**Bootstrap**

**Introduction:**

To start coding in Bootstrap we have to copy some links to the **Bootstrap** links in a HTML file

Those links are:

1. <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-QWTKZyjpPEjISv5WaRU9OFeRpok6YctnYmDr5pNlyT2bRjXh0JMhjY6hW+ALEwIH" crossorigin="anonymous">
2. <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js" integrity="sha384-YvpcrYf0tY3lHB60NNkmXc5s9fDVZLESaAA55NDzOxhy9GkcIdslK1eN7N6jIeHz" crossorigin="anonymous"></script>

To use it in a the **Javascript** copy these links to HTML file

1. <script src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.11.8/dist/umd/popper.min.js" integrity="sha384-I7E8VVD/ismYTF4hNIPjVp/Zjvgyol6VFvRkX/vR+Vc4jQkC+hVqc2pM8ODewa9r" crossorigin="anonymous"></script>
2. <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.min.js" integrity="sha384-0pUGZvbkm6XF6gxjEnlmuGrJXVbNuzT9qBBavbLwCsOGabYfZo0T0to5eqruptLy" crossorigin="anonymous"></script>

We can also download the packages to the bootstrap to use.

After downloading the source files we get the contents to the system. Those contents are

**Contents:**

1. Compiled Bootstrap
2. CSS files
3. JS files
4. Bootstrap Source Code

Bootstrap supports all types of major browsers and all Operating Systems

**Customization in Bootstrap**

**C**ustomization in **Bootstrap** includes **saas, color, options, color modes, components, CSS variables** and **optimize**

Customize Bootstrap with our built-in custom variables file and easily toggle global CSS preferences with new $enable-\* Sass variables. Override a variable’s value and recompile with npm run test as needed.

**Colors in Bootstrap:**

Bootstrap’s color palette has continued to expand and become more nuanced in v5.3.0. We’ve added new variables for secondary and tertiary text and background colors, plus {color}-bg-subtle, {color}-border-subtle, and {color}-text-emphasis for our theme colors.

The main Theme colors in the bootstrap are:

$theme-colors: {

"primary": $primary,

"secondary": $secondary,

"success": $success,

"info": $info,

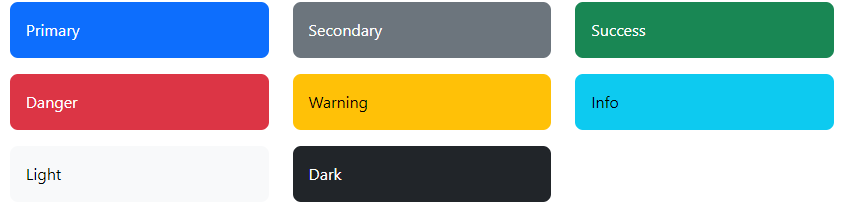
"warning": $warning,

"danger": $danger,

"light": $light,

"dark": $dark

};



Above mentioned are the theme colors of bootstrap

**Color modes**

Bootstrap now has the color modes. The main color mode is the **DARK** mode

Example:

<div class="dropdown" data-bs-theme="dark">

<button class="btn btn-secondary dropdown-toggle" type="button" id="dropdownMenuButtonDark" data-bs-toggle="dropdown" aria-expanded="false">

Dark dropdown

</button>

<ul class="dropdown-menu" aria-labelledby="dropdownMenuButtonDark">

<li><a class="dropdown-item active" href="#">Action</a></li>

<li><a class="dropdown-item" href="#">Action</a></li>

<li><a class="dropdown-item" href="#">Another action</a></li>

<li><a class="dropdown-item" href="#">Something else here</a></li>

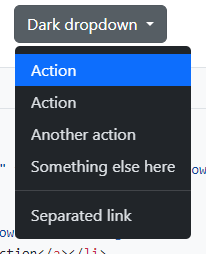
<li><hr class="dropdown-divider"></li>

<li><a class="dropdown-item" href="#">Separated link</a></li>

</ul>

</div>

Above mentioned is the code for using the dark mode. The output is



To use another color in the dark mode use the blow code

<div data-bs-theme="blue">

...

</div>

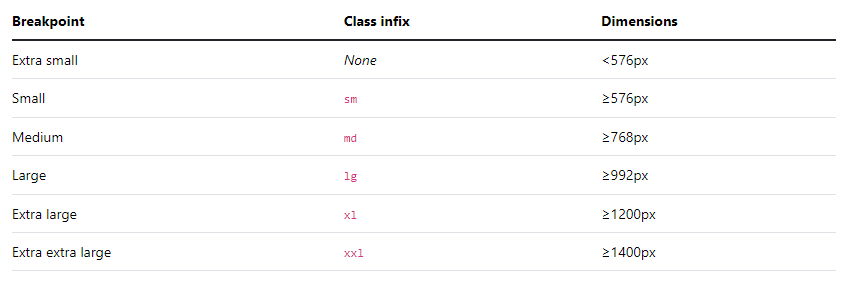
**Layouts in Bootstrap**

**Breakpoints:**

Breakpoints in bootstrap helps to create the responsive layouts according to the Device size or View ports

**Available Breakpoints:**

The Breakpoints available in the bootstrap are



**Containers in Bootstrap**

Containers are the building blocks in the bootstrap that holds the content within a given device or View port. There are three types of containers available in the Bootstrap.

They are:

1. **.container**, which sets a max-width at each responsive breakpoint
2. **.container-{breakpoint}**, which is width: 100% until the specified breakpoint
3. **.container-fluid**, which is width: 100% at all breakpoints

The “**container”** is the **default container** among those three containers.

**Responsive Container:**

The responsive container allow us to specify the class until the breakpoint is reached

Examples: <div class="container-sm">100% wide until small breakpoint</div>

<div class="container-md">100% wide until medium breakpoint</div>

<div class="container-lg">100% wide until large breakpoint</div>

<div class="container-xl">100% wide until extra large breakpoint</div>

<div class="container-xxl">100% wide until extra extra large breakpoint</div>

**Grid system in Bootstrap**

Grid system in bootstrap is used to align content and to layout the content

Example:

<div class="container text-center">

<div class="row">

<div class="col">

Column

</div>

<div class="col">

Column

</div>

<div class="col">

Column

</div>

</div>

</div>

The above snippet’s output is:



There are six default grid options in the bootstrap. These are

* Extra small (xs)
* Small (sm)
* Medium (md)
* Large (lg)
* Extra-large (xl)
* Extra extra large (xxl)

1. To have the Equal width of the grid layouts use the same number of col in the code.

<div class="row">

<div class="col">

1 of 2

</div>

<div class="col">

2 of 2

</div> </div>

**Setting the column Width:**

If we set a width of a column in the container other than the auto width, then the other columns automatically adjusts their width according to the container.

<div class="row">

<div class="col">

1 of 3

</div>

<div class="col-6">

2 of 3 (wider)

</div>

<div class="col">

3 of 3

</div>

</div>

Output:



**Variable width content:**

Use col-{breakpoint}-auto classes to size columns based on the natural width of their content.

If we use above highlighted code in the container the columns are automatically adjusted according to their siblings’ width.

**All Breakpoints:**

Use the col, col-width of column to specify the width of the column. If you don’t specify the column width in **col**, then the columns are automatically get adjusted.

Example:

<div class="container text-center">

<div class="row">

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

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

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

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

</div>

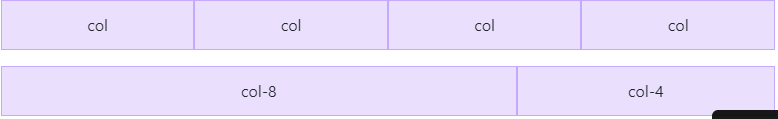
<div class="row">

<div class="col-8">col-8</div>

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

</div>

</div>



**Row Columns**

If we want to have two rows in a container with columns use the below code

<div class="container text-center">

<div class="row row-cols-2">

<div class="col">Column</div>

<div class="col">Column</div>

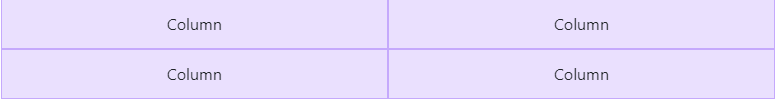
<div class="col">Column</div>

<div class="col">Column</div>

</div>

</div>

Output:



**Nesting:**

To insert a grid inside another grid which is called nesting add a new .row and set of **.col-sm-\*** columns within an existing **.col-sm-\*** column

Example:

<div class="container text-center">

<div class="row">

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

Level 1: .col-sm-3

</div>

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

<div class="row">

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

Level 2: .col-8 .col-sm-6

</div>

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

Level 2: .col-4 .col-sm-6

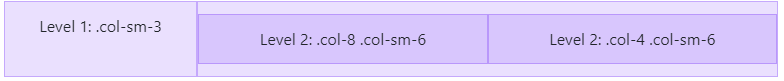
</div>

</div>

</div>

</div>

</div>

Output: 

**Columns**

* While using the grid layouts all the content goes into the columns
* Bootstrap has some predefined classes for creating fast and responsive layouts

**Alignment**

Here we use the utilities to align columns vertically and horizontally

**Vertical alignment**

To align columns vertically in the container we use the utilities called **“align-items”** and **“align-self”**

Example:

<div class="container text-center">

<div class="row align-items-start">

<div class="col">

One of three columns

</div>

<div class="col">

One of three columns

</div>

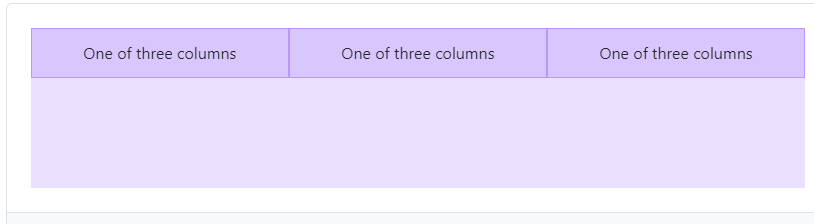
<div class="col">

One of three columns

</div>

</div>

</div>



<div class="container text-center">

<div class="row">

<div class="col align-self-start">

One of three columns

</div>

<div class="col align-self-center">

One of three columns

</div>

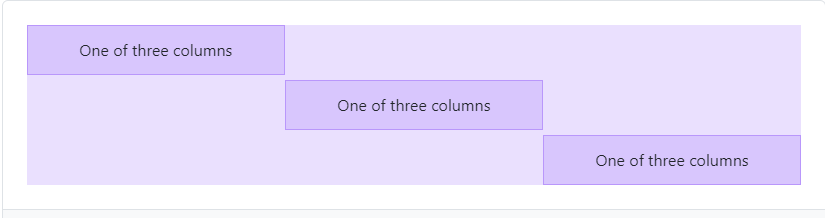
<div class="col align-self-end">

One of three columns

</div>

</div>

</div>



**Horizontal Alignment**

Horizontal alignment can be achieved by using the **“justify-content”**

**Example:**

<div class="container text-center">

<div class="row justify-content-start">

<div class="col-4">

One of two columns

</div>

<div class="col-4">

One of two columns

</div>

</div>

<div class="row justify-content-center">

<div class="col-4">

One of two columns

</div>

<div class="col-4">

One of two columns

</div>

</div>

<div class="row justify-content-end">

<div class="col-4">

One of two columns

</div>

<div class="col-4">

One of two columns

</div>

</div>

<div class="row justify-content-around">

<div class="col-4">

One of two columns

</div>

<div class="col-4">

One of two columns

</div>

</div>

<div class="row justify-content-between">

<div class="col-4">

One of two columns

</div>

<div class="col-4">

One of two columns

</div>

</div>

<div class="row justify-content-evenly">

<div class="col-4">

One of two columns

</div>

<div class="col-4">

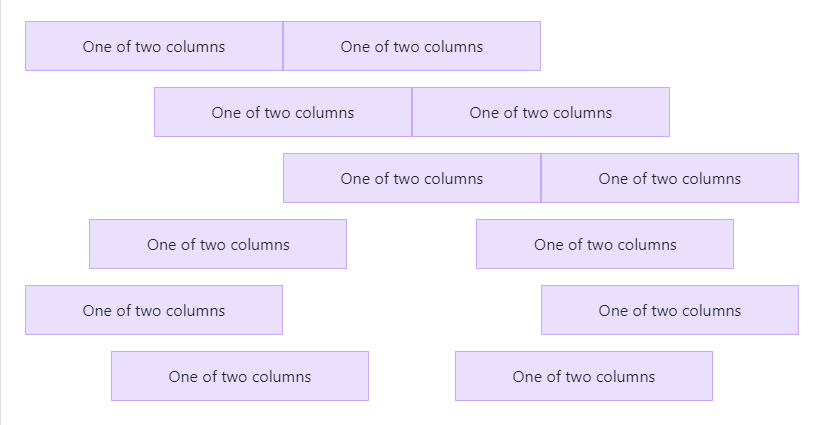
One of two columns

</div>

</div>

</div>

**Output**



**Column Wrapping**

If there are more than 12 columns in a single row and are mentioned in groups, then the extra columns will take the next row.

**Example:**

<div class="container">

<div class="row">

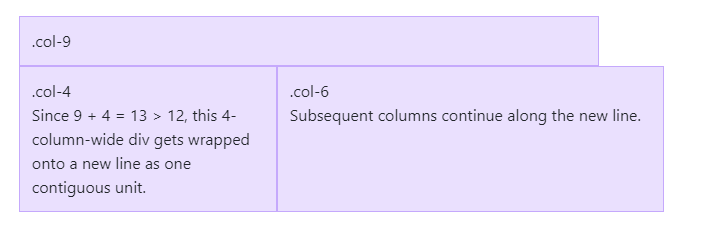
<div class="col-9">.col-9</div>

<div class="col-4">.col-4<br>Since 9 + 4 = 13 &gt; 12, this 4-column-wide div gets wrapped onto a new line as one contiguous unit.</div>

<div class="col-6">.col-6<br>Subsequent columns continue along the new line.</div>

</div>

</div>



**Column Breaks**

If you want to go to the next row even if there is some space in the first row, you have to add an element with the max-width

**Example**

<div class="container text-center">

<div class="row">

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

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

<!-- Force next columns to break to new line -->

<div class="w-100"></div>

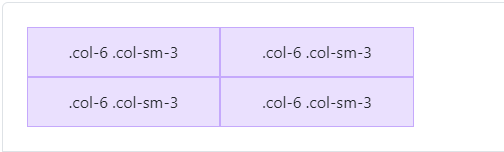
<div class="col-6 col-sm-3">.col-6 .col-sm-3</div>

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

</div>

</div>

**Output**



**Gutters**

Gutters are the padding between the columns used to responsively space and align the content

**Horizontal Gutters**

Horizontal gutters are used to control the spacing between the columns horizontally. **“.gx-“**  is used for the Horizontal gutter

You also have an alternative for horizontal gutter. Use .overflow-hidden in the container instead.

Example:

<div class="container px-4 text-center">

<div class="row gx-5">

<div class="col">

<div class="p-3">Custom column padding</div>

</div>

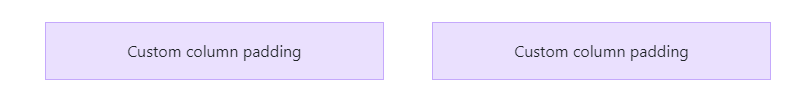
<div class="col">

<div class="p-3">Custom column padding</div>

</div>

</div>

</div>



2.

<div class="container overflow-hidden text-center">

<div class="row gx-5">

<div class="col">

<div class="p-3">Custom column padding</div>

</div>

<div class="col">

<div class="p-3">Custom column padding</div>

</div>

</div>

</div>

Both of the above examples will have the same output

**Vertical Gutters**

Vertical gutters are used to give space between the rows. **“.gy-“** is used to apply vertical gutter.

Example:

<div class="container overflow-hidden text-center">

<div class="row gy-5">

<div class="col-6">

<div class="p-3">Custom column padding</div>

</div>

<div class="col-6">

<div class="p-3">Custom column padding</div>

</div>

<div class="col-6">

<div class="p-3">Custom column padding</div>

</div>

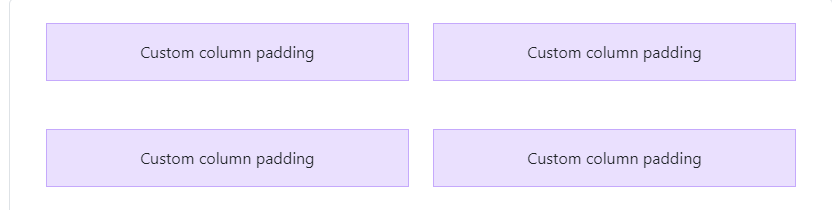
<div class="col-6">

<div class="p-3">Custom column padding</div>

</div>

</div>

</div>



**Horizontal and Vertical Gutters**

**“.g-“** is used as a short hand for both horizontal and vertical gutters at the same time.

Example:

<div class="container text-center">

<div class="row g-2">

<div class="col-6">

<div class="p-3">Custom column padding</div>

</div>

<div class="col-6">

<div class="p-3">Custom column padding</div>

</div>

<div class="col-6">

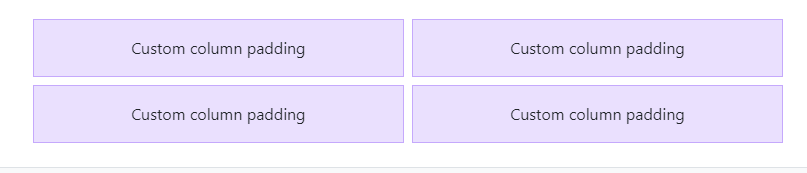
<div class="p-3">Custom column padding</div>

</div>

<div class="col-6">

<div class="p-3">Custom column padding</div></div></div></div>

Output:



**Utilities**

**U**tilities are used for faster mobile friendly and responsive development.

* Using the display utilities you can have the responsively toggling common values in toggling property
* Flexbox display uses d-flex to allow sizing, spacing etc
* Margin and Padding are used to control the space and sizes of the elements and components

**Z-Index**

**Z**-Index provides a third axis to arrange the content. We use the Z-index to properly layer navigation, tooltips, modals and more

**CSS Grid**

Key differences

Compared to the default grid system:

* Flex utilities don’t affect the CSS Grid columns in the same way.
* Gaps replaces gutters. The gap property replaces the horizontal padding from our default grid system and functions more like margin.
* As such, unlike .rows, .grids have no negative margins and margin utilities cannot be used to change the grid gutters. Grid gaps are applied horizontally and vertically by default. See the [customizing section](https://getbootstrap.com/docs/5.3/layout/css-grid/#customizing) for more details.
* Inline and custom styles should be viewed as replacements for modifier classes (e.g., style="--bs-columns: 3;" vs class="row-cols-3").
* Nesting works similarly, but may require you to reset your column counts on each instance of a nested .grid. See the [nesting section](https://getbootstrap.com/docs/5.3/layout/css-grid/#nesting) for details.

**Content**

**Typography**

**Headings:**

All the headings from <h1> to <h6> are available.

To customize the heading we can use the “body-secondary” to fade the text.

**Display Headings:**

Use **Display headings** instead of normal headings to stand out from others

**Inline text Elements:**

There are some styling inline elements in Bootstrap. Those are

1. <mark>: Use this tag to highlight thr text
2. <del>: Use this tag to treat as the deleted text
3. <ins>: This tag is used to treat the mentioned text as the addition to the document
4. <u>: This tag is used to underline the text
5. <small>: This tag is used to treat the text as fine print
6. <strong>: It is used to give **bold** styling
7. <em>: This tag is used to put *Italic* styling

**Blockquotes:**

This is used to quote the blocks of content from another source within the document.

**Naming a Source:**

To give the source to a figure use **blockquote-footer** after the figure.

**LISTS:**

To put the list item in the same line we will use .list-inline and .list-inline-item.

**Description list alignment:**

We can use <dl> tag to create description lists and <dt>, <dd> are used as list elements.

**IMAGES**

**Responsive Images:**

Images in Bootstrap can be made responsive by using .img-fluid. This will give the image max-width: 100%; and height: auto; to the image

**Image Thumbnails:**

.img-thumbnail gives the image a rounded 1px border even without giving the stying

**Aligning Images:**

* Use float class and text alignment classes to align the images to start, end and center
* Also you can use mx-auto d-block to fit the image in the center

**Tables**

**W**e can give the colors to the tables using the css options.

Example:

<!-- On tables -->

<table class="table-primary">...</table>

<table class="table-secondary">...</table>

<table class="table-success">...</table>

<table class="table-danger">...</table>

<table class="table-warning">...</table>

<table class="table-info">...</table>

<table class="table-light">...</table>

<table class="table-dark">...</table>

<!-- On rows -->

<tr class="table-primary">...</tr>

<tr class="table-secondary">...</tr>

<tr class="table-success">...</tr>

<tr class="table-danger">...</tr>

<tr class="table-warning">...</tr>

<tr class="table-info">...</tr>

<tr class="table-light">...</tr>

<tr class="table-dark">...</tr>

<!-- On cells (`td` or `th`) -->

<tr>

<td class="table-primary">...</td>

<td class="table-secondary">...</td>

<td class="table-success">...</td>

<td class="table-danger">...</td>

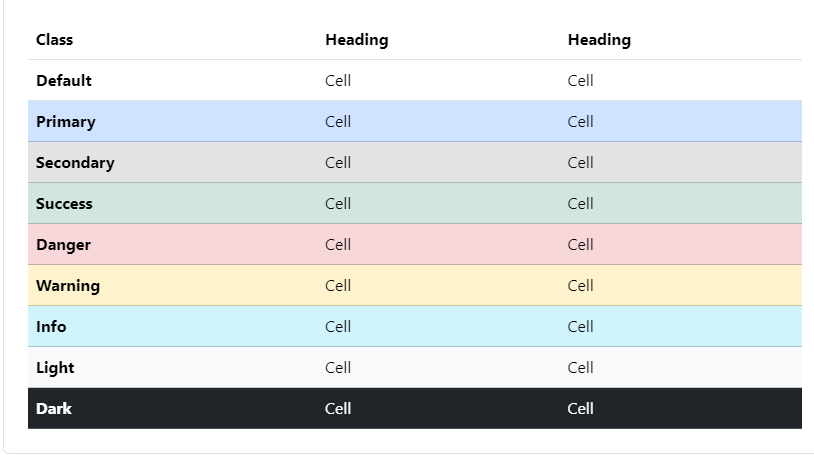
<td class="table-warning">...</td>

<td class="table-info">...</td>

<td class="table-light">...</td>

<td class="table-dark">...</td>

</tr>



**Accented Tables**

**Striped Rows**

**U**sing .table-striped will give striped rows in the table

**Striped columns**

**U**sing .table-striped-column will give zebra-striping to any table column in the table

Use **table-dark** to give dark theme to the table

**Hoverable Rows**

Use .table-hover to enable the hover effect in the table rows

**Active Tables**

To highlight a table row or cell by adding .table-active class

**Bordered table**

**A**dd .table-bordered to give the borders on sides of tables and cells.

**Tables without Borders**

Use .table-borderless to remove all the borders

**Small Tables**

Adding .table-sm in table class to make the padding of the table into half

**Figures**

To use an image with a caption use the figure class.

Example:

<figure class="figure">

<img src="..." class="figure-img img-fluid rounded" alt="...">

<figcaption class="figure-caption">A caption for the above image.</figcaption>

</figure>

**Forms**

**Form Control:**

**Sizing of the content:**

Set the heights using classes like .form-control-lg and .form-control-sm.

**Form Text**

* The text form of the form can be created using .form-text
* If a block-level element will be used, a top margin is added for easy spacing from the inputs above.
* Inline text can use any typical inline HTML element (be it a <span>, <small>, or something else) with nothing more than the .form-text class.

**Form Sizing:**

**Sizing:**

You may also choose from small and large custom selects to match our similarly sized text inputs.

Example:

<select class="form-select form-select-lg mb-3" aria-label="Large select example">

<option selected>Open this select menu</option>

<option value="1">One</option>

<option value="2">Two</option>

<option value="3">Three</option>

</select>

<select class="form-select form-select-sm" aria-label="Small select example">

<option selected>Open this select menu</option>

<option value="1">One</option>

<option value="2">Two</option>

<option value="3">Three</option>

</select>

**Checks**

**Checked checkbox:**

This checkbox is checked on default.



<div class="form-check">

<input class="form-check-input" type="checkbox" value="" id="flexCheckChecked" checked>

<label class="form-check-label" for="flexCheckChecked">

Checked checkbox

</label>

</div>

**Default checkbox:**

This is a default checkbox



<div class="form-check">

<input class="form-check-input" type="checkbox" value="" id="flexCheckDefault">

<label class="form-check-label" for="flexCheckDefault">

Default checkbox

</label>

</div>

**Radios**

**Default radio**

<div class="form-check">

<input class="form-check-input" type="radio" name="flexRadioDefault" id="flexRadioDefault1">

<label class="form-check-label" for="flexRadioDefault1">

Default radio

</label>

</div>



**Checked Radio**

<div class="form-check">

<input class="form-check-input" type="radio" name="flexRadioDefault" id="flexRadioDefault2" checked>

<label class="form-check-label" for="flexRadioDefault2">

Default checked radio

</label>

</div>

****

**Switches**

<div class="form-check form-switch">

<input class="form-check-input" type="checkbox" role="switch" id="flexSwitchCheckDefault">

<label class="form-check-label" for="flexSwitchCheckDefault">Default switch checkbox input</label>

</div>

<div class="form-check form-switch">

<input class="form-check-input" type="checkbox" role="switch" id="flexSwitchCheckChecked" checked>

<label class="form-check-label" for="flexSwitchCheckChecked">Checked switch checkbox input</label>

</div>

<div class="form-check form-switch">

<input class="form-check-input" type="checkbox" role="switch" id="flexSwitchCheckDisabled" disabled>

<label class="form-check-label" for="flexSwitchCheckDisabled">Disabled switch checkbox input</label>

</div>

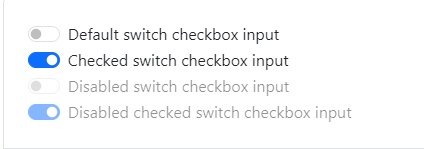
<div class="form-check form-switch">

<input class="form-check-input" type="checkbox" role="switch" id="flexSwitchCheckCheckedDisabled" checked disabled>

<label class="form-check-label" for="flexSwitchCheckCheckedDisabled">Disabled checked switch checkbox input</label>

</div>

**Output**

****

**Toggle buttons**

<input type="checkbox" class="btn-check" id="btn-check" autocomplete="off">

<label class="btn btn-primary" for="btn-check">Single toggle</label>

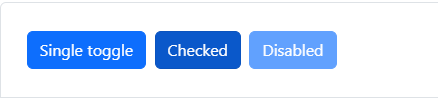
<input type="checkbox" class="btn-check" id="btn-check-2" checked autocomplete="off">

<label class="btn btn-primary" for="btn-check-2">Checked</label>

<input type="checkbox" class="btn-check" id="btn-check-3" autocomplete="off" disabled>

<label class="btn btn-primary" for="btn-check-3">Disabled</label>

**Output**

****

**Range**

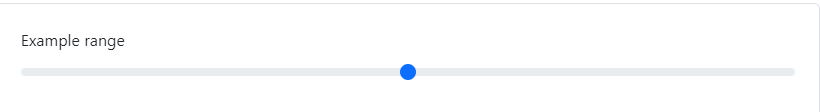
Create custom <input type="range"> controls with .form-range.

Example:

<label for="customRange1" class="form-label">Example range</label>

<input type="range" class="form-range" id="customRange1">

Output:



**Input Group**

Extend the Form controls by adding text, buttons on either side of textual inputs, custom selects and custom file inputs.

**Basic Example:**

<div class="input-group mb-3">

<span class="input-group-text" id="basic-addon1">@</span>

<input type="text" class="form-control" placeholder="Username" aria-label="Username" aria-describedby="basic-addon1">

</div><div class="input-group mb-3">

<input type="text" class="form-control" placeholder="Recipient's username" aria-label="Recipient's username" aria-describedby="basic-addon2">

<span class="input-group-text" id="basic-addon2">@example.com</span>

</div><div class="mb-3">

<label for="basic-url" class="form-label">Your vanity URL</label>

<div class="input-group">

<span class="input-group-text" id="basic-addon3">https://example.com/users/</span>

<input type="text" class="form-control" id="basic-url" aria-describedby="basic-addon3 basic-addon4">

</div>

<div class="form-text" id="basic-addon4">Example help text goes outside the input group.</div>

</div>

<div class="input-group mb-3">

<span class="input-group-text">$</span>

<input type="text" class="form-control" aria-label="Amount (to the nearest dollar)">

<span class="input-group-text">.00</span>

</div><div class="input-group mb-3">

<input type="text" class="form-control" placeholder="Username" aria-label="Username">

<span class="input-group-text">@</span>

<input type="text" class="form-control" placeholder="Server" aria-label="Server">

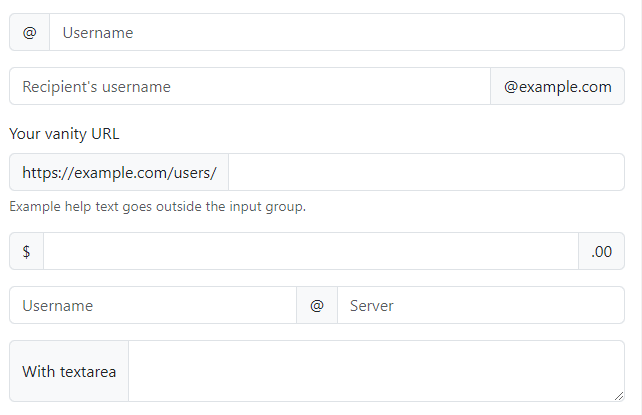
</div><div class="input-group">

<span class="input-group-text">With textarea</span>

<textarea class="form-control" aria-label="With textarea"></textarea>

</div>

**Output**

****

**Wrapping:**

Input groups wrap by default. But in order to accommodate custom form field validation within an input group we use **flex-wrap: wrap**

**Example:**

<div class="input-group flex-nowrap">

<span class="input-group-text" id="addon-wrapping">@</span>

<input type="text" class="form-control" placeholder="Username" aria-label="Username" aria-describedby="addon-wrapping">

</div>

**Output**

****

**Sizing:**

To give sizing to the form classes we can add the sizing to the relative classes itself and the contents will automatically resize themselves without repeating the size classes on each element.

**Example:**

<div class="input-group input-group-sm mb-3">

<span class="input-group-text" id="inputGroup-sizing-sm">Small</span>

<input type="text" class="form-control" aria-label="Sizing example input" aria-describedby="inputGroup-sizing-sm">

</div>

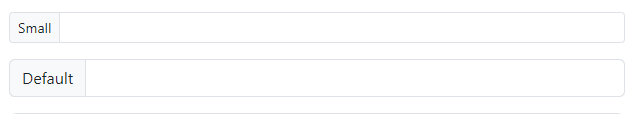
<div class="input-group mb-3">

<span class="input-group-text" id="inputGroup-sizing-default">Default</span>

<input type="text" class="form-control" aria-label="Sizing example input" aria-describedby="inputGroup-sizing-default">

</div>

**Output**

****

**Checkboxes and radios:**

To add a checkbox or radio option in the form give the checkbox or radio option to the .form-check-input when there is no visible text.

**Example:**

<div class="input-group mb-3">

<div class="input-group-text">

<input class="form-check-input mt-0" type="checkbox" value="" aria-label="Checkbox for following text input">

</div>

<input type="text" class="form-control" aria-label="Text input with checkbox">

</div>

<div class="input-group">

<div class="input-group-text">

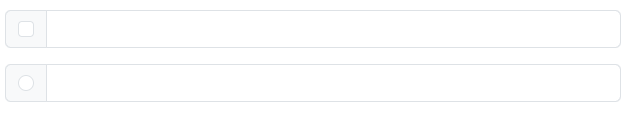
<input class="form-check-input mt-0" type="radio" value="" aria-label="Radio button for following text input">

</div>

<input type="text" class="form-control" aria-label="Text input with radio button">

</div>

**Output**

****

**Multiple Inputs:**

Use <input-group> to add multiple inputs to the form

**Example:**

<div class="input-group">

<span class="input-group-text">First and last name</span>

<input type="text" aria-label="First name" class="form-control">

<input type="text" aria-label="Last name" class="form-control"></div>

**Output:**

****

**Custom Forms:**

Input groups include support for custom selects and custom file inputs. Browser default versions of these are not supported.

**Example:**

<div class="input-group mb-3">

<label class="input-group-text" for="inputGroupSelect01">Options</label>

<select class="form-select" id="inputGroupSelect01">

<option selected>Choose...</option>

<option value="1">One</option>

<option value="2">Two</option>

<option value="3">Three</option>

</select>

</div>

<div class="input-group mb-3">

<select class="form-select" id="inputGroupSelect02">

<option selected>Choose...</option>

<option value="1">One</option>

<option value="2">Two</option>

<option value="3">Three</option>

</select>

<label class="input-group-text" for="inputGroupSelect02">Options</label>

</div>

<div class="input-group mb-3">

<button class="btn btn-outline-secondary" type="button">Button</button>

<select class="form-select" id="inputGroupSelect03" aria-label="Example select with button addon">

<option selected>Choose...</option>

<option value="1">One</option>

<option value="2">Two</option>

<option value="3">Three</option>

</select>

</div>

<div class="input-group">

<select class="form-select" id="inputGroupSelect04" aria-label="Example select with button addon">

<option selected>Choose...</option>

<option value="1">One</option>

<option value="2">Two</option>

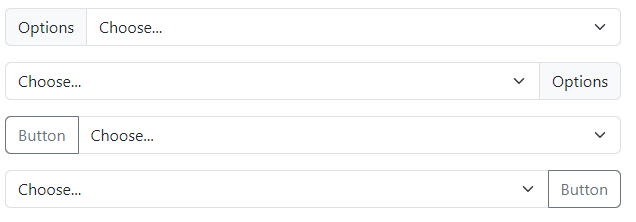
<option value="3">Three</option>

</select>

<button class="btn btn-outline-secondary" type="button">Button</button>

</div>

**Output:**

****

**Floating labels**

Create beautifully simple form labels that float over your input fields

**Textareas:**

By default, <textarea> with .form-control will be the same height as <input>

To set a custom height on the <textarea> use the **height** to increase the textarea.

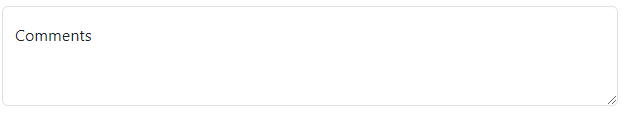
**Example**

<div class="form-floating">

<textarea class="form-control" placeholder="Leave a comment here" id="floatingTextarea2" style="height: 100px"></textarea>

<label for="floatingTextarea2">Comments</label></div>

**Output**

****

**Selects**

Other than .form-control, Floating labels are only available on .form-select. Both work on the same way.

**Example**

<div class="form-floating">

<select class="form-select" id="floatingSelect" aria-label="Floating label select example">

<option selected>Open this select menu</option>

<option value="1">One</option>

<option value="2">Two</option>

<option value="3">Three</option>

</select>

<label for="floatingSelect">Works with selects</label>

</div>

**Output**

****

**Readonly plaintext:**

Floating labels also support plaintext class, which can be helpful for toggling an editable input to a plaintext value without affecting the page layout.

**Example**

<div class="form-floating mb-3">

<input type="email" readonly class="form-control-plaintext" id="floatingEmptyPlaintextInput" placeholder="name@example.com">

<label for="floatingEmptyPlaintextInput">Empty input</label></div>

<div class="form-floating mb-3">

<input type="email" readonly class="form-control-plaintext" id="floatingPlaintextInput" placeholder="name@example.com" value="name@example.com">

<label for="floatingPlaintextInput">Input with value</label>

</div>

**Output**



**Input Groups.**

Floating labels also support .input-group

**Example:**

<div class="input-group mb-3">

<span class="input-group-text">@</span>

<div class="form-floating">

<input type="text" class="form-control" id="floatingInputGroup1" placeholder="Username">

<label for="floatingInputGroup1">Username</label>

</div>

</div>

**Output**

****

**Components**

**Accordian**

Accordian is a type of text documents which uses collapse internally to make it collapsible.

**Example**

<div class="accordion" id="accordionExample">

<div class="accordion-item">

<h2 class="accordion-header">

<button class="accordion-button" type="button" data-bs-toggle="collapse" data-bs-target="#collapseOne" aria-expanded="true" aria-controls="collapseOne">

Accordion Item #1

</button>

</h2>

<div id="collapseOne" class="accordion-collapse collapse show" data-bs-parent="#accordionExample">

<div class="accordion-body">

<strong>This is the first item's accordion body.</strong> It is shown by default, until the collapse plugin adds the appropriate classes that we use to style each element. These classes control the overall appearance, as well as the showing and hiding via

</div>

</div> </div>

<div class="accordion-item">

<h2 class="accordion-header">

<button class="accordion-button collapsed" type="button" data-bs-toggle="collapse" data-bs-target="#collapseTwo" aria-expanded="false" aria-controls="collapseTwo">

Accordion Item #2

</button>

</h2>

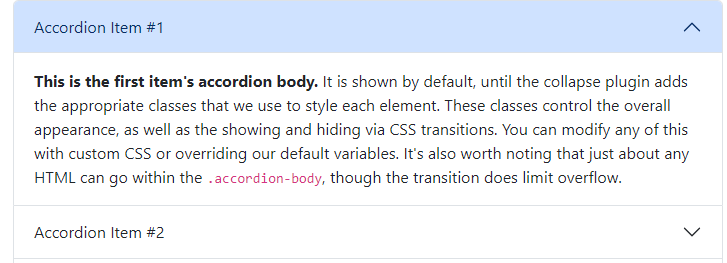
<div id="collapseTwo" class="accordion-collapse collapse" data-bs-parent="#accordionExample">

<div class="accordion-body">

<strong>This is the second item's accordion body.</strong> It is hidden by default, until the collapse plugin adds the appropriate classes that we use to style each element.

</div> </div></div>

**Output:**

****

**Flush**

Add .accordian-flush to remove some borders and rounded corners to give output according to their parent container.

**Example:**

<div class="accordion accordion-flush" id="accordionFlushExample">

<div class="accordion-item">

<h2 class="accordion-header">

<button class="accordion-button collapsed" type="button" data-bs-toggle="collapse" data-bs-target="#flush-collapseOne" aria-expanded="false" aria-controls="flush-collapseOne">

Accordion Item #1

</button>

</h2>

<div id="flush-collapseOne" class="accordion-collapse collapse" data-bs-parent="#accordionFlushExample">

<div class="accordion-body">Placeholder content for this accordion, which is intended to demonstrate the <code>.accordion-flush</code> class. This is the first item's accordion body.</div>

</div>

</div>

<div class="accordion-item">

<h2 class="accordion-header">

<button class="accordion-button collapsed" type="button" data-bs-toggle="collapse" data-bs-target="#flush-collapseTwo" aria-expanded="false" aria-controls="flush-collapseTwo">

Accordion Item #2

</button>

</h2>

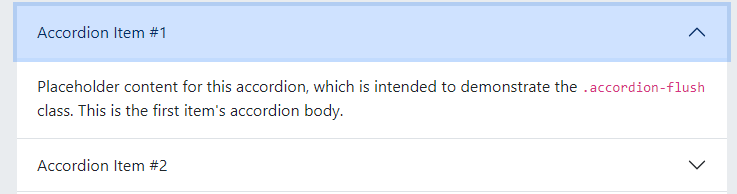
<div id="flush-collapseTwo" class="accordion-collapse collapse" data-bs-parent="#accordionFlushExample">

<div class="accordion-body">Placeholder content for this accordion, which is intended to demonstrate the <code>.accordion-flush</code> class. This is the second item's accordion body. Let's imagine this being filled with some actual content.</div>

</div>

</div>

**Output:**

****

**Always open**

Remove the data-bs-parent attribute on each .accordian-collapse to make accordion item stay open even if another item is opened.

**Example**

<div class="accordion" id="accordionPanelsStayOpenExample">

<div class="accordion-item">

<h2 class="accordion-header">

<button class="accordion-button" type="button" data-bs-toggle="collapse" data-bs-target="#panelsStayOpen-collapseOne" aria-expanded="true" aria-controls="panelsStayOpen-collapseOne">

Accordion Item #1

</button>

</h2>

<div id="panelsStayOpen-collapseOne" class="accordion-collapse collapse show">

<div class="accordion-body">

<strong>This is the first item's accordion body.</strong> It is shown by default, until the collapse plugin adds the appropriate classes that we use to style each element. These classes control the overall appearance, as well as the showing and hiding via CSS transitions. You can modify any of this with custom CSS or overriding our default variables. It's also worth noting that just about any HTML can go within the <code>.accordion-body</code>, though the transition does limit overflow.

</div>

</div>

</div>

<div class="accordion-item">

<h2 class="accordion-header">

<button class="accordion-button collapsed" type="button" data-bs-toggle="collapse" data-bs-target="#panelsStayOpen-collapseTwo" aria-expanded="false" aria-controls="panelsStayOpen-collapseTwo">

Accordion Item #2

</button>

</h2>

<div id="panelsStayOpen-collapseTwo" class="accordion-collapse collapse">

<div class="accordion-body">

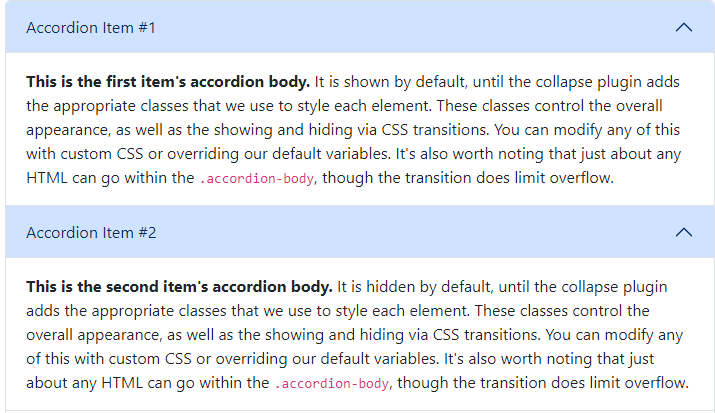
<strong>This is the second item's accordion body.</strong> It is hidden by default, until the collapse plugin adds the appropriate classes that we use to style each element. These classes control the overall appearance, as well as the showing and hiding via CSS transitions. You can modify any of this with custom CSS or overriding our default variables. It's also worth noting that just about any HTML can go within the <code>.accordion-body</code>, though the transition does limit overflow.

</div>

</div>

</div>

**Output**

****

**Alerts**

Alerts provide feedback messages for user actions which are not correct or if there is an error in their action

**Example alerts**

<div class="alert alert-primary" role="alert">

A simple primary alert—check it out!

</div>

<div class="alert alert-secondary" role="alert">

A simple secondary alert—check it out!

</div>

<div class="alert alert-success" role="alert">

A simple success alert—check it out!

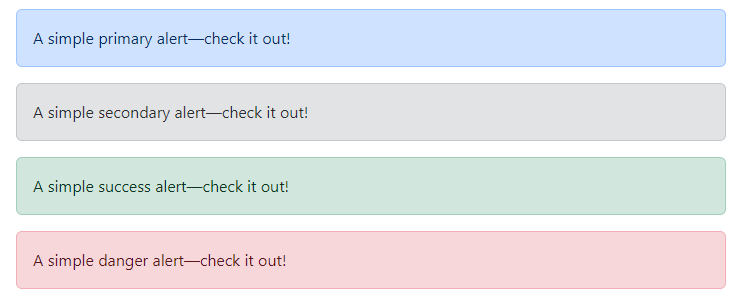
</div>

<div class="alert alert-danger" role="alert">

A simple danger alert—check it out!

</div>

**Output**

****

**Link color:**

Use the .alert-link utility class to quickly provide matching colored links within any alert

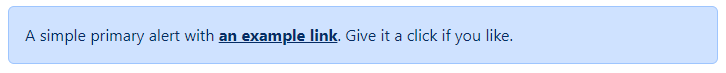
**Example:**

<div class="alert alert-primary" role="alert">

A simple primary alert with <a href="#" class="alert-link">an example link</a>. Give it a click if you like.

</div>

**Output:**

****

**Dismissing alert:**

By using this dismiss alert you can dismiss the alert message by clicking on **X** button.

**Example**

<div class="alert alert-warning alert-dismissible fade show" role="alert">

<strong>Holy guacamole!</strong> You should check in on some of those fields below.

<button type="button" class="btn-close" data-bs-dismiss="alert" aria-label="Close"></button>

</div>

**Output**

****

**Badges**

Badges are used as notification symbols for the labelling components

**Buttons**

Badges are used as a apart of links or buttons to provide a counter

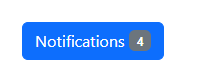
**Example**

<button type="button" class="btn btn-primary">

Notifications <span class="badge text-bg-secondary">4</span>

</button>

**Output**

****

**Positioned**

Use utilities to modify a badge and its position it in the corner of link or button

**Example**

<button type="button" class="btn btn-primary position-relative">

Inbox

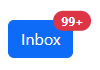
<span class="position-absolute top-0 start-100 translate-middle badge rounded-pill bg-danger">

99+

<span class="visually-hidden">unread messages</span>

</span></button>

**Output**

****

**Background Colors**

You can set a background color for the badges with the contrasting foreground color

**Example**

<span class="badge text-bg-primary">Primary</span>

<span class="badge text-bg-secondary">Secondary</span>

<span class="badge text-bg-success">Success</span>

<span class="badge text-bg-danger">Danger</span>

<span class="badge text-bg-warning">Warning</span>

<span class="badge text-bg-info">Info</span>

<span class="badge text-bg-light">Light</span>

<span class="badge text-bg-dark">Dark</span>

**Output**



**Breadcrumb**

Breadcrumb is used to indicate on which page you are on in the navigation bar

**Example;**

<nav aria-label="breadcrumb">

<ol class="breadcrumb">

<li class="breadcrumb-item active" aria-current="page">Home</li>

</ol>

</nav>

<nav aria-label="breadcrumb">

<ol class="breadcrumb">

<li class="breadcrumb-item"><a href="#">Home</a></li>

<li class="breadcrumb-item active" aria-current="page">Library</li>

</ol>

</nav>

<nav aria-label="breadcrumb">

<ol class="breadcrumb">

<li class="breadcrumb-item"><a href="#">Home</a></li>

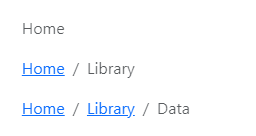
<li class="breadcrumb-item"><a href="#">Library</a></li>

<li class="breadcrumb-item active" aria-current="page">Data</li>

</ol>

</nav>

**Output**



**Dividers**

Dividers are used through the CSS custom Property --bs-breadcrumb-divider

**Example:**

<nav style="--bs-breadcrumb-divider: '>';" aria-label="breadcrumb">

<ol class="breadcrumb">

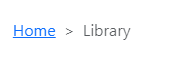
<li class="breadcrumb-item"><a href="#">Home</a></li>

<li class="breadcrumb-item active" aria-current="page">Library</li>

</ol>

</nav>

**Output**



**Buttons**

Use bootstrap custom button styles for forms, dialogs and more with support for multiple sizes, states etc.

**Base class**

Bootstrap has a base button class that sets up basic style

**Example**

<button type="button" class="btn">Base class</button>

**Variants**

Bootstrap has some button variants like colors, link type etc which serves their own purposes

**Example:**

<button type="button" class="btn btn-primary">Primary</button>

<button type="button" class="btn btn-link">Link</button>

**Output**



**Outline buttons**

The outline buttons attribute removes the background color of the button. The background color only appears when you hover on it.

**Example:**

<button type="button" class="btn btn-outline-primary">Primary</button>

<button type="button" class="btn btn-outline-secondary">Secondary</button>

**Output**



**Sizes**

You can give the button sizes using .btn-lg, .btn-sm

**Example:**

<button type="button" class="btn btn-primary btn-lg">Large button</button>

<button type="button" class="btn btn-primary btn-sm">Small button</button>

**Output**



**Button Group;**

Use .btn-group instead of .btn in class to use a group of buttons.

**CARDS**

Cards support variety of content including images, text, list groups, links and many more. Here are some examples,

1. Body- This card contains only body content
2. Titles, Text and link- This card consists of card title, card text and 1 or 2 links
3. Images- This card contains an image and text representing that image
4. List Groups- This card consists of a list of content with a flush list
5. Kitchen Sink- This card consists of all the elements mentioned above
6. Header and Footer- This card will have an optional header and footer
7. Featured- This card helps us feature some content

These cards also uses CSS utilities to have styling for the elements and aligning the content in the card etc

**Carousel**

Carousel is nothing but a group of images which slides one after another.

**Example:**

<div id="carouselExample" class="carousel slide">

<div class="carousel-inner">

<div class="carousel-item active">

<img src="..." class="d-block w-100" alt="...">

</div>

<div class="carousel-item">

<img src="..." class="d-block w-100" alt="...">

</div>

<div class="carousel-item">

<img src="..." class="d-block w-100" alt="...">

</div>

</div>

<button class="carousel-control-prev" type="button" data-bs-target="#carouselExample" data-bs-slide="prev">

<span class="carousel-control-prev-icon" aria-hidden="true"></span>

<span class="visually-hidden">Previous</span>

</button>

<button class="carousel-control-next" type="button" data-bs-target="#carouselExample" data-bs-slide="next">

<span class="carousel-control-next-icon" aria-hidden="true"></span>

<span class="visually-hidden">Next</span>

</button>

</div>

**Output:** 

Note: You must add the .active class to one of the slide to make it a carousel, otherwise the carousel will not be visible and also set the unique ids for all the items in carousel.

**Types of Carousels**

1. **Indicators:** This will indicate in which slide of the carousel you are in
2. **Caption:** This will add caption to the image in the carousel using .carousel-caption
3. **Crossfade:** Adding .carousel-fade to the carousel gives a fade transistion instaed of sliding
4. **Autoplay:** You can make the carousel autoplay one-by-one by adding ride option to the carousel
5. **Individual Carousel interval:** Adding data-bs-interval=”” to the carousel-item gives automatic change in the slide without any manual control

**Close button:**This close button is used to dismiss the content like modals and alerts

Provide an option to dismiss or close a component with .btn-close.

**Collapse**

**Collapse** is used to toggle the visibility of content using a few classes.

These collapse button types are

1. .collapse hides content
2. .collapsing is applied during transitions
3. .collapse.show shows content

Example:

<p class="d-inline-flex gap-1">

<a class="btn btn-primary" data-bs-toggle="collapse" href="#collapseExample" role="button" aria-expanded="false" aria-controls="collapseExample">

Link with href

</a>

<button class="btn btn-primary" type="button" data-bs-toggle="collapse" data-bs-target="#collapseExample" aria-expanded="false" aria-controls="collapseExample">

Button with data-bs-target

</button>

</p>

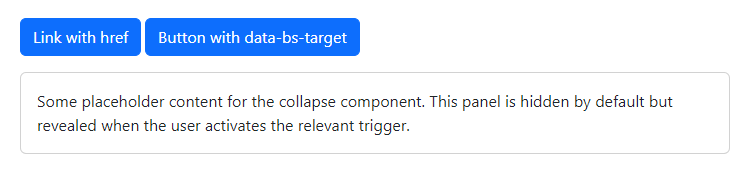
<div class="collapse" id="collapseExample">

<div class="card card-body">

Some placeholder content for the collapse component. This panel is hidden by default but revealed when the user activates the relevant trigger.

</div></div>

Output:



## Horizontal

The collapse plugin supports horizontal collapsing. Add the .collapse-horizontal modifier class to transition the width instead of height and set a width on the immediate child element. Feel free to write your own custom Sass, use inline styles, or use our [width utilities](https://getbootstrap.com/docs/5.3/utilities/sizing/).

**Dropdown:-** dropdown is used to select the list of an items.in dropdown we can also provide the links.

**Eg:-<div class="dropdown">**

**<button class="btn btn-secondary dropdown-toggle" type="button" data-bs-toggle="dropdown" aria-expanded="false">**

**Dropdown button**

**</button>**

**<ul class="dropdown-menu">**

**<li><a class="dropdown-item" href="#">Action</a></li>**

**<li><a class="dropdown-item" href="#">Another action</a></li>**

**<li><a class="dropdown-item" href="#">Something else here</a></li>**

**</ul>**

**</div>**

**List Group:**

List groups are a flexible and powerful component for displaying series of content.

**Types of List Group**

**Active items:**

Add .active to a .list-group-item to see the current list item

Example:

<ul class="list-group">

<li class="list-group-item active" aria-current="true">An active item</li>

<li class="list-group-item">A second item</li>

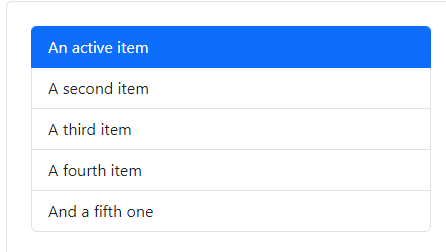
<li class="list-group-item">A third item</li>

<li class="list-group-item">A fourth item</li>

<li class="list-group-item">And a fifth one</li>

</ul>

**Output:**



**Links and Buttons:**

Use <a>s or <button>s to create actionable list group items with hover, disabled, and active states by adding .list-group-item-action.

**Example:**

<div class="list-group">

<a href="#" class="list-group-item list-group-item-action active" aria-current="true">

The current link item

</a>

<a href="#" class="list-group-item list-group-item-action">A second link item</a>

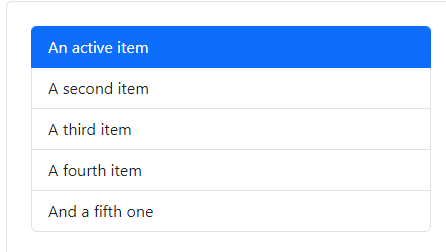
<a href="#" class="list-group-item list-group-item-action">A third link item</a>

<a href="#" class="list-group-item list-group-item-action">A fourth link item</a>

<a class="list-group-item list-group-item-action disabled" aria-disabled="true">A disabled link item</a>

</div>

**Output**



**Flush:**

Adding .list-group-flush will remove some borders and rounded corners

**Example:**

<ul class="list-group list-group-flush">

<li class="list-group-item">An item</li>

<li class="list-group-item">A second item</li>

<li class="list-group-item">A third item</li>

<li class="list-group-item">A fourth item</li>

<li class="list-group-item">And a fifth one</li>

</ul>

**Output**



**Numbered:**

Adding .list-group-numbered class will give the numbering to the list items

**Example:**

<ol class="list-group list-group-numbered">

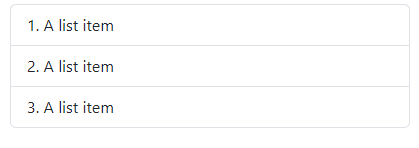
<li class="list-group-item">A list item</li>

<li class="list-group-item">A list item</li>

<li class="list-group-item">A list item</li>

</ol>

**Output**



**Modal:-** modal can give the information like a alert box.in modal we have title and body we have two buttons close and save changes.modal is like a container we use any other elements like login forms etc.

**eg:-<div class="dropdown">**

**<button class="btn btn-secondary dropdown-toggle" type="button" data-bs-toggle="dropdown" aria-expanded="false">**

**Dropdown button**

**</button>**

**<ul class="dropdown-menu">**

**<li><a class="dropdown-item" href="#">Action</a></li>**

**<li><a class="dropdown-item" href="#">Another action</a></li>**

**<li><a class="dropdown-item" href="#">Something else here</a></li>**

**</ul>**

**</div>**

**Navbar:-** Navbars come with built-in support for a handful of sub-components. Choose from the following as needed.in navigation we can also give the logo brandname, list, dropdown etc.

* .navbar-brand for your company, product, or project name.
* .navbar-nav for a full-height and lightweight navigation (including support for dropdowns).
* .navbar-toggler for use with our collapse plugin and other [navigation toggling](https://getbootstrap.com/docs/5.3/components/navbar/#responsive-behaviors) behaviors.
* Flex and spacing utilities for any form controls and actions.
* .navbar-text for adding vertically centered strings of text.
* .collapse.navbar-collapse for grouping and hiding navbar contents by a parent breakpoint.
* Add an optional .navbar-scroll to set a max-height and [scroll expanded navbar content](https://getbootstrap.com/docs/5.3/components/navbar/#scrolling).

**eg:-<!-- As a link -->**

**<nav class="navbar bg-body-tertiary">**

**<div class="container-fluid">**

**<a class="navbar-brand" href="#">Navbar</a>**

**</div>**

**</nav>**

**<!-- As a heading -->**

**<nav class="navbar bg-body-tertiary">**

**<div class="container-fluid">**

**<span class="navbar-brand mb-0 h1">Navbar</span>**

**</div>**

**</nav>**

## Border spinner:-

Use the border spinners for a lightweight loading indicator.we can also give clors for spinners and use growing spinner and align the spinner etc.

**eg:-<div class="spinner-border" role="status">**

**<span class="visually-hidden">Loading...</span>**

**</div>**

# **Progress:-** progress are used to give the progress bar ensuring you can stack progress bars, animate them, and place text labels over them. For progress bar we must specify the value which we want

**Eg:- <div class="progress" role="progressbar" aria-label="Basic example" aria-valuenow="25" aria-valuemin="0" aria-valuemax="100">**

**<div class="progress-bar" style="width: 25%"></div>**

**</div>**

# **Toasts:-**Toasts are used to Push notifications to your visitors with a toast, a lightweight and easily customizable alert message.

**Eg:-<div class="toast" role="alert" aria-live="assertive" aria-atomic="true">**

**<div class="toast-header">**

**<img src="..." class="rounded me-2" alt="...">**

**<strong class="me-auto">Bootstrap</strong>**

**<small>11 mins ago</small>**

**<button type="button" class="btn-close" data-bs-dismiss="toast" aria-label="Close"></button>**

**</div>**

**<div class="toast-body"> Hello, world! This is a toast message.</div>**

**</div>**

**Tooltips:-** Tooltips are used to guide the user we can give the direction like left,right,top,bottom.

**eg:-<button type="button" class="btn btn-secondary"**

**data-bs-toggle="tooltip" data-bs-placement="top"**

**data-bs-custom-class="custom-tooltip"**

**data-bs-title="This top tooltip is themed via CSS variables."> Custom tooltip**

**</button>**

**IconLink:-**Take a regular <a> element, add .icon-link, and insert an icon on either the left or right of your link text. The icon is automatically sized, placed, and colored.

**eg:-<a class="icon-link" href="#">**

**<svg class="bi" aria-hidden="true"><use xlink:href="#box-seam"></use></svg> Icon link**

**</a>**

**Helper Classes:-**

**Bootstrap 5 has a lot of utility/helper classes to quickly style elements without using any CSS code.**

1.Background 9.object-fit 16.z-index

2.Colors 10.Flex 17.Overflow

3.text-bg-color 11.Display 18.Text

4.Borders 12.Opacity 19.Height

5.Border Radius 13.Position 20.width

6.Margin 14.Shadow

7.Padding 15.Visibility

8.Float

**1.Background Color:-** To give the background color we use this property.background colors do not set the text color, so in some cases you'll want to use them together with a .text-\* class.

The classes for background colors are:-

.bg-primary, .bg-success, .bginfo, .bg-warning, .bg-danger, .bg-secondary, .bg-dark and .bg-light.

**eg:-<div class="p-3 bg-primary text-white">Pavani</div> Syntax:- .bg-color**

**2.Colors:-** To give the color for the text we can use this property.The classes for text color are:-

.text-primary, .text-primary-emphasis, .text-secondary, .text-secondary-emphasis .text-success, .text-success-emphasis, .text-danger, .text-danger-emphasis

**Syntax:- .text-color eg:-<p class="text-primary">Pavani</p>**

**3.Text-bg-color:-** If we want to give both text color and background color we can use this property. Based on background color text color will be display.

**Syntax:-.text-bg-color eg:-<div class="text-bg-primary p-3">pavani</div>**

**4.Borders:-** The border property is used to add or remove borders from an element.we use the classes for apply the border they are:- **border border-primary, border border-primary-subtle, border border-secondary, border border-secondary-subtle, border border-success,border** .The values are **border, border border-top-0 , border border-left-0 • border border-bottom-0 Syntax:-**.**border-value eg:-<span class="border border-1"></span>**

**5.Border Radius:-** To change the corners of an elements we use the border radius property with using the classes.

**rounded-0 ,rounded , rounded-sm ,rounded-lg ,Rounded-top, Rounded-right ,Rounded-left , Rounded-bottom, Rounded-circle**

**Syntax:-.rounded-value**  **eg:-<img src="..." class="rounded" alt="img">**

**6.Margin and padding:-** To align the elements we use the padding and margin property.

**• m - sets margin •p - sets padding**

**•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**

**8.Float:-** Float an element to right or left.This property is used to align the elements side by side by using  **Left , right , none**

**Syntax:-.float-value eg:- <div class="float-start">Float start on all viewport sizes</div>**

# **9.Object fit:-** Use the object fit utilities to modify how the content of a replaced element such as an <img> or <video>, should be resized to fit its container.

The values are:-object-fit-contain,

object-fit-contain

object-fit-cover

Object-fit-fill

object-fit-scale (for scale-down)

Object-fit-none

**eg:-<img src="..." class="object-fit-contain border rounded" alt="...">**

**10.Display:-** Quickly and responsively toggle the display value of components and more with our display utilities. Includes support for some of the more common values, as well as some extras for controlling display when printing.

i)d-inline:- converting block-level element to inline-line element.

ii)d-block:- converting inline element to block-level. Element.

iii)d-inline-block:-Element act likes a inline but it will accept block-level properties.

**11.Flex:-**

iv)d-flex:-By using this property we are arranging all the elements into one dimension.

•Justify-content:- around, start, end, between.

• Align-items:- center, start, end, stretch, baseline.

• Flex:- wrap, no-wrap.

• Flex-direction:- column, row, row-reverse, column-reverse.

• Flex-fill:- flex items to force them into equal widths

**12.Opacity:-**  The opacity property sets the opacity level for an element. The opacity level describes the transparency level, where 1 is not transparent at all, .5 is 50% visible, and 0 is completely transparent.

**eg:-<div class="opacity-25">...</div> <div class="opacity-0">...</div>**

**13.Position:-** By Using Position property we can arrange the elements where we want .By using this below values.

. Static, Fixed, Sticky, Relative, Absolute

Whenever we are using position properties along with those we can able to use top, bottom, start, end.

**eg:-<div class="position-fixed">...</div>**

**14.shadow :-**  Shadow property is used to Add or remove shadows to elements with box-shadow utilities.

box-shadow , box-shadow-sm, box-shadow-lg, box-shadow-inset

**eg:-<div class="shadow-none p-3 mb-5 bg-body-tertiary rounded">No shadow</div>**

**15.Visiblity:-**  This visibility have two values.

i)visible:-the element should be visible on UI

II)invisible:-The element should be invisible on UI

**eg:-<div class="visible">...</div> ,<div class="invisible">...</div>**

**16.Z-index:-** Use z-index utilities to stack elements on top of one another. Requires a position value other than static. Z index is used to align the elements on the top of the screen if we use the z index value for the element that element will be display top when the comapring with other elements

**eg:-<div class="z-3 position-absolute p-5 rounded-3"><span>z-3</span></div>**

**Syntax:- .z-values • Z-3 • Z-2 • Z-1 • Z-0 • Z-n**

**17.Overflow:-** The overflow property on the fly with four default values and classes. Auto , Hidden ,Visible , scroll

**Syntax:- .overflow-values eg:-<div class="overflow-visible">...</div>**

**18.text:-** This property is used to Aligning the text.by using the values**:-** Start,Center ,end

**Syntax:- .text-value eg:-<p class="text-center">Center aligned text on all viewport sizes.</p>**

**19.Height:-** Set the width of an element with the h-\* classes (.h-25, .h-50, .h-75, .h100, .mh-100)

**20.Width:-** To Set the width of the elements we use this propertywidth classes are **(.w-25, .w-50, .w-75, .w100, .mw-100)**