[Lesson 5-6]

DAN.IT EDUCATION

What we learnt last time?

- Gulp
- Using Gulp for automation
- Gulp plugins

Our targets for today

- Introduction to Bootstrap
- Installing Bootstrap
- Design templates

[Introduction]

- ➔ Bootstrap is an open source toolkit for developing with HTML, CSS, and JS
- ➔ It helps to build an application in a faster way
- ➔ It is integrated with SASS and jQuery
- ➔ Bootstrap is already responsive and mobile-first, making development of such applications easier

- ➔ Bootstrap 4 comes as CSS library and a separate JS library
- ➔ It is possible to use only CSS part of Bootstrap
- ➔ A full Bootstrap toolkit also uses jQuery, popper.js and its own JS plugin

Introduction

- ➔ Bootstrap changes few things about CSS
- ➔ A major difference with plain CSS is that Bootstrap sets box-sizing of every element on a page as border-box
- ➔ This may cause problems with some third party software like Google Maps
- ➔ In this case, you will need to override these changes to the target widget, you may use the following code:

```css
.selector-for-some-widget {
 box-sizing: content-box;
}
```

Adding via CDN

- Bootstrap can be added via CDN
  - CSS part:

```
<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css"
integrity="sha384-MCw98/SFnGE8fJT3GXwEOngsV7Zt27NXFoaoApmYm81iuXoPkFOJwJ8ERdknLPMO" crossorigin="anonymous">
```

  - JS part and dependencies:

```
<script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"
integrity="sha384-q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo" crossorigin="anonymous"></script>

<script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/popper.min.js"
integrity="sha384-ZMP7rVo3mIykV+2+9J3UJ46jBk0WLaUAdn689aCwoqbBJiSnjAK/l8WvCWPIPm49" crossorigin="anonymous"></script>

<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/js/bootstrap.min.js"
integrity="sha384-ChfqqxuZUCnJSK3+MXmPNIyE6ZbWh2IMqE241rYiqJxyMiZ6OW/JmZQ5stwEULTy" crossorigin="anonymous"></script>
```

- Links to the latest version can be found at https://getbootstrap.com/ in the “Introduction” section of documentation

[Adding via NPM]

- ➔ It also can be installed via NPM:

```
npm install bootstrap
```

- ➔ require('bootstrap') will load all of Bootstrap’s jQuery plugins onto the jQuery object
- ➔ The bootstrap module itself does not export anything. You can manually load Bootstrap’s jQuery plugins individually by loading the /js/*.js files under the package’s top-level directory
- ➔ Bootstrap’s package.json contains some additional metadata under the following keys:
  - → sass - path to Bootstrap’s main Sass source file
  - → style - path to Bootstrap’s non-minified CSS that’s been precompiled using the default settings (no customization)
- ➔ In both cases you are also expected to add meta tag for responsive support:

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

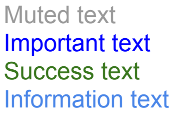
[Typography]

- ➔ Bootstrap 4 default `font-size` is 16px, and its `line-height` is 1.5
- ➔ The default `font-family` is "Helvetica Neue", Helvetica, Arial, sans-serif
- ➔ It changes and extends many tags, like:
  - → `<p>` elements have `margin-top: 0` and `margin-bottom: 1rem` (16px by default)
  - → Headings like `<h1>`, `<h2>` have bolder font-weight and an increased font-size
  - → `<small>` element is used to create a lighter, secondary text in any heading
  - → `<mark>` element is a yellow background color element with some padding
  - → `<abbr>` element have a dotted border bottom
  - → And many other changes

[Text Colors]

- ➔ Bootstrap 4 has built in contextual color classes, like:
- ➔ Classes for text: `.text-muted`, `.text-primary`, `.text-success`, `.text-info`, `.text-warning`, `.text-danger`, `.text-secondary`, `.text-white`,`.text-dark`, `.text-body` and `.text-light`:

```
<p class="text-muted">Muted text</p>
<p class="text-primary">Important text</p>
<p class="text-success">Success text</p>
<p class="text-info">Information text</p>
```



Muted text
Important text
Success text
Information text

- ➔ Contextual text classes can also be used on links, which will add a darker hover color:

```
<a href="#" class="text-muted">Muted link</a>
<a href="#" class="text-primary">Primary link</a>
<a href="#" class="text-success">Success link</a>
```

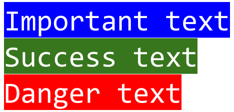
- ➔ You can also add 50% opacity for black or white text with the `.text-black-50` or `.text-white-50` classes:

```
<p class="text-black-50">Black text with 50% opacity on white background</p>
<p class="text-white-50 bg-dark">White text with 50% opacity on black background</p>
```

[Background Colors]

- ➔ There are classes for background colors like: `.bg-primary`, `.bg-success`, `.bg-info`, `.bg-warning`, `.bg-danger`, `.bg-secondary`, `.bg-dark` and `.bg-light`.
- ➔ They don’t set text color, so they can be used in any combinations with `.text-*` classes.

```html
<p class="bg-primary text-white">Important text</p>
<p class="bg-success text-white">Success text</p>
<p class="bg-danger text-white">Danger text</p>
```



Important text

Success text

Danger text

Basic Table

- ➔ Bootstrap 4 changes table and adds light padding and horizontal dividers.
- ➔ `.table` class adds basic styling to a table:
  - → `.table-striped` class adds zebra-stripes to a table
  - → `.table-bordered` class adds borders on all sides of the table and cells
  - → `.table-hover` class adds a hover effect (grey background color) on table rows
  - → `.table-dark` class adds a black background to the table
  - → `.table-borderless` class removes borders from the table

Contextual Classes

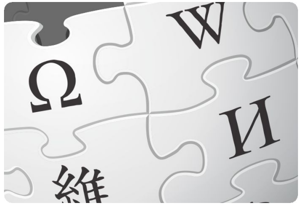
- Bootstrap contextual classes are used to color the whole table (`<table>`), table rows (`<tr>`) or table cells (`<td>`)
- The contextual classes that can be used are:

Class	Description
`.table-primary`	Blue: important action
`.table-success`	Green: successful action
`.table-danger`	Red: dangerous action
`.table-info`	Light blue: informative action
`.table-warning`	Orange: warning
`.table-active`	Grey: hover color to the table row or table cell
`.table-secondary`	Grey: slightly less important action
`.table-light`	Light grey table or table row background
`.table-dark`	Dark grey table or table row background

[Image Shapes]

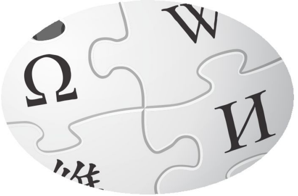
- ➔ Bootstrap provides number of classes for styling and positioning images
- ➔ `.rounded` class adds rounded corners:

```
<img src="wiki.jpg" class="rounded" alt="Cinque Terre">
```



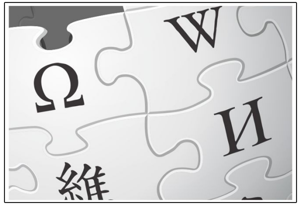
- ➔ `.rounded-circle` class makes image a circle:

```
<img src="wiki.jpg" class="rounded-circle" alt="Cinque Terre">
```



- ➔ `.img-thumbnail` class makes image a thumbnail:

```
<img src="wiki.jpg" class="img-thumbnail" alt="Cinque Terre">
```



Aligning Images

- Images can be aligned to the right with the `.float-right` class and to the left with `.float-left`:

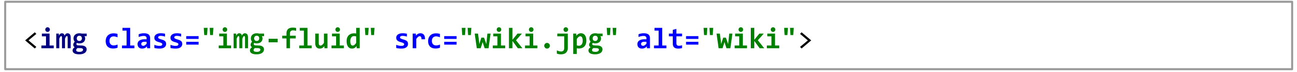
```
<img src="wiki.jpg" class="float-left">
<img src="wiki.jpg" class="float-right">
```

- To center an image add `.mx-auto` class (will set “margin: auto”) or `.d-block` class (will set “display: block”):

```
<img src="wiki.jpg" class="mx-auto d-block">
```

[Responsive Images]

- ➔ Bootstrap’s responsive images automatically adjust to fit the size of the screen
- ➔ Image can become responsive by adding an `.img-fluid` class to the `<img>` tag
- ➔ `.img-fluid` class applies `max-width: 100%;` and `height: auto;` to the image:

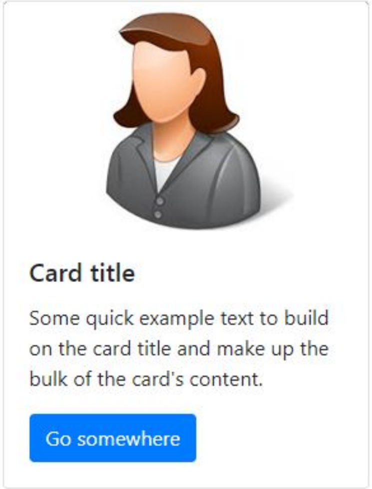


```
<img class="img-fluid" src="wiki.jpg" alt="wiki">
```

[Cards]

➔ Card is a flexible and extensible content container. It includes options for headers and footers, a wide variety of content, contextual background colors, and powerful display options

```
<div class="card" style="width: 18rem;">
  <img class="card-img-top" src=".../100px180/" alt="Card image cap">
  <div class="card-body">
     <h5 class="card-title">Card title</h5>
     <p class="card-text">Some quick example text to build on the card
title and make up the bulk of the card's content.</p>
     <a href="#" class="btn btn-primary">Go somewhere</a>
  </div>
</div>
```

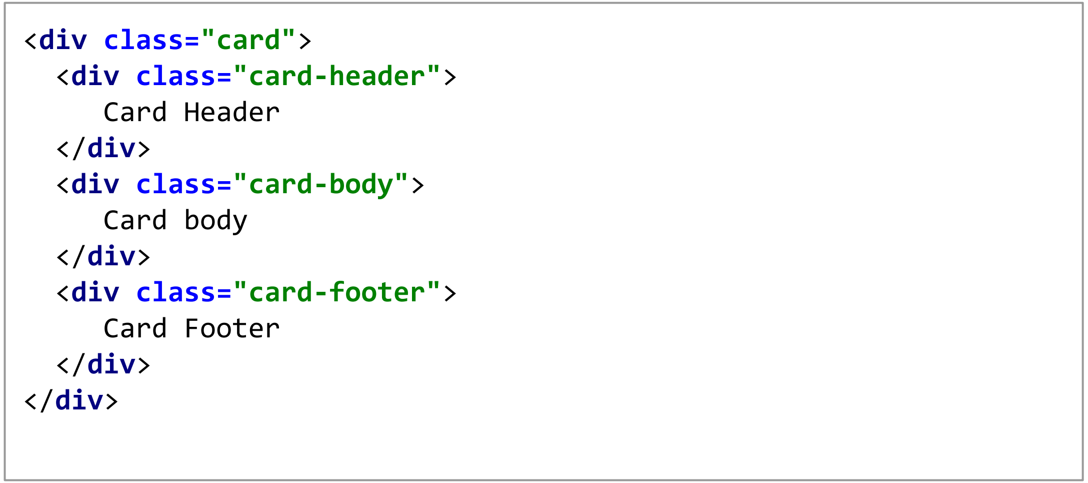


Card title

Some quick example text to build on the card title and make up the bulk of the card's content.

[Cards]

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



```html
<div class="card">
  <div class="card-header">
     Card Header
  </div>
  <div class="card-body">
     Card body
  </div>
  <div class="card-footer">
     Card Footer
  </div>
</div>
```

[Cards]

- ➔ Card titles are used by adding `.card-title` to a `<h*>` tag. In the same way, links are added and placed next to each other by adding `.card-link` to an `<a>` tag
- ➔ Subtitles are used by adding a `.card-subtitle` to a `<h*>` tag. If the `.card-title` and the `.card-subtitle` items are placed in a `.card-body` item, the card title and subtitle are aligned nicely

```
<div class="card" style="width: 18rem;">
  <div class="card-body">
     <h5 class="card-title">Card title</h5>
     <h6 class="card-subtitle mb-2 text-muted">Card subtitle</h6>
     <p class="card-text">Some quick example text to build on the card
title and make up the bulk of the card's content.</p>
     <a href="#" class="card-link">Card link</a>
     <a href="#" class="card-link">Another link</a>
  </div>
</div>
```

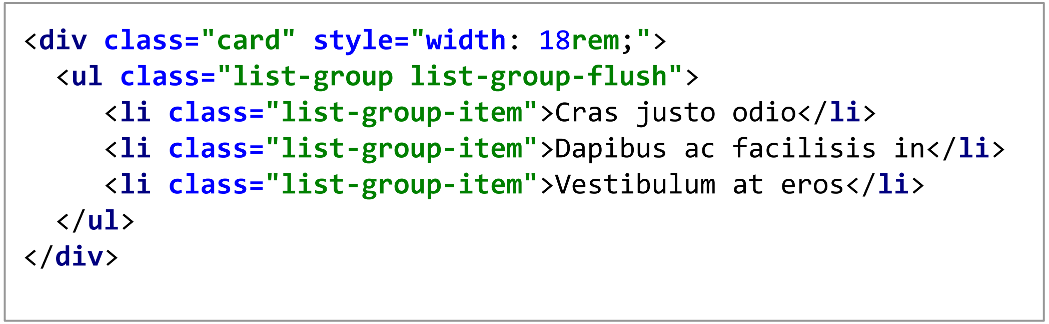
[Cards]

➔ `.card-img-top` places an image to the top of the card. With `.card-text`, text can be added to the card. Text within `.card-text` can also be styled with the standard HTML tags

```html
<div class="card" style="width: 18rem;">
  <img class="card-img-top" src=".../100px180/?text=Image cap" alt="Card image cap">
  <div class="card-body">
     <p class="card-text">Some quick example text to build on the card title and make up
the bulk of the card's content.</p>
  </div>
</div>
```

[Cards]

- ➔ Create lists of content in a card with a flush list group

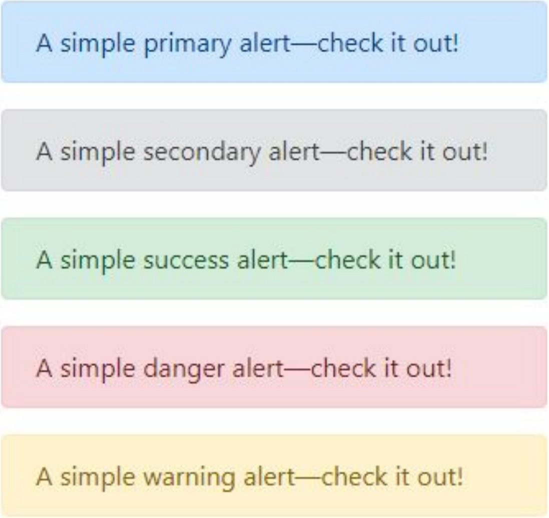


```
<div class="card" style="width: 18rem;">
  <ul class="list-group list-group-flush">
     <li class="list-group-item">Cras justo odio</li>
     <li class="list-group-item">Dapibus ac facilisis in</li>
     <li class="list-group-item">Vestibulum at eros</li>
  </ul>
</div>
```

Bootstrap 4 Alerts

- Bootstrap 4 provides a number of classes for alert messages
- `.alert` class creates a styler alert element in combination with one of the contextual classes `.alert-success`, `.alert-info`, `.alert-warning`, `.alert-danger`, `.alert-primary`, `.alert-secondary`, `.alert-light` or `.alert-dark`:

```html
<div class="alert alert-success">
  <strong>Success!</strong>
</div>
```



A simple primary alert—check it out!

A simple secondary alert—check it out!

A simple success alert—check it out!

A simple danger alert—check it out!

A simple warning alert—check it out!

Bootstrap 4 Alerts

- ➔ Adding `alert-link` class to link inside the alert box will create matching colored link:

```
<div class="alert alert-success">
  <strong>Success!</strong>
</div>
```

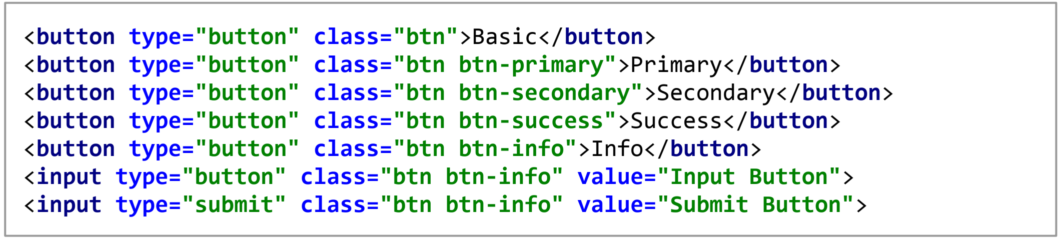
- ➔ To close the alert message, add a `.alert-dismissible` class to the alert container. Then add `class="close"` and `data-dismiss="alert"` to a link or a button element (when you click on this the alert box will disappear)

```
<div class="alert alert-success">
  <strong>Success!</strong> <a href="#" class="alert-link">Read more</a>.
</div>
```

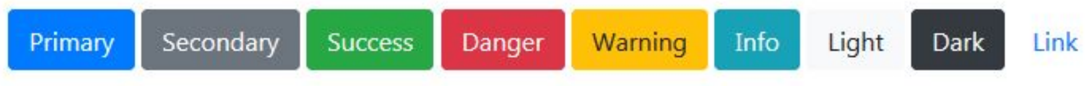
```
<div class="alert alert-success alert-dismissible">
  <button type="button" class="close" data-dismiss="alert">
      &times;
  </button>
  <strong>Success!</strong> Successful or positive action.
</div>
```

[Buttons]

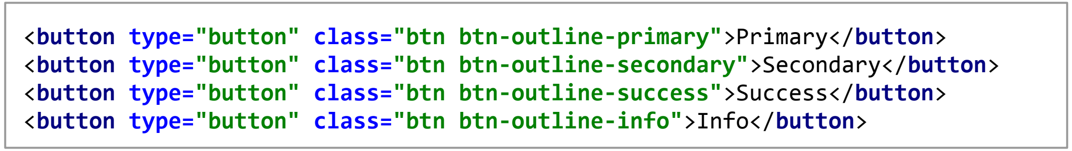
➔ Bootstrap 4 allows to style button elements (both <button> and <input type=”button”>)



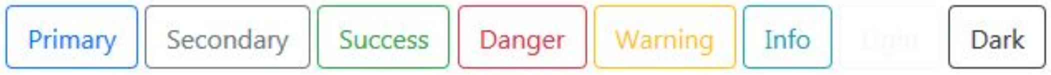
```
<button type="button" class="btn">Basic</button>
<button type="button" class="btn btn-primary">Primary</button>
<button type="button" class="btn btn-secondary">Secondary</button>
<button type="button" class="btn btn-success">Success</button>
<button type="button" class="btn btn-info">Info</button>
<input type="button" class="btn btn-info" value="Input Button">
<input type="submit" class="btn btn-info" value="Submit Button">
```



➔ All buttons also have outline versions



```
<button type="button" class="btn btn-outline-primary">Primary</button>
<button type="button" class="btn btn-outline-secondary">Secondary</button>
<button type="button" class="btn btn-outline-success">Success</button>
<button type="button" class="btn btn-outline-info">Info</button>
```



[Buttons]

➔ Buttons can have different sizes:

```
<button type="button" class="btn btn-primary btn-lg">Large</button>
<button type="button" class="btn btn-primary">Default</button>
<button type="button" class="btn btn-primary btn-sm">Small</button>
```

➔ `.btn-block` creates button that takes the entire width of the parent element

```
<button type="button" class="btn btn-primary btn-block">Full-Width Button</button>
```

➔ Button can be styled as active or disabled

➔ `.active` makes a button active, and `disabled` attribute makes button unclickable

```
<button type="button" class="btn btn-primary active">Active Primary</button>
<button type="button" class="btn btn-primary" disabled>Disabled Primary</button>
<a href="#" class="btn btn-primary disabled">Disabled Link</a>
```

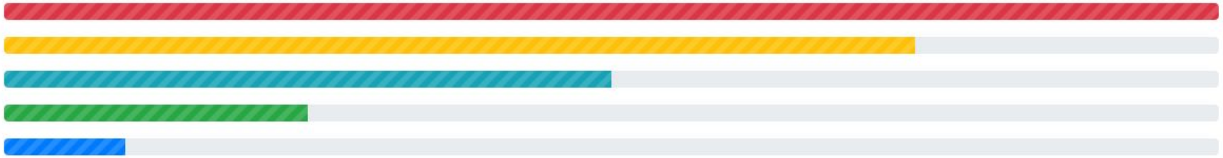
[Progress Bars]

- ➔ Bootstrap can create a progress bar
- ➔ `.progress` class sets a container element as a progression bar parent and `.progress-bar` class required to be added to the child element with `width` style used as a percentage of progression

```
<div class="progress">
  <div class="progress-bar" style="width:15%"></div>
</div>
```

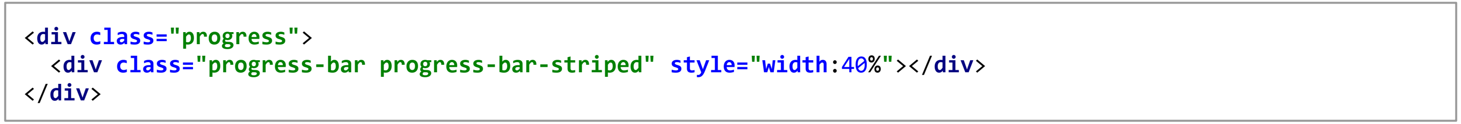
- ➔ It is possible to show the progression percentage as a test:

```
<div class="progress">
  <div class="progress-bar" style="width:50%">50%</div>
</div>
```



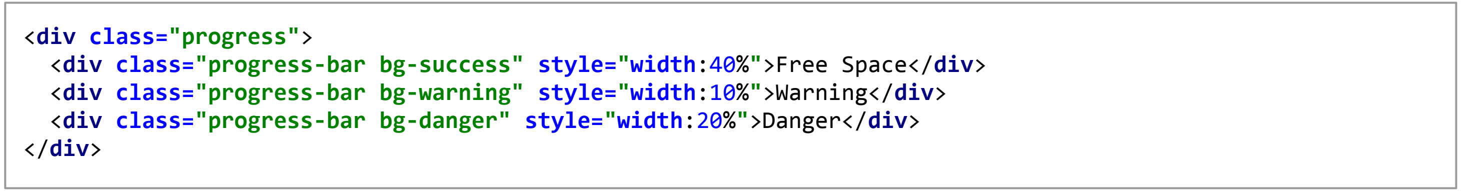
[Progress Bars]

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

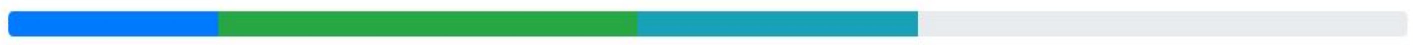


```
<div class="progress">
  <div class="progress-bar progress-bar-striped" style="width:40%"></div>
</div>
```

- ➔ Progress bars can also be stacked:



```
<div class="progress">
  <div class="progress-bar bg-success" style="width:40%">Free Space</div>
  <div class="progress-bar bg-warning" style="width:10%">Warning</div>
  <div class="progress-bar bg-danger" style="width:20%">Danger</div>
</div>
```

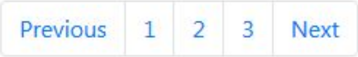


- ➔ Add `.progress-bar-animated` class to `.progress-bar` to animate the stripes right to left via CSS3 animations

[Pagination]

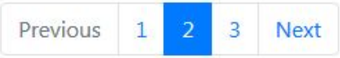
- ➔ Bootstrap allows to easily create a pagination on a site
- ➔ It can be created by adding `.pagination` class to `<ul>` element.
- ➔ `.page-item` must be added then to `<li>` elements and a `.page-link` class to each link inside `<li>`:

```
<ul class="pagination">
  <li class="page-item"><a class="page-link" href="#">Previous</a></li>
  <li class="page-item"><a class="page-link" href="#">1</a></li>
  <li class="page-item"><a class="page-link" href="#">2</a></li>
  <li class="page-item"><a class="page-link" href="#">3</a></li>
  <li class="page-item"><a class="page-link" href="#">Next</a></li>
</ul>
```



- ➔ `.active` class is used to set a page as active:

```
<li class="page-item active"><a class="page-link" href="#">2</a></li>
```



- ➔ `.disabled` class makes page unclickable:

```
<li class="page-item disabled"><a class="page-link" href="#">Previous</a></li>
```

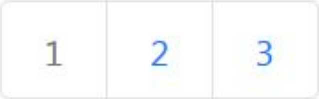
Pagination

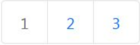
➔ Add class `.pagination-lg` for larger blocks or `.pagination-sm` for smaller blocks:

```
<ul class="pagination pagination-lg">
  <li class="page-item"><a class="page-link" href="#">Previous</a></li>
  <li class="page-item"><a class="page-link" href="#">Next</a></li>
</ul>

<ul class="pagination pagination-sm">
  <li class="page-item"><a class="page-link" href="#">Previous</a></li>
  <li class="page-item"><a class="page-link" href="#">Next</a></li>
</ul>
```

➔ Use utility classes to change the alignment of the pagination:

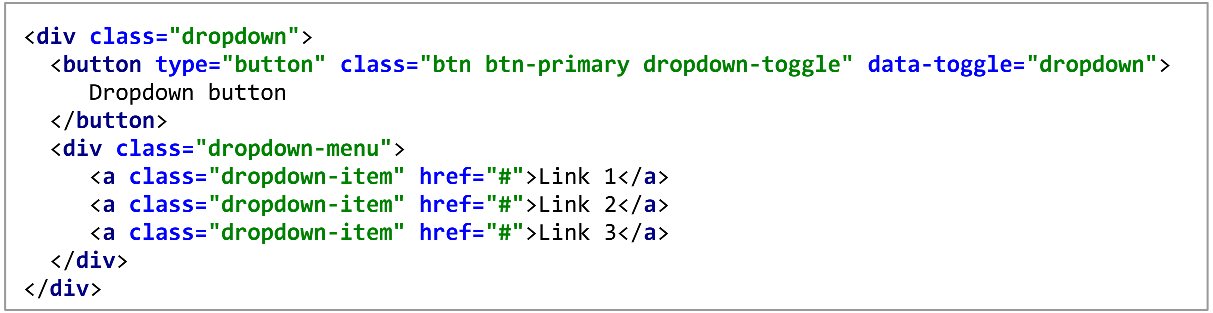
```
<ul class="pagination" style="margin:20px 0">
  <li class="page-item">...</li>
</ul>

<ul class="pagination justify-content-center" style="margin:20px 0">
  <li class="page-item">...</li>
</ul>

<ul class="pagination justify-content-end" style="margin:20px 0">
  <li class="page-item">...</li>
</ul>
```



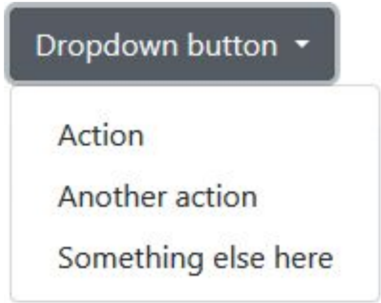


Dropdowns

- ➔ Bootstrap allows creating dropdown menus



```
<div class="dropdown">
  <button type="button" class="btn btn-primary dropdown-toggle" data-toggle="dropdown">
     Dropdown button
  </button>
  <div class="dropdown-menu">
     <a class="dropdown-item" href="#">Link 1</a>
     <a class="dropdown-item" href="#">Link 2</a>
     <a class="dropdown-item" href="#">Link 3</a>
  </div>
</div>
```



- ➔ `.dropdown` class sets a dropdown menu container
- ➔ `.dropdown-toggle` and the `data-toggle="dropdown"` attribute sets element that will open a menu
- ➔ `.dropdown-menu` class sets a dropdown menu.
- ➔ `.dropdown-item` class sets dropdown menu items

Dropdowns

- ➔ `.dropdown-divider` class is used to separate links inside the dropdown menu:

```
<div class="dropdown-divider"></div>
```

- ➔ `.dropdown-header` class is used to add headers inside the dropdown menu:

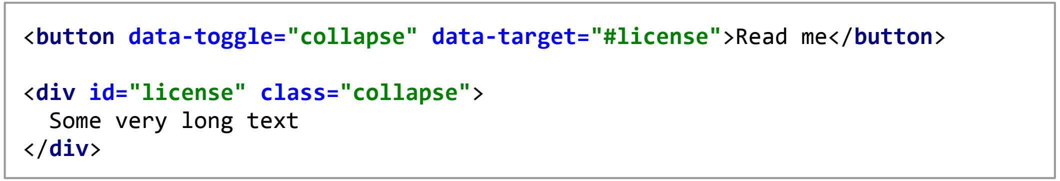
```
<div class="dropdown-header">Dropdown header 1</div>
```

- ➔ `.active` class sets dropdown menu item as active
- ➔ `.disabled` class sets dropdown menu item as disabled:

```
<a class="dropdown-item active" href="#">Active</a>
<a class="dropdown-item disabled" href="#">Disabled</a>
```

[Collapse]

➔ Collapsibles are useful when you want to hide and show large amount of content:



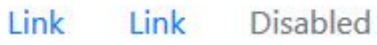
```
<button data-toggle="collapse" data-target="#license">Read me</button>

<div id="license" class="collapse">
  Some very long text
</div>
```

- `.collapse` class sets container as as collapsible element
- `data-toggle="collapse"` attribute added to the <a> or <button> elements will make them collapsible-container toggle buttons
- `data-target="#id"` or `href` (for links only) attributes sets target of the collapsible content
- By default, the collapsible content is hidden. However, you can add the .show class to show the content by default

[Navs]

- ➔ Horizontal menu can be created with a `.nav` class set to `<ul>` element
- ➔ `.nav-item` class for `<li>` elements will set them as menu items
- ➔ `.nav-link` class must be added to the links

```
<ul class="nav">
  <li class="nav-item">
     <a class="nav-link" href="#">Link</a>
  </li>
  <li class="nav-item">
     <a class="nav-link" href="#">Link</a>
  </li>
  <li class="nav-item">
     <a class="nav-link disabled" href="#">Disabled</a>
  </li>
</ul>
```



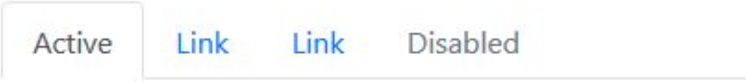
- ➔ Add the `.justify-content-center` class to center the nav, and the `.justify-content-end` class to right-align the nav

```
<ul class="nav justify-content-center">

<ul class="nav justify-content-end">
```

[Navs]

- ➔ `.nav-tabs` class makes container a navigation tab
- ➔ `.active` class sets link as selected menu item
- ➔ `.disabled` makes link unclickable

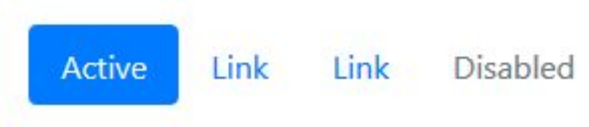
```html
<ul class="nav nav-tabs">
  <li class="nav-item">
     <a class="nav-link active" href="#">Active</a>
  </li>
  <li class="nav-item">
     <a class="nav-link" href="#">Link</a>
  </li>
  <li class="nav-item">
     <a class="nav-link" href="#">Link</a>
  </li>
  <li class="nav-item">
     <a class="nav-link disabled" href="#">Disabled</a>
  </li>
</ul>
```



[Navs]

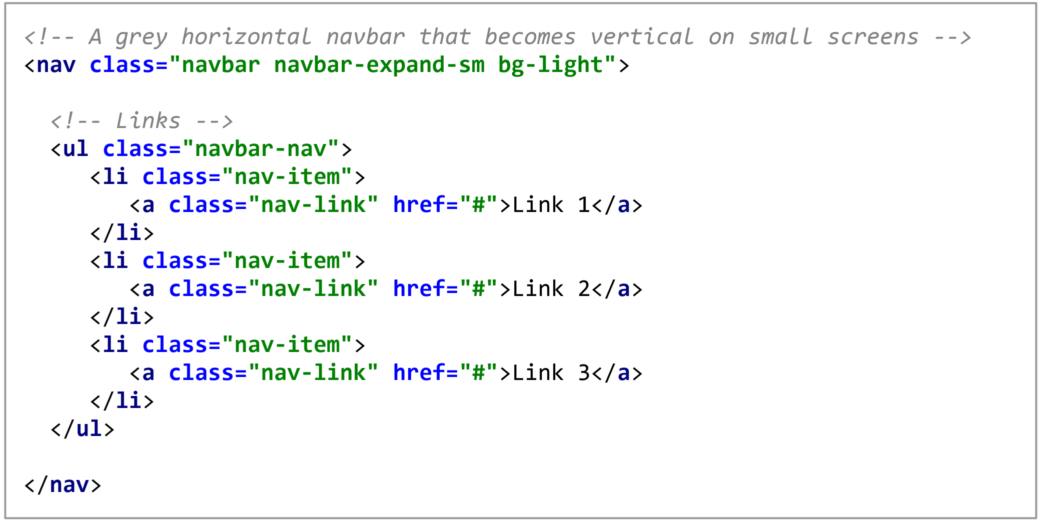
- ➔ `.nav-pills` class turns container into navigation pills

```
<ul class="nav nav-pills">
  <li class="nav-item">
     <a class="nav-link active" href="#">Active</a>
  </li>
  <li class="nav-item">
     <a class="nav-link" href="#">Link</a>
  </li>
  <li class="nav-item">
     <a class="nav-link" href="#">Link</a>
  </li>
  <li class="nav-item">
     <a class="nav-link disabled" href="#">Disabled</a>
  </li>
</ul>
```

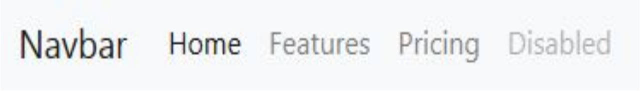


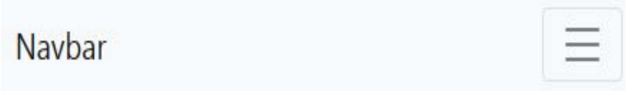
[Navbar]

- ➔ Bootstrap allows creating an adaptive navigation bars
- ➔ `.navbar` class creates navigation bar
- ➔ `.navbar-expand-xl|lg|md|sm` class stacks the navbar vertically on extra large, large, medium or small screens
- ➔ `.nav-item` class is used with `<li>` elements for menu items
- ➔ .`navbar-nav` class with `<ul>` is used for navigation links



```html
<!-- A grey horizontal navbar that becomes vertical on small screens -->
<nav class="navbar navbar-expand-sm bg-light">

  <!-- Links -->
  <ul class="navbar-nav">
     <li class="nav-item">
        <a class="nav-link" href="#">Link 1</a>
     </li>
     <li class="nav-item">
        <a class="nav-link" href="#">Link 2</a>
     </li>
     <li class="nav-item">
        <a class="nav-link" href="#">Link 3</a>
     </li>
  </ul>

</nav>
```





[Forms]

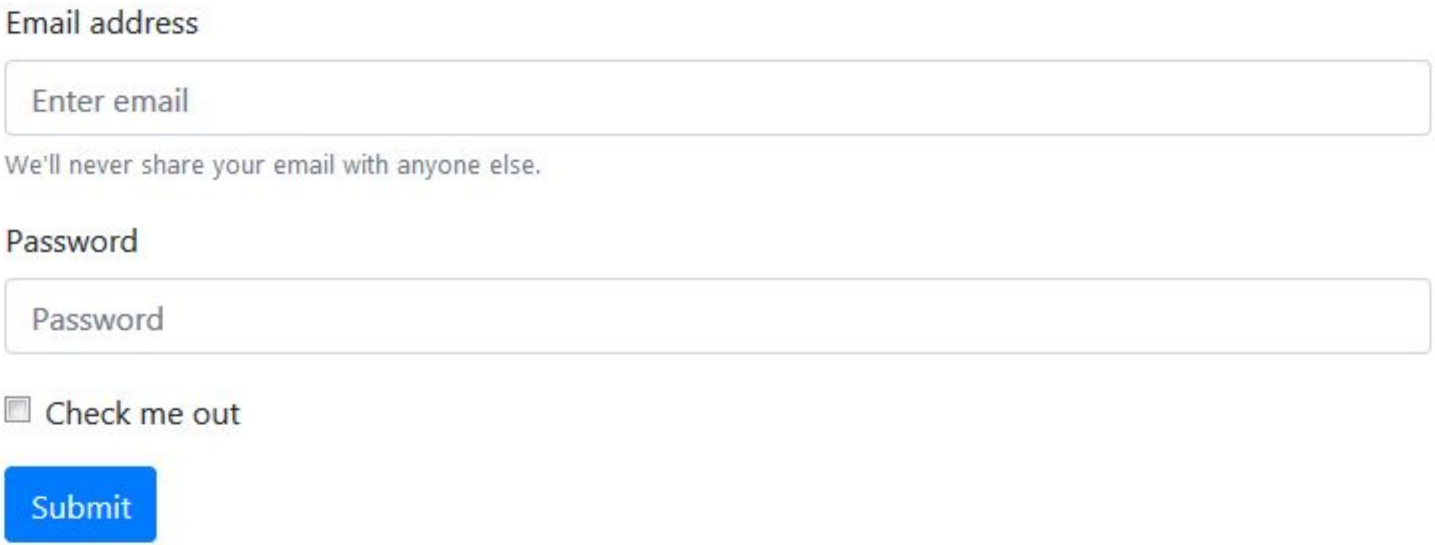
- ➔ Bootstrap can also style forms
- ➔ `<input>`, `<textarea>`, and `<select>` elements with class `.form-control` have a width of 100%
- ➔ Bootstrap provides two types of form layouts:
  - → Stacked (full-width) form
  - → Inline form

Stacked Form

- ➔ Forms are stacked by default
- ➔ `.form-group` class sets margins for the form elements:

```html
<form>
  <div class="form-group">
     <label for="exampleInputEmail1">Email address</label>
     <input type="email" class="form-control" id="exampleInputEmail1" aria-describedby="emailHelp" placeholder="Enter email">
     <small id="emailHelp" class="form-text text-muted">We'll never share your email with anyone else.</small>
  </div>
  <div class="form-group">
     <label for="exampleInputPassword1">Password</label>
     <input type="password" class="form-control" id="exampleInputPassword1" placeholder="Password">
  </div>
  <div class="form-group form-check">
     <input type="checkbox" class="form-check-input" id="exampleCheck1">
     <label class="form-check-label" for="exampleCheck1">Check me out</label>
  </div>
  <button type="submit" class="btn btn-primary">Submit</button>
</form>
```

Stacked Form



Email address

Enter email

We'll never share your email with anyone else.

Password

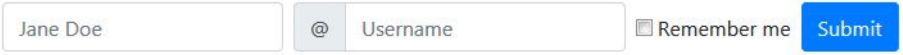
Password

□ Check me out

[Inline Form]

- ➔ .form-inline class will create an inline form that will align input elements horizontally
- ➔ If screen size will be less that 576px, form will become usual vertical form

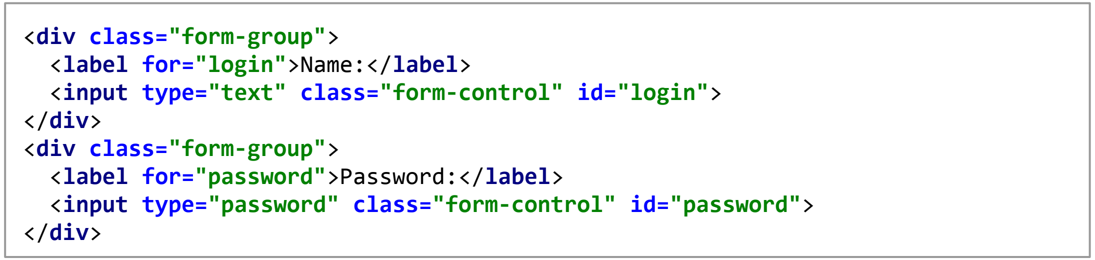
Inline Form



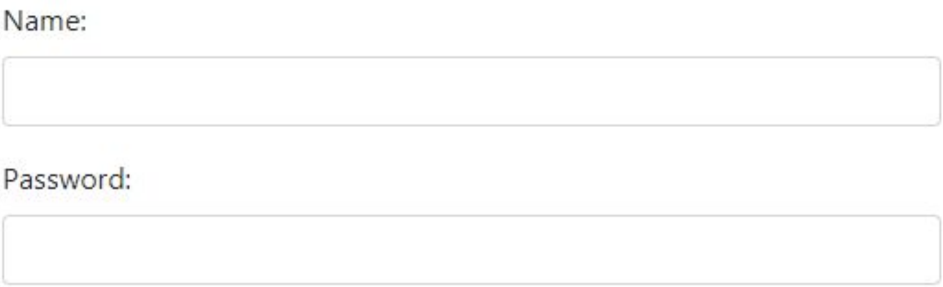
```
<form class="form-inline">
  <label class="sr-only" for="inlineFormInputName2">Name</label>
  <input type="text" class="form-control mb-2 mr-sm-2" id="inlineFormInputName2" placeholder="Jane Doe">

  <label class="sr-only" for="inlineFormInputGroupUsername2">Username</label>
  <div class="input-group mb-2 mr-sm-2">
     <div class="input-group-prepend">
        <div class="input-group-text">@</div>
     </div>
     <input type="text" class="form-control" id="inlineFormInputGroupUsername2" placeholder="Username">
  </div>

  <div class="form-check mb-2 mr-sm-2">
     <input class="form-check-input" type="checkbox" id="inlineFormCheck">
     <label class="form-check-label" for="inlineFormCheck">
        Remember me
     </label>
  </div>

  <button type="submit" class="btn btn-primary mb-2">Submit</button>
</form>
```

Form Inputs

- ➔ Bootstrap supports such input elements like input, textarea, checkbox, radio and select
- ➔ And HTML5 input types like text, password, datetime, datetime-local, date, month, time, week, number, email, url, search, tel, and color as well.



```html
<div class="form-group">
  <label for="login">Name:</label>
  <input type="text" class="form-control" id="login">
</div>
<div class="form-group">
  <label for="password">Password:</label>
  <input type="password" class="form-control" id="password">
</div>
```



Name:

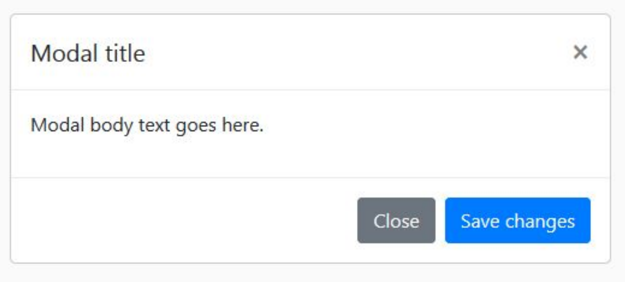
Password:

[Modal]

- ➔ You can create a modal window using Bootstrap
- ➔ It can be used to add dialogs to your web-site for lightboxes, user notifications, or completely custom content
  - → Modals are built with HTML, CSS, and JavaScript. They’re positioned over everything else in the document and remove scroll from the <body> so that modal content scrolls instead
  - → Clicking on the modal “backdrop” will automatically close the modal
  - → Bootstrap only supports one modal window at a time. Nested modals aren’t supported as they are believed to be a poor user experience

[Modal]

```
<div class="modal" tabindex="-1" role="dialog">
  <div class="modal-dialog" role="document">
     <div class="modal-content">
        <div class="modal-header">
           <h5 class="modal-title">Modal title</h5>
           <button type="button" class="close" data-dismiss="modal"
aria-label="Close">
              <span aria-hidden="true">&times;</span>
           </button>
        </div>
        <div class="modal-body">
           <p>Modal body text goes here.</p>
        </div>
        <div class="modal-footer">
           <button type="button" class="btn btn-secondary"
data-dismiss="modal">Close</button>
           <button type="button" class="btn btn-primary">Save
changes</button>
        </div>
     </div>
  </div>
</div>
```



Modal title

Modal body text goes here.

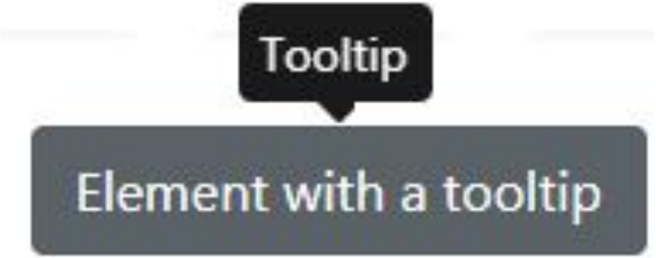
[Tooltips]

- ➔ Tooltips are small pop-ups that is shown when user hovers an element
- ➔ Tooltips rely on the **Popper.js** for positioning. You must include **popper.min.js** before bootstrap.js or use `bootstrap.bundle.min.js` / `bootstrap.bundle.js` which contains **Popper.js** in order for tooltips to work!
- ➔ Tooltips are opt-in for performance reasons, so you must initialize them yourself
- ➔ Tooltips with zero-length titles are never displayed
- ➔ Specify `container: 'body'` to avoid rendering problems in more complex components (like our input groups, button groups, etc)
- ➔ Triggering tooltips on hidden elements will not work
- ➔ Tooltips for `.disabled` or `disabled` elements must be triggered on a wrapper element
- ➔ When triggered from hyperlinks that span multiple lines, tooltips will be centered. Use `white-space: nowrap;` on your `<a>` to avoid this behavior
- ➔ Tooltips must be hidden before their corresponding elements have been removed from the DOM

[Tooltips]

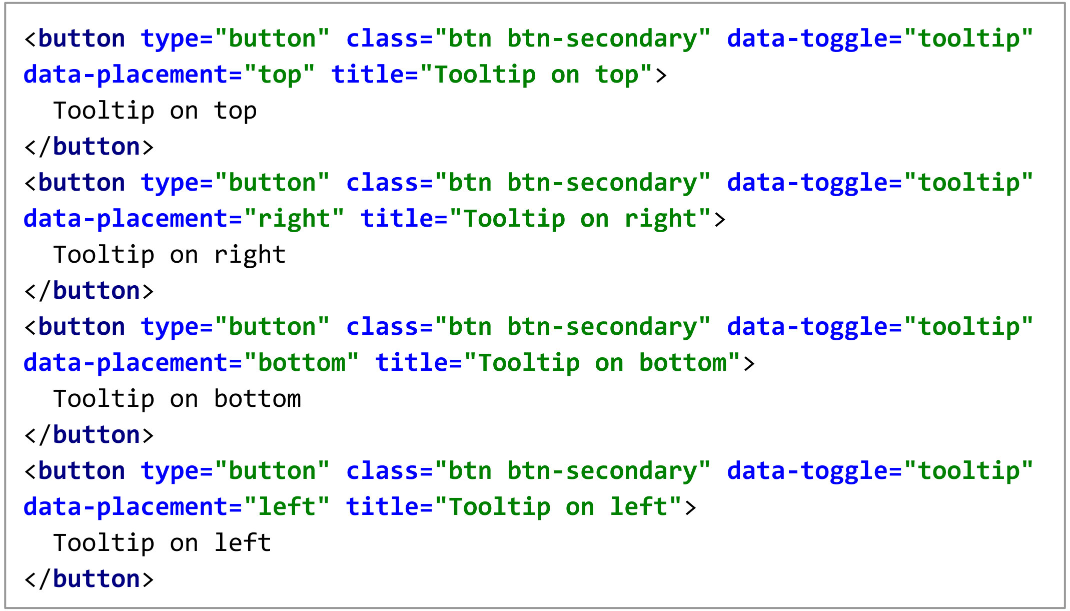
- ➔ One way to initialize all tooltips on a page would be to select them by their `data-toggle` attribute:

```
$(function () {
   $('[data-toggle="tooltip"]').tooltip()
})
```



[Tooltips]

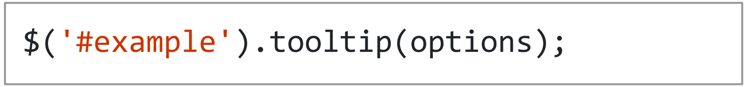
- ➔ Here are some examples:



```
<button type="button" class="btn btn-secondary" data-toggle="tooltip"
data-placement="top" title="Tooltip on top">
  Tooltip on top
</button>
<button type="button" class="btn btn-secondary" data-toggle="tooltip"
data-placement="right" title="Tooltip on right">
  Tooltip on right
</button>
<button type="button" class="btn btn-secondary" data-toggle="tooltip"
data-placement="bottom" title="Tooltip on bottom">
  Tooltip on bottom
</button>
<button type="button" class="btn btn-secondary" data-toggle="tooltip"
data-placement="left" title="Tooltip on left">
  Tooltip on left
</button>
```

[Tooltips]

- ➔ The tooltip plugin generates content and markup on demand, and by default places tooltips after their trigger element
- ➔ Trigger the tooltip via JavaScript:



```
$('#example').tooltip(options);
```

[Tooltips]

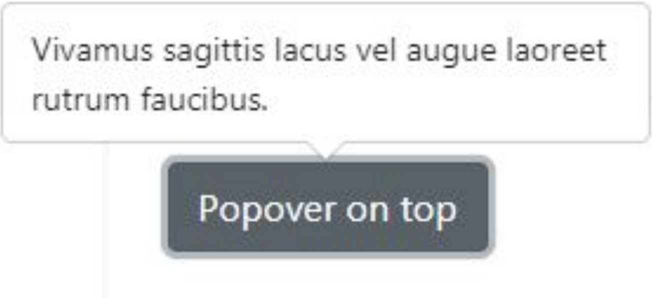
- ➔ The required markup for a tooltip is only a `data` attribute and `title` on the HTML element you wish to have a tooltip. The generated markup of a tooltip is rather simple, though it does require a position (by default, set to `top` by the plugin)

```html
<!-- HTML to write -->
<a href="#" data-toggle="tooltip" title="Some tooltip text!">Hover over me</a>

<!-- Generated markup by the plugin -->
<div class="tooltip bs-tooltip-top" role="tooltip">
  <div class="arrow"></div>
  <div class="tooltip-inner">
     Some tooltip text!
  </div>
</div>
```

[Popovers]

- ➔ The Popover component is similar to tooltips; it is a pop-up box that appears when the user clicks on an element. The difference is that the popover can contain much more content
- ➔ One way to initialize all popovers on a page would be to select them by their `data-toggle` attribute:

```
$(function () {
   $('[data-toggle="popover"]').popover()
})
```



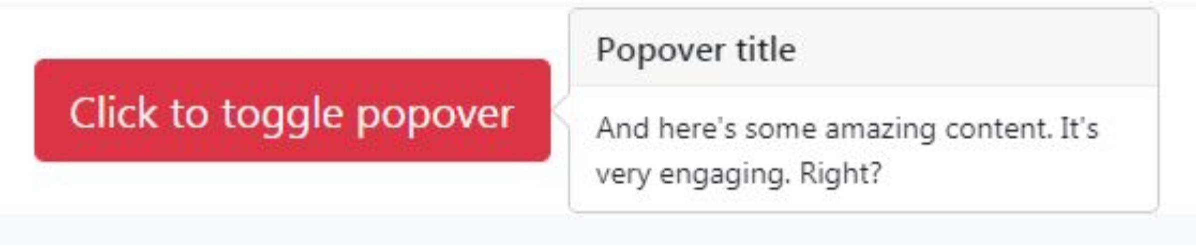
Vivamus sagittis lacus vel augue laoreet rutrum faucibus.

[Popovers]

- ➔ When you have some styles on a parent element that interfere with a popover, you'll want to specify a custom `container` so that the popover's HTML appears within that element instead

```
$(function () {

$('.example-popover').popover({
        container: 'body'
   })
})
```

```
<button type="button" class="btn btn-lg btn-danger"
data-toggle="popover" title="Popover title"
data-content="And here's some amazing content. It's very
engaging. Right?">Click to toggle popover</button>
```



Popover title

And here's some amazing content. It's very engaging. Right?

[Popovers]

➔ Use the `focus` trigger to dismiss popovers on the user’s next click of a different element than the toggle element

```
<a tabindex="0" class="btn btn-lg btn-danger" role="button" data-toggle="popover"
data-trigger="focus" title="Dismissible popover" data-content="And here's some amazing content.
It's very engaging. Right?">Dismissible popover</a>
```

```
$('.popover-dismiss').popover
({
   trigger: 'focus'
})
```

[Popovers]

- ➔ Elements with the `disabled` attribute aren’t interactive, meaning users cannot hover or click them to trigger a popover (or tooltip). As a workaround, you’ll want to trigger the popover from a wrapper `<div>` or `<span>` and override the `pointer-events` on the disabled element
- ➔ For disabled popover triggers, you may also prefer `data-trigger="hover"` so that the popover appears as immediate visual feedback to your users as they may not expect to *click* on a disabled element

```
<span class="d-inline-block" data-toggle="popover" data-content="Disabled popover">
 <button class="btn btn-primary" style="pointer-events: none;" type="button" disabled>
Disabled button
 </button>
</span>
```

- ➔ Enable popovers via JavaScript:

```
$('#example').popover(options)
```

[Carousel]

- The carousel is a slideshow for cycling through a series of content, built with CSS 3D transforms and a bit of JavaScript. It works with a series of images, text, or custom markup. It also includes support for previous/next controls and indicators
- Carousels don’t automatically normalize slide dimensions. As such, you may need to use additional utilities or custom styles to appropriately size content. While carousels support previous/next controls and indicators, they’re not explicitly required. Add and customize as you see fit
- The `.active` class needs to be added to one of the slides otherwise the carousel will not be visible. Also be sure to set a unique id on the `.carousel` for optional controls, especially if you’re using multiple carousels on a single page. Control and indicator elements must have a `data-target` attribute (or `href` for links) that matches the id of the `.carousel` element

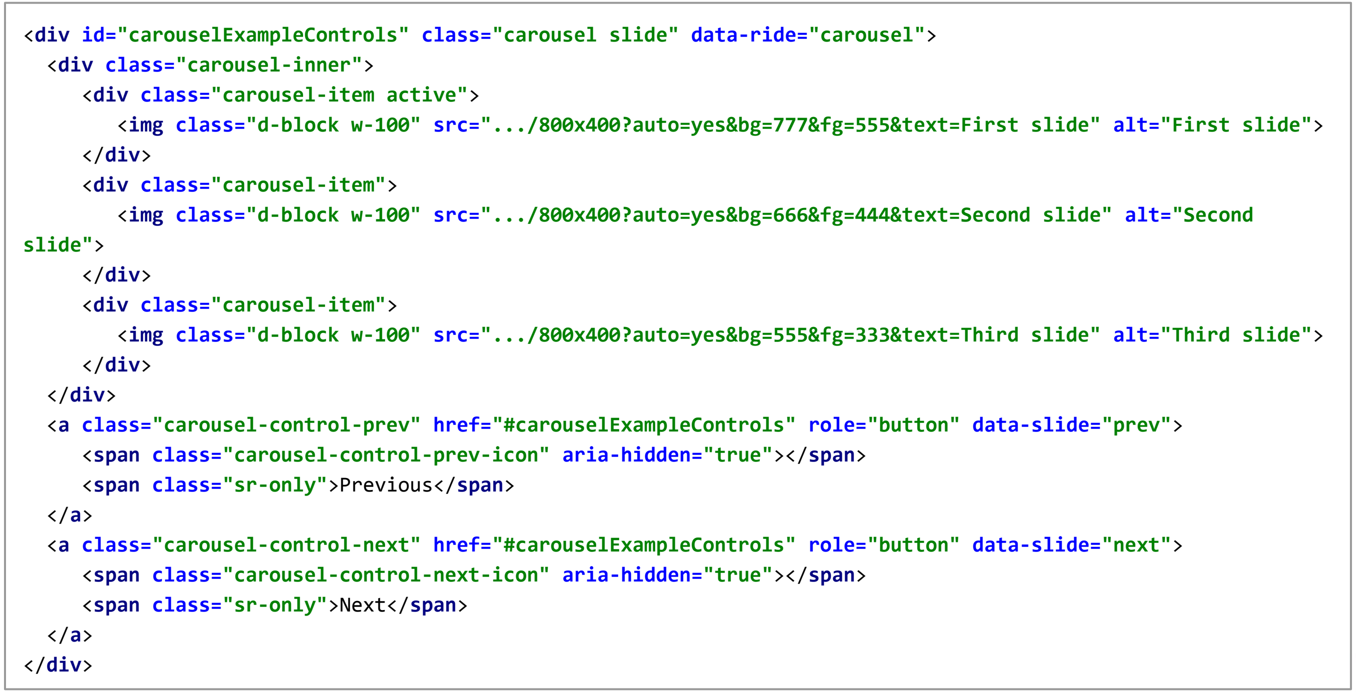
Carousel

- ➔ Here’s a carousel with slides only. Note the presence of the `.d-block` and `.w-100` on carousel images to prevent browser default image alignment

```
<div id="carouselExampleSlidesOnly" class="carousel slide" data-ride="carousel">
  <div class="carousel-inner">
     <div class="carousel-item active">
        <img class="d-block w-100"
src=".../800x400?auto=yes&bg=777&fg=555&text=First slide" alt="First slide">
     </div>
     <div class="carousel-item">
        <img class="d-block w-100"
src=".../800x400?auto=yes&bg=666&fg=444&text=Second slide" alt="Second slide">
     </div>
     <div class="carousel-item">
        <img class="d-block w-100"
src=".../800x400?auto=yes&bg=555&fg=333&text=Third slide" alt="Third slide">
     </div>
  </div>
</div>
```

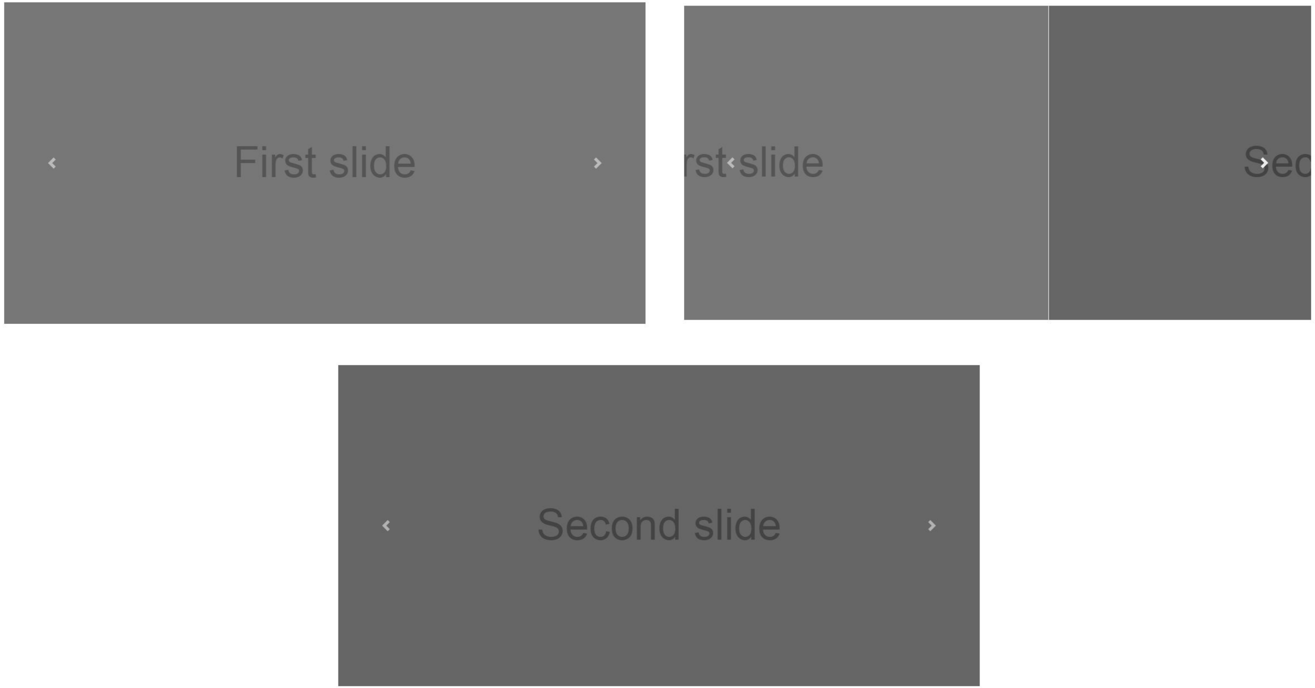
[Carousel]

- ➔ Adding in the previous and next controls:



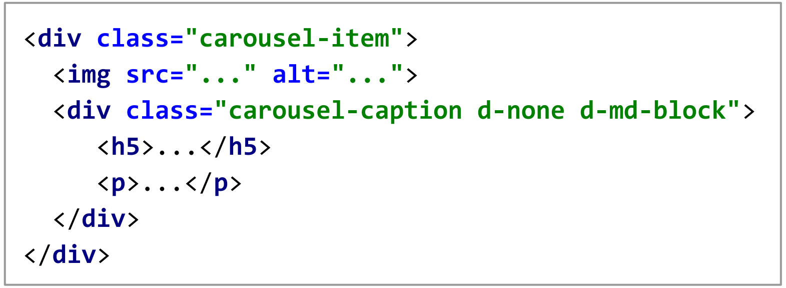
```
<div id="carouselExampleControls" class="carousel slide" data-ride="carousel">
  <div class="carousel-inner">
     <div class="carousel-item active">
        <img class="d-block w-100" src=".../800x400?auto=yes&bg=777&fg=555&text=First slide" alt="First slide">
     </div>
     <div class="carousel-item">
        <img class="d-block w-100" src=".../800x400?auto=yes&bg=666&fg=444&text=Second slide" alt="Second
slide">
     </div>
     <div class="carousel-item">
        <img class="d-block w-100" src=".../800x400?auto=yes&bg=555&fg=333&text=Third slide" alt="Third slide">
     </div>
  </div>
  <a class="carousel-control-prev" href="#carouselExampleControls" role="button" data-slide="prev">
     <span class="carousel-control-prev-icon" aria-hidden="true"></span>
     <span class="sr-only">Previous</span>
  </a>
  <a class="carousel-control-next" href="#carouselExampleControls" role="button" data-slide="next">
     <span class="carousel-control-next-icon" aria-hidden="true"></span>
     <span class="sr-only">Next</span>
  </a>
</div>
```

Carousel



Carousel

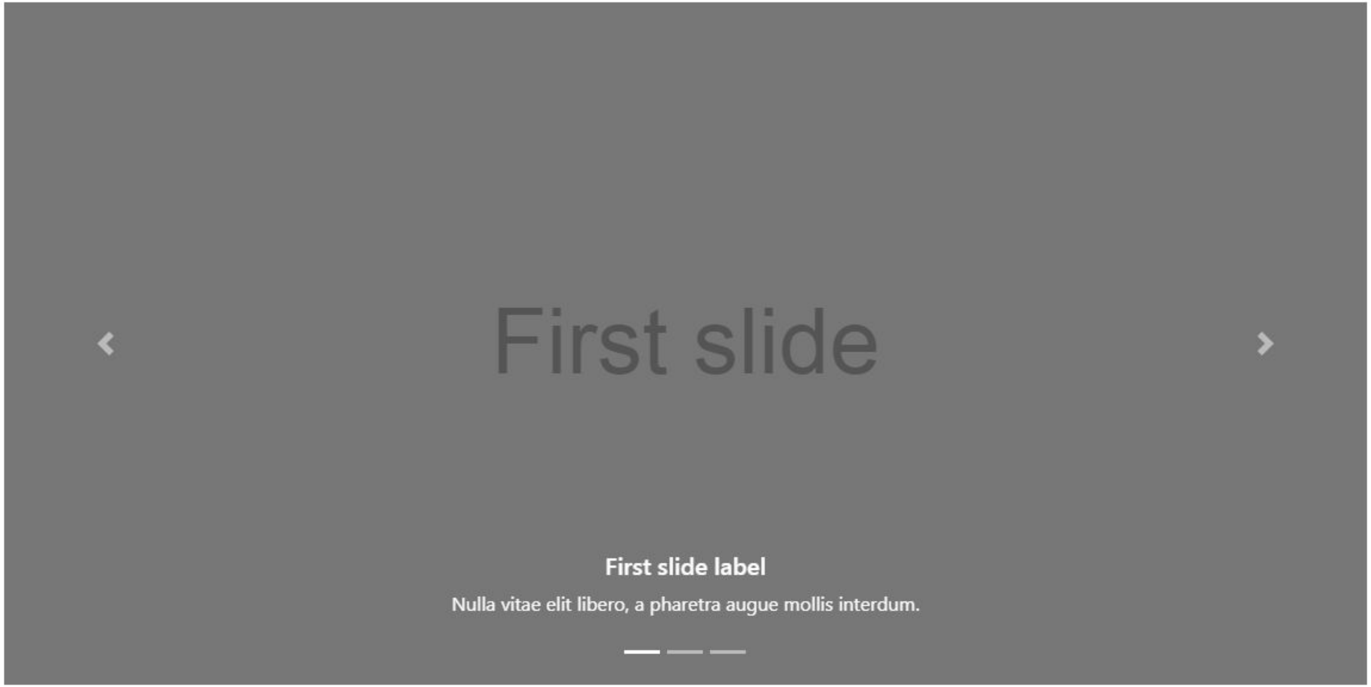
- ➔ Add captions to your slides easily with the `.carousel-caption` element within any `.carousel-item`. They can be easily hidden on smaller viewports, as shown below, with optional **display utilities**. We hide them initially with `.d-none` and bring them back on medium-sized devices with `.d-md-block`



```
<div class="carousel-item">
  <img src="..." alt="...">
  <div class="carousel-caption d-none d-md-block">
     <h5>...</h5>
     <p>...</p>
  </div>
</div>
```

- ➔ Add `.carousel-fade` to your carousel to animate slides with a fade transition instead of a slide

Carousel



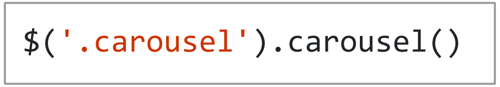
First slide

First slide label

Nulla vitae elit libero, a pharetra augue mollis interdum.

Carousel

- Use data attributes to easily control the position of the carousel. `data-slide` accepts the keywords `prev` or `next`, which alters the slide position relative to its current position. Alternatively, use `data-slide-to` to pass a raw slide index to the carousel `data-slide-to="2"`, which shifts the slide position to a particular index beginning with `0`
- The `data-ride="carousel"` attribute is used to mark a carousel as animating starting at page load. **It cannot be used in combination with (redundant and unnecessary) explicit JavaScript initialization of the same carousel**
- You can as well call carousel manually with JavaScript:



```
$('.carousel').carousel()
```

[And more]

- ➔ These are only some of the features that Bootstrap provides
- ➔ Whole list of Bootstrap features will be too big for one presentation
- ➔ Much more can be found at Bootstrap official documentation https://getbootstrap.com/docs

[Control questions]

1. Which components (files) does Bootstrap 4 include?
2. What kind of tasks does Bootstrap solve?
3. How can we make Bootstrap 4 responsive?
4. Name some features provided by Bootstrap 4.
5. Can we use Bootstrap to create simple scripting tasks like modal windows and popups?