1.1 - Welcome to Front-End Web UI Frameworks and Tools: Bootstrap 4: Additional Resources

Bootstrap Resources

- Bootstrap Site

Coursera Resources

- Coursera Learner Help
- Switching to a Different Session

1.2 - Full Stack Web Development: Additional Resources

PDFs of Presentations

FSWD-BigPicture.pdf

Useful Links

- What is a Full Stack developer?
- Wait, Wait… What is a Full-stack Web Developer After All?
- The Myth of the Full-stack Developer
- Multi-tier Architecture
- What is the 3-Tier Architecture?

2.1 - Exercise (Instructions): Setting up Git

Objectives and Outcomes

In this exercise you will learn to install Git on your computer. Git is required for using all the remaining Node.js and Node based tools that we encounter in the rest of the course. At the end of this exercise, you would be able to:

- Install Git on your computer
- Ensure that Git can be used from the command-line or command-prompt on your computer
- Set up some of the basic global configuration for Git

Downloading and Installing Git

- To install Git on your computer, go to https://git-scm.com/downloads to download the Git installer for your specific computing platform.
- Then, follow the installation steps as you install Git using the installer.
- You can find more details about installing Git at https://git-scm.com/book/en/v2/Getting-Started-Installing-Git. This document lists several ways of installing Git on various platforms.
- Installing some of the GUI tools like GitHub Desktop will also install Git on your computer.
- On a Mac, setting up XCode command-line tools also will set up Git on your computer.
- You can choose any of the methods that is most convenient for you.

Some Global Configuration for Git

- Open a cmd window or terminal on your computer.
- Check to make sure that Git is installed and available on the command line, by typing the following at the command prompt:

```
git --version
```

- To configure your user name to be used by Git, type the following at the prompt:

```
git config --global user.name "Your Name"
```

- To configure your email to be used by Git, type the following at the prompt:

```
git config --global user.email <your email address>
```

- You can check your default Git global configuration, you can type the following at the prompt:

```
git config --list
```

Conclusions

At the end of this exercise you should have Git available on the command-line of your computer.

2.2 - Exercise (Instructions): Basic Git Commands

Objectives and Outcomes

In this exercise you will get familiar with some basic Git commands. At the end of this exercise you will be able to:

- Set up a folder as a Git repository
- Perform basic Git operations on your Git repository

Basic Git Commands

- At a convenient location on your computer, create a folder named **git-test**.
- Open this git-test folder in your favorite editor.
- Add a file named *index.html* to this folder, and add the following HTML code to this file:

```
<!DOCTYPE html>

<html>

  <head></head>



  <body>

    <h1>This is a Header</h1>

  </body>

</html>
```

Initializing the folder as a Git repository

- Go to the git-test folder in your cmd window/terminal and type the following at the prompt to initialize the folder as a Git repository:

```
git init
```

Checking your Git repository status

- Type the following at the prompt to check your Git repository's status:

```
git status
```

Adding files to the staging area

- To add files to the staging area of your Git repository, type:

```
git add .
```

Commiting to the Git repository

- To commit the current staging area to your Git repository, type:

```
git commit -m "first commit"
```

Checking the log of Git commits

- To check the log of the commits to your Git repository, type

```
git log --oneline
```

- Now, modify the *index.html* file as follows:

```
<!DOCTYPE html>

<html>

  <head></head>

  <body>

    <h1>This is a Header</h1>

    <p>This is a paragraph</p>

  </body>

</html>
```

- Add a sub-folder named **templates** to your **git-test** folder, and then add a file named *test.html* to the templates folder. Then set the contents of this file to be the same as the *index.html* file above.
- Then check the status and add all the files to the staging area.
- Then do the second commit to your repository
- Now, modify the *index.html* file as follows:

```
<!DOCTYPE html>

<html>

    <head></head>

    <body>

        <h1>This is a Header</h1>

        <p>This is a paragraph</p>

        <p>This is a second paragraph</p>

    </body>

</html>
```

- Now add the modified index.html file to the staging area and then do a third commit.

Checking out a file from an earlier commit

- To check out the index.html from the second commit, find the number of the second commit using the git log, and then type the following at the prompt:

```
git checkout <second commit's number> index.html
```

Resetting the Git repository

- To discard the effect of the previous operation and restore index.html to its state at the end of the third commit, type:

```
git reset HEAD index.html
```

- Then type the following at the prompt:

```
git checkout -- index.html
```

- You can also use *git reset* to reset the staging area to the last commit without disturbing the working directory.

Conclusions

At the end of this exercise you should have learnt some basic Git commands. Experiment with these commands until you fully understand how to use Git.

2.3 - Exercise (Instructions): Online Git Repositories

Objectives and Outcomes

In this exercise you will learn about how to set up and use an online Git repository and synchronize your local Git repository with your online repository. At the end of this exercise, you will be able to:

- Set up the online repository as a remote repository for your local Git repository
- Push your commits to the online repository
- Clone an online Git repository to your computer

Setting up an Online Git repository

- Sign up for an account either at Bitbucket (https://bitbucket.org) or GitHub (https://github.com).
- Then set up an online Git repository named **git-test**. Note the URL of your online Git repository. Note that private repositories on GitHub requires a paid account, and is not available for free accounts.

Set the local Git repository to set its remote origin

- At the prompt, type the following to set up your local repository to link to your online Git repository:

```
git remote add origin <repository URL>
```

Pushing your commits to the online repository

- At the prompt, type the following to push the commits to the online repository:

```
git push -u origin master
```

Cloning an online repository

- To clone an online repository to your computer, type the following at the prompt:

```
git clone <repository URL>
```

Conclusions

In this exercise you have learnt to set up an online Git repository, synchronize your local repository with the remote repository, and clone an online repository.

2.5 - Exercise (Instructions): Setting up Node.js and NPM

Note: Make sure you have installed Git on your machine before you install Node.js. Please complete the previous Git installation exercise before proceeding with this exercise.

Objectives and Outcomes

In this exercise, you will learn to set up the Node.js environment, a popular Javascript based server framework, and node package manager (NPM) on your machine. To learn more about NodeJS, you can visit https://nodejs.org. For this course, you just need to install Node.js on your machine and make use of it for running some front-end tools. You will learn more about the server-side support using Node.js in a subsequent course. At the end of this exercise, you will be able to:

- Complete the set up of Node.js and NPM on your machine
- Verify that the installation was successful and your machine is ready for using Node.js and NPM.

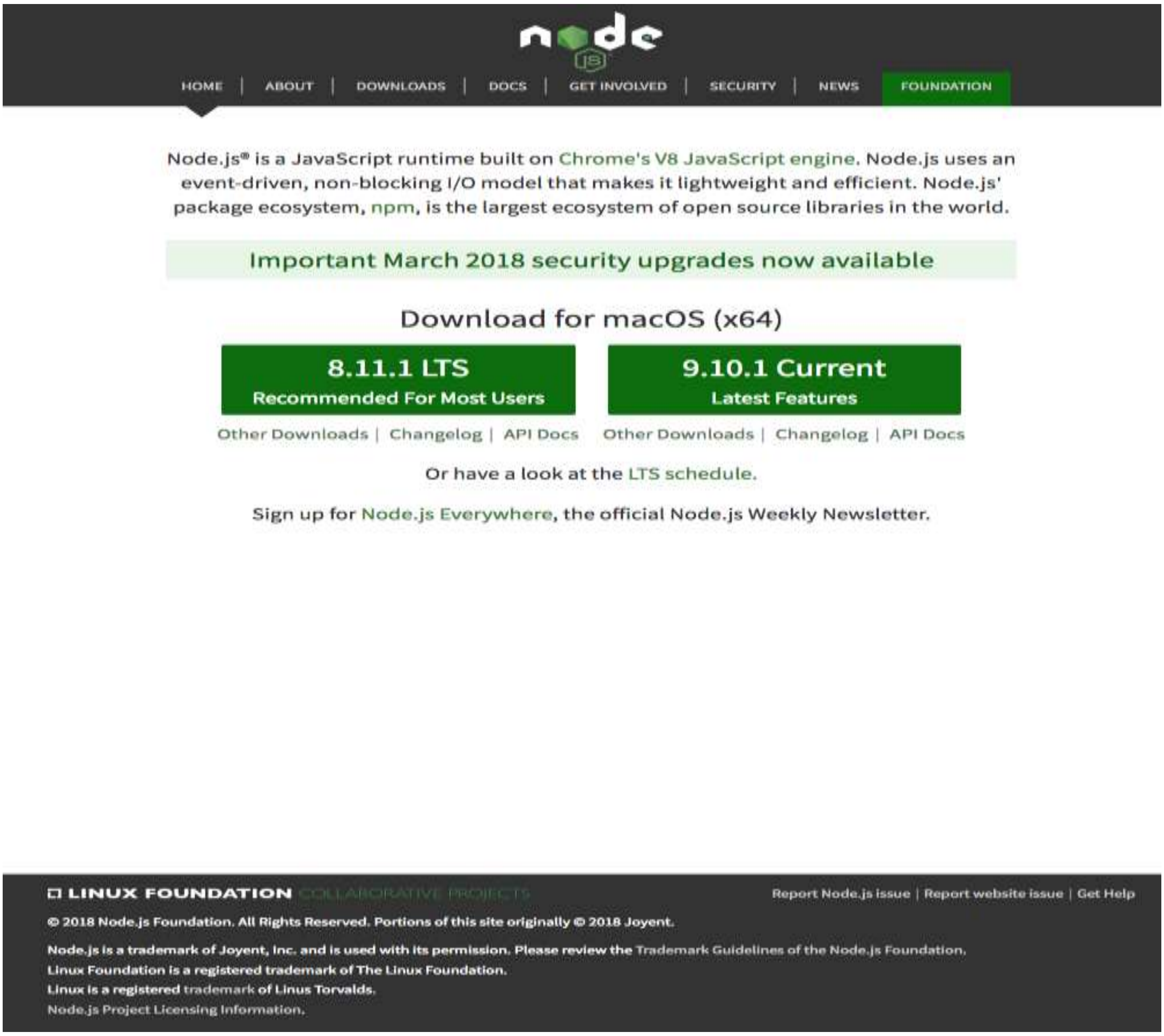
Installing Node

- To install Node on your machine, go to https://nodejs.org and click on the Download button. Depending on your computer's platform (Windows, MacOS or Linux), the appropriate installation package is downloaded.
- As an example, on a Mac, you will see the following web page. Click on the Download button. Follow along the instructions to install Node on your machine. (Note: Now Node gives you the option of installing a mature and dependable LTS version and a more newer stable version. You should to install the LTS version. I will use this version in the course.)

Note: On Windows machines, you may need to configure your PATH environmental variable in case you forgot to turn on the add to PATH during the installation steps.

Verifying the Node Installation

- Open a terminal window on your machine. If you are using a Windows machine, open a cmd window or PowerShell window with **admin** privileges.
- To ensure that your NodeJS setup is working correctly, type the following at the command prompt to check for the version of **Node** and **NPM**


node JS
HOME | ABOUT | DOWNLOADS | DOCS | GET INVOLVED | SECURITY | NEWS | FOUNDATION

Node.js® is a JavaScript runtime built on Chrome's V8 JavaScript engine. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient. Node.js' package ecosystem, npm, is the largest ecosystem of open source libraries in the world.

Important March 2018 security upgrades now available

Download for macOS (x64)

8.11.1 LTS
Recommended For Most Users
9.10.1 Current
Latest Features
Other Downloads | Changelog | API Docs
Other Downloads | Changelog | API Docs

Or have a look at the LTS schedule.

Sign up for Node.js Everywhere, the official Node.js Weekly Newsletter.

LINUX FOUNDATION COLLABORATIVE PROJECTS
Report Node.js issue | Report website issue | Get Help

© 2018 Node.js Foundation. All Rights Reserved. Portions of this site originally © 2018 Joyent.

Node.js is a trademark of Joyent, Inc. and is used with its permission. Please review the Trademark Guidelines of the Node.js Foundation.
Linux Foundation is a registered trademark of The Linux Foundation.
Linux is a registered trademark of Linus Torvalds.
Node.js Project Licensing Information.

```
node -v

npm -v
```

Conclusions

At the end of this exercise, your machine is now ready with the Node installed for further development. We will examine web development tools next.

2.6 - Exercise (Instructions): Basics of Node.js and NPM

Objectives and Outcomes

In this exercise you will learn the basics of Node and NPM. At the end of this exercise, you will be able to:

- Set up package.json file in the project folder for configuring your Node and NPM for this project
- Install a NPM module and make use of it within your project

Initializing package.json

- At the command prompt in your **git-test** folder, type

```
npm init
```

- Follow along the prompts and answer the questions as follows: accept the default values for most of the entries, except set the entry point to index.html
- This should create a *package.json* file in your **git-test** folder.

Installing an NPM Module

- Install an NPM module, lite-server, that allows you to run a Node.js based development web server and serve up your project files. To do this, type the following at the prompt:

```
npm install lite-server --save-dev
```

- You can check out more documentation on lite-server here.
- Next, open package.json in your editor and modify it as shown below. Note the addition of two lines, line 7 and line 9.

```
{
  "name": "git-test",
  "version": "1.0.0",
  "description": "This is the Git and Node basic learning project",
  "main": "index.html",
  "scripts": {
    "start": "npm run lite",
    "test": "echo \"Error: no test specified\" && exit 1",
    "lite": "lite-server"

  },

  "repository": {

    "type": "git",

    "url": "git+https://jogesh_k_muppala@bitbucket.org/jogesh_k_muppala/git-test.git"

  },

  "author": "",

  "license": "ISC",

  "homepage": "https://bitbucket.org/jogesh_k_muppala/git-test#readme",

  "devDependencies": {

    "lite-server": "^2.2.2"

  }

}
```

- Next, start the development server by typing the following at the prompt:

```
npm start
```

- This should open your *index.html* page in your default browser.
- If you now open the *index.html* page in an editor and make changes and save, the browser should immediately refresh to reflect the changes.

Setting up .gitignore

- Next, create a file in your project directory named *.gitignore* (**Note**: the name starts with a period)Then, add the following to the .gitignore file

```
node_modules
```

- Then do a git commit and push the changes to the online repository. You will note that the node_modules folder will not be added to the commit, and will not be uploaded to the repository.

Conclusions

In this exercise you learnt to set up package.json, install a npm package and start a development server.

2.7 - Setting up your Development Environment: Git and Node: Additional Resources

PDFs of Presentations

Git.pdf

Git-Exercises.pdf

NodeJS.pdf

Exercises-Node-NPM.pdf

Additional Resources (Git)

- Git site http://git-scm.com.
- Installing Git chapter from Pro Git
- Git reference manual
- Quick reference guides: GitHub Cheat Sheet (PDF) | Visual Git Cheat Sheet (SVG | PNG)
- Atlassian comprehensive Git tutorial

Additional Resources (Node.js and NPM)

- Nodejs.org
- Npmjs.com
- Node API Documentation
- NPM Documentation
- lite-server

3.1 - Exercise: Getting Started with Bootstrap

Exercise Resources

Bootstrap4-starter.zip

Objectives and Outcomes

This exercise introduces the first set of steps to set up your web page to make use of Bootstrap classes and components. At the end of this exercise, you will be able to:

- Download Bootstrap using NPM and include it in your project
- Understand how to set up a web project to use Bootstrap
- Include the Bootstrap CSS and JS classes into a web page

Note: Please remember to retain the folder and all the files that you create in this exercise. Further exercises will build upon the files that you create in this exercise. DO NOT DELETE the files at the end of the exercise.

Setting up the Project Folder

- Go to a convenient folder location on your computer and download the ***Bootstrap4-starter.zip*** file using the link provided at the top of this page.
- Unzip the file to see a folder named ***Bootstrap4*** and a sub-folder under it named ***conFusion*** created. Move to the *conFusion* folder.
- Open a cmd window/terminal and move to the conFusion folder.
- At the prompt type

```
npm install
```

- This will install the lite-server node module to your project.
- Next, initialize a Git repository in the project folder, and then set up a .gitignore file with the contents as shown below:

```
node_modules
```

- Now do a commit of your project folder to the Git repository with the message "Initial Setup". You will be doing a commit of your project at the end of each exercise so that you retain the completed files of each exercise.
- Set up an online Git repository and synchronize your project folder with the online repository.

Downloading Bootstrap

- You will use npm to fetch the Bootstrap files for use within your project. Thereafter you need to install JQuery and Popper.js as shown below since Bootstrap 4 depends on these two. At the prompt, type the following to fetch Bootstrap files to your project folder:

```
npm install bootstrap@4.0.0 --save

npm install jquery@3.3.1 popper.js@1.12.9 –save
```

- This will fetch the Bootstrap files and store is in your node_modules folder in a bootstrap folder. The bootstrap->dist folder contains the precompiled Bootstrap CSS and JS files for use within your project.

- Open your project folder in your editor, and then open the index.html file in the *conFusion* folder. This is your starting web page for the project. We have already created the web page with some content to get you started. We will use Bootstrap to style this web page, and learn Bootstrap features, classes and components along the way.
- Start your lite-server by typing **npm start** at the prompt. The *index.html* file should now be loaded into your default browser.

Getting your Web page Bootstrap ready

- Open the *index.html* file in your favourite text editor. If you are using Visual Studio Code, Brackets, Sublime Text or similar editors, you can open the project folder in the editor and then view index.html.
- Insert the following code in the <*head*> of *index.html* file before the title.

```
<!-- Required meta tags always come first -->

<meta charset="utf-8">

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

<meta http-equiv="x-ua-compatible" content="ie=edge">



<!-- Bootstrap CSS -->

<link rel="stylesheet" href="node_modules/bootstrap/dist/css/bootstrap.min.css">
```

- This will include Bootstrap CSS into your web page. Note the subtle change in the fonts of the content of the web page. This is the Bootstrap typography effect coming into play. The default Bootstrap typography sets the font to Helvetica Neue and selects the appropriate font size based on the choice of the heading style and paragraph style for the content.
- At the bottom of the page, just before the end of the body tag, add the following code to include the JQuery library, popper.js library and Bootstrap's Javascript plugins. Bootstrap by default uses the JQuery Javascript library for its Javascript plugins. Hence the need to include JQuery library in the web page.

```
<!-- jQuery first, then Popper.js, then Bootstrap JS. -->

<script src="node_modules/jquery/dist/jquery.slim.min.js"></script>

<script src="node_modules/popper.js/dist/umd/popper.min.js"></script>

<script src="node_modules/bootstrap/dist/js/bootstrap.min.js"></script>
```

- Now, do a Git commit with the message "Intro. to Bootstrap". You may push the commit to your online repository.

Conclusion

We have now understood how to set up a web project to use Bootstrap. In the next lecture, we will explore further on responsive design and Bootstrap's grid system.

3.2 - Introduction to Bootstrap: Additional Resources

PDFs of the Presentations

1-Web-UI-Frameworks.pdf

2-Intro-Bootstrap.pdf

Exercise Resources

- (required for the exercise)

Bootstrap4-starter.zip

Bootstrap Official Resources

- Bootstrap 4 Home Page
- Bootstrap typography
- Migrating from Bootstrap 3 to Bootstrap 4

Front-end Web UI Frameworks

- Top 10 Front-End Frameworks of 2018
- The 5 Most Popular Front-end Frameworks Compared

4.1 - Exercise: Responsive Design and Bootstrap Grid System Part 1

Objectives and Outcomes

This exercise introduces you to responsive design and Bootstrap support for mobile first responsive design through the use of the grid system. At the end of this exercise, you will be able to:

- Create responsive websites using the Bootstrap grid system
- Reordering content using push, pull and offset classes

Note: In this exercise we will continue to update the *index.html* file in the *conFusion* folder that we created and edited in the previous lecture.

Bootstrap Grid System and Responsive Design

Bootstrap is designed to be mobile first, meaning that the classes are designed such that we can begin by targeting mobile device screens first and then work upwards to larger screen sizes. The starting point for this is first through media queries. We have already added the support for media queries in the last lesson, where we added this line to the head:

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

The *viewport* meta tag ensures that the screen width is set to the device width and the content is rendered with this width in mind. This brings us to the second issue, designing the websites to be responsive to the size of the viewport. This is where the Bootstrap grid system comes to our aid. Bootstrap makes available four sizes, xs for extra small, sm for small, md for medium and lg for large screen sizes. We have already seen the basics of responsive design. In this exercise, we will employ the Bootstrap grid classes to design the websites. We would like our website to have the content stacked on extra small devices, but become horizontal within each row for smaller devices and beyond. Towards this goal, we will make use of the classes .col-*, .col-sm-*, col-md-*, and .col-lg-* for defining the layouts for the various device sizes. We can specify how many columns each piece of content will occupy within a row, all adding up to 12 or a multiple thereof.

Using a Container class

- We use the container class to keep content within a fixed width on the screen, determined by the size of the screen. The alternative is to use the container-fluid class to make the content automatically to span the full width of the screen. We will discuss further about this when we discuss the Bootstrap grid system in the next lecture. Add the container class to the first div right after the </header> in the file as follows.

```
<div class="container"> ...
```

Dividing the content into rows

- Let us now add the class *row* to the first-level inner *div* elements inside the container. This organizes the page into rows of content. In the next exercise, we will see how we can add other classes to the rows.

```
    <div class="row"> ...
```

Creating a Jumbotron

- Let us add the class jumbotron to the header class as shown below. This turns the header element into a Bootstrap component named Jumbotron. A jumbotron is used to showcase key content on a website. In this case we are using it to highlight the name of the restaurant.

```
        <header class="jumbotron"> ...
```

- In the header add a **container** class to the first inner div and a row class to the second inner div.

Creating a footer

- Finally, in the footer add a **container** class to the first inner div and a row class to the second inner div.

Applying column classes within each row

- In the header row, we will display the restaurant name and the description to occupy 6 columns, while we will leave six columns for displaying the restaurant logo in the future. Let us go into the jumbotron and define the classes for the inner divs as follows:

```
                <div class="col-12 col-sm-6"> ... </div>



                <div class="col-12 col-sm"> ... </div>
```

- For the remaining three div rows that contain the content, let us define the classes for the inner divs as follows:

```
            <div class="col-12 col-sm-4 col-md-3"> ... </div>



            <div class="col col-sm col-md"> ... </div>
```

- For the footer, let us define the classes for the inner divs as follows:

```
<div class="col-4 col-sm-2"> ... </div>



<div class="col-7 col-sm-5"> ... </div>



<div class="col-12 col-sm-4"> ... </div>



<div class="col-auto"> ... </div>
```

Now you can see how the web page has been turned into a mobile-first responsive design layout.

Using Order and Offset with column layout classes

- In the content rows, we would like to have the title and description to alternate so that it gives an interesting look to the web page. For extra small screens, the default stacked layout works best. This can be accomplished by using the .order-sm-last and .order-sm-first for the first and the third rows as follows:

```
<div class="col-12 col-sm-4 order-sm-last col-md-3"> ... </div>



<div class="col col-sm order-sm-first col-md"> ... </div>
```

- For the div containing the <ul> with the site links, update the class as follows:

```
  <div class="col-4 offset-1 col-sm-2">
```

- After saving all the changes, you can do a Git commit with the message "Bootstrap Grid Part 1" and push your changes to the online repository.

Conclusion

In this exercise, we reviewed responsive design and the Bootstrap grid system.

4.2 - Exercise: Responsive Design and Bootstrap Grid System Part 2

Objectives and Outcomes

This exercise continues the examination of responsive design and Bootstrap support for mobile first responsive design through the use of the grid system. We also learn how to customize some of the Bootstrap classes through defining our own modifications in a separate CSS file. At the end of this exercise, you will be able to:

- Customize the CSS classes through your own additions in a separate CSS file
- Centering the content both vertically and horizontally within a row

List styles

- You can use several list styles to display lists in different formats. In this exercise, we will use the unordered list style *list-unstyled* to display the links at the bottom of the page without the bullets. To do this, go to the links in the footer and update the ul as follows

```
<ul class="list-unstyled"> ... </ul>
```

Using Custom CSS classes

We can define our own custom CSS classes in a separate CSS file, and also customize some of the built-in CSS classes. We will now attempt to do this in this part of the exercise.

- Create a folder named ***css***. Then create a file named *styles.css* in the ***css*** folder. Open this file to edit the contents. Add the following CSS code to the file:

```
.row-header{

    margin:0px auto;

    padding:0px;

}



.row-content {

    margin:0px auto;

    padding: 50px 0px 50px 0px;
    border-bottom: 1px ridge;

    min-height:400px;

}



.footer{

    background-color: #D1C4E9;

    margin:0px auto;

    padding: 20px 0px 20px 0px;

}
```

- Include the *styles.css* file into the head of the *index.html* file as follows:

```
    <link href="css/styles.css" rel="stylesheet">
```

- Then add these classes to the corresponding rows in the *index.html* file as follows. See the difference in the *index.html* file in the browser. The first one is for the row in the <header>, the next three for the rows in the content, and the last one directly to the <footer> tag.

```
    <div class="row row-header"> ... </div>



    <div class="row row-content"> ... </div>



    <div class="row row-content"> ... </div>



    <div class="row row-content"> ... </div>



    <footer class="footer"> ... </footer>
```

- Our next set of customization is to the jumbotron and the address. Add the following to *styles.css* file:

```
.jumbotron {
    padding:70px 30px 70px 30px;
    margin:0px auto;
    background: #9575CD ;
    color:floralwhite;
}


address{
    font-size:80%;
    margin:0px;
    color:#0f0f0f;
}
```

Vertically Centering the Content

- In the content section, update all the rows as follows:

```
        <div class="row row-content align-items-center">
```

- In the footer, update the third column div that contains the social media links as follows:

```
                <div class="col-12 col-sm-4 align-self-center">
```

Horizontally Centering the Content

- Update the copyright paragraph as follows:

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

                <div class="col-auto">
```

- Update the inner div containing the social media links as follows:

```
                    <div class="text-center">
```

- After saving all the changes, you can do a Git commit with the message "Bootstrap Grid Part 2" and push your changes to the online repository.

Conclusion

In this exercise, we continued our review of responsive design and the Bootstrap grid system. We also learnt how to customize using our own CSS classes.

4.3 - Responsive Design and Bootstrap Grid System: Additional Resources

PDFs of Presentations

3-Responsive-Design.pdf

4-Bootstrap-Grid.pdf

Bootstrap Official Documentation

- Bootstrap Grid System

Responsive Design and Bootstrap Grid Resources

- CSS Flexible Box Layout Module Level 1 (W3C Documentation)
- A Complete Guide to Flexbox
- A Visual Guide to CSS3 Flexbox Properties
- The Bootstrap 4 Grid: What's New?
- How the Bootstrap Grid Really Works
- The Subtle Magic Behind Why the Bootstrap 3 Grid Works (a detailed explanation of why the Bootstrap grid system works the way it does, a delight to read!)
- What The Heck Is Responsive Web Design? (a short presentation that introduces responsive web design)
- Beginner’s Guide to Responsive Web Design (simple introduction to responsive web design)
- The 2014 Guide to Responsive Web Design (an updated guide to responsive design)

Assignment 1 Resources

Assignment 1 Starter Files

aboutus.html.zip

Bootstrap Resources

- Bootstrap grid
- Bootstrap display utilities (documentation here about the d-none and d-sm-block classes)

Chrome extension

- Full Page Screen Capture.