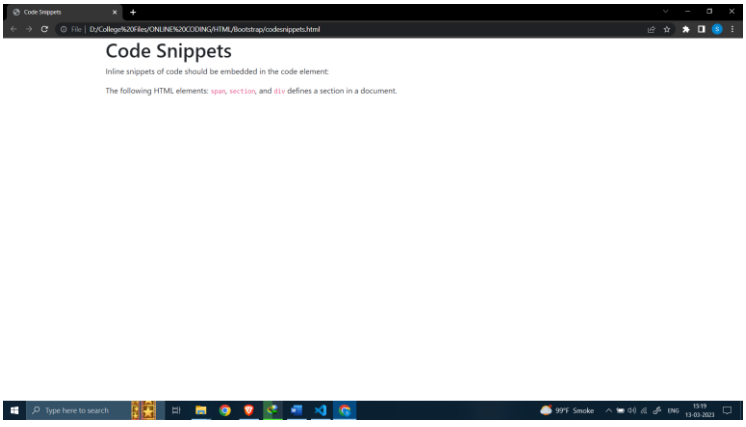
Add the .blockquote class to a <blockquote> when quoting blocks of content from another source:



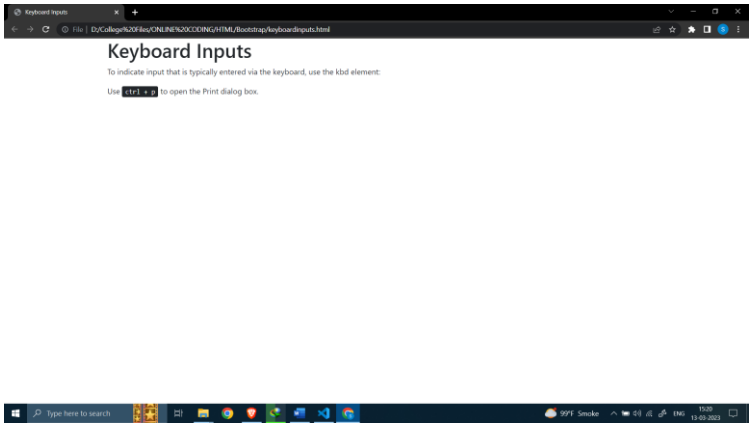
Bootstrap 4 will style the HTML <dl> element in the following way:



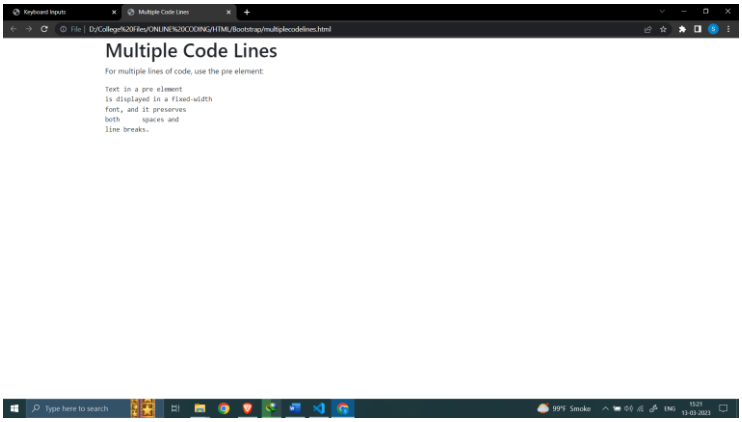
Bootstrap 4 will style the HTML <code> element in the following way:



Bootstrap 4 will style the HTML <kbd> element in the following way:

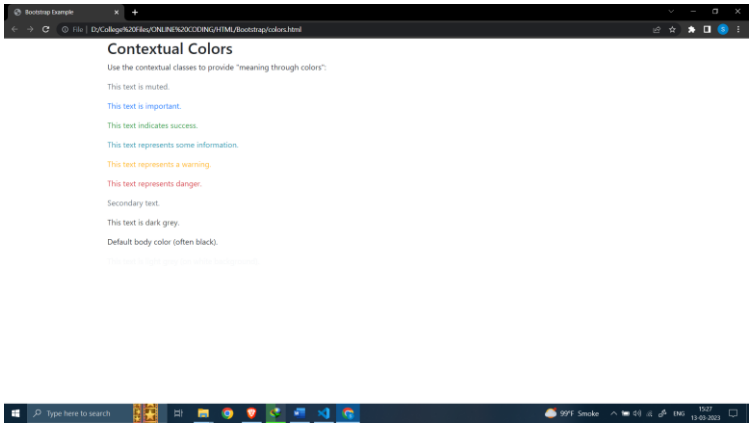


Bootstrap 4 will style the HTML <pre> element in the following way:

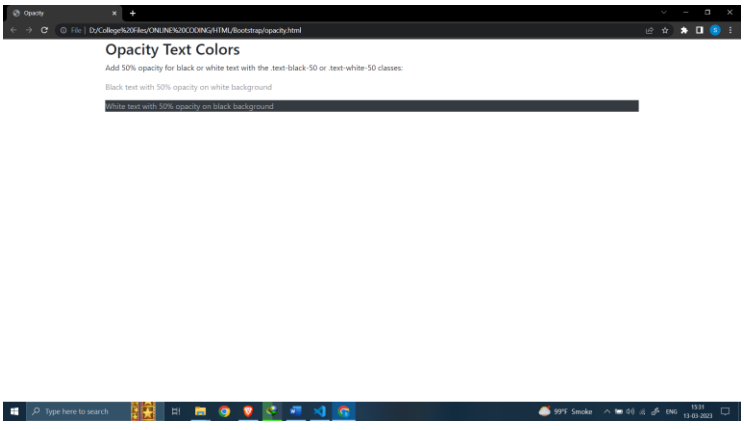


Colors

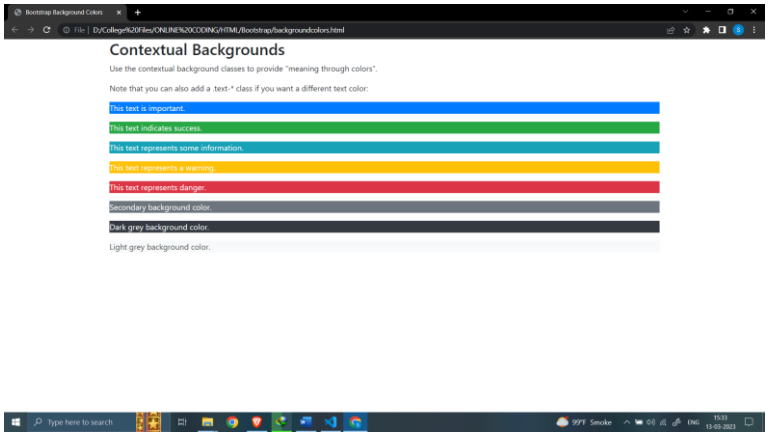
Bootstrap 4 has some contextual classes that can be used to provide meaning through colors. The classes for text colors are, .text-muted, .text-primary, .text-success, .text-info, .text-warning, .text-danger, .text-secondary, .text-white, .text-dark, .text-body (default body color/often black) and .text-light:



You can also add 50% opacity for black or white text with the .text-black-50 or .text-white-50 classes:

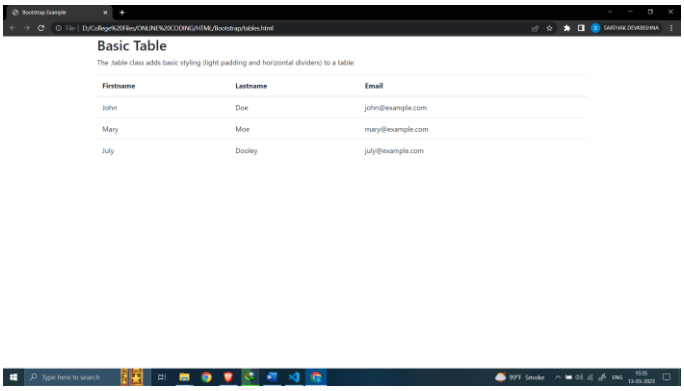


The classes for background colors are: .bg-primary, .bg-success, .bg-info, .bg-warning, .bg-danger, .bg-secondary, .bg-dark and .bg-light. Note that background colors do not set the text color, so in some cases you'll want to use them together with a .text-* class.

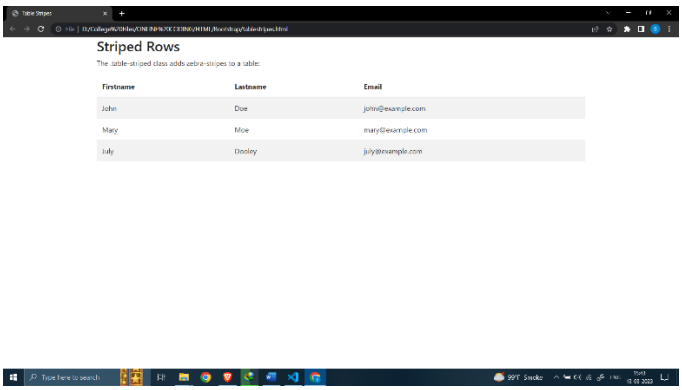


Tables

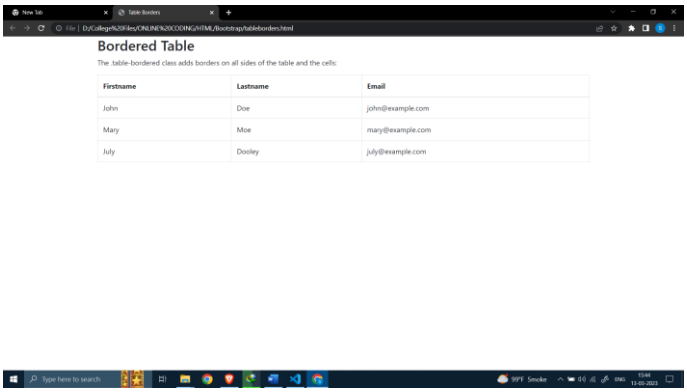
A basic Bootstrap 4 table has a light padding and horizontal dividers. The **.table** class adds basic styling to a table:



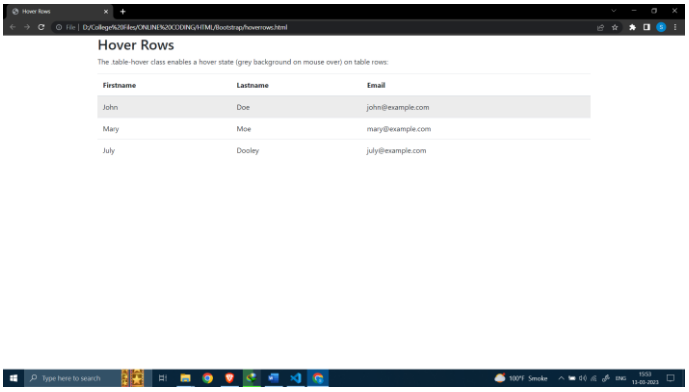
The .table-striped class adds zebra-stripes to a table:



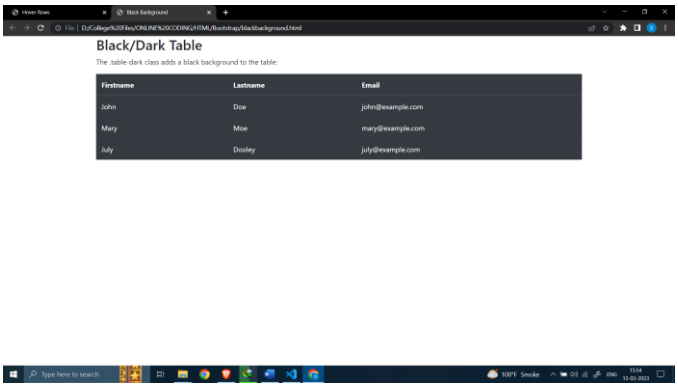
The .table-bordered class adds borders on all sides of the table and cells:



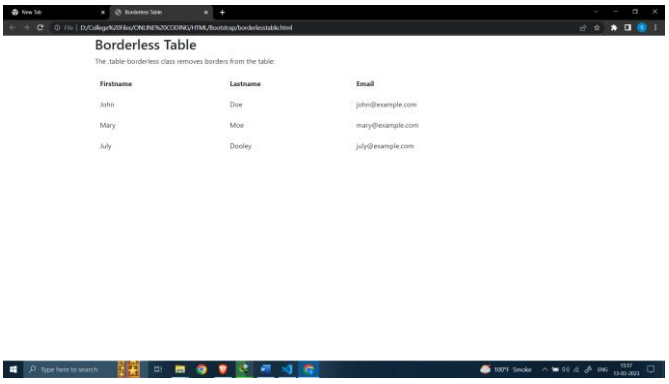
The .table-hover class adds a hover effect (grey background color) on table rows:



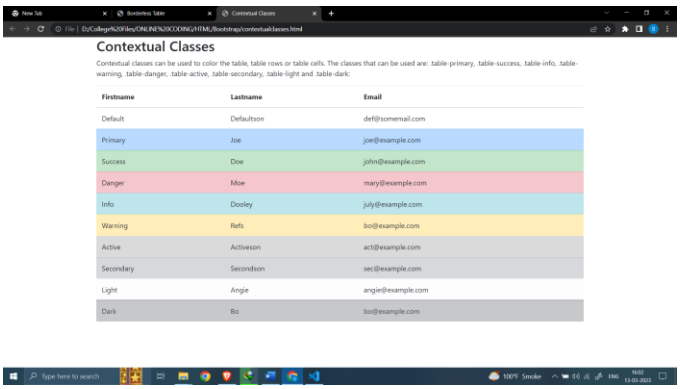
The .table-dark class adds a black background to the table:



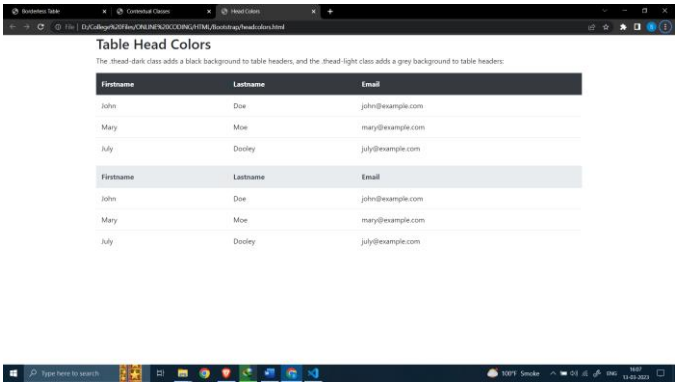
The .table-borderless class removes borders from the table:



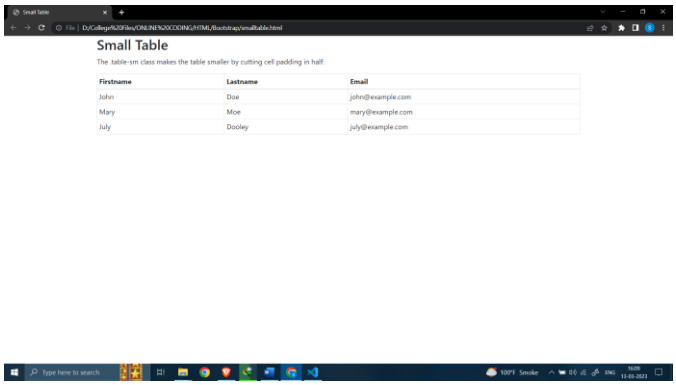
Contextual classes can be used to color the whole table (<table>), the table rows (<tr>) or table cells (<td>):



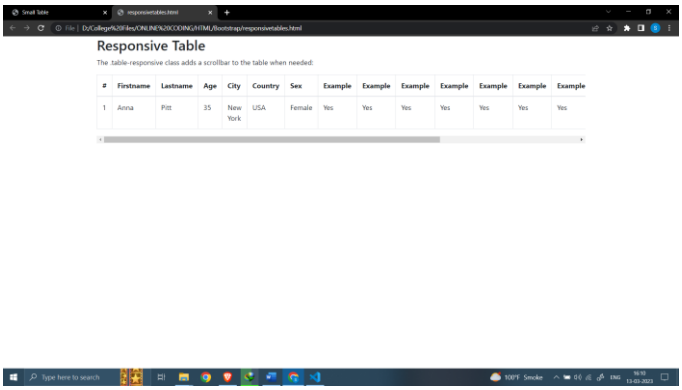
The .thead-dark class adds a black background to table headers, and the .thead-light class adds a grey background to table headers:



The .table-sm class makes the table smaller by cutting cell padding in half:

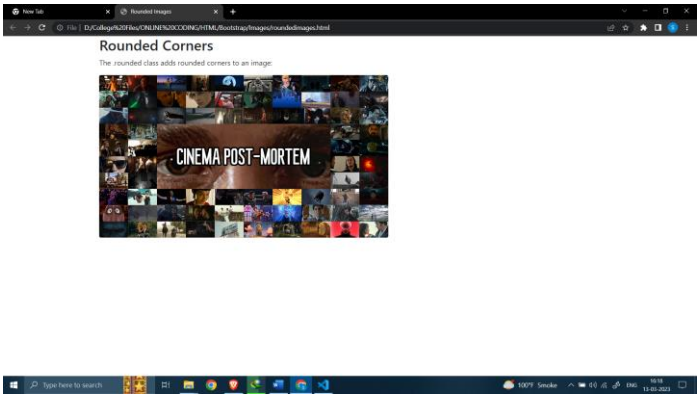


The .table-responsive class adds a scrollbar to the table when needed (when it is too big horizontally):

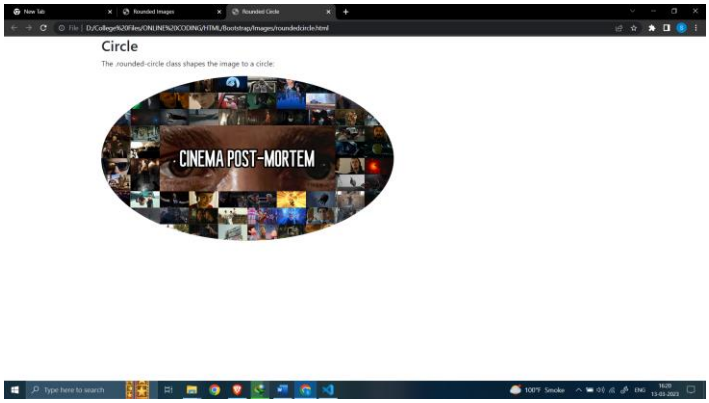


Images

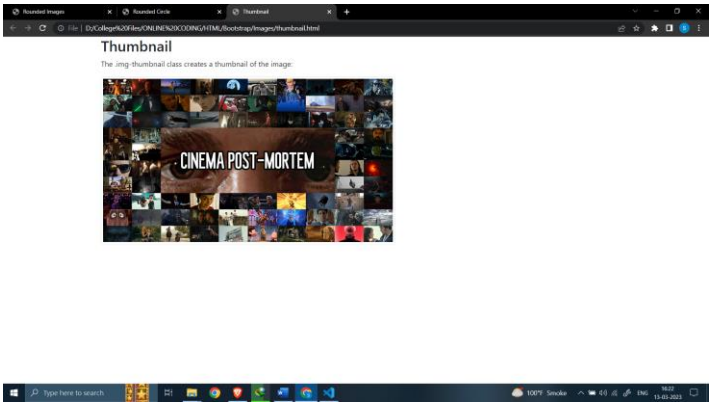
The .rounded class adds rounded corners to an image:



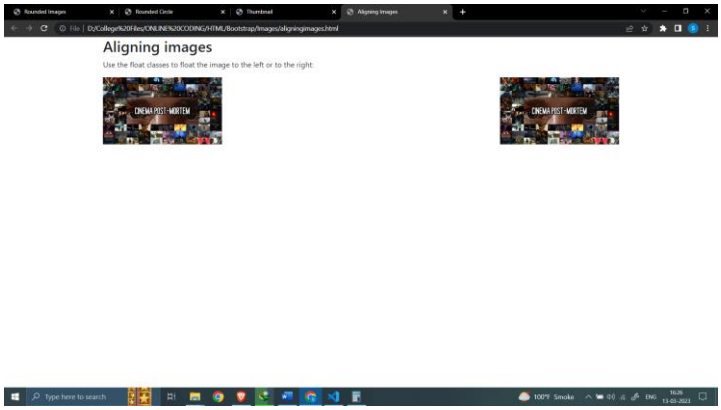
The .rounded-circle class shapes the image to a circle:



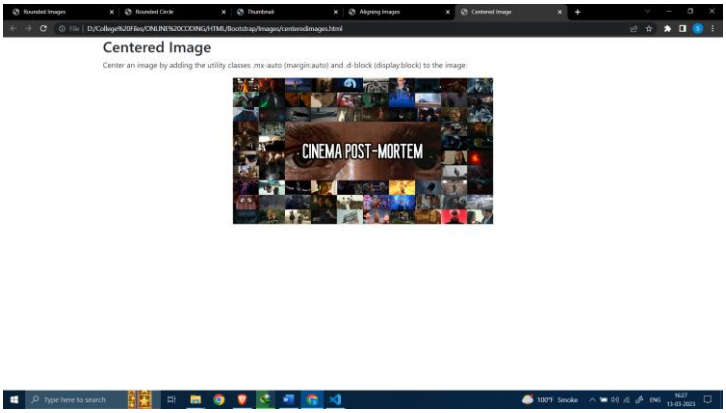
The .img-thumbnail class shapes the image to a thumbnail (bordered):



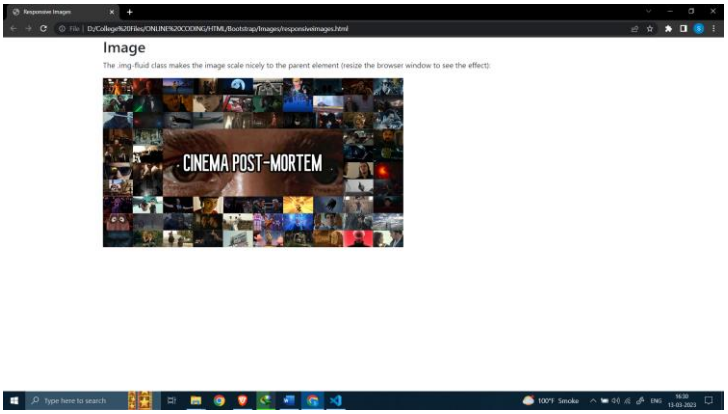
Float an image to the right with the .float-right class or to the left with .float-left:



Center an image by adding the utility classes .mx-auto (margin:auto) and .d-block (display:block) to the image:

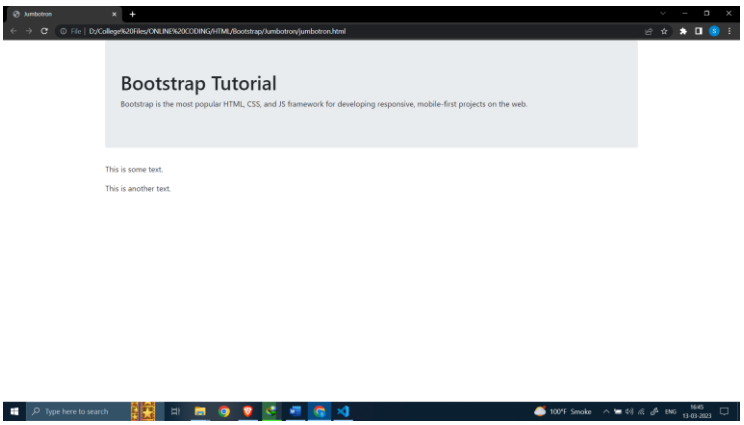


Images come in all sizes. So do screens. Responsive images automatically adjust to fit the size of the screen. Create responsive images by adding an .img-fluid class to the <img> tag. The image will then scale nicely to the parent element. The .img-fluid class applies max-width: 100%; and height: auto; to the image:

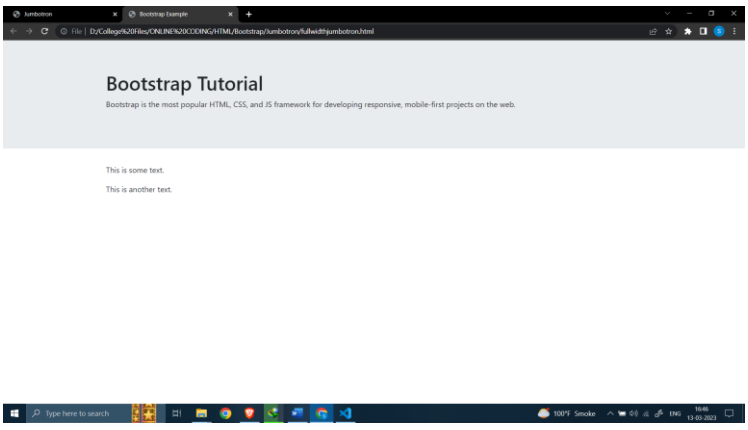


Jumbotron

A jumbotron indicates a big grey box for calling extra attention to some special content or information. Inside a jumbotron you can put nearly any valid HTML, including other Bootstrap elements/classes. Use a element with class .jumbotron to create a jumbotron:

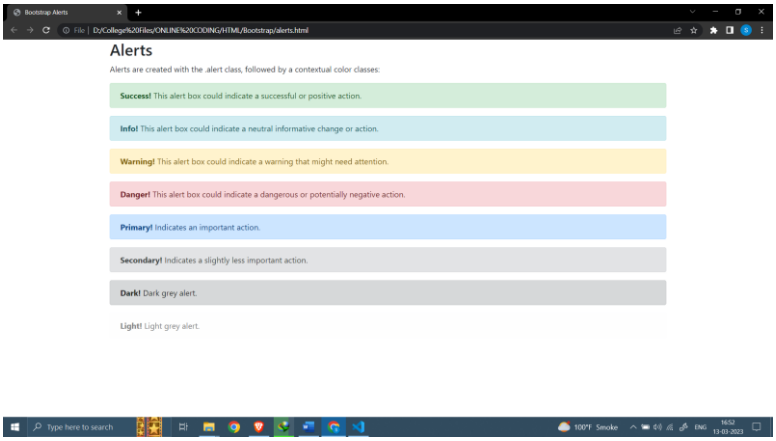


If you want a full-width jumbotron without rounded borders, add the .jumbotron-fluid class and a .container or .container-fluid inside of it:



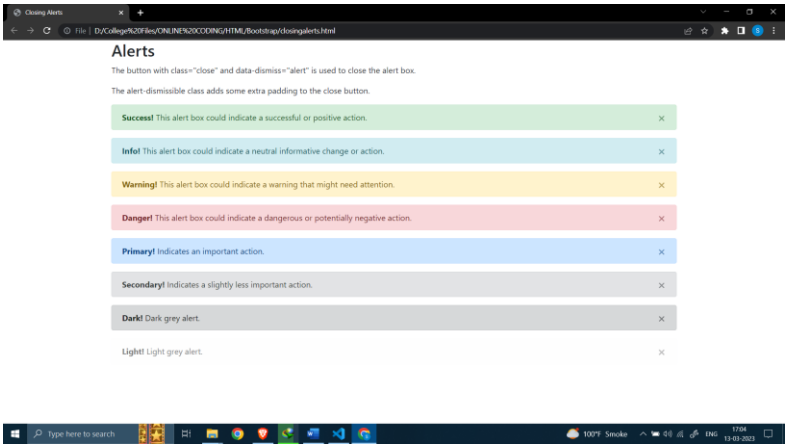
Alerts

Bootstrap 4 provides an easy way to create predefined alert messages. Alerts are created with the .alert class, followed by one of the contextual classes .alert-success, .alert-info, .alert-warning, .alert-danger, .alert-primary, .alert-secondary, .alert-light or .alert-dark:



Add the **alert-link** class to any links inside the alert box to create matching colored links.

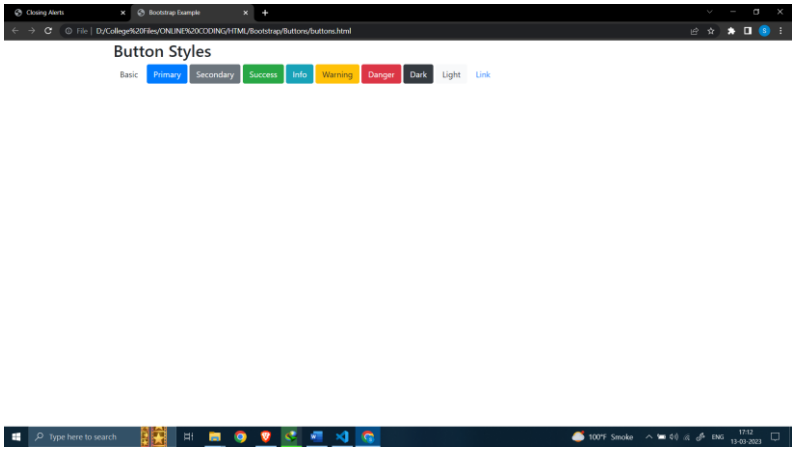
To close the alert message, add a **.alert-dismissible** class to the alert container. Then add class="close" and data-dismiss="alert" to a link or a button element (when you click on this the alert box will disappear).



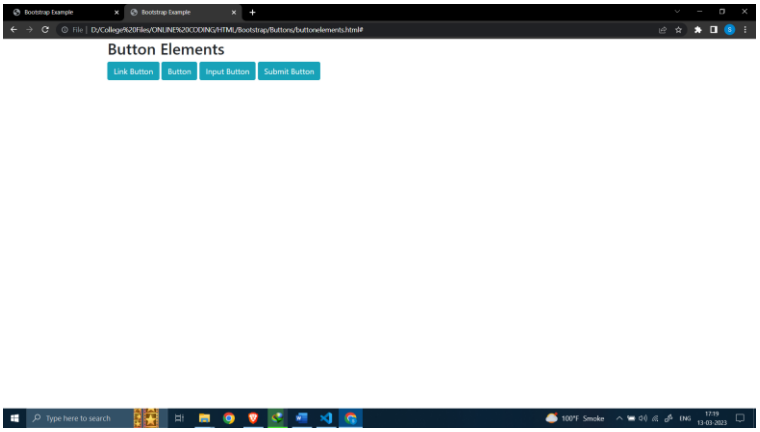
The **.fade** and **.show** classes adds a fading effect when closing the alert message.

Buttons

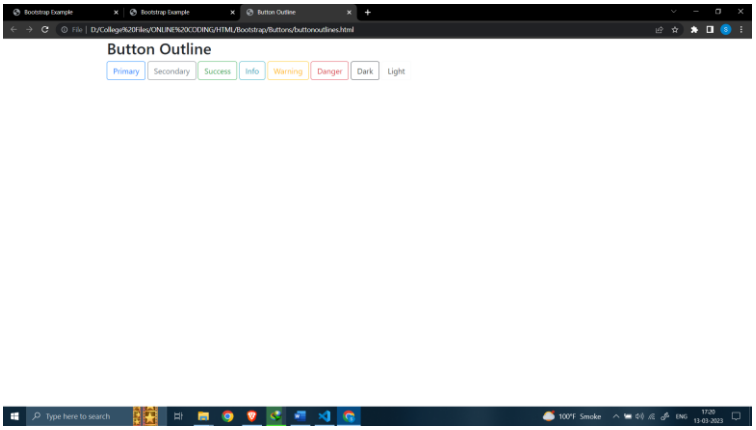
Bootstrap 4 provides different styles of buttons:



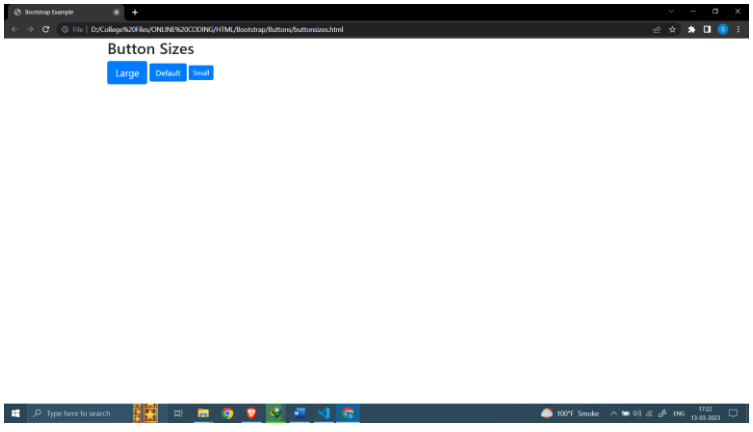
The button classes can be used on <a>, <button>, or <input> elements:



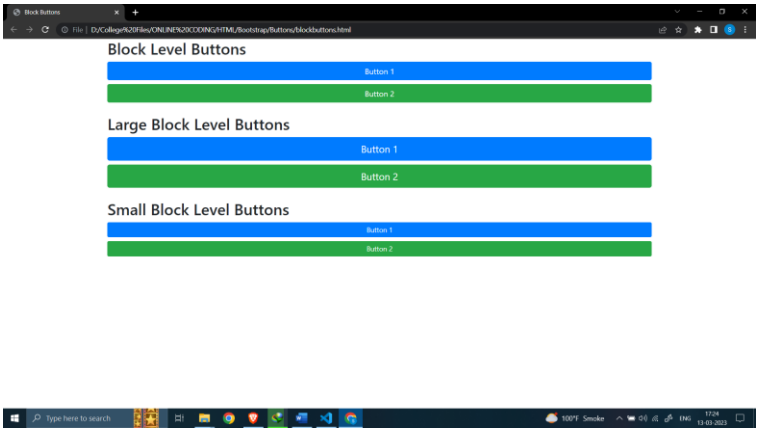
Bootstrap 4 provides eight outline/bordered buttons:



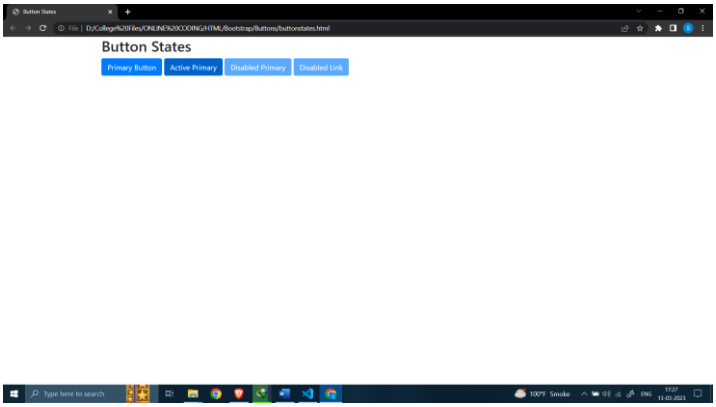
Use the **.btn-lg** class for large buttons or **.btn-sm** class for small buttons:



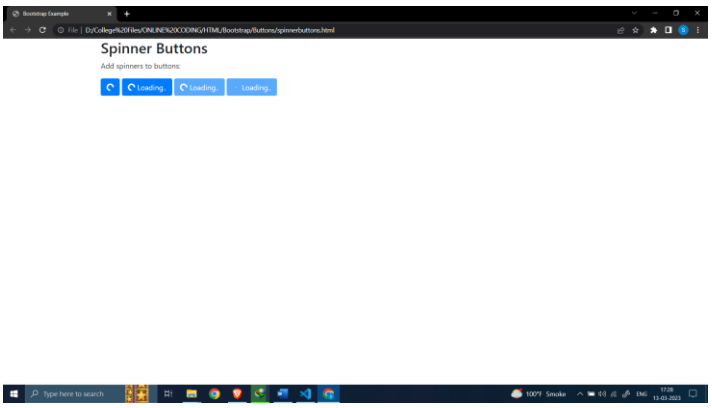
Add class **.btn-block** to create a block level button that spans the entire width of the parent element:



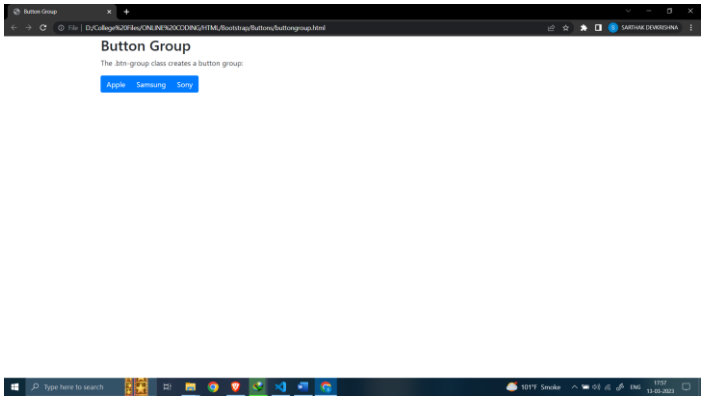
A button can be set to an active (appear pressed) or a disabled (unclickable) state. The class **.active** makes a button appear pressed, and the disabled attribute makes a button unclickable. Note that <a> elements do not support the disabled attribute and must therefore use the **.disabled** class to make it visually appear disabled:



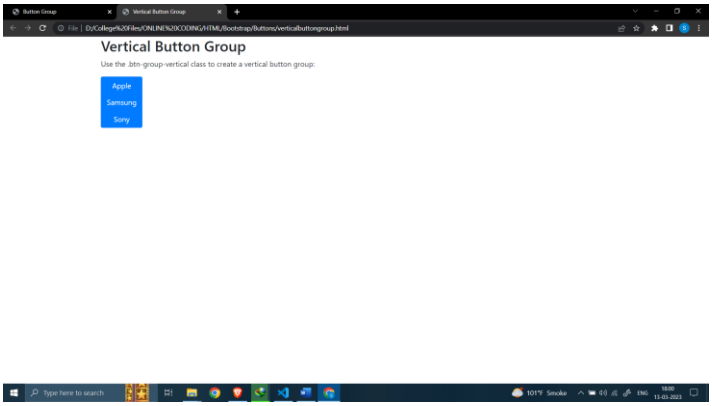
You can also add **spinners** to a button:



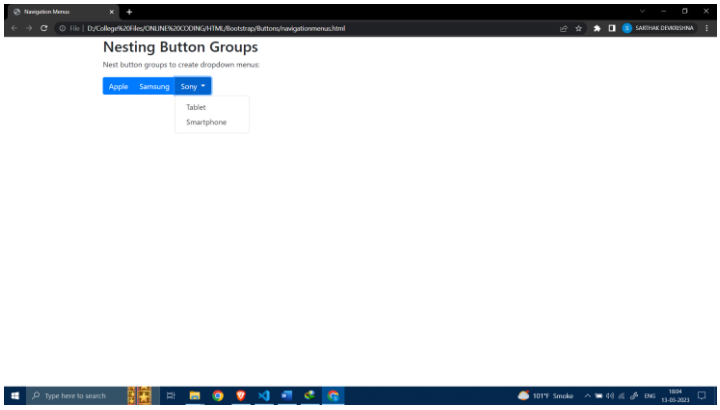
Bootstrap 4 allows you to group a series of buttons together (on a single line) in a button group. Use a element with class **.btn-group** to create a button group. Instead of applying button sizes to every button in a group, use class .btn-group-lg for a large button group or the .btn-group-sm for a small button group.



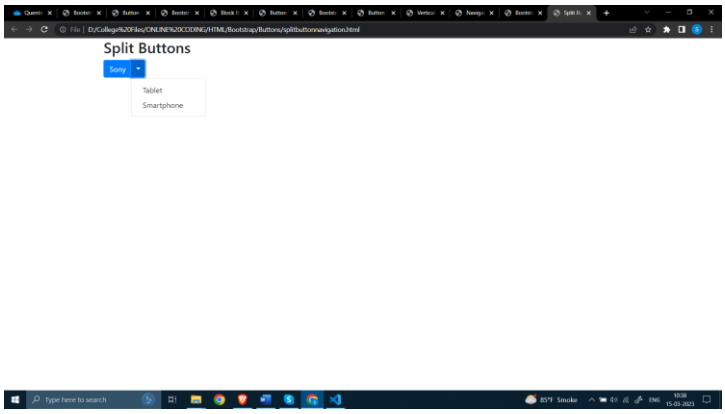
Bootstrap 4 also supports vertical button groups: Use the class **.btn-group-vertical** to create a vertical button group:



Nest button groups to create dropdown menus:

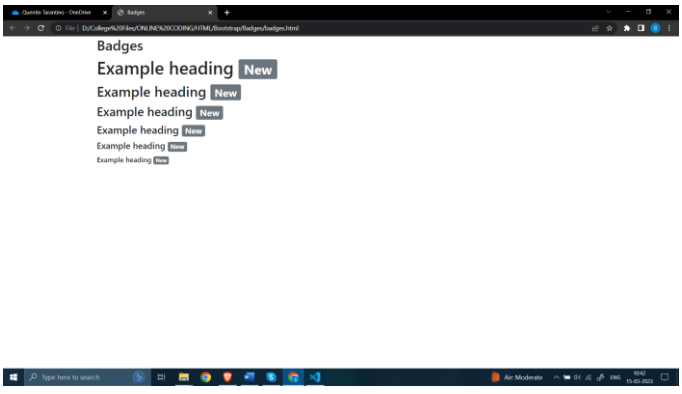


The navigation menu can be shifted from the entire button tab to just the arrow as below:

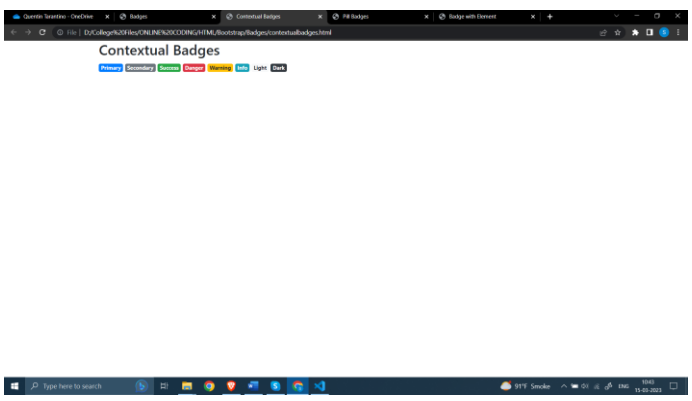


Badges

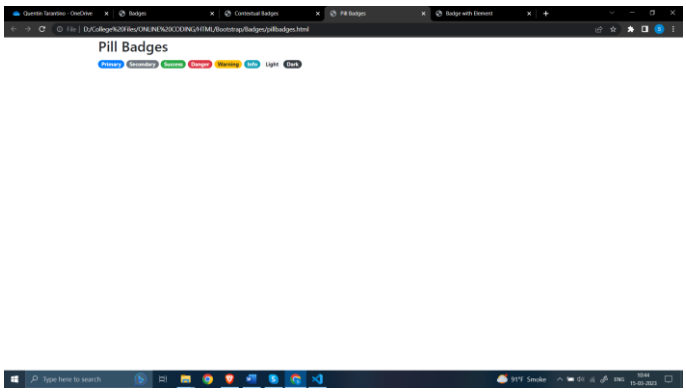
Badges are used to add additional information to any content. Use the **.badge** class together with a contextual class (like .badge-secondary) within elements to create rectangular badges. Note that badges scale to match the size of the parent element (if any):



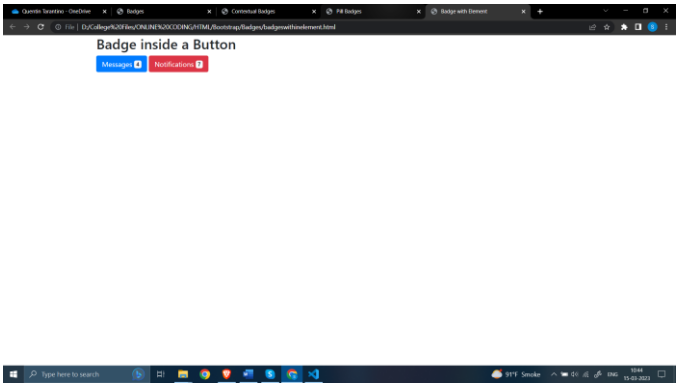
Use any of the contextual classes (**.badge-***) to change the color of a badge:



Use the **.badge-pill** class to make the badges more round:

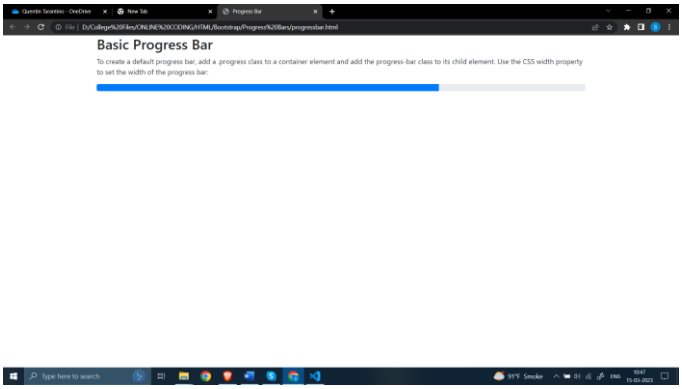


An example of using a badge inside a button:

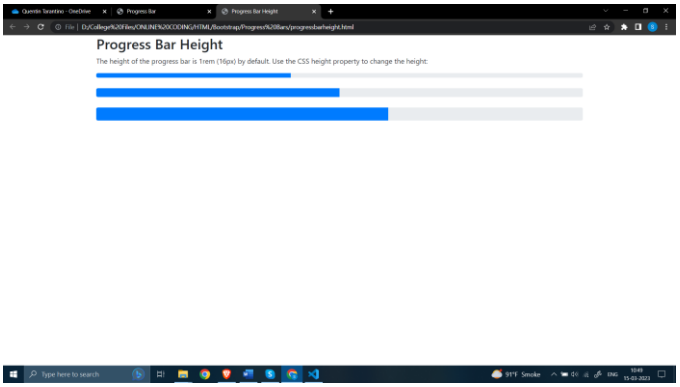


Progress Bars

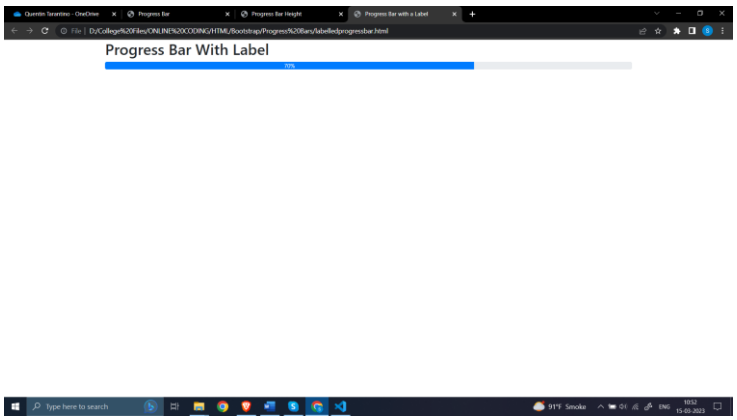
A progress bar can be used to show a user how far along he/she is in a process. To create a default progress bar, add a **.progress** class to a container element and add the .progress-bar class to its child element. Use the CSS width property to set the width of the progress bar:



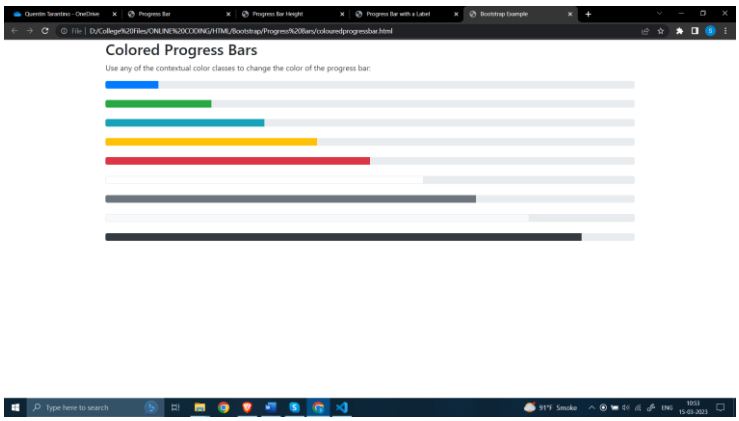
The height of the progress bar is 16px by default. Use the CSS height property to change it. Note that you must set the same height for the progress container and the progress bar:



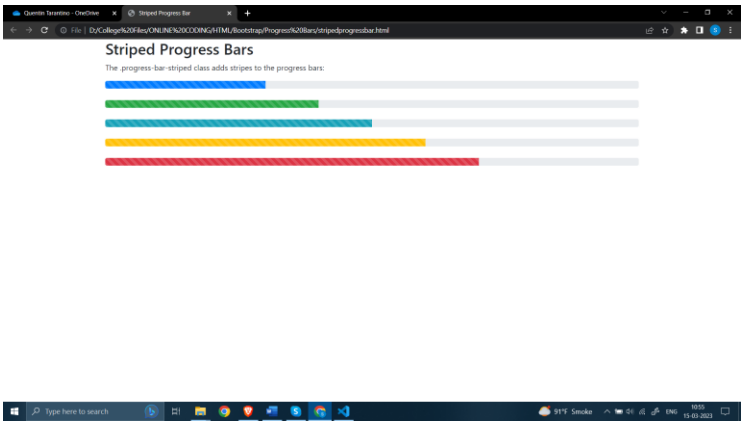
Add text inside the progress bar to show the visible percentage:



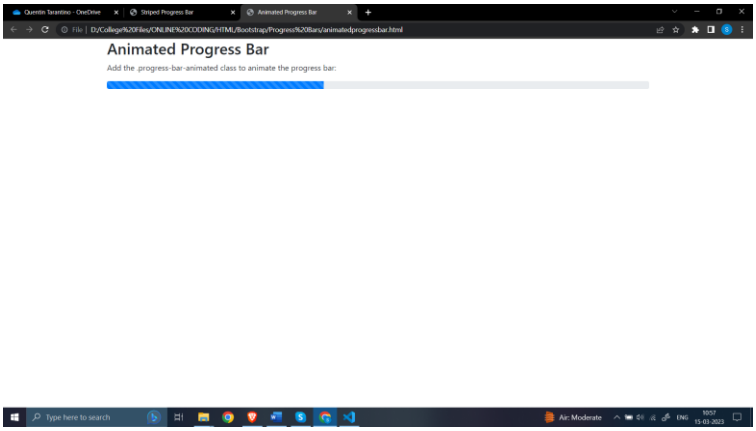
By default, the progress bar is blue (primary). Use any of the Bootstrap 4 contextual background classes to change its color:



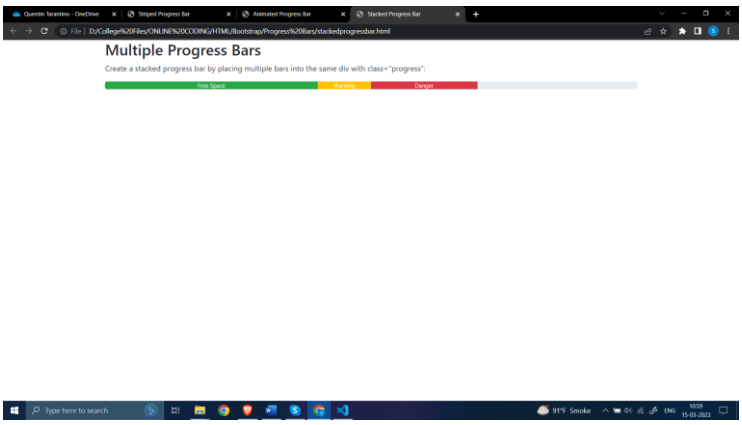
Use the **.progress-bar-striped** class to add stripes to the progress bars:



Add the **.progress-bar-animated** class to animate the progress bar:

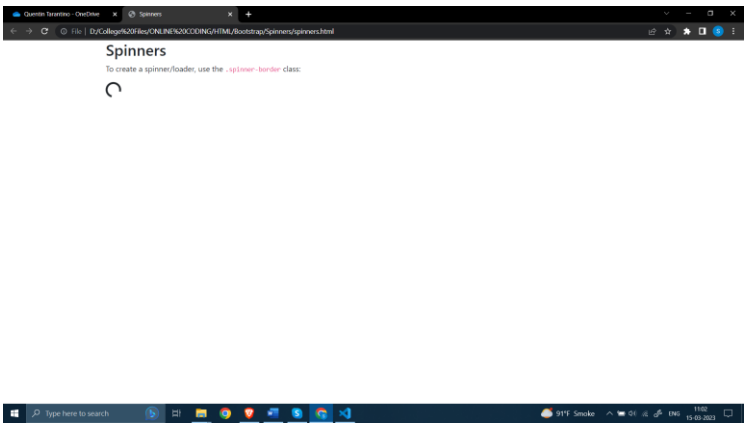


Progress bars can also be stacked:

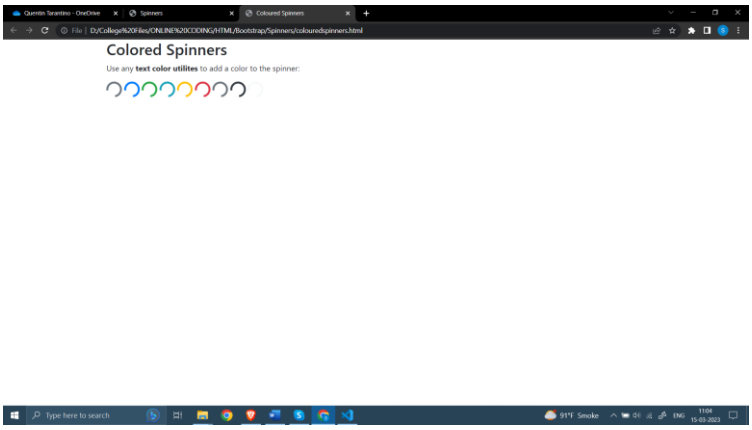


Spinners

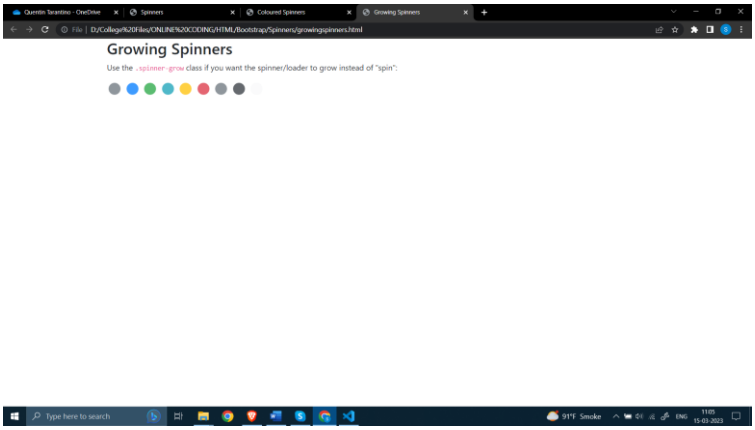
To create a spinner/loader, use the **.spinner-border** class:



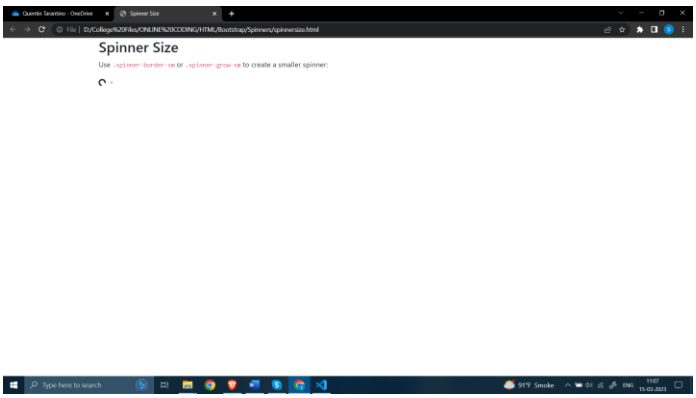
Use any text color utilites to add a color to the spinner:



Use the **.spinner-grow** class if you want the spinner/loader to grow instead of "spin":

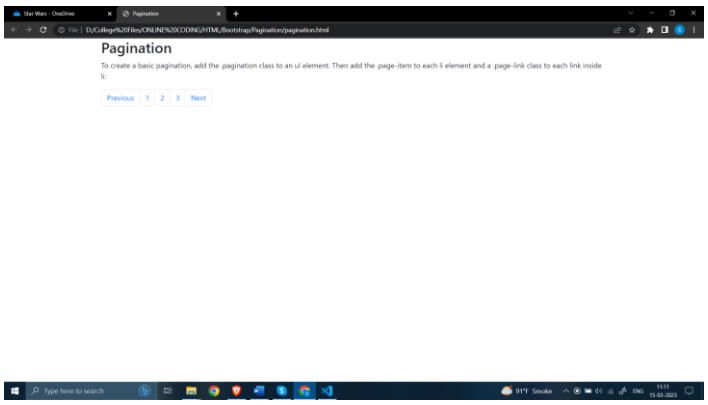


Use **.spinner-border-sm** or **.spinner-grow-sm** to create a smaller spinner:

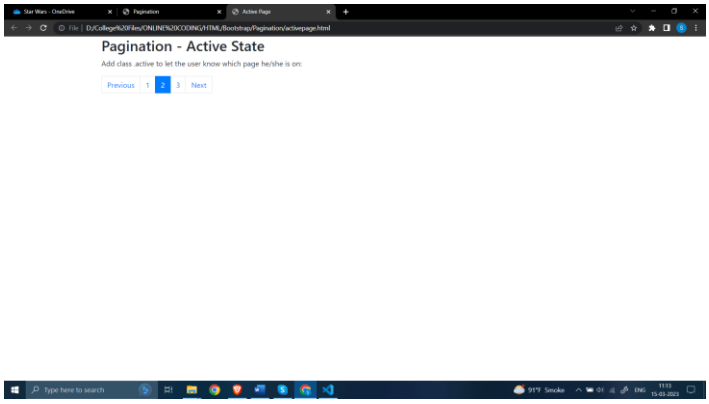


Pagination

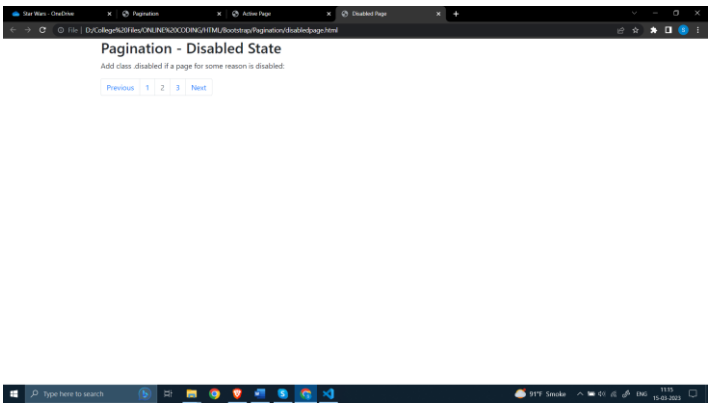
If you have a web site with lots of pages, you may wish to add some sort of pagination to each page. To create a basic pagination, add the **.pagination** class to an <ul> element. Then add the **.page-item** to each <li> element and a .page-link class to each link inside <li>:



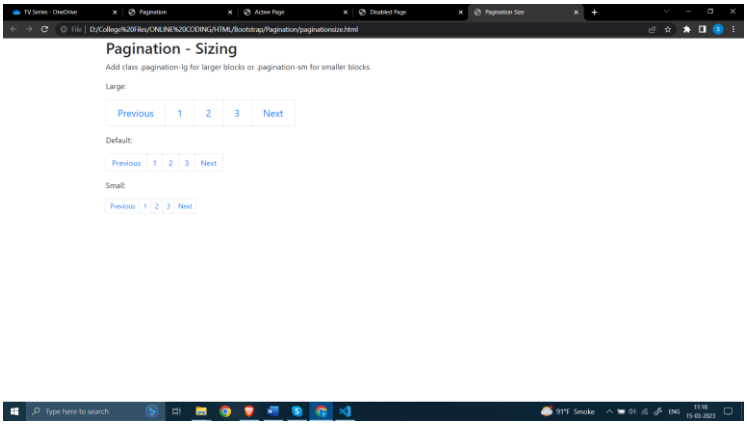
The **.active** class is used to "highlight" the current page:



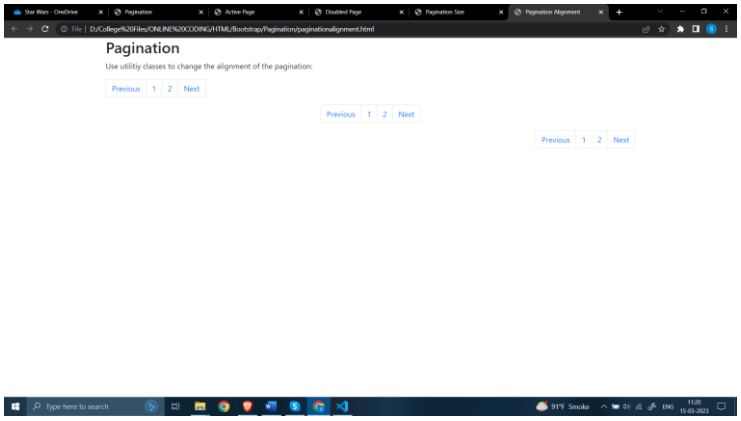
The **.disabled** class is used for un-clickable links:



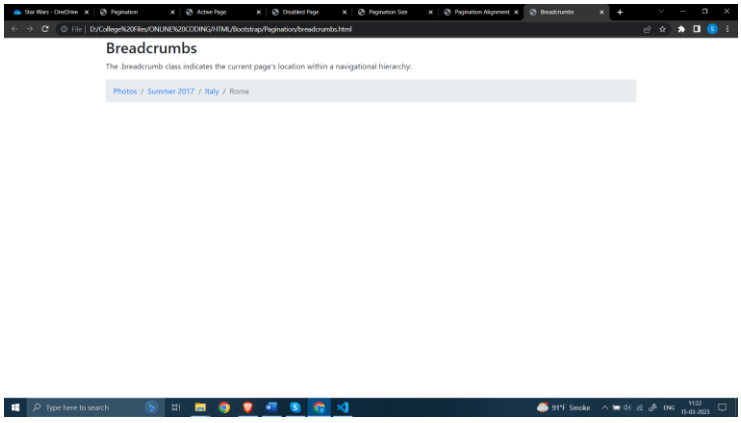
Pagination blocks can also be sized to a larger or a smaller size. Add class **.pagination-lg** for larger blocks or **.pagination-sm** for smaller blocks:



Use utility classes to change the alignment of the pagination:

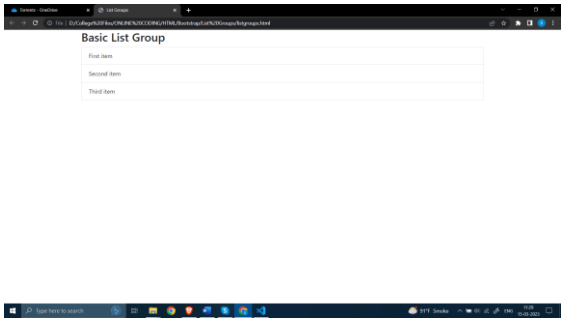


Another form for pagination, is breadcrumbs. The **.breadcrumb** and **.breadcrumb-item** classes indicates the current page's location within a navigational hierarchy:

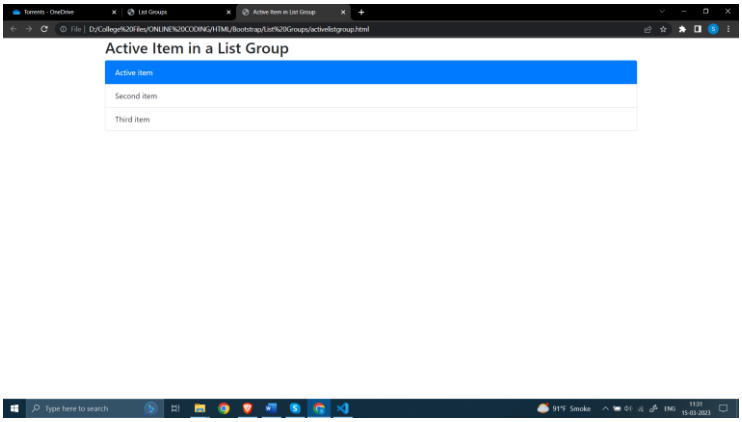


List Groups

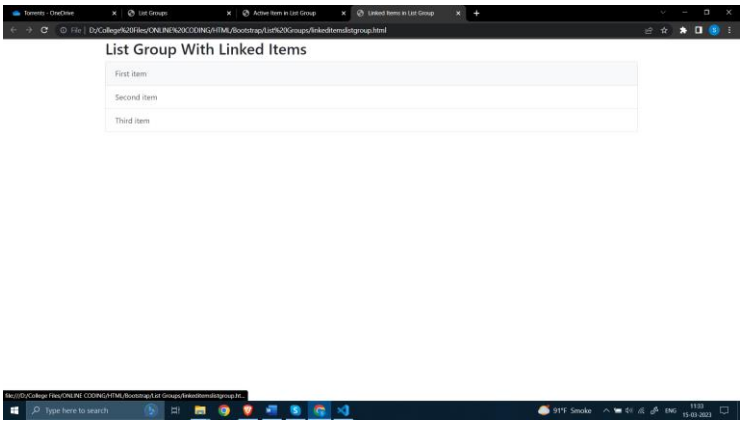
The most basic list group is an unordered list with list items. To create a basic list group, use an <ul> element with class **.list-group**, and <li> elements with class .list-group-item:



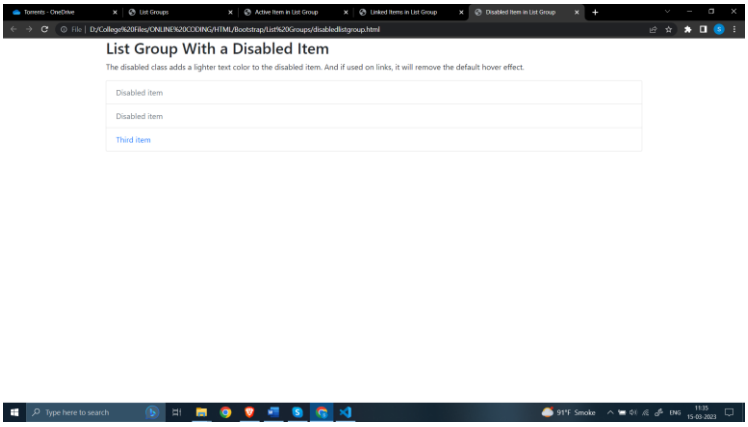
Use the **.active** class to highlight the current item:



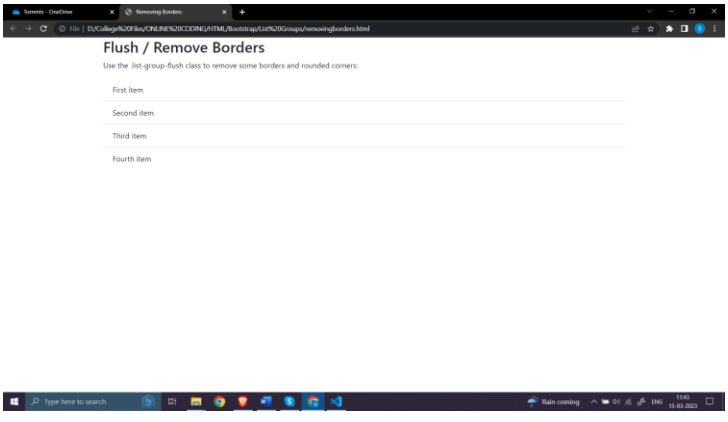
To create a list group with linked items, use ** ** instead of **<ul>** and **<a>** instead of **<li>**. Optionally, add the **.list-group-item-action** class if you want a grey background color on hover:



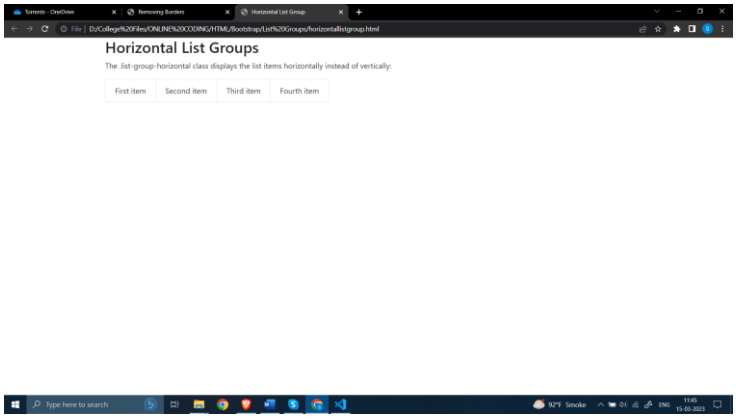
The **.disabled** class adds a lighter text color to the disabled item. And when used on links, it will remove the hover effect:



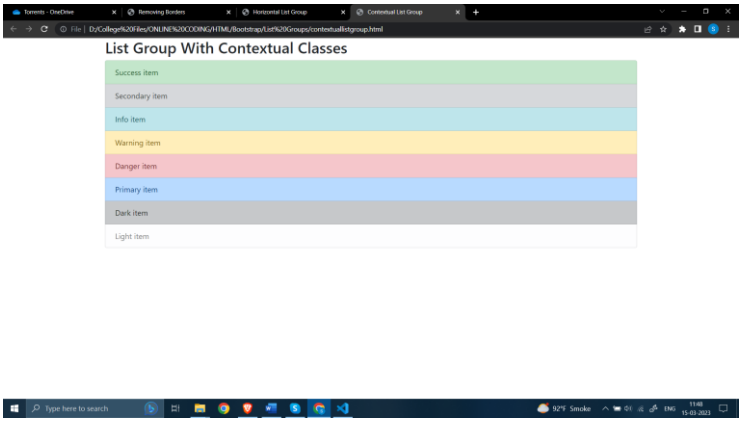
Use the **.list-group-flush** class to remove some borders and rounded corners:



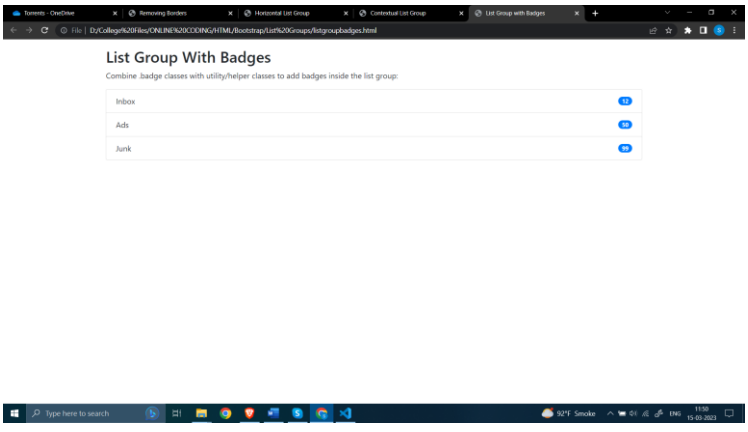
If you want the list items to display horizontally instead of vertically (side by side instead of on top of each other), add the **.list-group-horizontal** class to **.list-group**:



Contextual classes can be used to color list items. The classes for coloring list-items are: **.list-group-item-success**, **.list-group-item-secondary**, **.list-group-item-info**, **.list-group-item-warning**, **.list-group-item-danger**, **.list-group-item-primary**, **.list-group-item-dark** and **.list-group-item-light**:

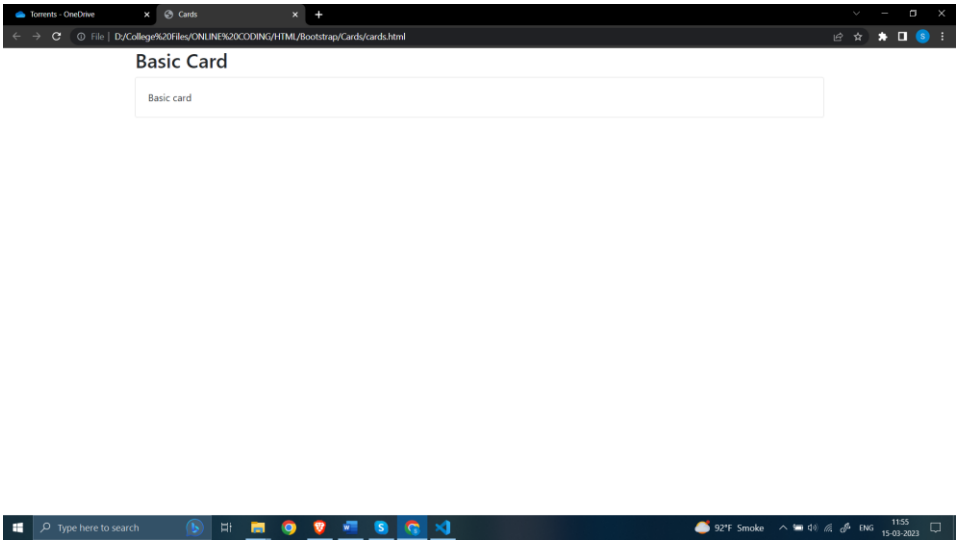


Combine **.badge** classes with utility/helper classes to add badges inside the list group:

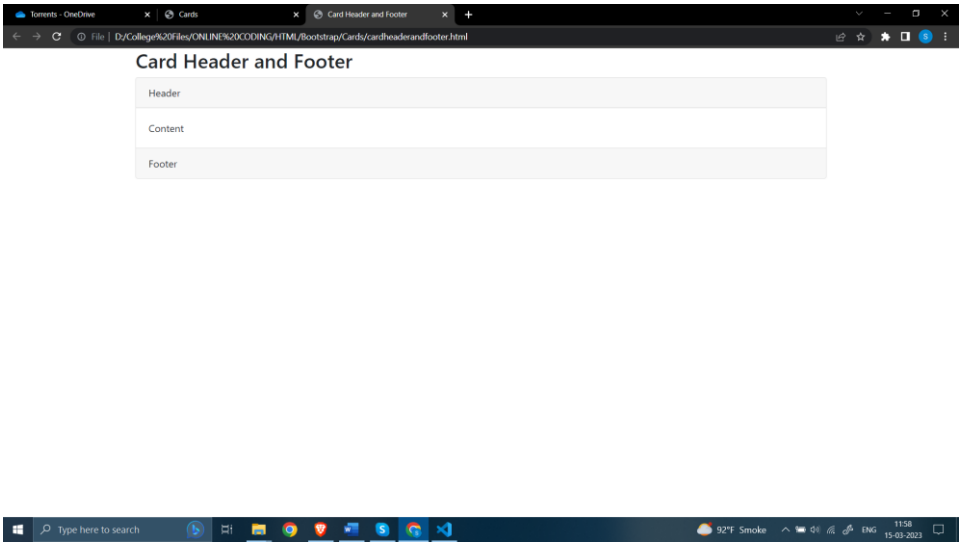


Cards

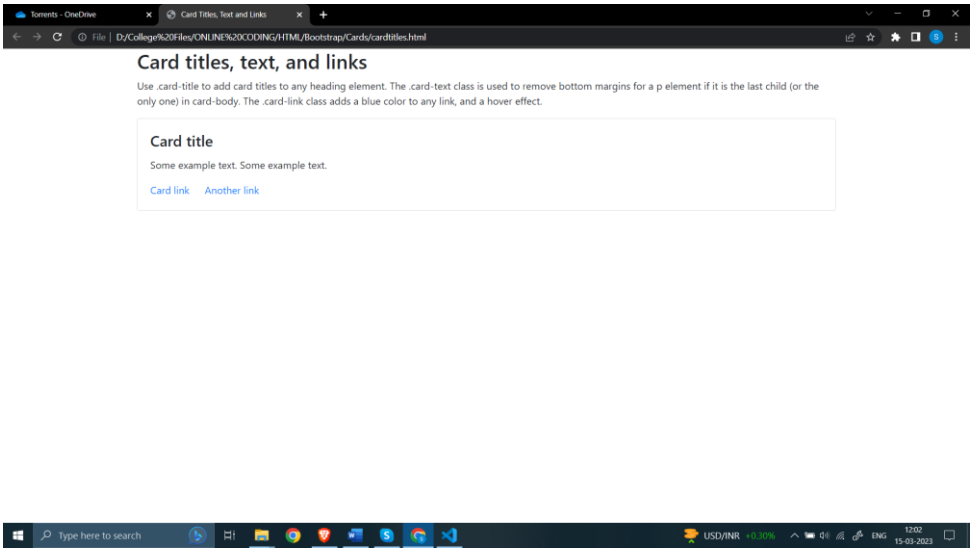
A card in Bootstrap 4 is a bordered box with some padding around its content. It includes options for headers, footers, content, colors, etc. A basic card is created with the **.card** class, and content inside the card has a .card-body class:



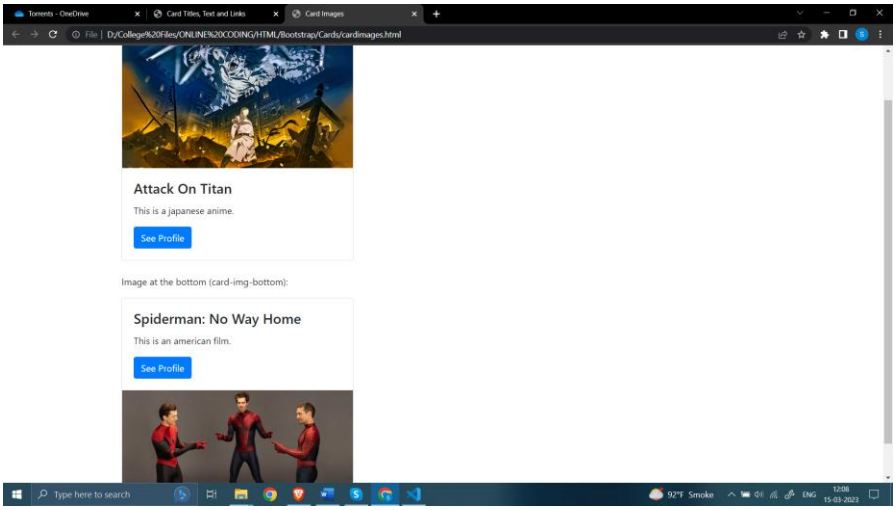
The **.card-header** class adds a heading to the card and the **.card-footer** class adds a footer to the card:



To add a background color the card, use contextual classes. Use **.card-title** to add card titles to any heading element. The **.card-text** class is used to remove bottom margins for a <p> element if it is the last child (or the only one) inside **.card-body**. The **.card-link** class adds a blue color to any link, and a hover effect.



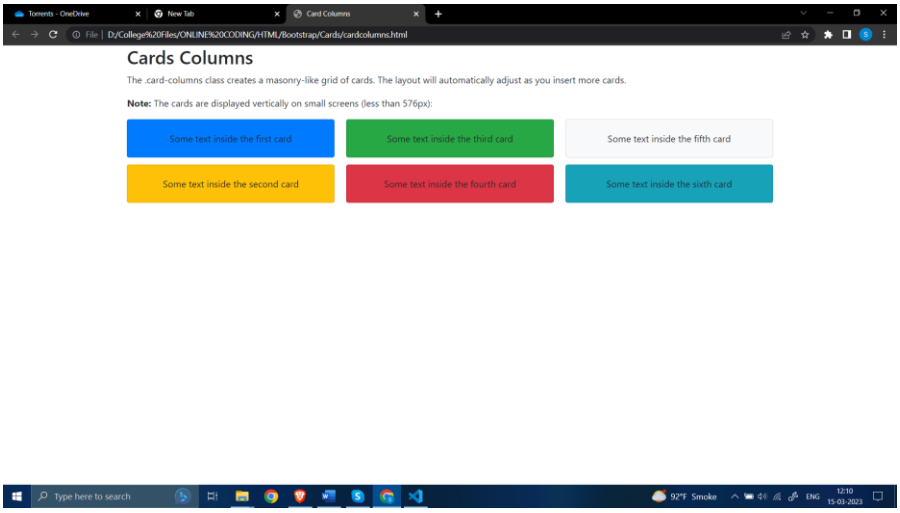
Add **.card-img-top** or **.card-img-bottom** to an <img> to place the image at the top or at the bottom inside the card. Note that we have added the image outside of the **.card-body** to span the entire width:



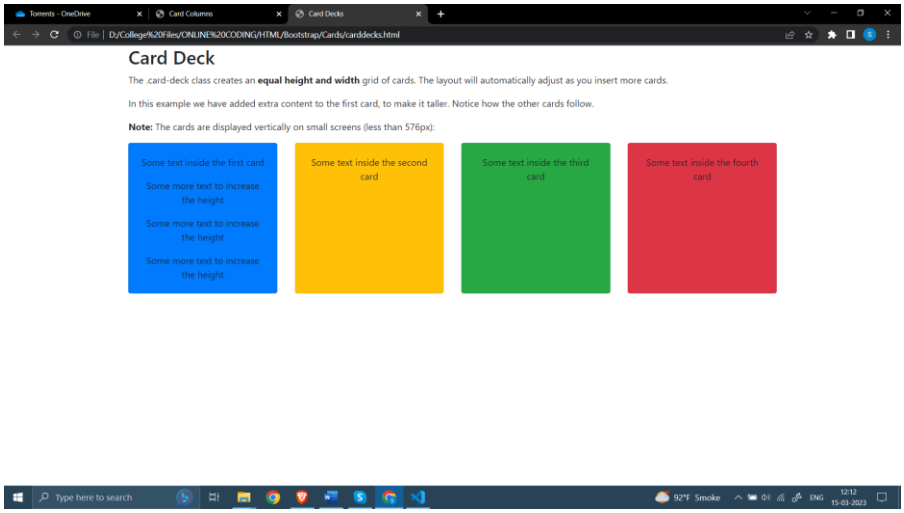
Add the **.stretched-link** class to a link inside the card, and it will make the whole card clickable and hoverable (the card will act as a link).

Turn an image into a card background and use **.card-img-overlay** to add text on top of the image.

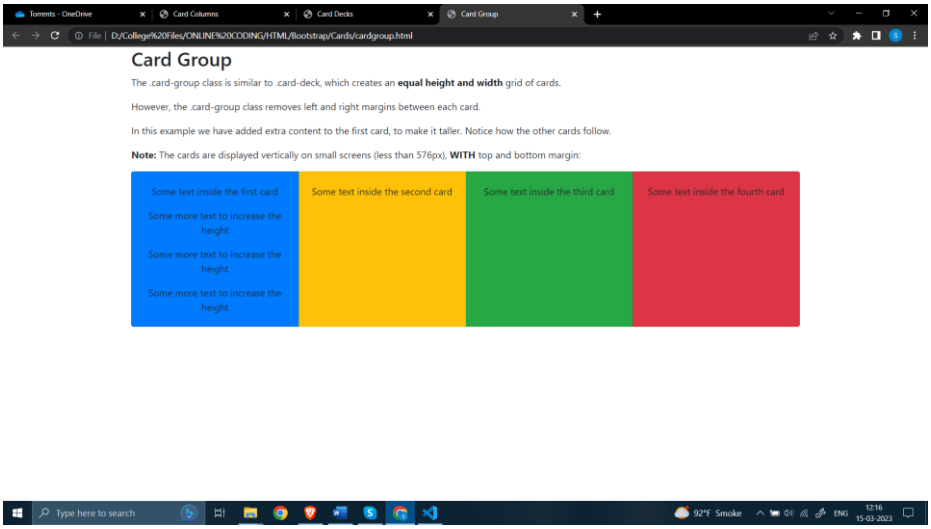
The **.card-columns** class creates a masonry-like grid of cards (like pinterest). The layout will automatically adjust as you insert more cards. The cards are displayed vertically on small screens (less than 576px).



The **.card-deck** class creates a grid of cards that are of equal height and width. The layout will automatically adjust as you insert more cards. The cards are displayed vertically on small screens (less than 576px).

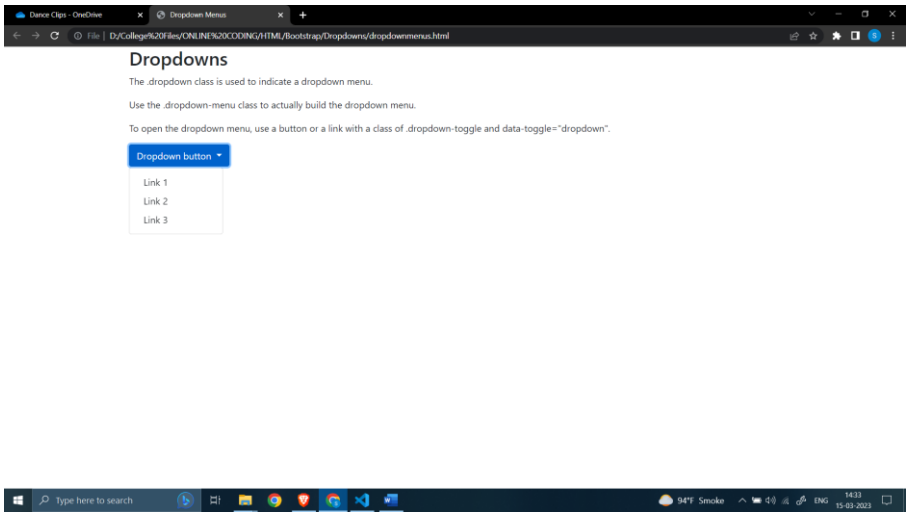


The **.card-group** class is similar to **.card-deck**. The only difference is that the .card-group class removes left and right margins between each card. The cards are displayed vertically on small screens (less than 576px), WITH top and bottom margin.

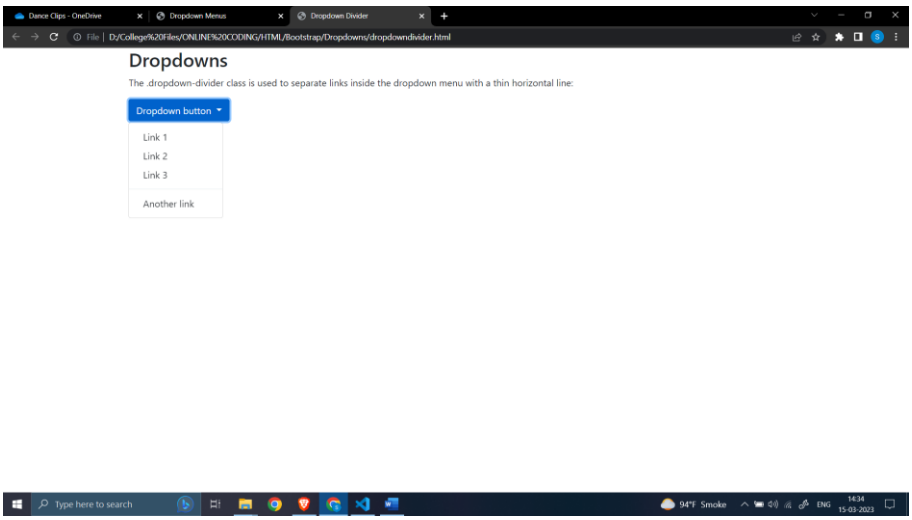


Dropdowns

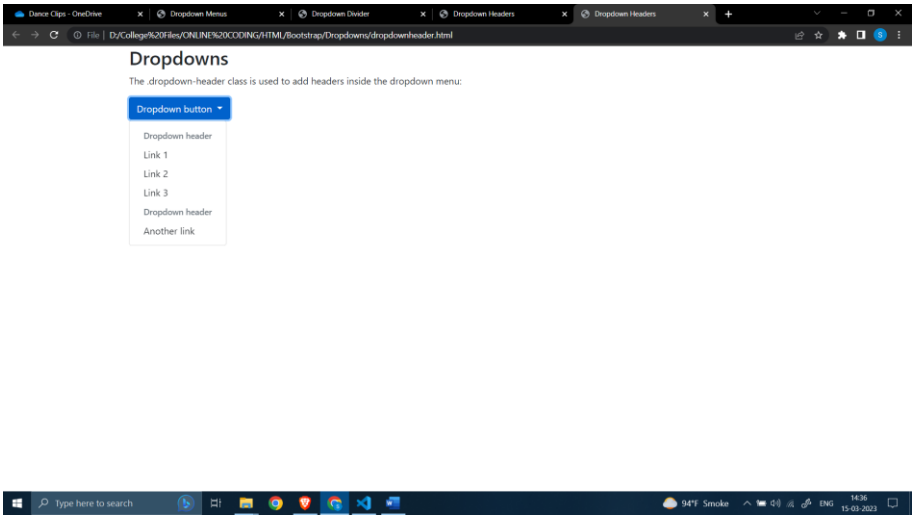
The **.dropdown** class indicates a dropdown menu. To open the dropdown menu, use a button or a link with a class of **.dropdown-toggle** and the **data-toggle="dropdown"** attribute. Add the **.dropdown-menu** class to a element to actually build the dropdown menu. Then add the **.dropdown-item** class to each element (links or buttons) inside the dropdown menu.



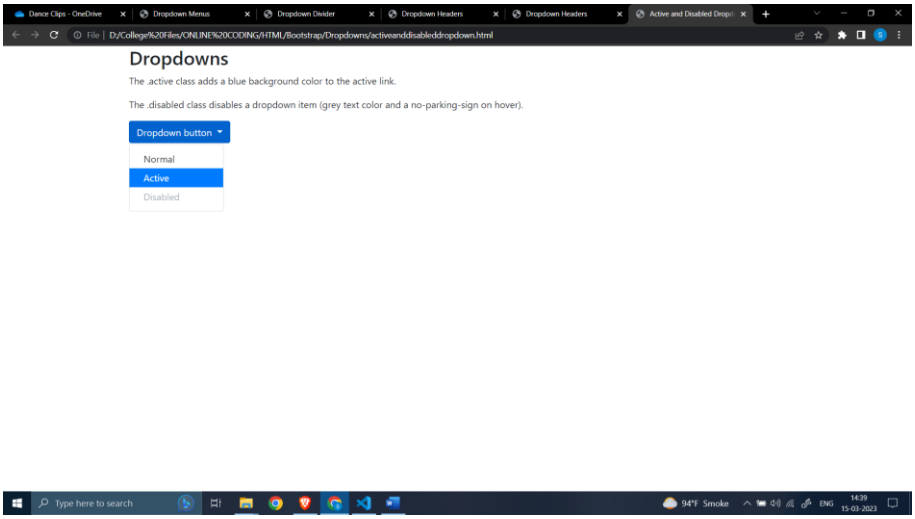
The **.dropdown-divider** class is used to separate links inside the dropdown menu with a thin horizontal border.



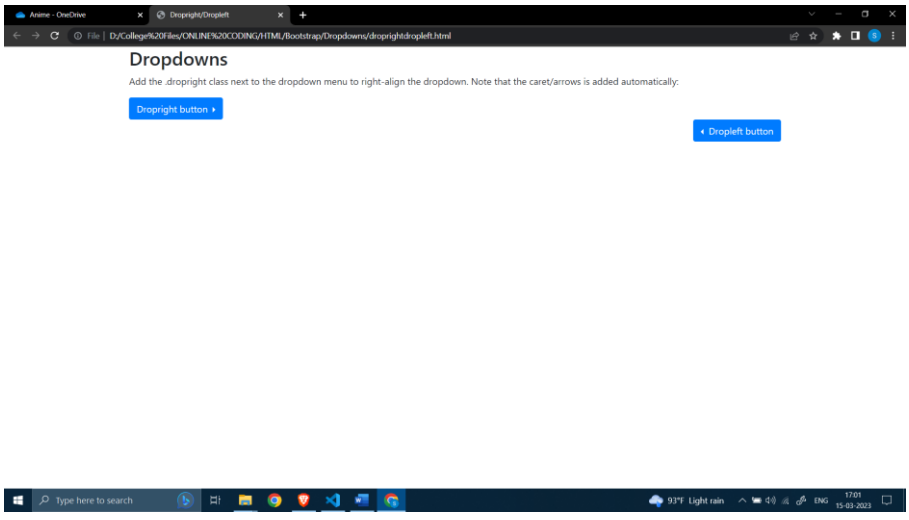
The **.dropdown-header** class is used to add headers inside the dropdown menu:



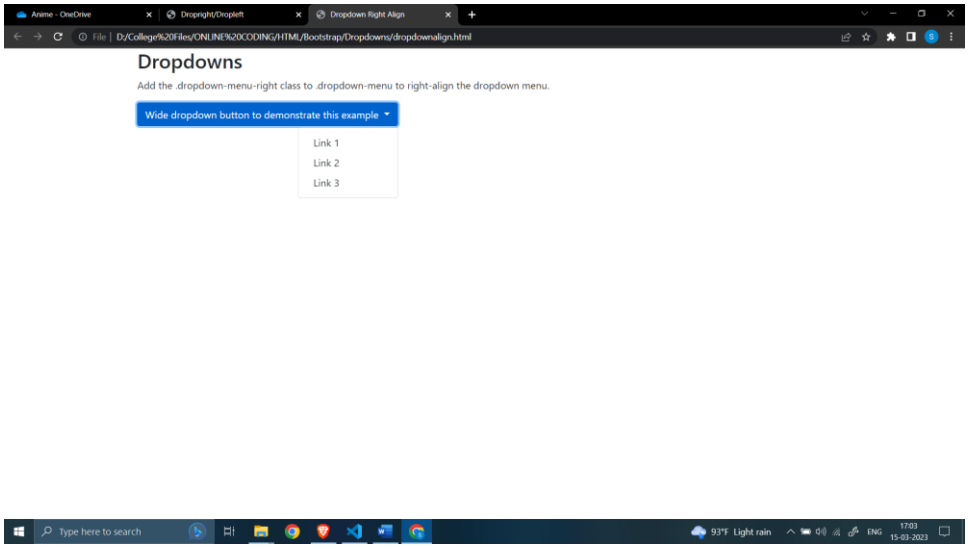
Highlight a specific dropdown item with the **.active** class (adds a blue background color). To disable an item in the dropdown menu, use the **.disabled** class (gets a light-grey text color and a "no-parking-sign" icon on hover):



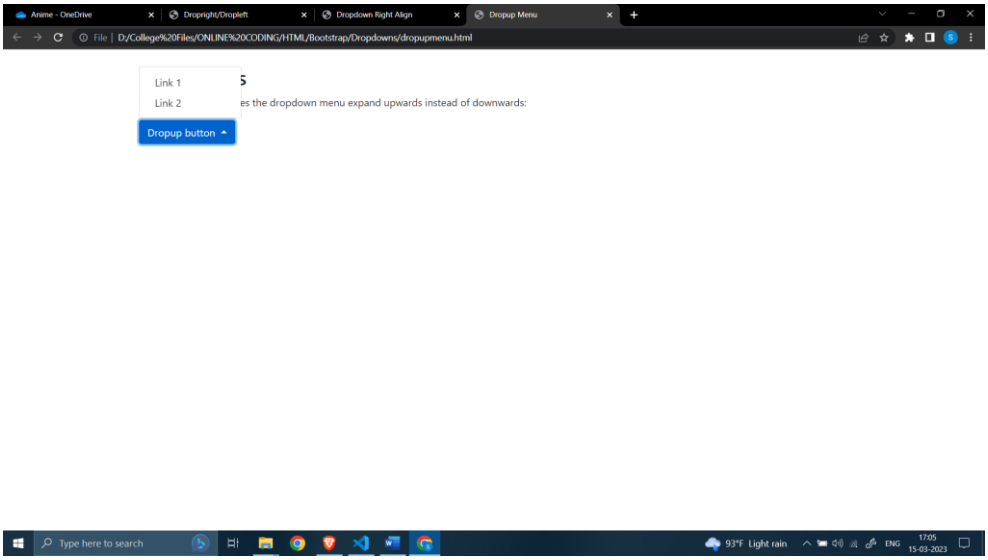
You can also create a "dropright" or "dropleft" menu, by adding the **.dropright** or **.dropleft** class to the dropdown element. Note that the caret/arrow is added automatically:



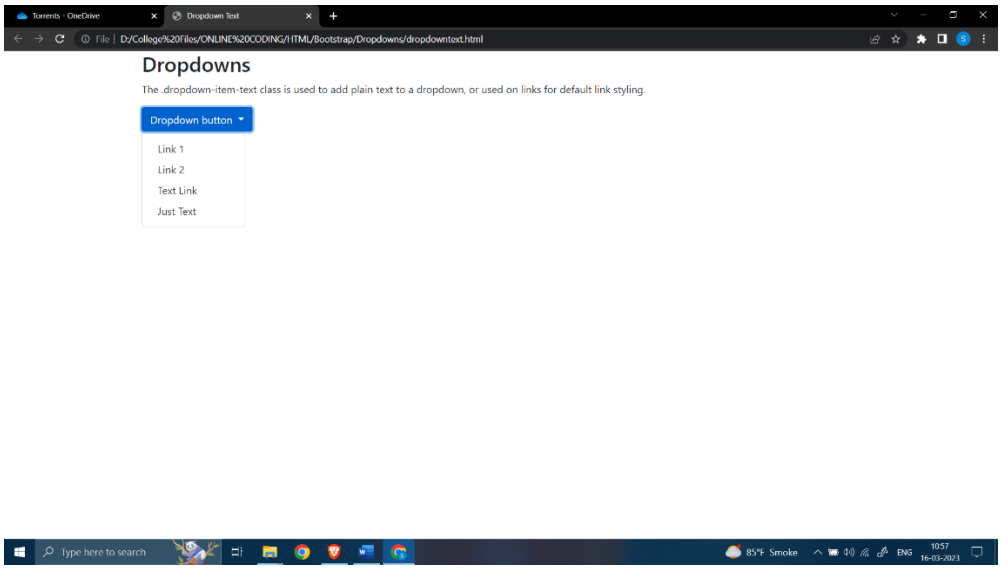
To right-align the dropdown menu, add the **.dropdown-menu-right** class to the element with .dropdown-menu:



If you want the dropdown menu to expand upwards instead of downwards, change the element with class="dropdown" to "dropup":

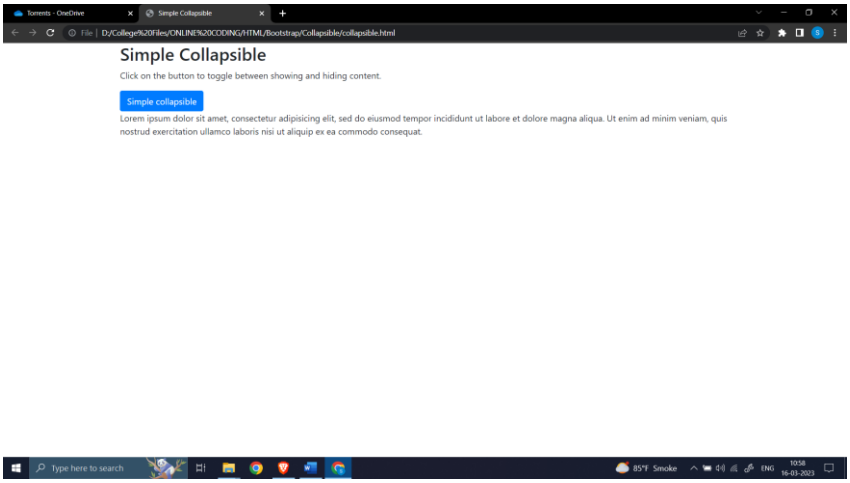


The **.dropdown-item-text** class is used to add plain text to a dropdown item, or used on links for default link styling:

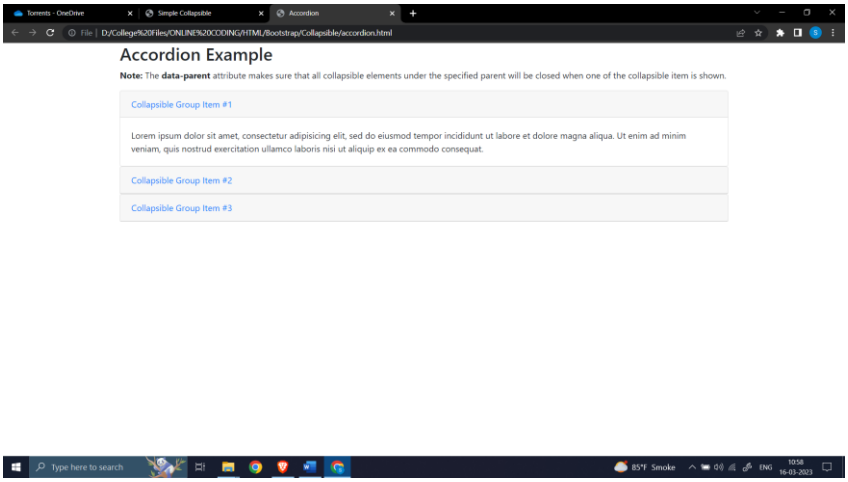


Collapsible

The **.collapse** class indicates a collapsible element (a in our example); this is the content that will be shown or hidden with a click of a button. To control (show/hide) the collapsible content, add the data-toggle="collapse" attribute to an <a> or a <button> element. Then add the data-target="#id" attribute to connect the button with the collapsible content ( ). For <a> elements, you can use the href attribute instead of the data-target attribute:

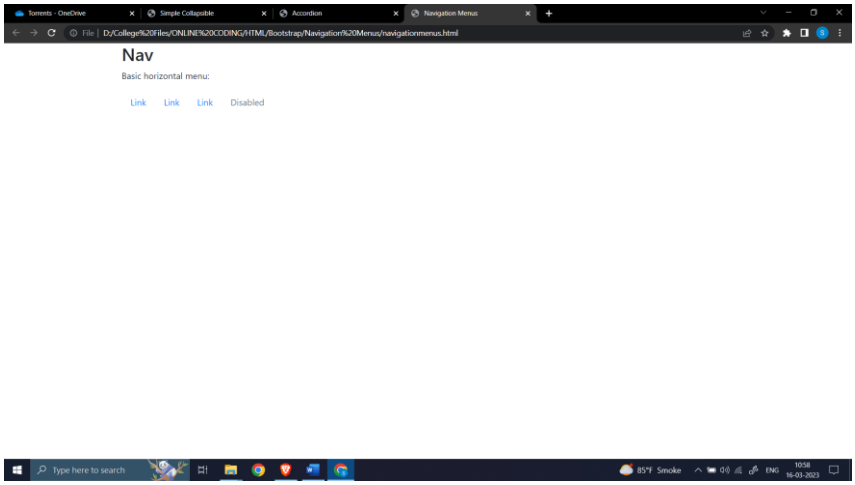


The following example shows a simple accordion by extending the card component. Use the data-parent attribute to make sure that all collapsible elements under the specified parent will be closed when one of the collapsible item is shown.

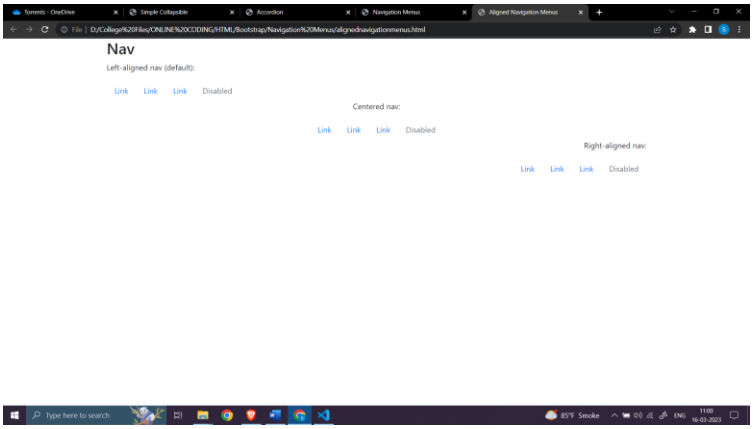


Navigation Menus

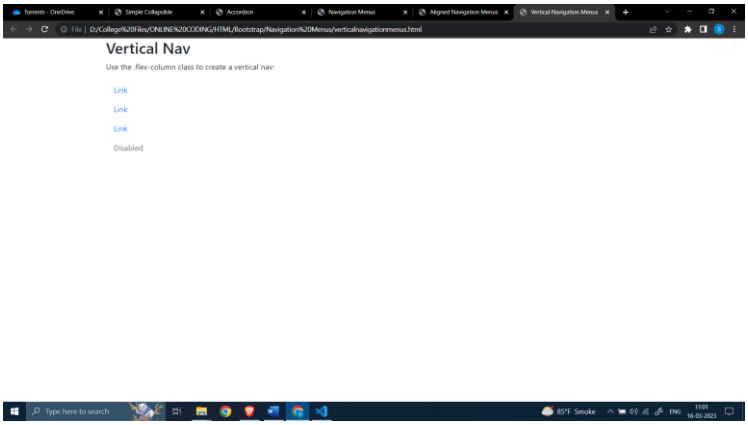
If you want to create a simple horizontal menu, add the **.nav** class to a <ul> element, followed by **.nav-item** for each <li> and add the **.nav-link** class to their links:



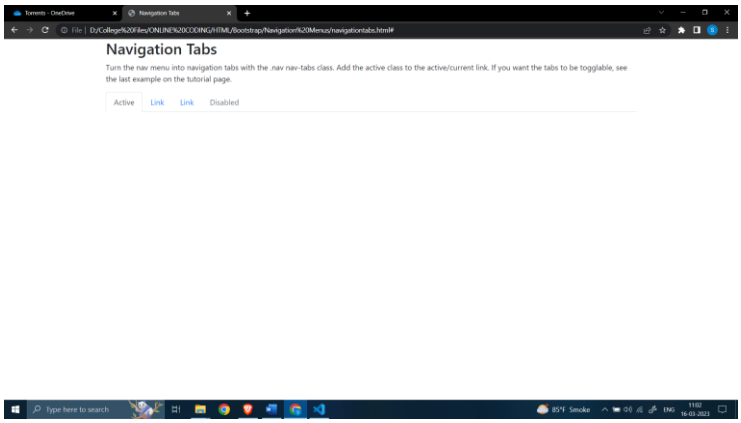
Add the **.justify-content-center** class to center the nav, and the **.justify-content-end** class to center-align/right-align the nav:



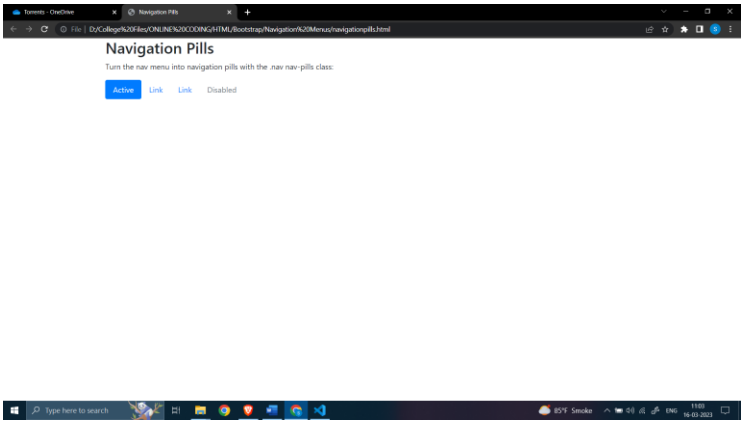
Add the **.flex-column** class to create a vertical nav:



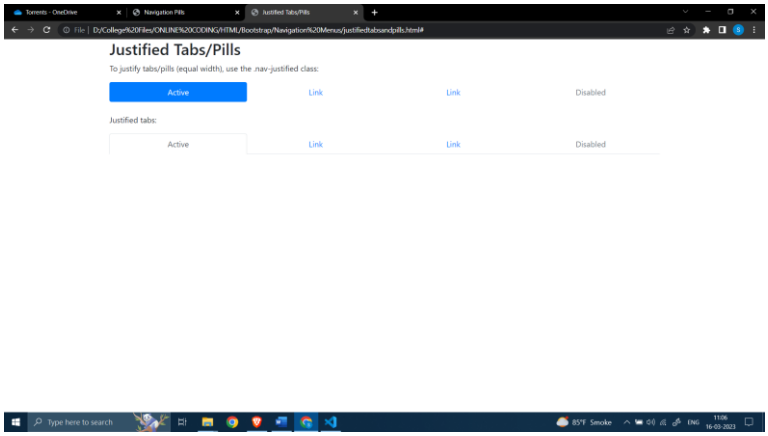
Turn the nav menu into navigation tabs with the .nav-tabs class. Add the **.active** class to the active/current link:



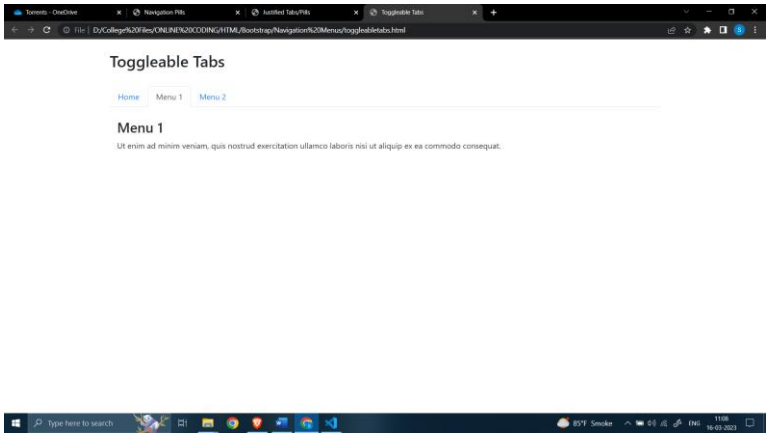
urn the nav menu into navigation pills with the **.nav-pills** class:



Justify the tabs/pills with the **.nav-justified** class (equal width):



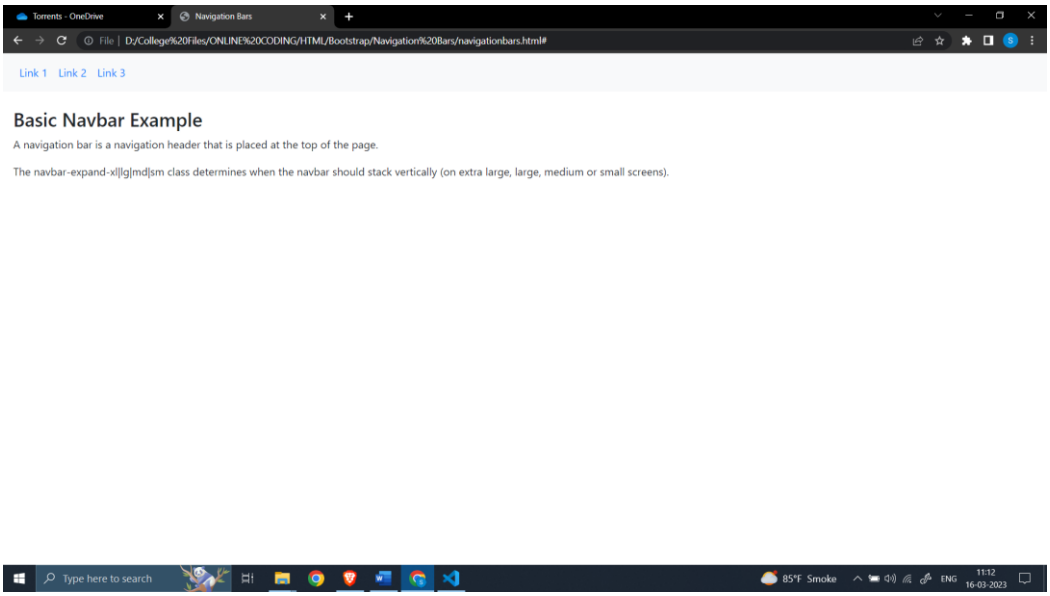
To make the tabs toggleable, add the **data-toggle="tab"** attribute to each link. Then add a **.tab-pane** class with a unique ID for every tab and wrap them inside a element with class **.tab-content**. If you want the tabs to fade in and out when clicking on them, add the **.fade** class to **.tab-pane**:



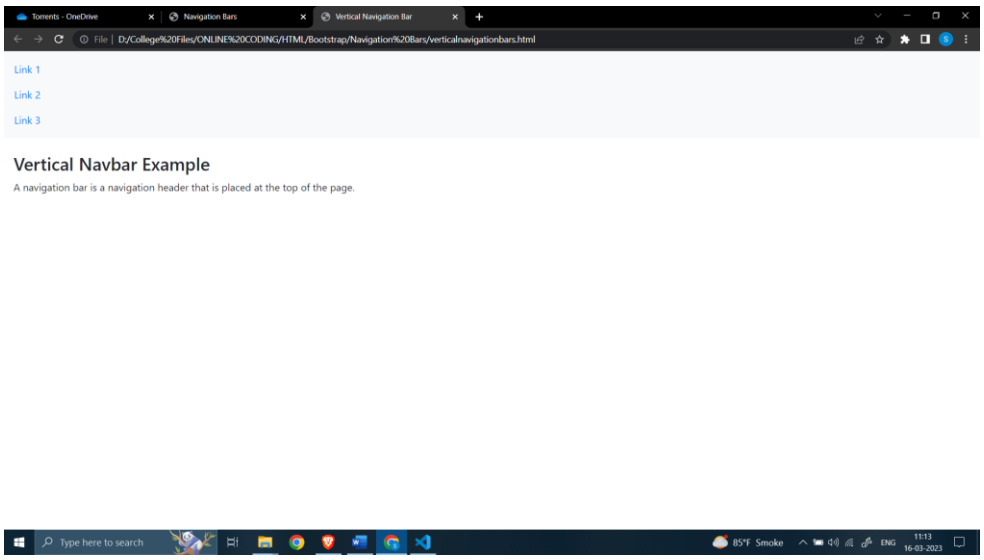
The same code applies to pills; only change the data-toggle attribute to **data-toggle="pill"**.

Navigation Bar

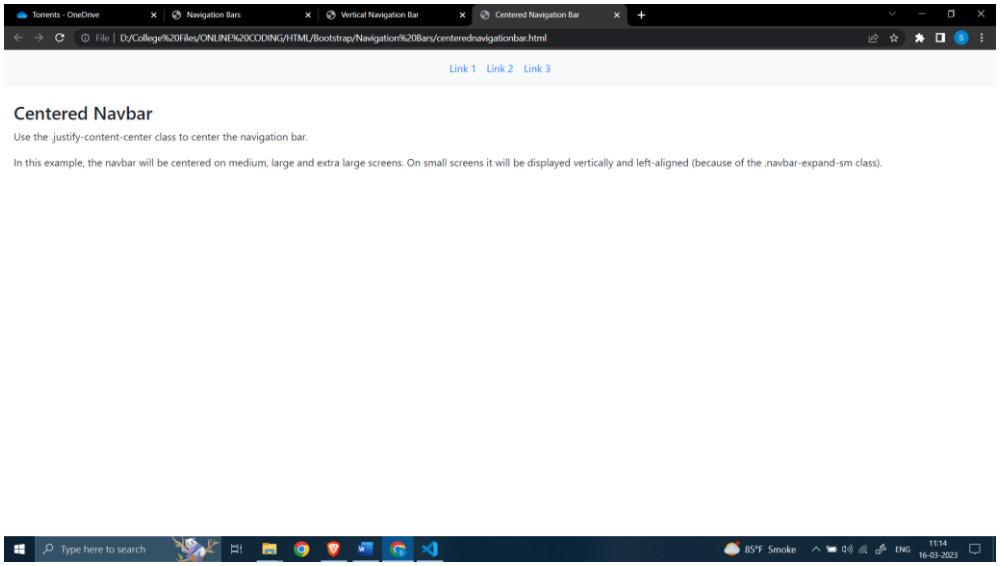
A navigation bar is a navigation header that is placed at the top of the page. With Bootstrap, a navigation bar can extend or collapse, depending on the screen size. A standard navigation bar is created with the **.navbar** class, followed by a responsive collapsing class: **.navbar-expand-xl|lg|md|sm** (stacks the navbar vertically on extra large, large, medium or small screens). To add links inside the navbar, use a <ul> element with **class="navbar-nav"**. Then add <li> elements with a .nav-item class followed by an <a> element with a **.nav-link class**:



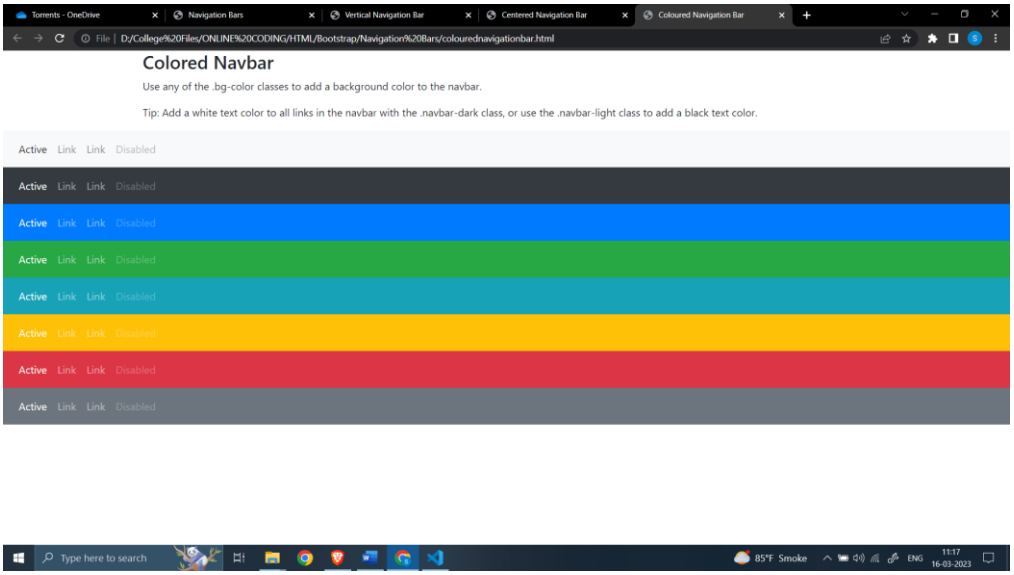
Remove the **.navbar-expand-xl|lg|md|sm** class to create a vertical navigation bar:



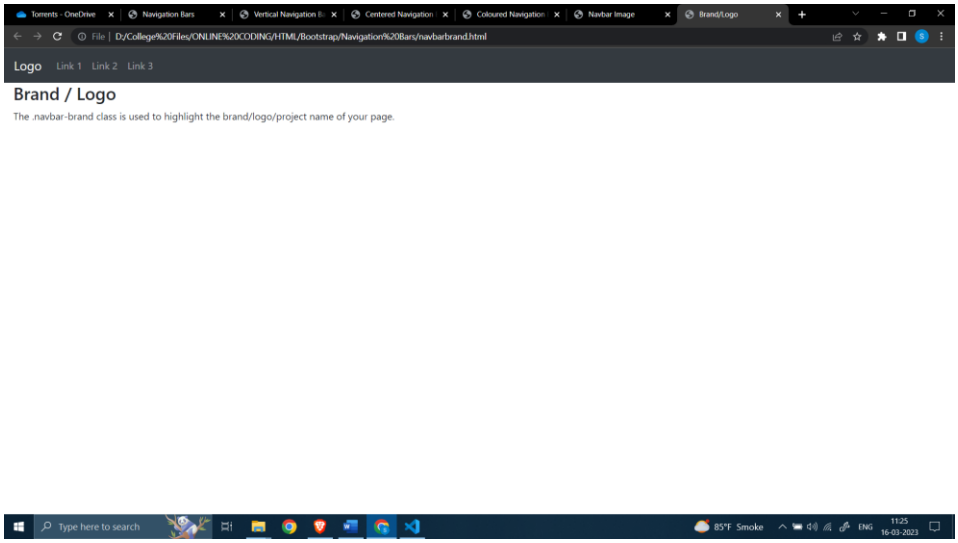
Add the **.justify-content-center** class to center the navigation bar. The following example will center the navigation bar on medium, large and extra large screens. On small screens it will be displayed vertically and left-aligned (because of the **.navbar-expand-sm** class):



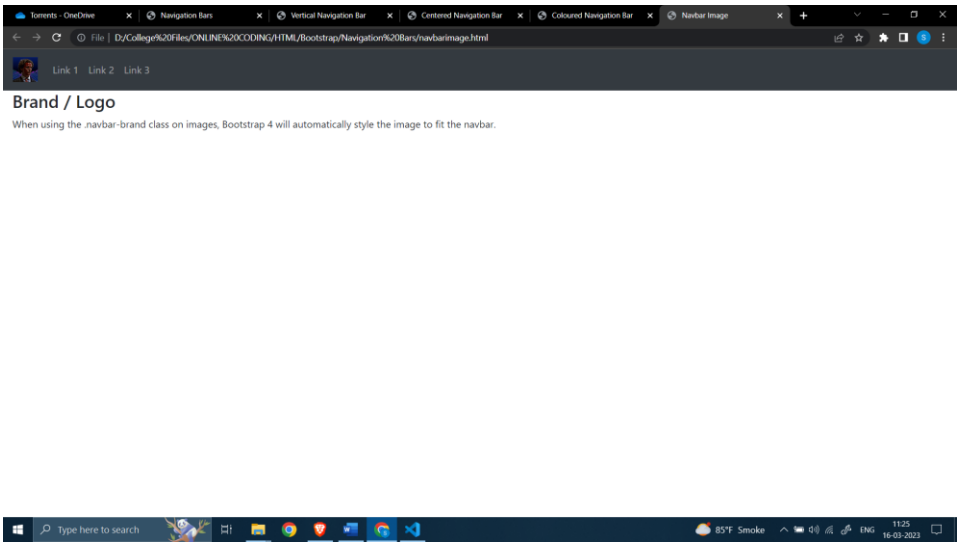
Use any of the **.bg-color** classes to change the background color of the navbar. Add a white text color to all links in the navbar with the **.navbar-dark** class, or use the **.navbar-light** class to add a black text color.



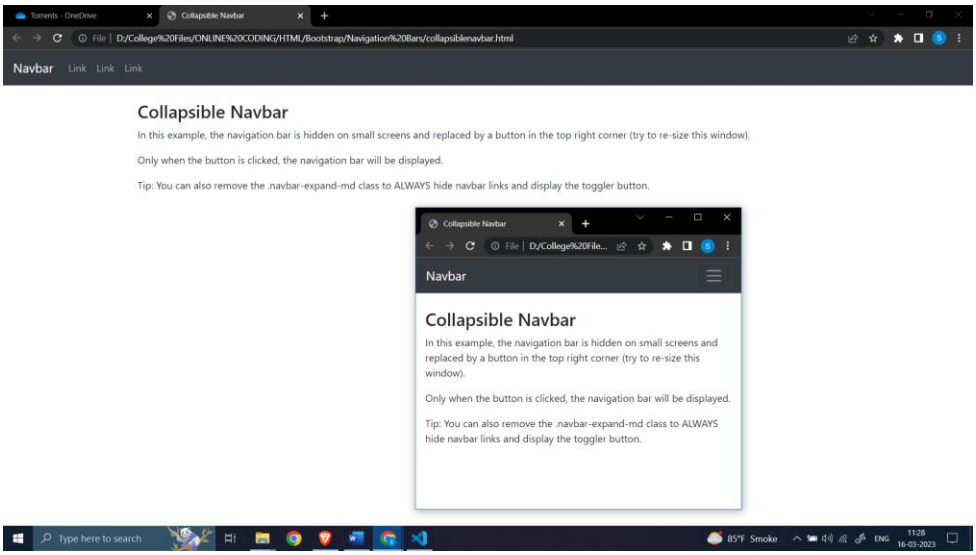
The **.navbar-brand** class is used to highlight the brand/logo/project name of your page.



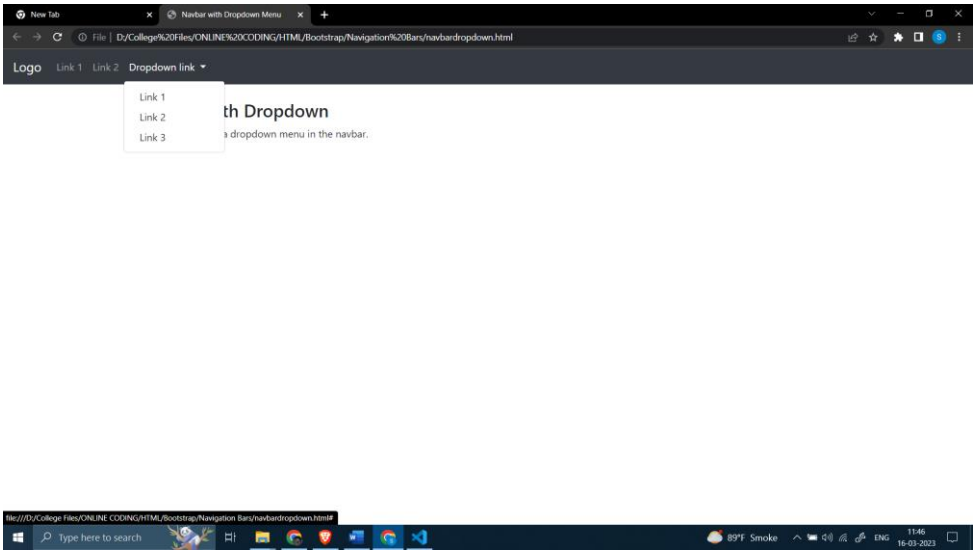
When using the **.navbar-brand** class on images, Bootstrap 4 will automatically style the image to fit the navbar vertically:



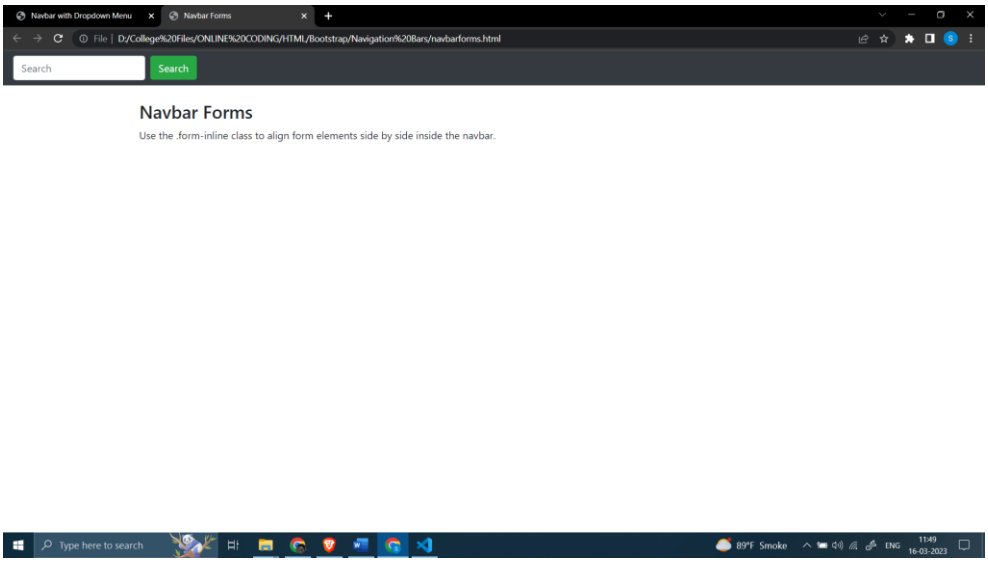
Very often, especially on small screens, you want to hide the navigation links and replace them with a button that should reveal them when clicked on. To create a collapsible navigation bar, use a button with **class="navbar-toggler"**, **data-toggle="collapse"** and **data-target="#thetarget"**. Then wrap the navbar content (links, etc) inside a div element with **class="collapse navbar-collapse"**, followed by an id that matches the data-target of the **button: "thetarget"**.



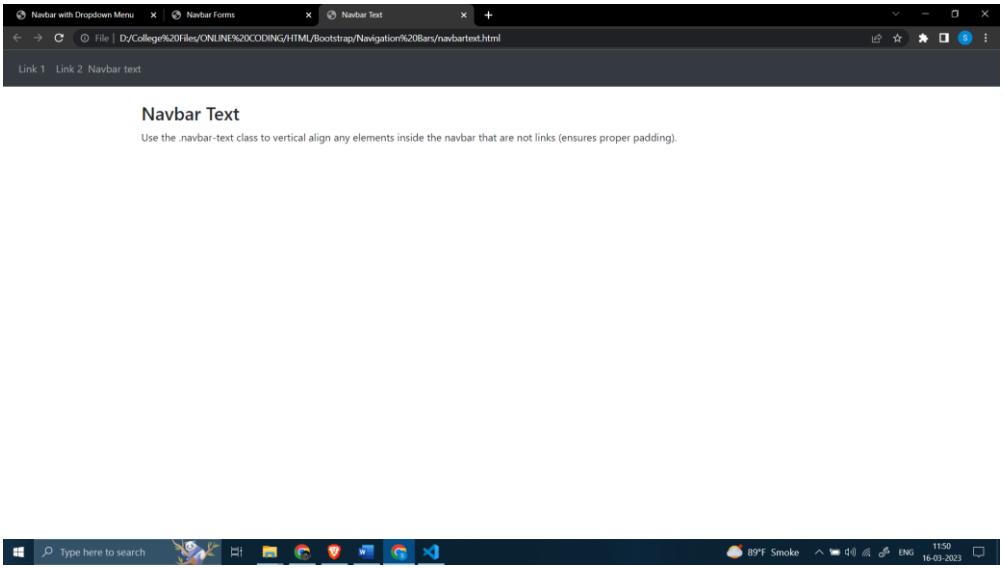
Navbars can also hold dropdown menus:



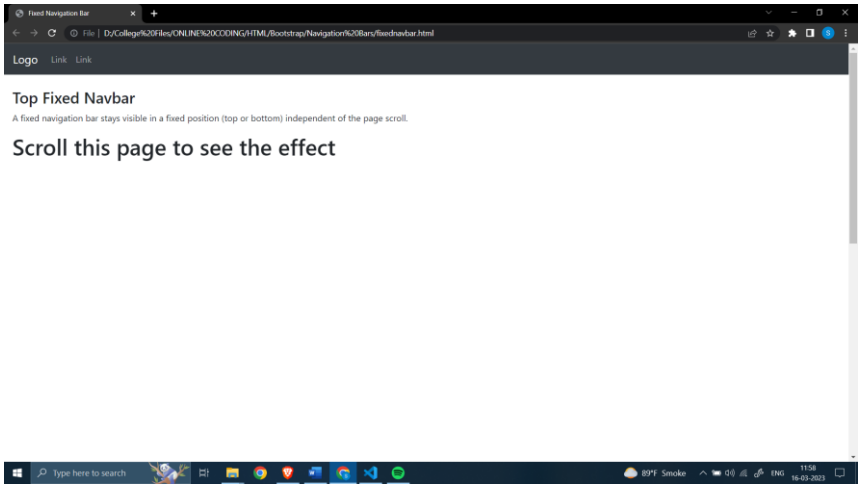
Add a <form> element with class=**"form-inline"** to group inputs and buttons side-by-side. You can also use other input classes, such as **.input-group-prepend** or **.input-group-append** to attach an icon or help text next to the input field.



Use the **.navbar-text** class to vertical align any elements inside the navbar that are not links (ensures proper padding and text color).



The navigation bar can also be fixed at the top or at the bottom of the page. A fixed navigation bar stays visible in a fixed position (top or bottom) independent of the page scroll. The **.fixed-top** class makes the navigation bar fixed at the top. Use the **.sticky-top** class to make the navbar fixed/stay at the top of the page when you scroll past it.

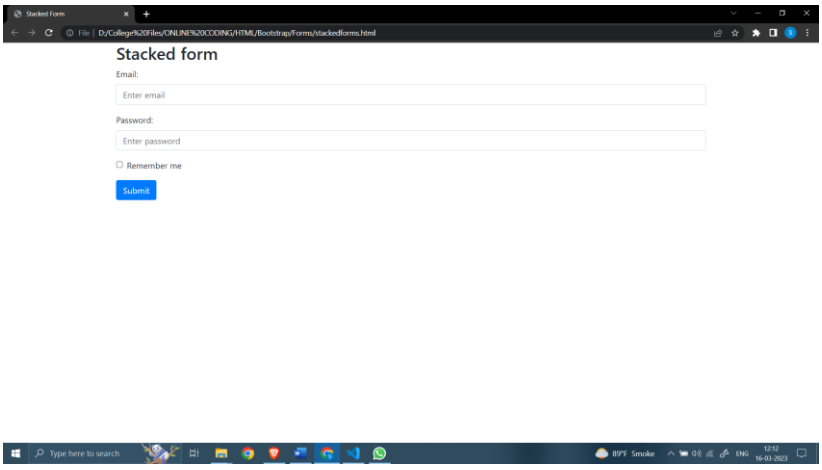


Forms

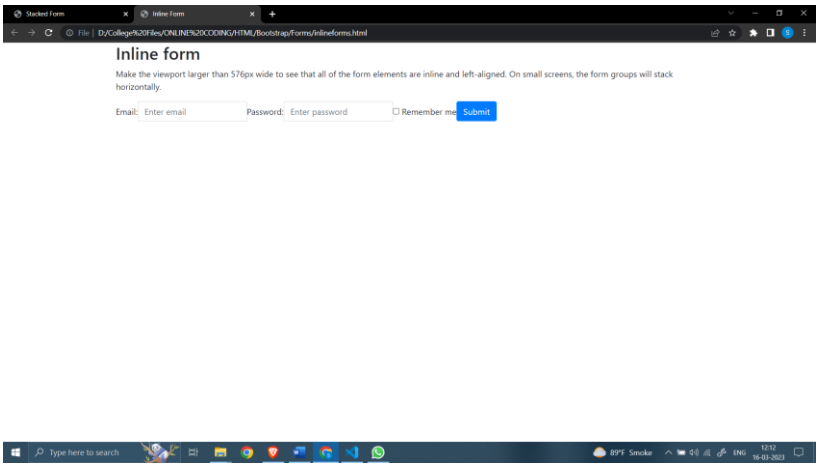
Bootstrap provides two types of form layouts:

- Stacked (full-width) form
- Inline form

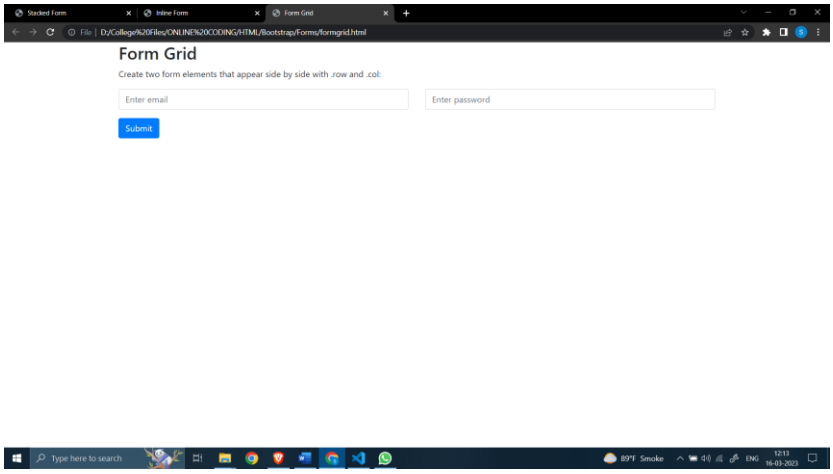
The following example creates a stacked form with two input fields, one checkbox, and a submit button. Add a wrapper element with **.form-group**, around each form control, to ensure proper margins:



The inline form above feels "compressed", and will look much better with Bootstrap's spacing utilities. The following example adds a right margin (.mr-sm-2) to each input on all devices (small and up). And a margin bottom class (.mb-2) is used to style the input field when it breaks (goes from horizontal to vertical due to not enough space/width):



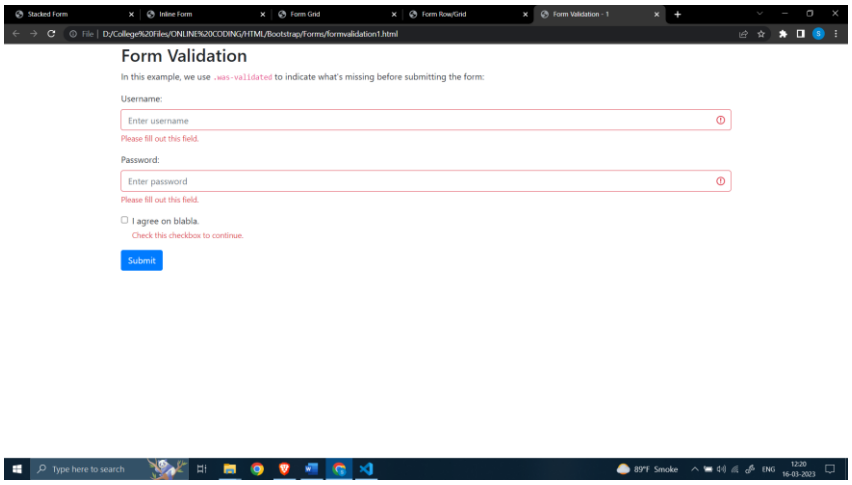
You can also use columns (**.col**) to control the width and alignment of form inputs without using spacing utilities. Just remember to put them inside a **.row** container. In the example below, we use two columns that will appear side by side:



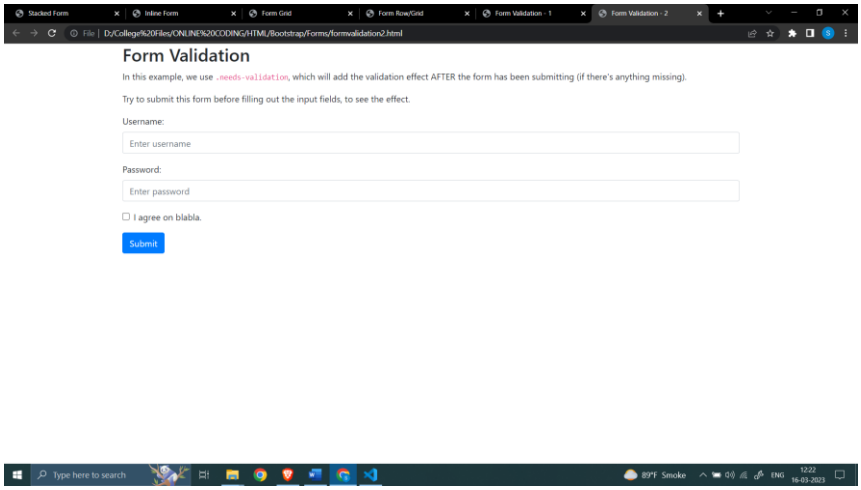
If you want less grid margins (override default column gutters), use **.form-row** instead of **.row**.

You can use different validation classes to provide valuable feedback to users. Add either **.was-validated** or **.needs-validation** to the <form> element, depending on whether you want to provide validation feedback before or after submitting the form. The input fields will have a green (valid) or red (invalid) border to indicate what's missing in the form. You can also add a **.valid-feedback** or **.invalid-feedback** message to tell the user explicitly what's missing, or needs to be done before submitting the form.

In this example, we use **.was-validated** to indicate what's missing before submitting the form:



In this example, we use **.needs-validation**, which will add the validation effect AFTER the form has been submitting (if there's anything missing):

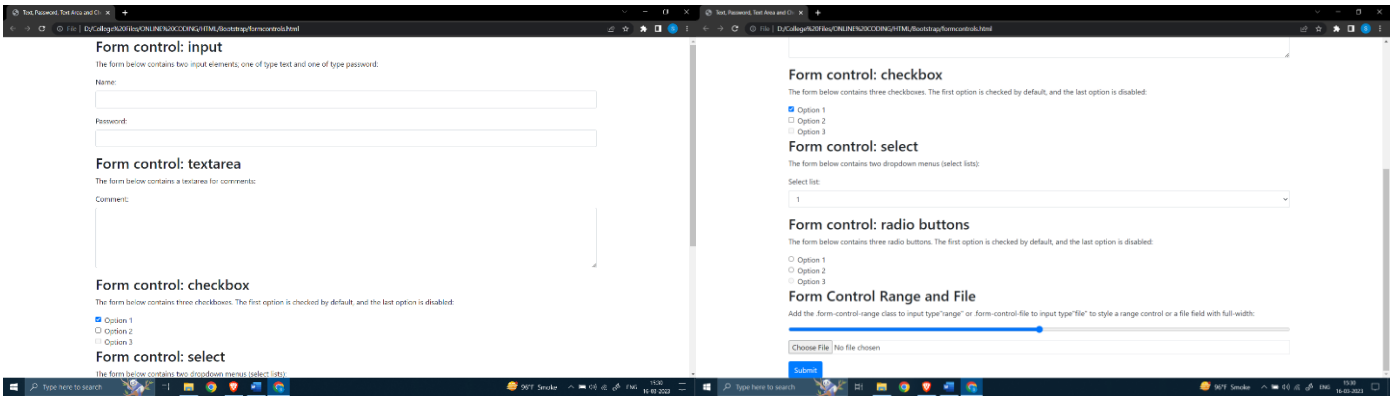


Bootstrap supports the following form controls:

- Input
- Textarea
- Checkbox
- Radio
- Select

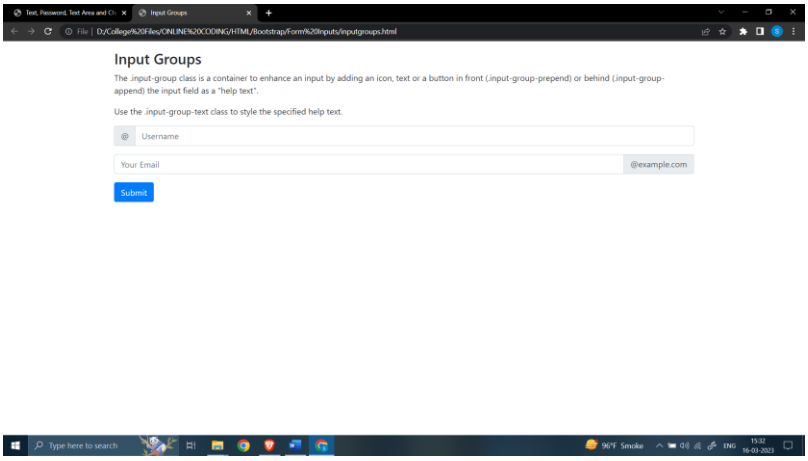
Bootstrap supports all the HTML5 input types: text, password, datetime, datetime-local, date, month, time, week, number, email, url, search, tel, and color.

Use a wrapper element with class="form-check" to ensure proper margins for labels and checkboxes. Add the **.form-check-label** class to label elements, and **.form-check-input** to style checkboxes properly inside the **.form-check container**. Use the **.form-check-inline** class if you want the checkboxes/radio buttons to appear on the same line.

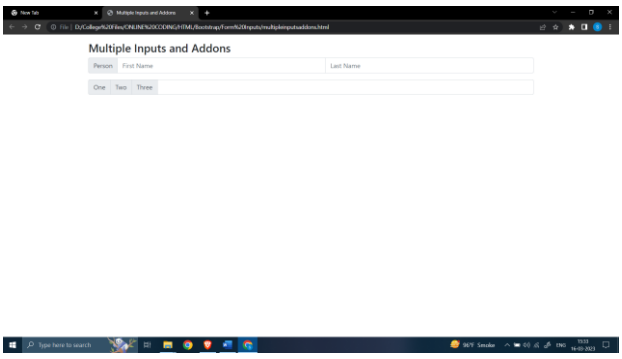


Input Groups

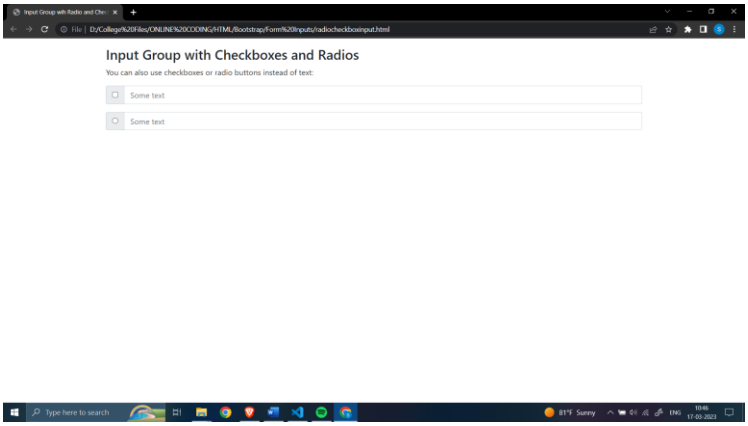
The **.input-group** class is a container to enhance an input by adding an icon, text or a button in front or behind the input field as a "help text". Use **.input-group-prepend** to add the help text in front of the input, and **.input-group-append** to add it behind the input. At last, add the **.input-group-text** class to style the specified help text.



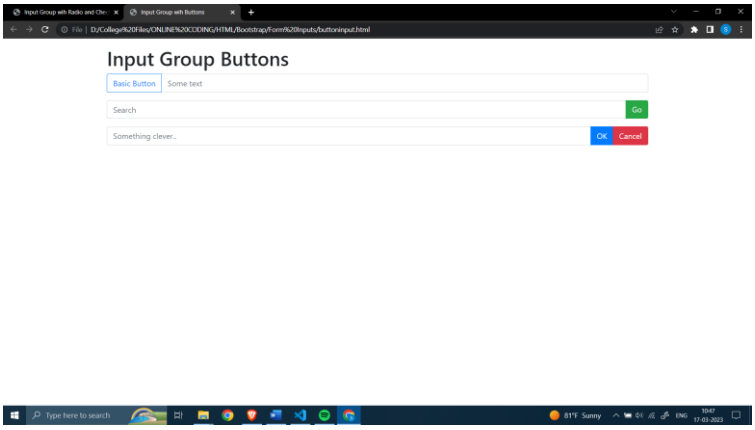
Use the **.input-group-sm** class for small input groups and **.input-group-lg** for large inputs groups. We can also add multiple inputs or addons:



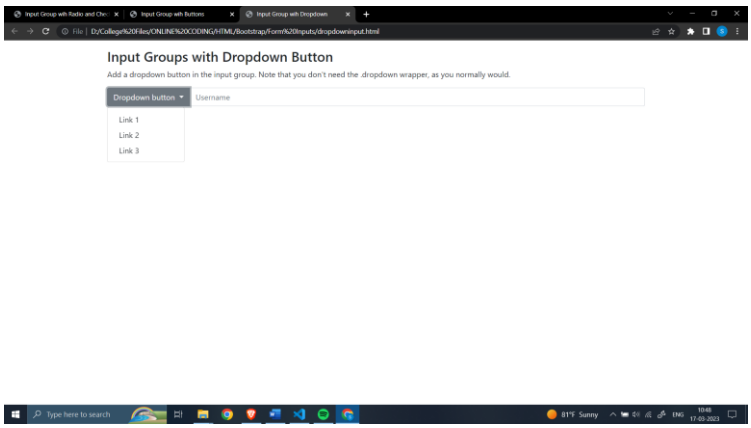
You can also use checkboxes or radio buttons instead of text:



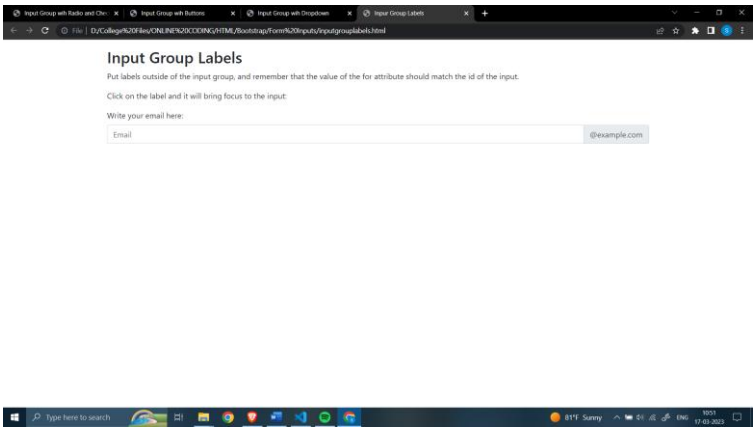
Buttons can also be used:



Similarly, dropdown menu can also be added as an input group:

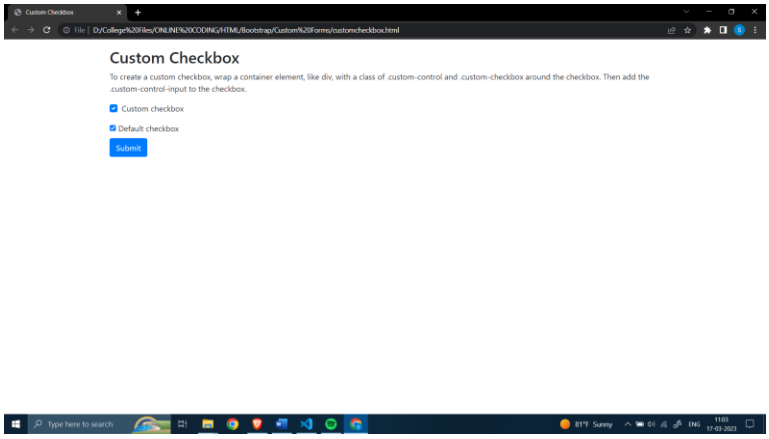


Put labels outside of the input group, and remember that the value of the for attribute should match the id of the input. Click on the label and it will bring focus to the input:

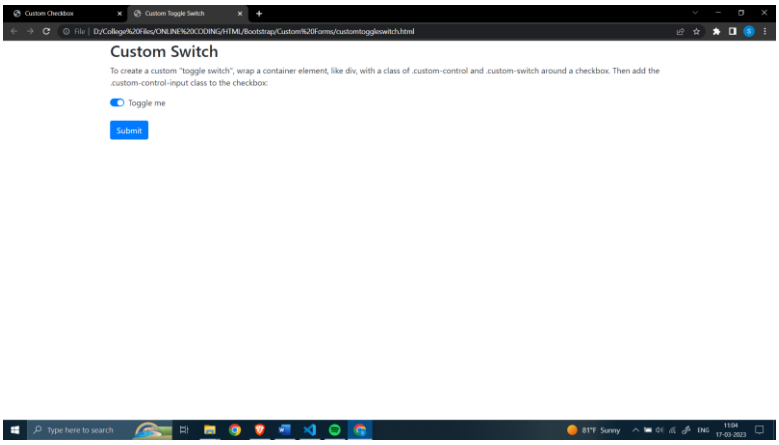


Bootstrap 4 comes with customized form elements, that are meant to replace browser defaults. To create a custom checkbox, wrap a container element, like , with a class of **.custom-control** and **.custom-checkbox** around the checkbox. Then add the **.custom-control-input** to the input with type="checkbox".

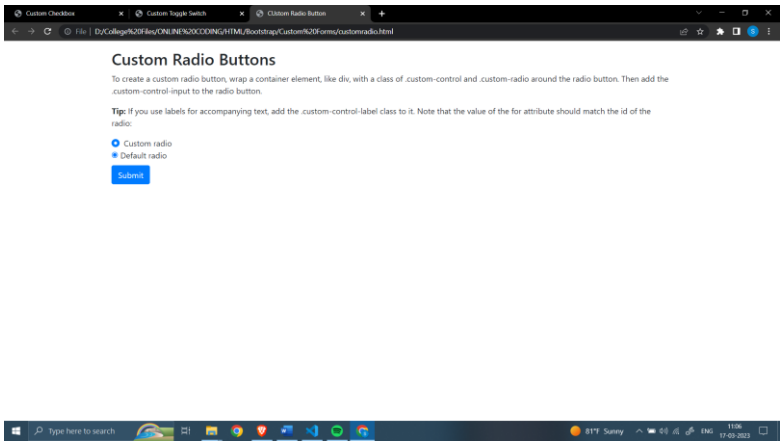
If you use labels for accompanying text, add the **.custom-control-label** class to it. Note that the value of the for attribute should match the id of the checkbox:



To create a custom "toggle switch", wrap a container element, like , with a class of **.custom-control** and **.custom-switch** around a checkbox. Then add the **.custom-control-input** class to the checkbox:

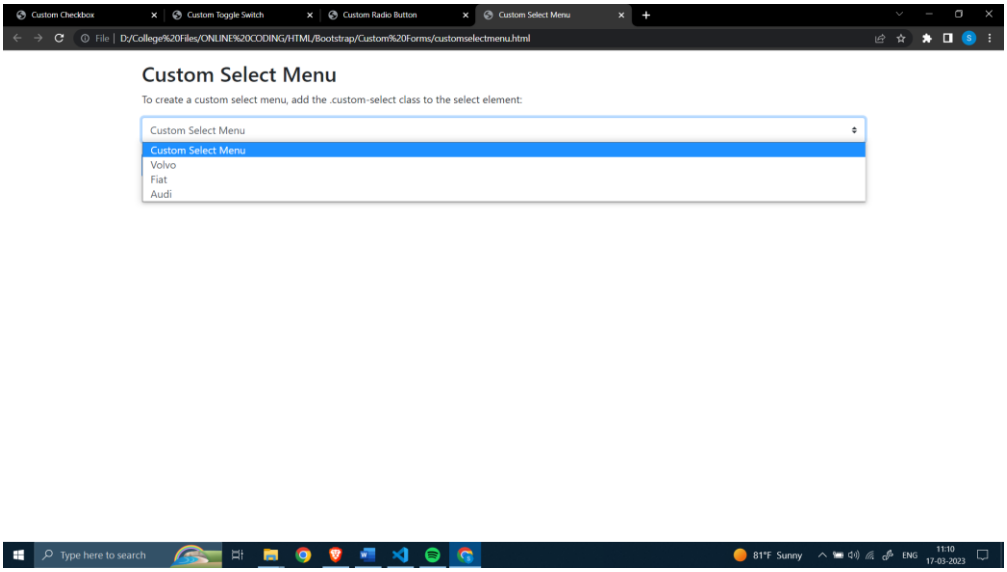


To create a custom radio button, wrap a container element, like , with a class of **.custom-control** and **.custom-radio** around the radio button. Then add the **.custom-control-input** to the input with type="radio":

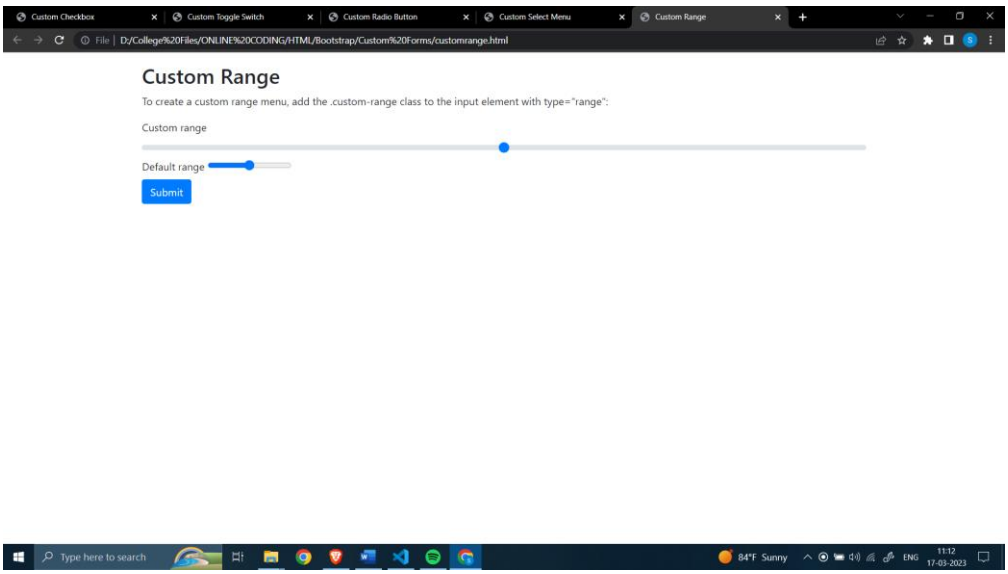


If you want the custom form controls to sit side by side (inline), add the **.custom-control-inline** to the wrapper/container.

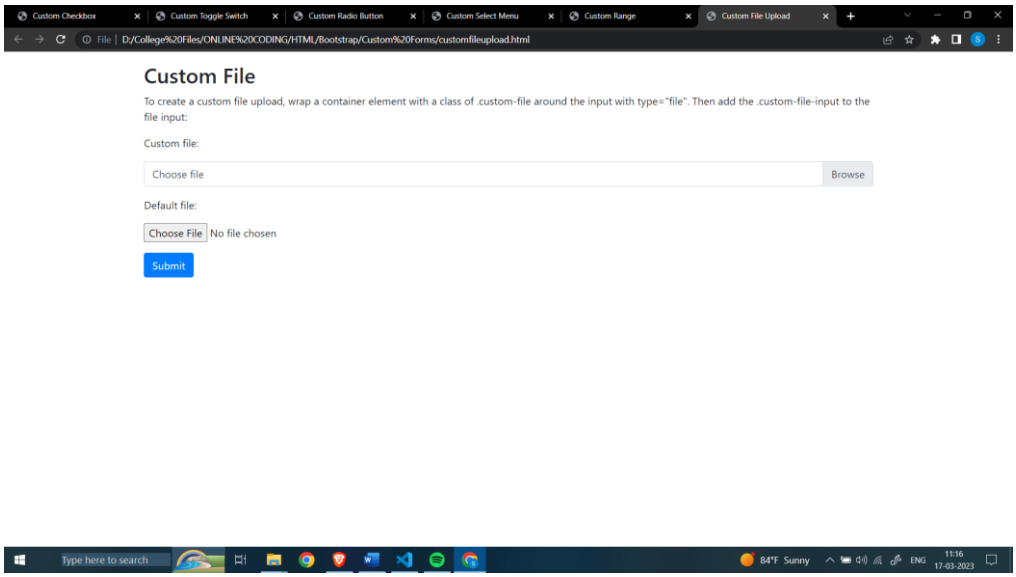
To create a custom select menu, add the **.custom-select** class to the <select> element. Use the **.custom-select-sm** class to create a small select menu and the **.custom-select-lg** class for a large one:



To create a custom range menu, add the **.custom-range** class to an input with type="<range>":

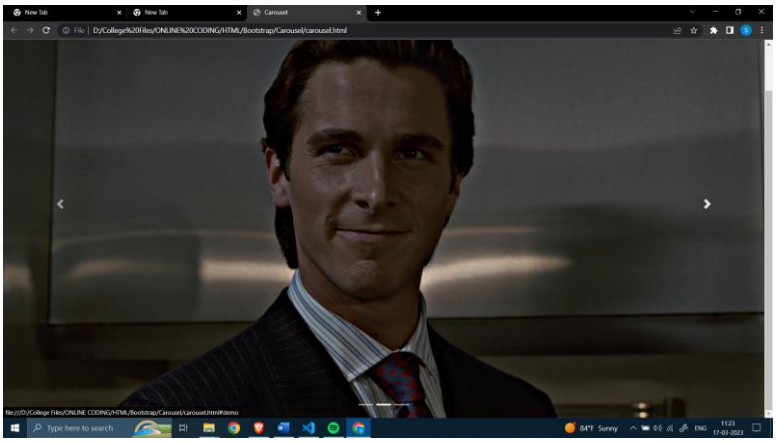


To create a custom file upload, wrap a container element with a class of **.custom-file** around the input with type="file". Then add the **.custom-file-input** to it. Note that you also have to include some jQuery code if you want the name of the file to appear when you select a specific file:



Carousel

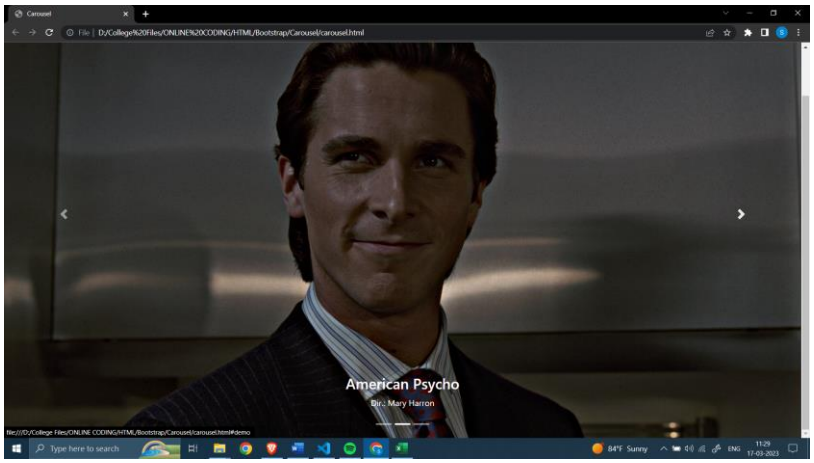
The following example shows how to create a basic carousel with indicators and controls.



A description of what each class from the example above do:

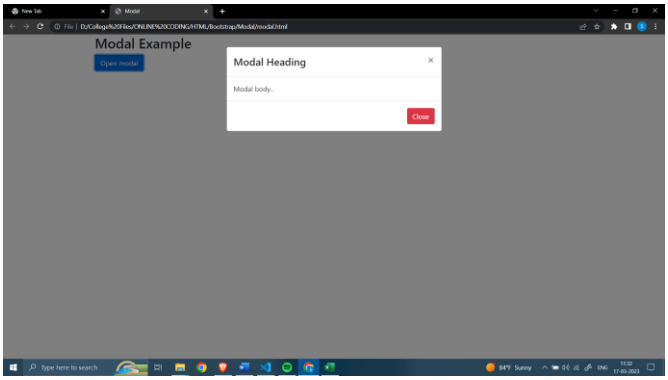
Class	Description
.carousel	creates a carousel
.carousel-indicators	adds indicators for the carousel. These are the little dots at the bottom of each slide (which indicates how many slides there are in the carousel, and which slide the user are currently viewing)
.carousel-inner	adds slides to the carousel
.carousel-item	specifies the content of each slide
.carousel-control-prev	adds a left (previous) button to the carousel, which allows the user to go back between the slides
.carousel-control-next	adds a right (next) button to the carousel, which allows the user to go forward between the slides
.carousel-control-prev-icon	used together with **.carousel-control-prev** to create a "previous" button
.carousel-control-next-icon	used together with **.carousel-control-next** to create a "next" button
.slide	adds a CSS transition and animation effect when sliding from one item to the next. Remove this class if you do not want this effect

Add elements inside ** ** within each ** ** to create a caption for each slide:



Modal

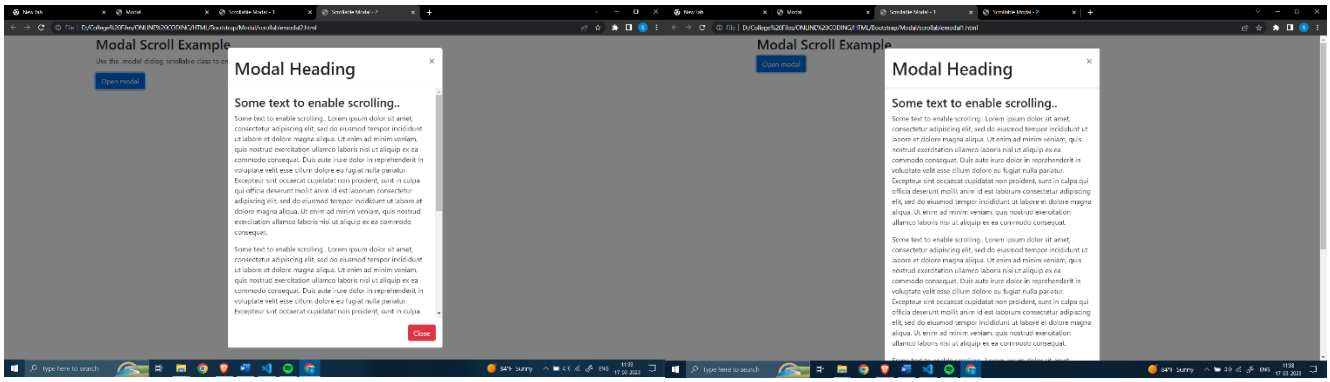
The Modal component is a dialog box/popup window that is displayed on top of the current page. The following example shows how to create a basic modal. Use the **.fade** class to add a fading effect when opening and closing the modal:



Change the size of the modal by adding the **.modal-sm** class for small modals, **.modal-lg** class for large modals, or **.modal-xl** for extra large modals. Add the size class to the element with class **.modal-dialog**.

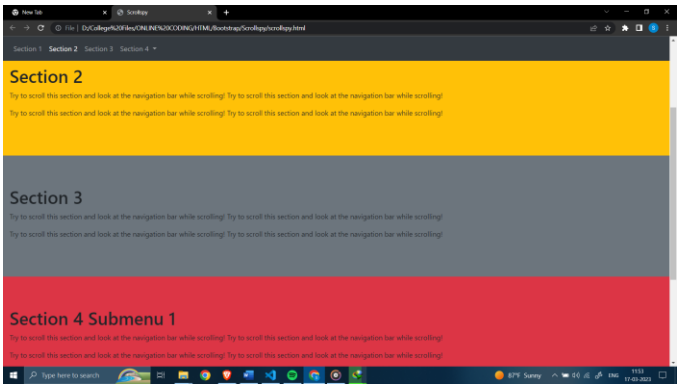
Center the modal vertically and horizontally within the page, with the **.modal-dialog-centered** class.

When you have a lot of content inside the modal, a scrollbar is added to the page. We can either have the entire dialog box displayed with the page scrolling option or we can also scroll inside the modal, instead of the page itself, by adding **.modal-dialog-scrollable** to **.modal-dialog**.

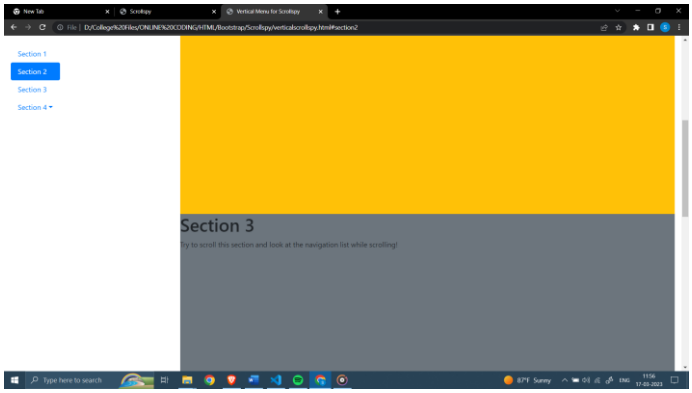


Scrollspy

Scrollspy is used to automatically update links in a navigation list based on scroll position. Add data-spy="scroll" to the element that should be used as the scrollable area (often this is the <body> element). Then add the data-target attribute with a value of the id or the class name of the navigation bar (.navbar). This is to make sure that the navbar is connected with the scrollable area. Note that scrollable elements must match the ID of the links inside the navbar's list items (matches <a href="#section1">). The optional data-offset attribute specifies the number of pixels to offset from top when calculating the position of scroll. This is useful when you feel that the links inside the navbar changes the active state too soon or too early when jumping to the scrollable elements. Default is 10 pixels.



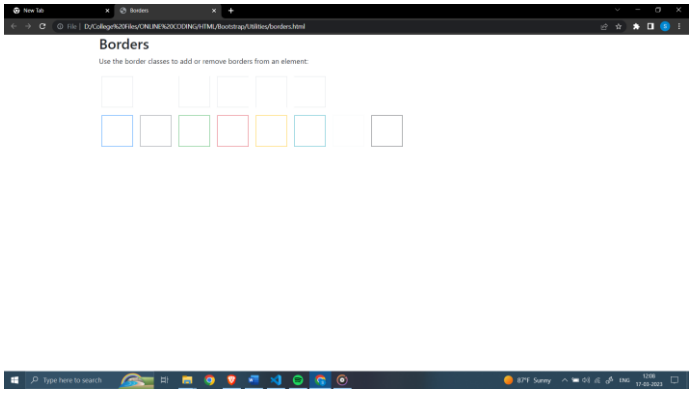
We can also use Navigation Pills to create a vertical menu for our scrollspy as below:



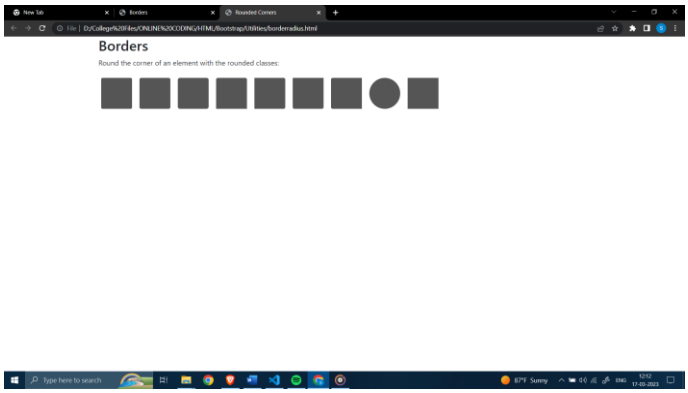
Utilities

Bootstrap 4 has a lot of utility/helper classes to quickly style elements without using any CSS code.

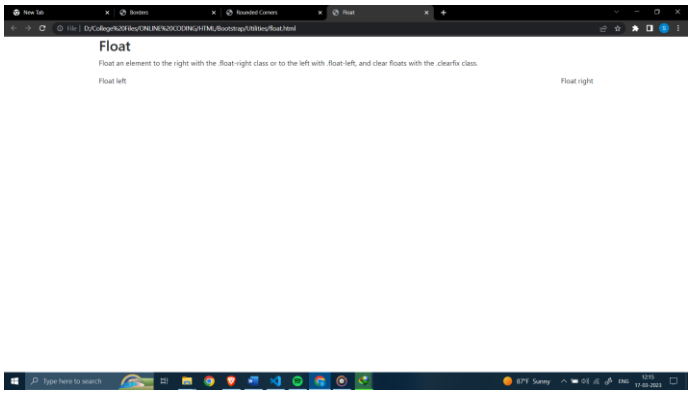
Use the border classes to add or remove borders from an element. We can also add a color to the border with any of the contextual border color classes.



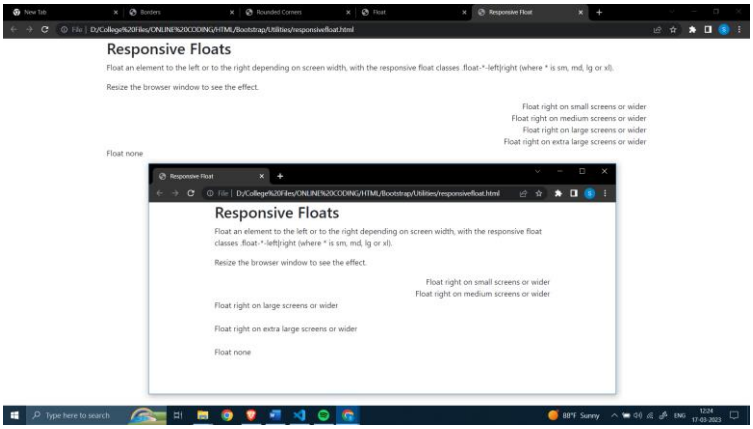
Rounded corners are also possible with the rounded classes:



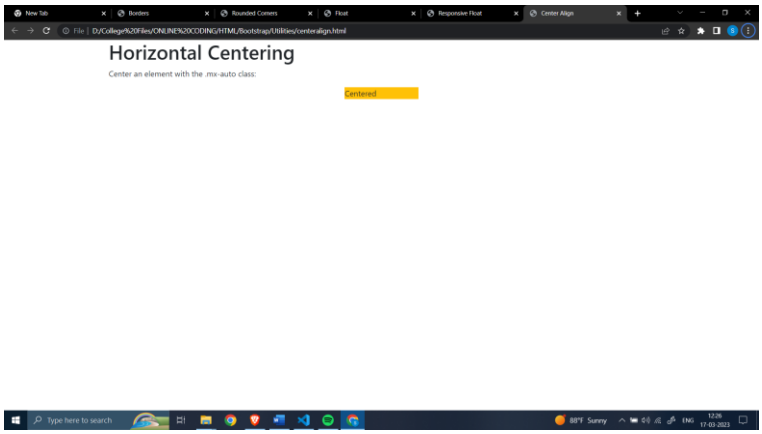
Float an element to the right with the **.float-right** class or to the left with **.float-left**, and clear floats with the **.clearfix** class:



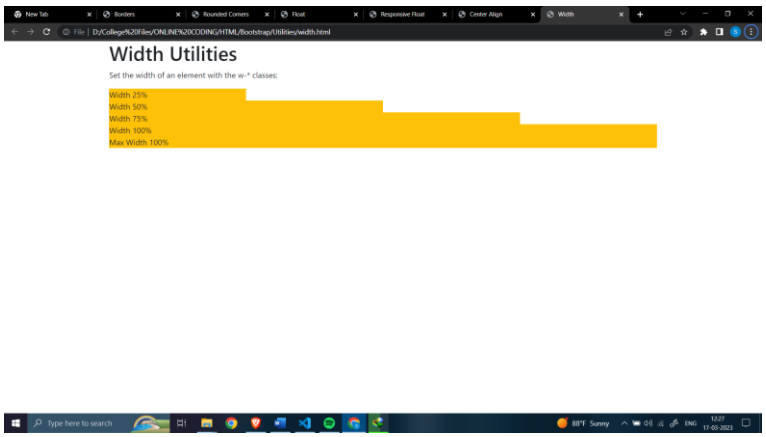
Float an element to the left or to the right depending on screen width, with the responsive float classes (.float-*-left|right - where * is sm (>=576px), md (>=768px), lg (>=992px) or xl (>=1200px)):



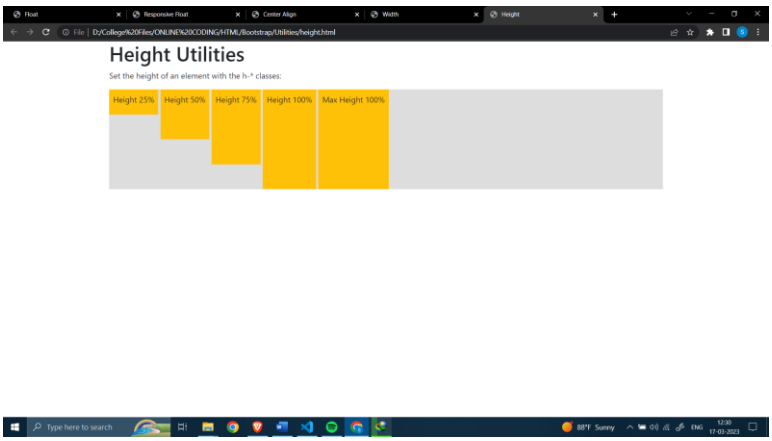
Center an element with the **.mx-auto** class (adds margin-left and margin-right: auto):



Set the width of an element with the w-* classes (.w-25, .w-50, .w-75, .w-100, .mw-100):



Set the height of an element with the h-* classes (.h-25, .h-50, .h-75, .h-100, .mh-100):



Bootstrap 4 has a wide range of responsive margin and padding utility classes. They work for all breakpoints: xs (<=576px), sm (>=576px), md (>=768px), lg (>=992px) or xl (>=1200px)). The classes are used in the format:

{property}{sides}-{size} for xs and {property}{sides}-{breakpoint}-{size} for sm, md, lg, and xl.

Where property is one of:

- m - sets margin
- p - sets padding

Where sides is one of:

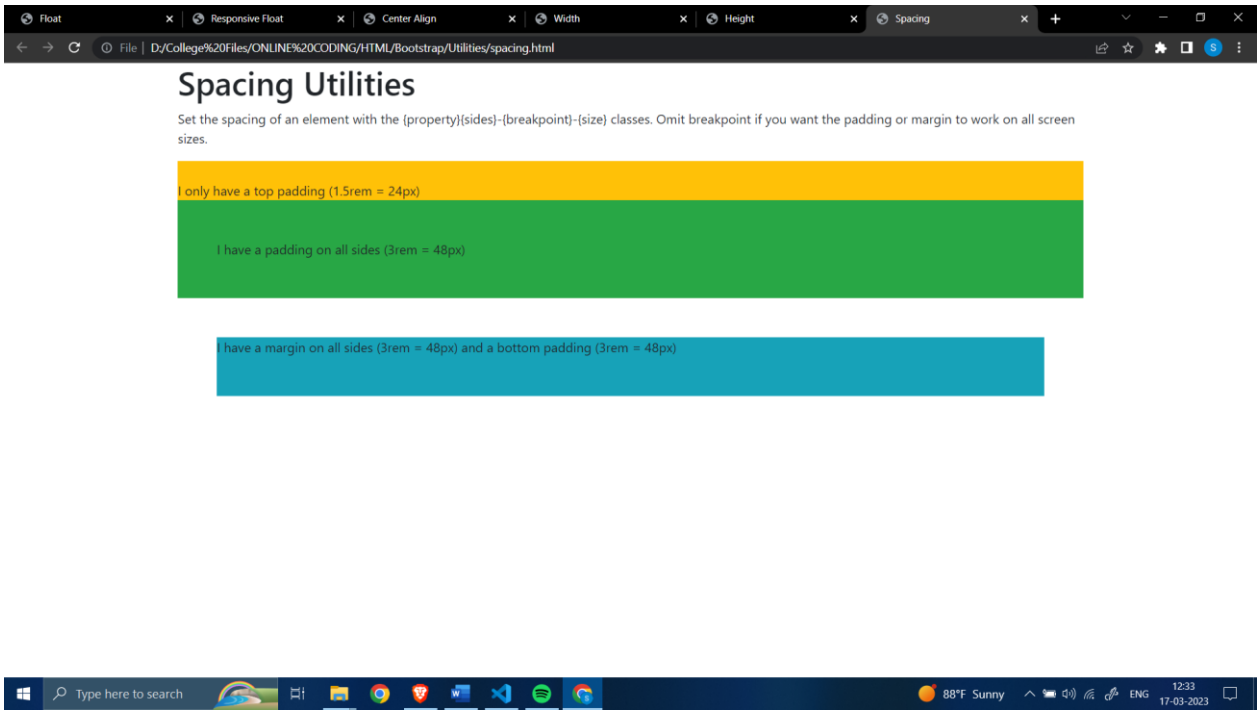
- t - sets margin-top or padding-top
- b - sets margin-bottom or padding-bottom
- l - sets margin-left or padding-left
- r - sets margin-right or padding-right
- x - sets both padding-left and padding-right or margin-left and margin-right
- y - sets both padding-top and padding-bottom or margin-top and margin-bottom
- blank - sets a margin or padding on all 4 sides of the element

Where size is one of:

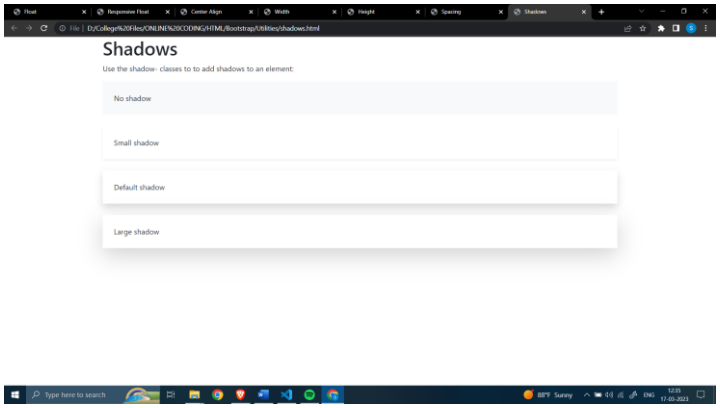
- 0 - sets margin or padding to 0
- 1 - sets margin or padding to .25rem (4px if font-size is 16px)
- 2 - sets margin or padding to .5rem (8px if font-size is 16px)
- 3 - sets margin or padding to 1rem (16px if font-size is 16px)
- 4 - sets margin or padding to 1.5rem (24px if font-size is 16px)
- 5 - sets margin or padding to 3rem (48px if font-size is 16px)
- auto - sets margin to auto

Margins can also be negative, by adding an "n" in front of size:

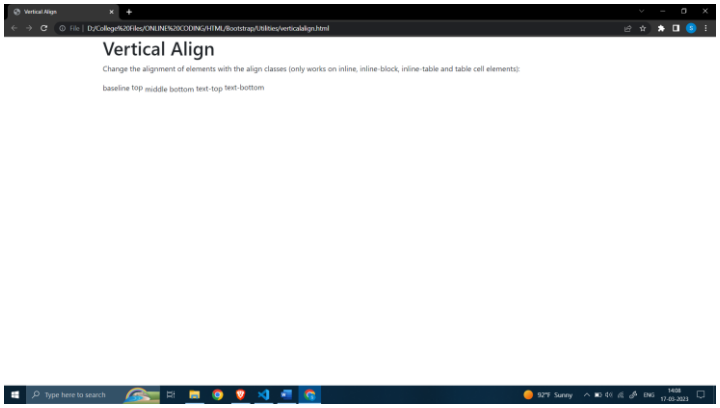
- n1 - sets margin to -.25rem (-4px if font-size is 16px)
- n2 - sets margin to -.5rem (-8px if font-size is 16px)
- n3 - sets margin to -1rem (-16px if font-size is 16px)
- n4 - sets margin to -1.5rem (-24px if font-size is 16px)
- n5 - sets margin to -3rem (-48px if font-size is 16px)



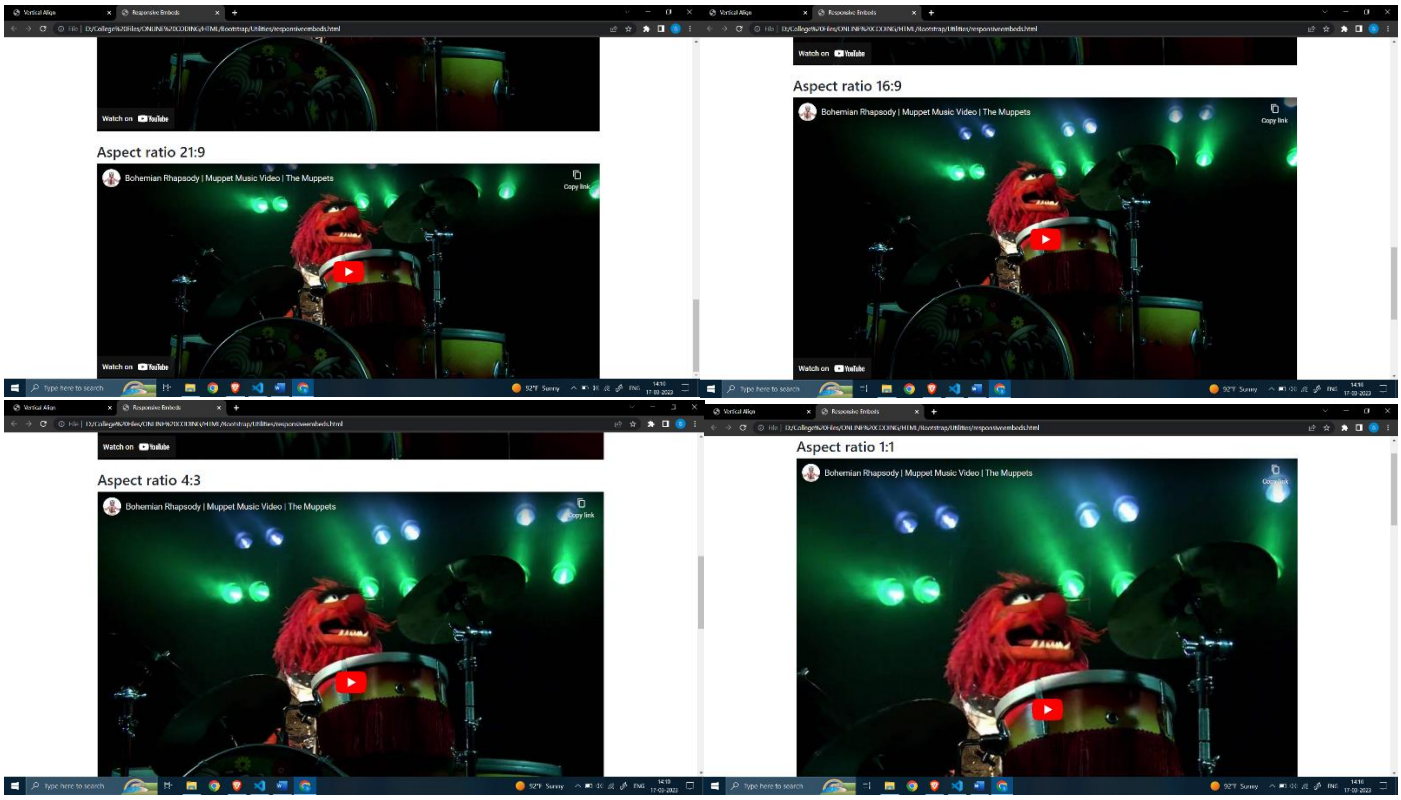
Use the shadow- classes to add shadows to an element:



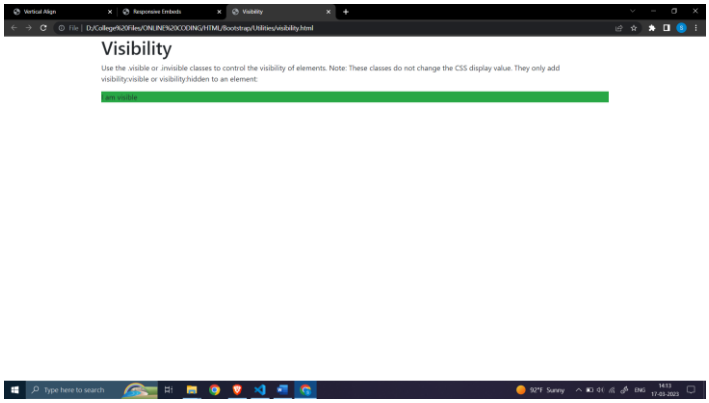
Use the align- classes to change the alignment of elements (only works on inline, inline-block, inline-table and table cell elements):



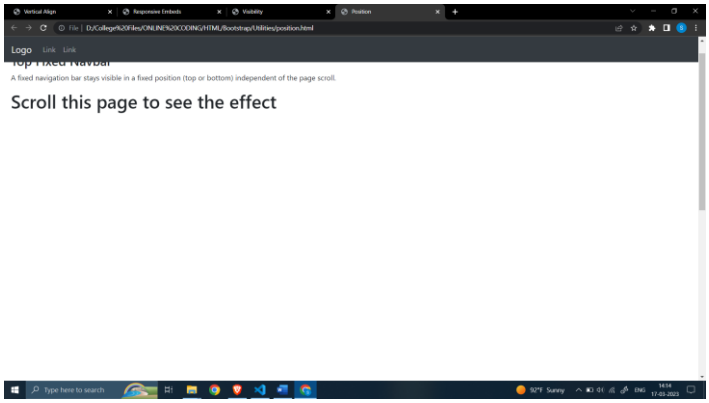
We can also create responsive video or slideshow embeds based on the width of the parent. Add the **.embed-responsive-item** to any embed elements (like <iframe> or <video>) in a parent element with **.embed-responsive** and an aspect ratio of your choice:



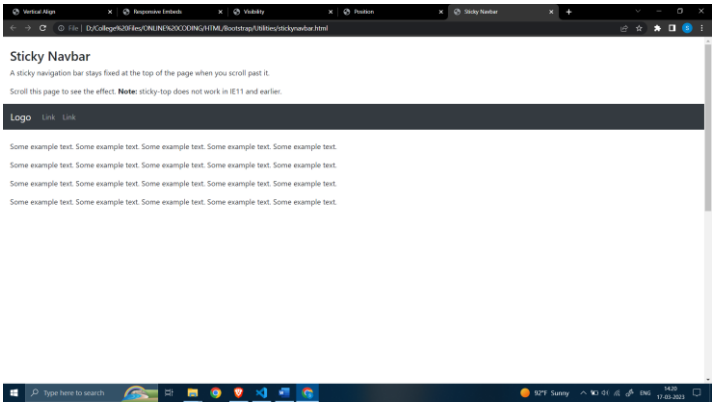
Use the **.visible** or **.invisible** classes to control the visibility of elements. Note: These classes do not change the CSS display value. They only add **visibility:visible** or **visibility:hidden**:



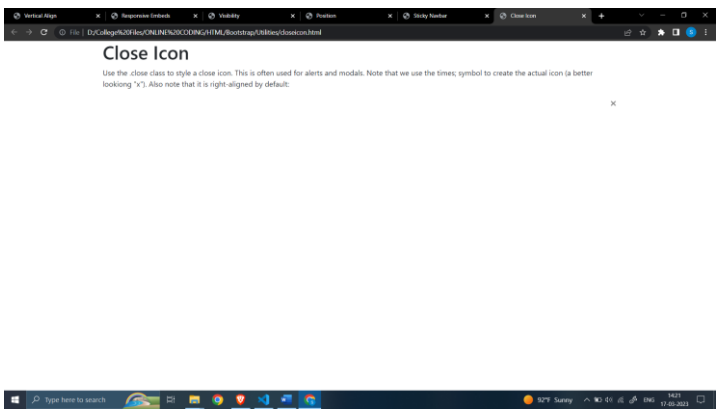
Use the **.fixed-top** class to make any element fixed/stay at the top of the page and use the **.fixed-bottom** class to make any element fixed/stay at the bottom of the page:



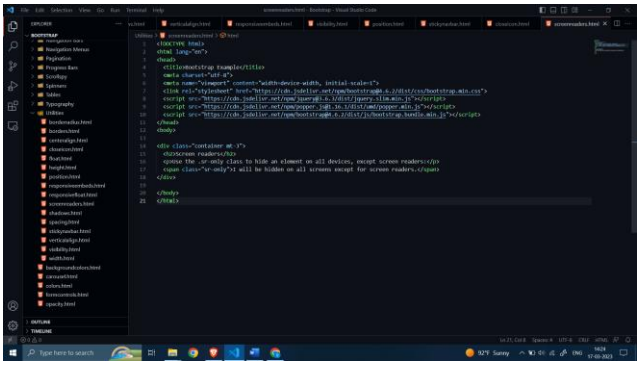
Use the **.sticky-top class** to make any element fixed/stay at the top of the page when you scroll past it:



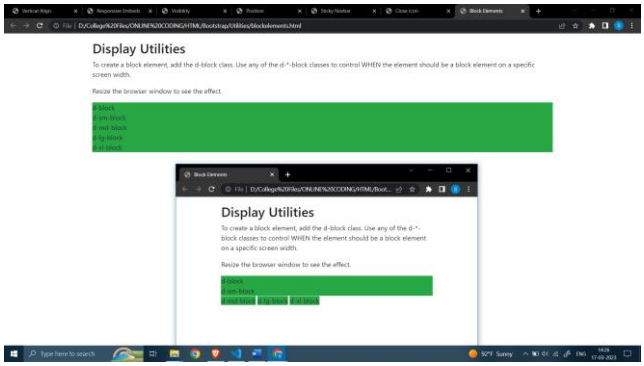
Use the **.close** class to style a close icon. This is often used for alerts and modals. Note that we use the × symbol to create the actual icon (a better looking "x"). Also note that it floats right by default:



Use the **.sr-only** class to hide an element on all devices, except screen readers:



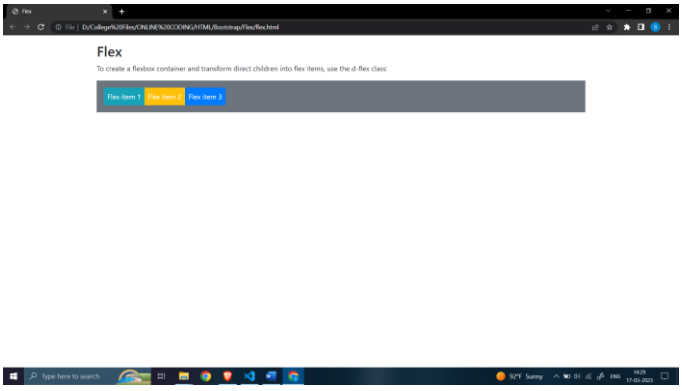
To make an element into a block element, add the **.d-block** class. Use any of the d-*-block classes to control WHEN the element should be a block element on a specific screen width:



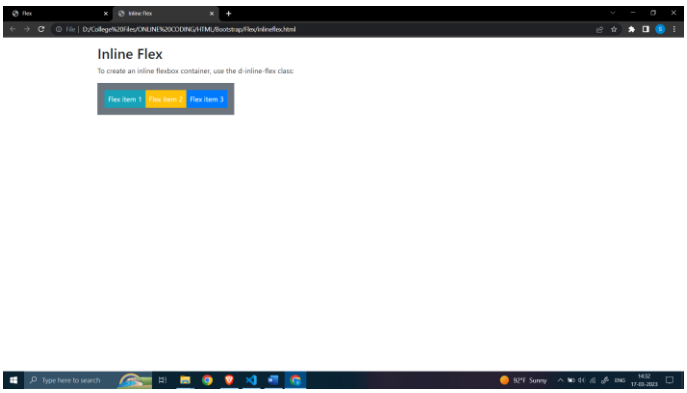
Flex

Use flex classes to control the layout of Bootstrap 4 components. The biggest difference between Bootstrap 3 and Bootstrap 4 is that Bootstrap 4 now uses flexbox, instead of floats, to handle the layout. The Flexible Box Layout Module, makes it easier to design flexible responsive layout structure without using float or positioning.

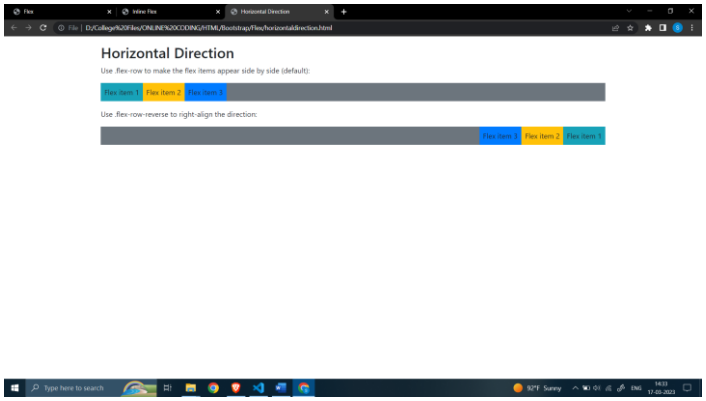
To create a flexbox container and to transform direct children into flex items, use the **d-flex** class:



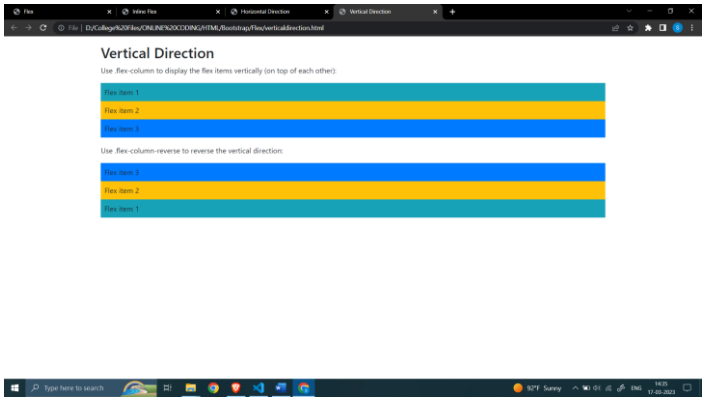
To create an inline flexbox container, use the **d-inline-flex** class:



Use **.flex-row** to display the flex items horizontally (side by side). This is default. Use **.flex-row-reverse** to right-align the horizontal direction:

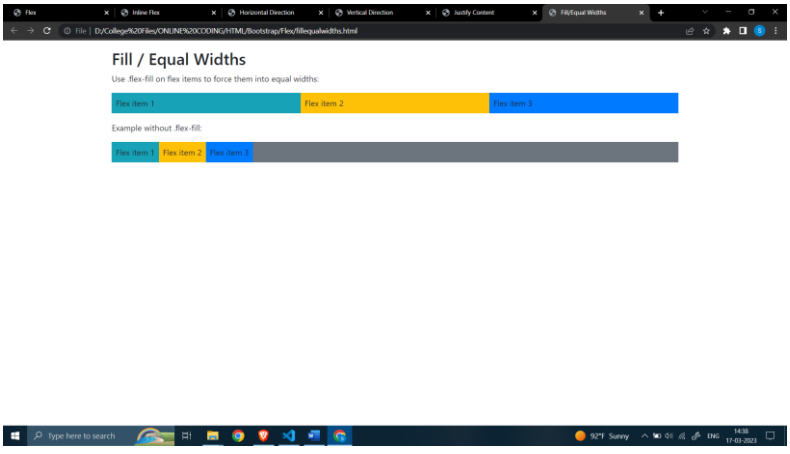


Use **.flex-column** to display the flex items vertically (on top of each other). This is default. Use **.flex-column-reverse** to reverse the vertical direction:

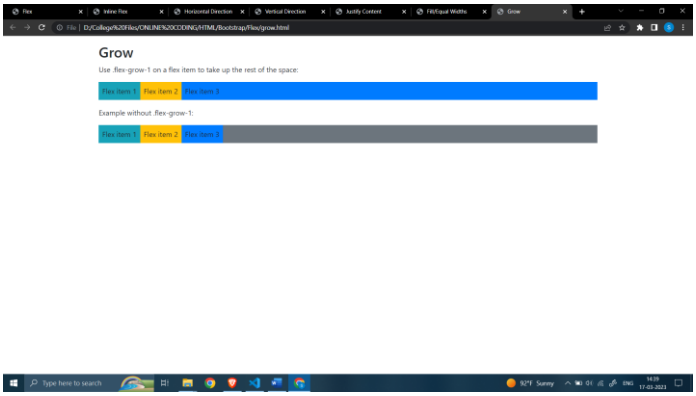


Use the **.justify-content-*** classes to change the alignment of flex items. Valid classes are start (default), end, center, between or around:

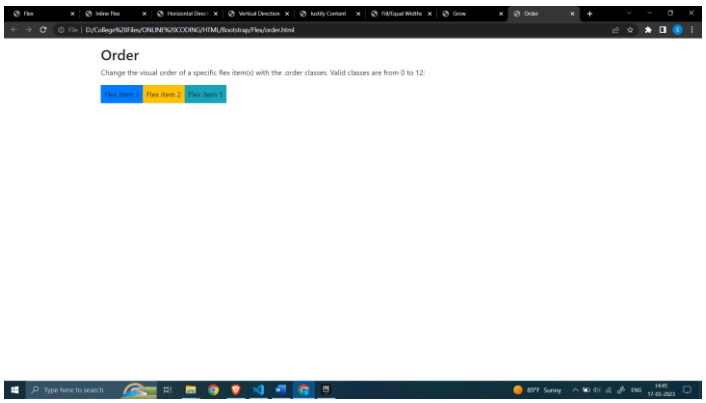
Use **.flex-fill** on flex items to force them into equal widths:



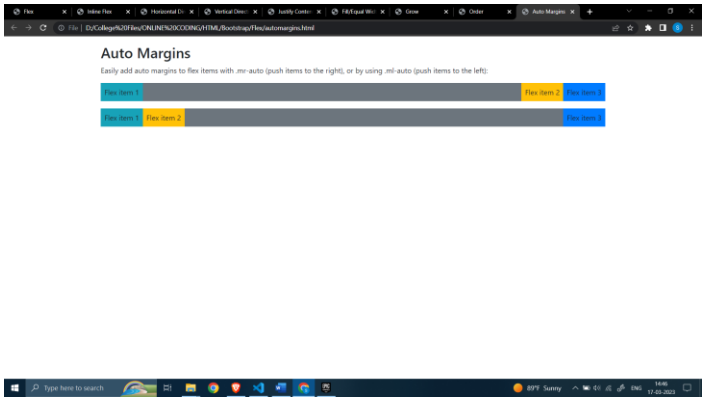
Use **.flex-grow-1** on a flex item to take up the rest of the space. In the example below, the first two flex items take up their necessary space, while the last item takes up the rest of the available space. Use **.flex-shrink-1** on a flex item to make it shrink if necessary:



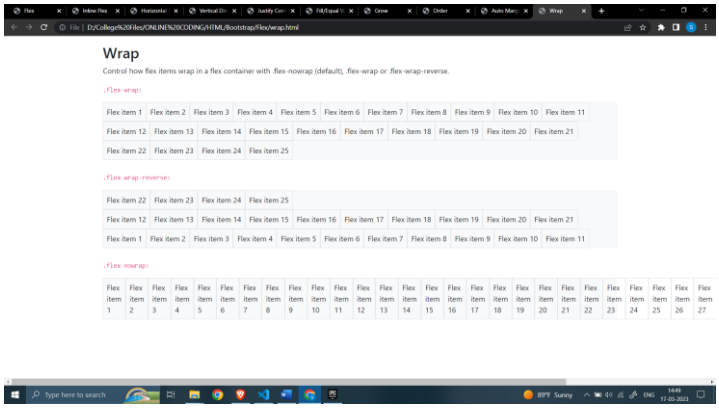
Change the visual order of a specific flex item(s) with the **.order** classes. Valid classes are from 0 to 12, where the lowest number has highest priority (order-1 is shown before order-2, etc..):



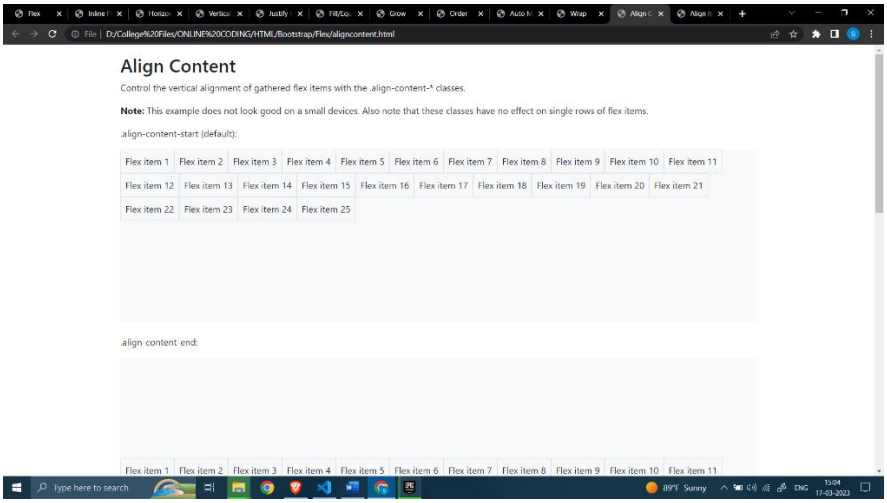
Easily add auto margins to flex items with **.mr-auto** (push items to the right), or by using **.ml-auto** (push items to the left):



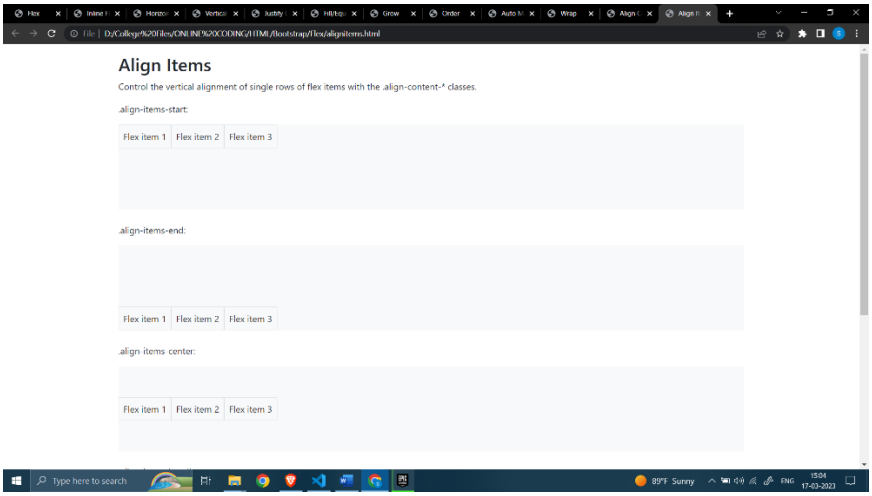
Control how flex items wrap in a flex container with **.flex-nowrap** (default), **.flex-wrap** or **.flex-wrap-reverse**:



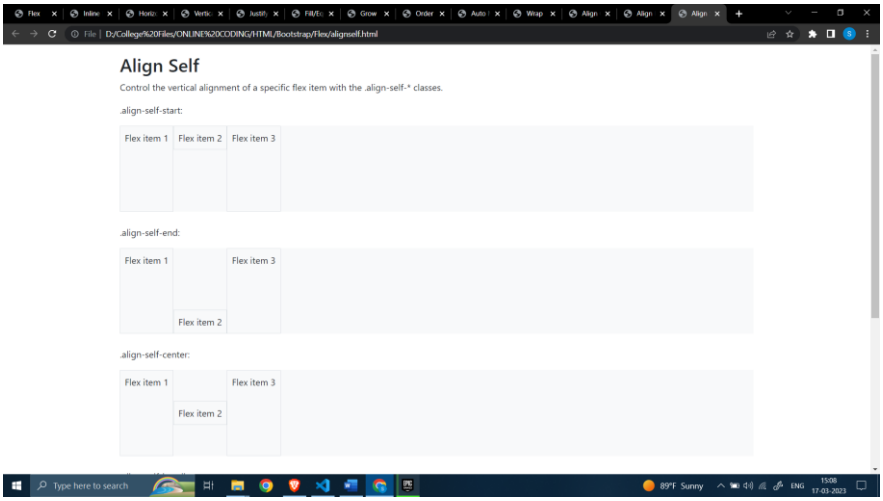
Control the vertical alignment of gathered flex items with the **.align-content-*** classes. Valid classes are **.align-content-start** (default), **.align-content-end**, **.align-content-center**, **.align-content-between**, **.align-content-around** and **.align-content-stretch**. These classes have no effect on single rows of flex items.



Control the vertical alignment of single rows of flex items with the **.align-items-*** classes. Valid classes are **.align-items-start**, **.align-items-end**, **.align-items-center**, **.align-items-baseline**, and **.align-items-stretch** (default).



Control the vertical alignment of a specified flex item with the **.align-self-*** classes. Valid classes are **.align-self-start**, **.align-self-end**, **.align-self-center**, **.align-self-baseline**, and **.align-self-stretch** (default).



All flex classes comes with additional responsive classes, which makes it easy to set a specific flex class on a specific screen size.

The * symbol can be replaced with sm, md, lg or xl, which represents small, medium, large or xlarge screens.

Class	Description
Flex Container	
.d-*-flex	Creates a flexbox container for different screens
.d-*-inline-flex	Creates an inline flexbox container for different screens
Direction	
.flex-*-row	Display flex items horizontally on different screens
.flex-*-row-reverse	Display flex items horizontally, and right-aligned, on different screens
.flex-*-column	Display flex items vertically on different screens
.flex-*-column-reverse	Display flex items vertically, with reversed order, on different screens screens
Justified Content	
.justify-content-*-start	Display flex items from the start (left-aligned) on different screens
.justify-content-*-end	Display flex items at the end (right-aligned) on different screens
.justify-content-*-center	Display flex items in the center of a flex container on different screens
.justify-content-*-between	Display flex items in "between" on different screens
.justify-content-*-around	Display flex items "around" on different screens
Fill / Equal Width	
.flex-*-fill	Force flex items into equal widths on different screens
Grow	
.flex-*-grow-0	Don't make the items grow on different screens
.flex-*-grow-1	Make items grow on different screens
Shrink	
.flex-*-shrink-0	Don't make the items shrink on diferent screens
.flex-*-shrink-1	Make items shrink on different screens
Order	
.order-*-0-12	Change the order from 0 to 12 on small screens
Wrap	
.flex-*-nowrap	Don't wrap items on different screens
.flex-*-wrap	Wrap items on different screens
.flex-*-wrap-reverse	Reverse the wrapping of items on different screens
Align Content	
.align-content-*-start	Align gathered items from the start on different screens
.align-content-*-end	Align gathered items at the end on different screens
.align-content-*-center	Align gathered items in the center on different screens
.align-content-*-around	Align gathered items "around" on different screens
.align-content-*-stretch	Stretch gathered items on different screens
Align Items	
.align-items-*-start	Align single rows of items from the start on different screens
.align-items-*-end	Align single rows of items at the end on different screens
.align-items-*-center	Align single rows of items in the center on different screens
.align-items-*-baseline	Align single rows of items on the baseline on different screens
.align-items-*-stretch	Stretch single rows of items on different screens
Align Self	
.align-self-*-start	Align a flex item from the start on different screens
.align-self-*-end	Align a flex item at the end on different screens
.align-self-*-center	Align a flex item in the center on different screens
.align-self-*-baseline	Align a flex item on the baseline on different screens
.align-self-*-stretch	Stretch a flex item on different screens

Grids

Bootstrap's grid system allows up to 12 columns across the page. If you do not want to use all 12 column individually, you can group the columns together to create wider columns:

span 1	span 1	span 1	span 1	span 1	span 1	span 1	span 1	span 1	span 1	span 1	span 1
span 4				span 4				span 4			
span 4				span 8							
span 6						span 6					
span 12											

Bootstrap's grid system is responsive, and the columns will re-arrange depending on the screen size: On a big screen it might look better with the content organized in three columns, but on a small screen it would be better if the content items were stacked on top of each other.

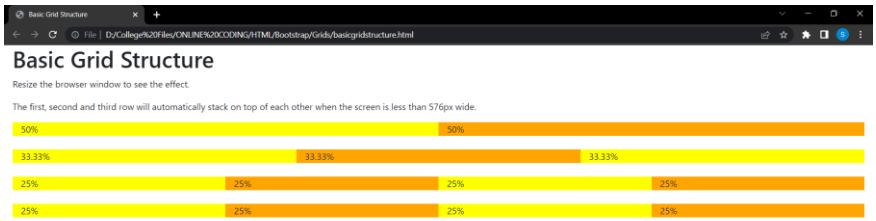
The Bootstrap 4 grid system has five classes:

- .col- (extra small devices - screen width less than 576px)
- .col-sm- (small devices - screen width equal to or greater than 576px)
- .col-md- (medium devices - screen width equal to or greater than 768px)
- .col-lg- (large devices - screen width equal to or greater than 992px)
- .col-xl- (xlarge devices - screen width equal to or greater than 1200px)

The classes above can be combined to create more dynamic and flexible layouts. Each class scales up, so if you wish to set the same widths for sm and md, you only need to specify sm.

Some Bootstrap 4 grid system rules:

- Rows must be placed within a **.container (fixed-width)** or **.container-fluid (full-width)** for proper alignment and padding
- Use rows to create horizontal groups of columns
- Content should be placed within columns, and only columns may be immediate children of rows
- Predefined classes like **.row** and **.col-sm-4** are available for quickly making grid layouts
- Columns create gutters (gaps between column content) via padding. That padding is offset in rows for the first and last column via negative margin on **.rows**
- Grid columns are created by specifying the number of 12 available columns you wish to span. For example, three equal columns would use three **.col-sm-4**
- Column widths are in percentage, so they are always fluid and sized relative to their parent element
- The biggest difference between Bootstrap 3 and Bootstrap 4 is that Bootstrap 4 now uses flexbox, instead of floats. One big advantage with flexbox is that grid columns without a specified width will automatically layout as "equal width columns" (and equal height). Example: Three elements with **.col-sm** will each automatically be 33.33% wide from the small breakpoint and up.

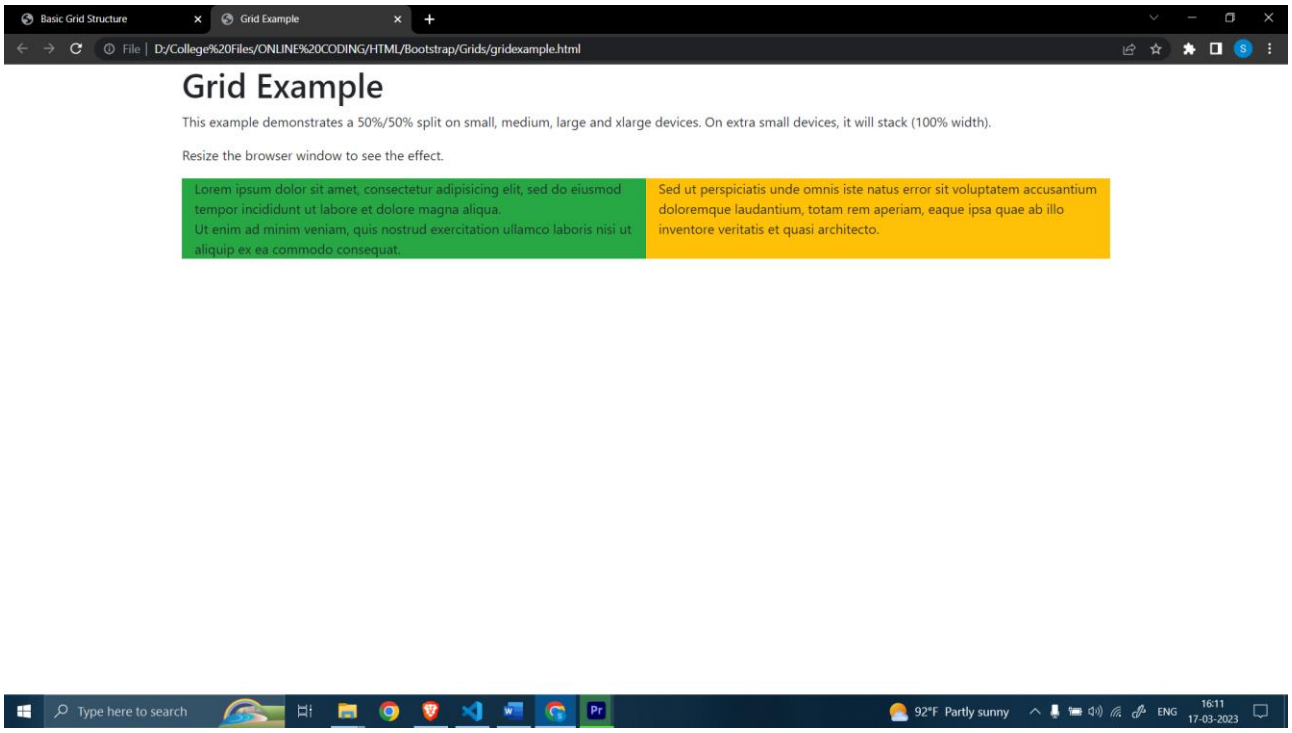


The following table summarizes how the Bootstrap 4 grid system works across different screen sizes:

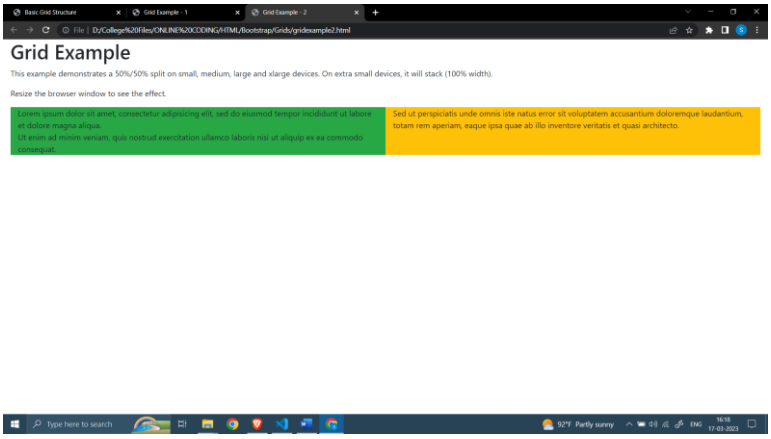
	Extra small (<576px)	Small (>=576px)	Medium (>=768px)	Large (>=992px)	Extra Large (>=1200px)
Class prefix	.col-	.col-sm-	.col-md-	.col-lg-	.col-xl-
Grid behaviour	Horizontal at all times	Collapsed to start, horizontal above breakpoints	Collapsed to start, horizontal above breakpoints	Collapsed to start, horizontal above breakpoints	Collapsed to start, horizontal above breakpoints
Container width	None (auto)	540px	720px	960px	1140px
Suitable for	Portrait phones	Landscape phones	Tablets	Laptops	Laptops and Desktops
# of columns	12	12	12	12	12
Gutter width	30px (15px on each side of a column)	30px (15px on each side of a column)	30px (15px on each side of a column)	30px (15px on each side of a column)	30px (15px on each side of a column)
Nestable	Yes	Yes	Yes	Yes	Yes
Offsets	Yes	Yes	Yes	Yes	Yes
Column ordering	Yes	Yes	Yes	Yes	Yes

We will create a basic grid system that starts out stacked on extra small devices, before becoming horizontal on larger devices.

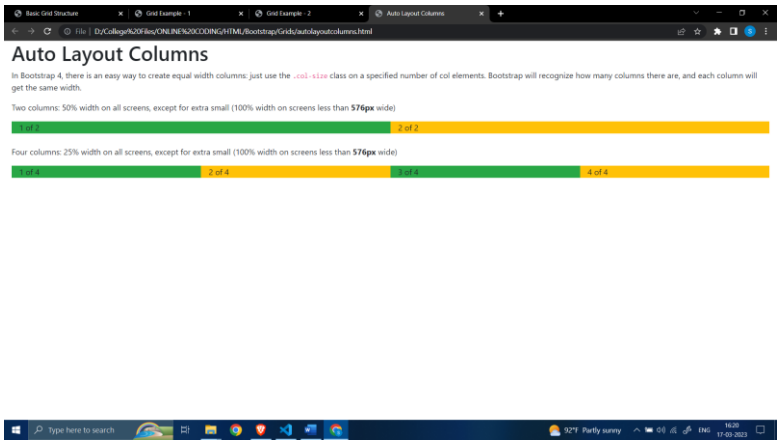
The following example shows a simple "stacked-to-horizontal" two-column layout, meaning it will result in a 50%/50% split on all screens, except for extra small screens, which it will automatically stack (100%):



You can turn any fixed-width layout into a full-width layout by changing the **.container** class to **.container-fluid**:



There is an easy way to create equal width columns for all devices: just remove the number from **.col-size-*** and only use the **.col-size** class on a specified number of col elements. Bootstrap will recognize how many columns there are, and each column will get the same width. The size classes determines when the columns should be responsive:



Use the **.col** class on a specified number of elements and Bootstrap will recognize how many elements there are (and create equal-width columns). In the example below, we use three col elements, which gets a width of 33.33% each.