Front-end web UI frameworks and tools: bootstrap 4

# 1st Course in the full-stack web development with react Specialization

# About Course

This course will give you an overview of client-side web UI frameworks, in particular Bootstrap 4. You will learn about grids and responsive design, Bootstrap CSS and javascript components. You will learn about CSS preprocessors, Less and Sass. You will also learn the basics of Node.js and NPM and task runners like Grunt and Gulp. At the end of this course, you will be able to a)Set up, design and style a web page using Bootstrap 4 and its components, b) Create a responsive web page design, and c) Make use of web tools to setup and manage web sites. This course also includes an honors track that enables you to work on your own project developing a website using Bootstrap 4. Students enrolling in this course should have prior good working knowledge of HTML, CSS and javascript.

# Skills You Will Gain

jquery

sass(style sheet language

node.js

bootstrap( front-end-framework

# Course Materials

General Links

<https://git-scm.com/about> (everything Git)

<https://getbootstrap.com/> (everything Bootstrap)

*Additional Reading*

*Beyond Scope*

# Week

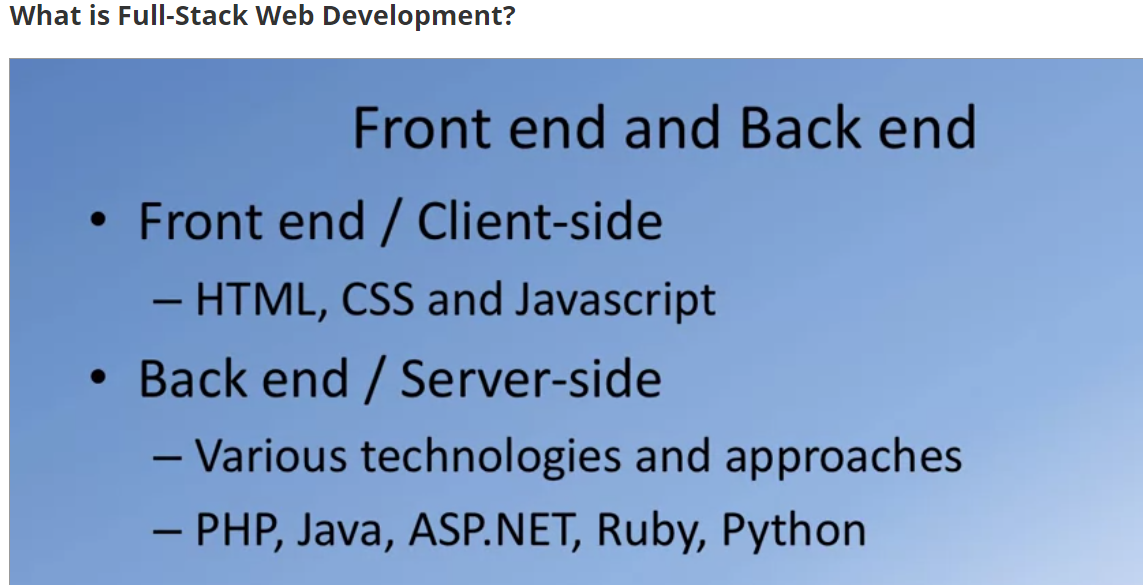
Week 1

Front-end Web UI Frameworks Overview: Bootstrap:

This module gives you a quick introduction to full-stack web development and the outline of the course. Then you will learn the basics of Bootstrap, setting up a web project using Bootstrap. You will learn about responsive design and the Bootstrap grid system. At the end of this module, you need to complete your first assignment.

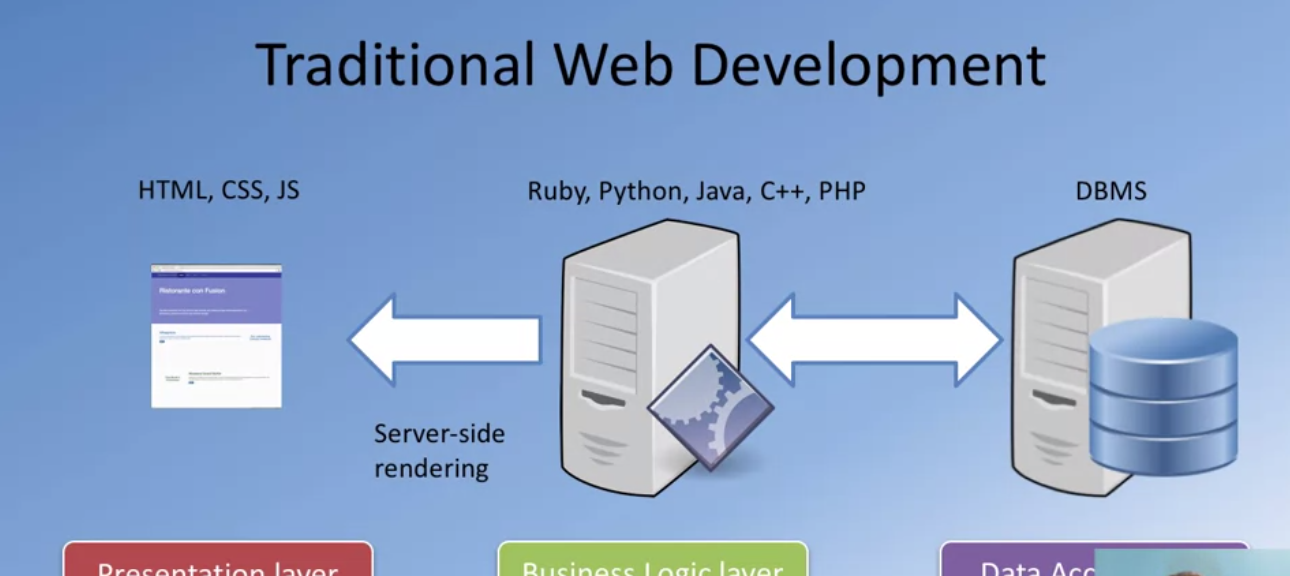
Notes

What is Full-Stack Web Development?

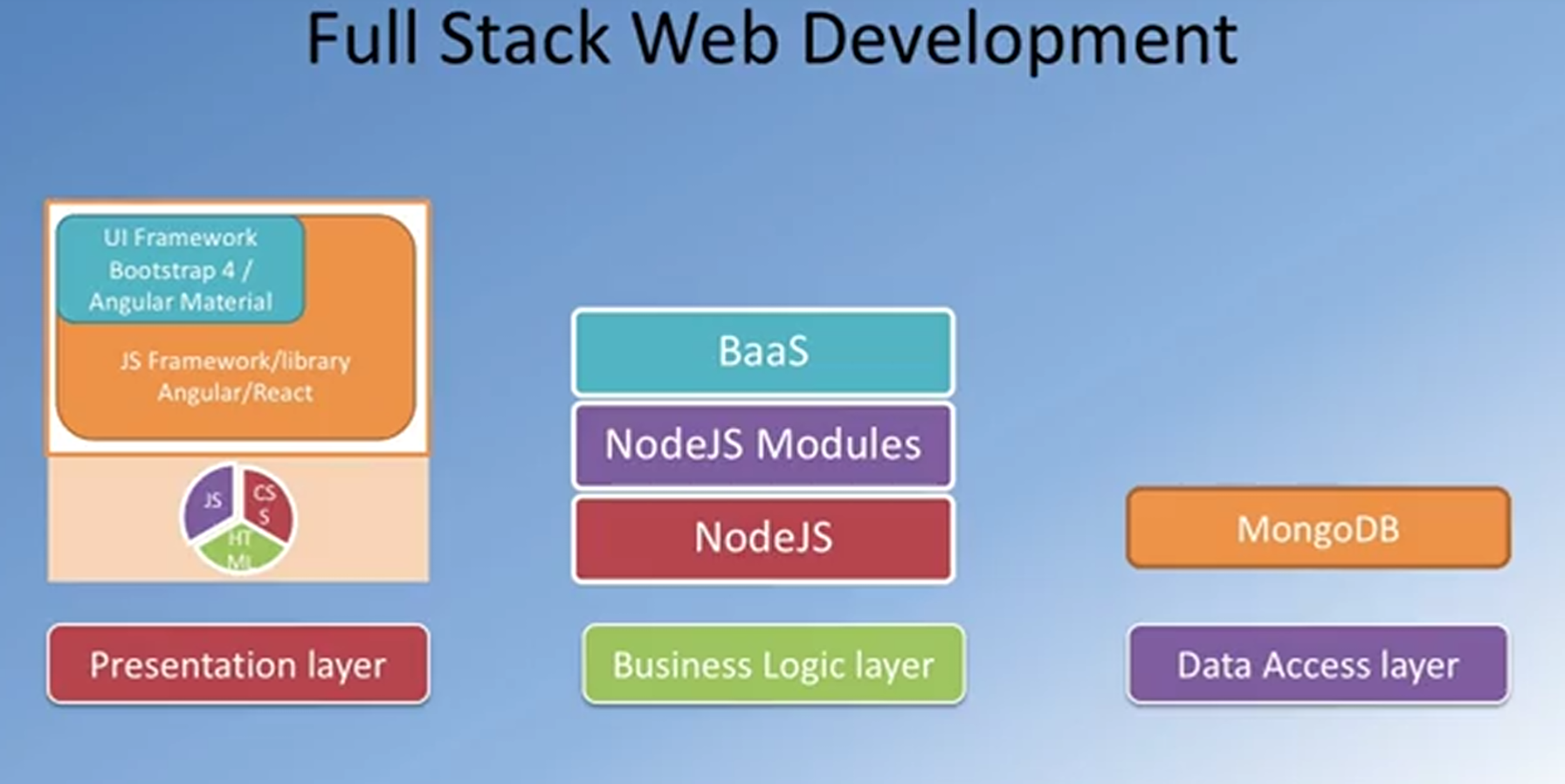


* Three Tier Architecture – Full Stack Development
  + Presentation Layer
  + Business Logic Layer
  + Data Access Layer

Full Web Development model

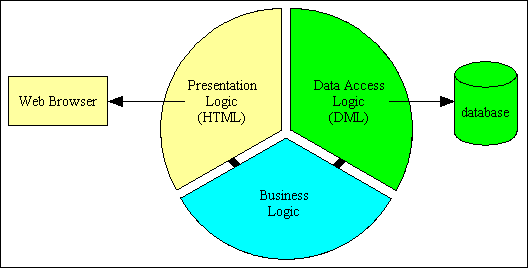






We often hear people talking about the three tier architecture for Web Development. In this approach, the entire web application is organized into three different layers. The presentation layer, which is concerned with delivering to the user, So, this is usually the UI-related concerns that are dealt with at the presentation layer. The Business Logic Layer, on the other hand, is concerned more about the data, the data validation, the dynamic content processing, and generating the content to be delivered to the user. This is backed up behind the scenes with the data persistence layer or the data access layer. So, this is concerned with how we store and interact with the data, typically, in the form of a database and access this data through an API.

The Business Logic Layer is also concerned with the rendering of information to the front side, typically, in the form of server-side rendering these days.



Note here that the presentation layer has no direct communication with the data access layer - it can only talk to the business layer.

Note also that you should not infer from this diagram that the entire application can be built with a single component in each of these three layers. There are several choices as follows:

* There should be a separate component in the Presentation layer for each user transaction.
* There should be a separate component in the Business layer for each business entity (database table).
* There should be a separate component in the Data Access layer for each supported DBMS.

With this structure it is easy to replace the component in one layer with another component without having to make any changes to any component in the other layers.

* You can change the UI component so that you can switch between a variety of different output formats, such as HTML, PDF or CSV.
* You can change the data access component so that you can switch between a variety of database engines, such as MySQL, Oracle or SQL Server.

This structure also provides more reusability as a single component in the Business layer can be shared by several components in the Presentation layer. This means that business logic can be defined in one place yet shared by multiple components.

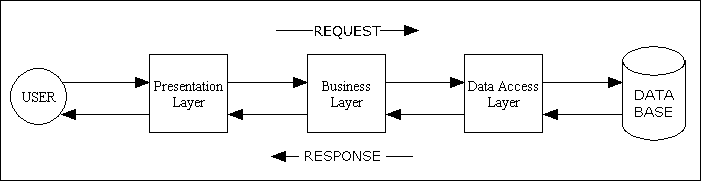
The Rules of the 3 Tier Architecture

It is simply not good enough to split the code for an application into 3 parts and call it "3 Tier" if the code within each tier does not behave in a certain way. There are rules to be followed, but these rules are pretty straightforward.

* The code for each layer must be contained within separate files which can be maintained separately, possibly by separate teams.
* Each layer may only contain code which belongs in that layer. Thus business logic can only reside in the Business layer, presentation logic in the Presentation layer, and data access logic in the Data Access layer.
* The Presentation layer can only receive requests from, and return responses to, an outside agent. This is usually a person, but may be another piece of software.
* The Presentation layer can only send requests to, and receive responses from, the Business layer. It cannot have direct access to either the database or the Data Access layer.
* The Business layer can only receive requests from, and return responses to, the Presentation layer.
* The Business layer can only send requests to, and receive responses from, the Data Access layer. It cannot access the database directly.
* The Data Access layer can only receive requests from, and return responses to, the Business layer. It cannot issue requests to anything other than the DBMS which it supports.
* Each layer should be totally unaware of the inner workings of the other layers. The Business layer, for example, must be database-agnostic and not know or care about the inner workings of the Data Access object. It must also be presentation-agnostic and not know or care how its data will be handled. It should not process its data differently based on what the receiving component will do with that data. The presentation layer may take the data and construct an HTML document, a PDF document, a CSV file, or process it in some other way, but that should be totally irrelevant to the Business layer.

This cycle of requests and their associated responses can be shown in the form of a simple diagram, as shown in [figure 6](http://www.tonymarston.net/php-mysql/3-tier-architecture.html#figure6):

Figure 6 - Requests and Responses in the 3 Tier Architecture



## Setting up your Development Environment: Git and Node

Getting started with Git

Some Global Configuration for Git:

At the command-line:

* Check version with [git] --version
* Config. User name with [git][config] --global [user.name] <”your input”>
* Config. User email with [git][config] --global [user.email] <”your input”>
* Check default Git global config. with [git][config] --list

Some Basic Concepts:

* Version Control: software tool(s) that enable the management of changes to source code
  + Maintaining version history
* Several version control tools: CVS, SVN,Git, etc

Basic Git Commands:

* Initializing the folder as a Git repository
  + Open command prompt and locate the directory to the git folder that you wish to initialize as a repository in this case (git-test) and type the following command at the prompt to initialize the folder as a Git repository: git init

This this creates what is known as a master branch (initial branch)

* Checking your Git repository status
  + Type the following command 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 you Git repository, type the following command: git add <filename> or git add . ( dot wildcard will add all pending files/folders )(note the space between the two commands [add] and [.] )
* Committing to the Git repository
  + To commit the current staging area to your Git repository, type command: git commit –m “first commit” ( -m “message” inputs a message or label identifying the specific commit command)
* Checking the log of Git commits
  + To check the log of the commits to your Git repository, type the command: git log --oneline (--oneline prints shorthand log)
* Checking out a file from an earlier commit ( for reverting to previous state )
  + To check out the file from a previous commit use the git log command to view log data:

Type the command: git checkout <previous commit id number>” filename”

* Unstage (start revert process) changes that were invoked by the checkout command
  + Type the command: git reset HEAD <filename>

Online Git Repository

Git Repository service providers:

* GitHub(<https://github.com>)
* Bitbucket(https://bitbucket.org)

Configure Git repository to mirror online repository (this command is followed by a push -u command )

* Type the command in the prompt: git remote add origin <**repository** URL>

Push local git repository to the online repository

* Type the command: git push –u origin master

**Cloning an online repository – at the command prompt change the directory to a location you want to clone the repository to.**

* **Type the command: git clone <repository URL>**

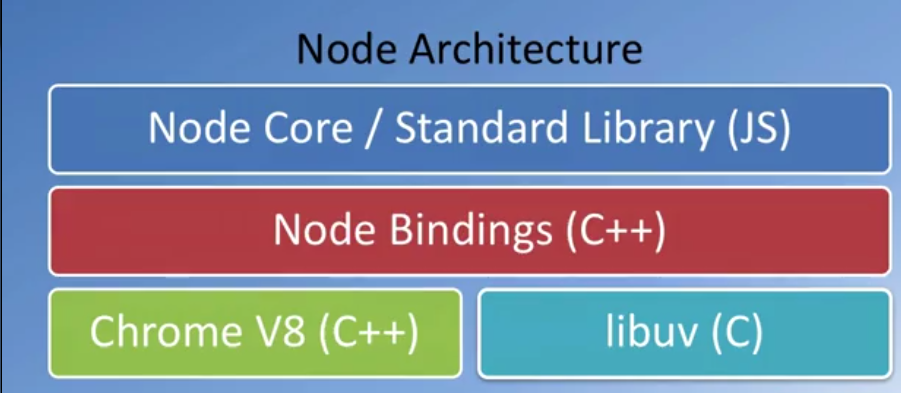
**What is Node.js?**

Node.js environment, a popular Javascript based server framework

JavaScript runtime built on Chrome V8 JavaScript Engine

Allows users to write and run programs in JavaScript ( a scripting language developed for web use ) on desktops.

**Node Architecture**



* Node Package Manager ( NPM )
  + Node package manager managers ecosystem of node modules / packages
  + A package contains – JS files and package.json (manifest)

Why package.json?

* It serves as documentation for what packages your project depends on.
* It allows you to specify the versions of a package that your project can use using semantic versioning rules.
* Makes your build reproducible, which means that its way easier to share with other developers.

Initializing package.json

* + To initialize a package.json file for your project, type at the prompt in your project directory: npm init

Using NPM to install a node module ( lite server )

* To access the NPM and install module(s) type the command in the project directory: npm install [module name] The command npm install lite-server –save-dev installs lite server module to host repository in real time (--save-dev add this info to the package.json/manifest)

Special Note: When you wish to exclude file/folders for being committed and pushed to the online mirrored repository create a file named <.gitignore.txt> and add the names of file/folder(s) you want ignored.

# Introduction to Bootstrap: Objectives and Outcomes

In this lesson, you will be given a quick overview of front-end UI frameworks, and an introduction to Bootstrap. The exercises will introduce you to getting started with Bootstrap for your web project. At the end of this lesson, you will be able to:

* Identify the purpose of using front-end UI frameworks in web design and development
* Set up a project with Bootstrap support
* Configure a web project to use Bootstrap
* Become familiar with the basic features of Bootstrap

Popular front-end web UI frameworks

* Bootstrap
* Semantic-UI
* Foundation
* Materialize

Why Front-End Web UI Frameworks?

* Responsive web design
  + Mobile first
* Cross-browser compatibility

## Setup Bootstrap

### 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

1. Create a project folder ( note: that this project should a have default index.html file for initial setup purposes)
2. Setup the project to use NPM with the “npm install” command at the cmd
3. Be sure to open the project in an editor and create a “.gitignore.txt” and add the node\_modules folder to the ignore list
4. Initialize a git repository either before or after the “npm install” and commit necessary changes
5. Setup the project to use Bootstrap with the “npm install bootstrap@<currentversion> --save” command
6. Follow the prior command with the “npm install jquery@<currentversion> popper.js@<currentversion> --save” command
7. Note the node\_modules\bootstrap\dist folder contains the precompiled Bootstrap CSS and JS files for use within your project
8. Add the scripts <”start”:”npm run lite”> and <”lite”:”lite-server”>to the scripts object in the package.json file.
9. Start with the “npm start” command at the command line
10. Open the index.html file in an editor and insert the following code in the head tag to include Bootstrap CSS in your web page
    1. <!-- 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">

1. 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.
   1. <!-- 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>

Note: And note the order in which I have included this. So the Bootstrap is at the bottom. Then, since Bootstrap is dependent upon both jQuery and Popper in that order, so I will first input jQuery. And then after that, we input Popper, and then finally, Bootstrap min.js at the bottom of the index.html file. Now, this is included at the bottom of the page. Because when you are loading the page from a web server, you want the CSS classes to be loaded immediately so that as the page starts rendering, when the JavaScript is fetched, the JavaScript needs to execute in order to make changes to your page with the JavaScript code, and that will take a little bit of time. So you don't want the user to be waiting for the entire page to be loaded before they see something in their browser window.

# Responsive Design and Bootstrap Grid System: Objectives and Outcomes

In this lesson, you will be given an overview of responsive web design and an introduction to the Bootstrap grid system. The exercises will concentrate on enhancing your web project using the Bootstrap grid in order to make it responsive. At the end of this lesson, you will be able to:

* Understand the reasons for using responsive web design in a web project
* Use the Bootstrap grid system to design responsive websites
* Add your own custom CSS classes to a Bootstrap based web project

Responsive design with Bootstraps grid system primary functionality is to allow your web page(s) to adapt to the user’s viewport” (the screen dimensions for the device).

This technique is always implemented in a mobile-first approach. This will ensure that your web page will provide a full access experience to users accessing the page with mobile devices, while meeting the constraints of reduced screen real-estate.

Several features that allow for viewport adaptability (responsive design):

* Grid system
* Fluid images
* Media queries

### Media Queries

CSS techniques that apply styles based on the size of the viewport.

An example of a media query:

@media (min-width: 992px) {

/\* CSS styles customized for desktop \*/

}

### Grid System

The viewport. Addressed in the head of your html doc. As meta

<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
* Designing the websites to be responsive to the size of the viewport
  + Bootstrap grid system

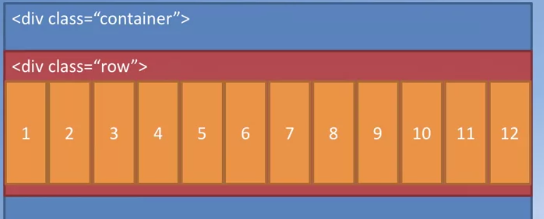
CSS Flexbox Layout ( discussed more in detail later )

* Simpler and flexible layout options in CSS
* Can easily handle dynamic/unknown size of content containers
* Direction-agnostic layout

#### Bootstrap Grid Techniques

Container class system with rows and columns ( think of a table ). Where the container (outermost) class would define the table borders and the rows and columns would define the cells.

* Note that the columns are limited to a maximum of 12



Bootstrap makes available five classes for rendering based on screen dimensions:

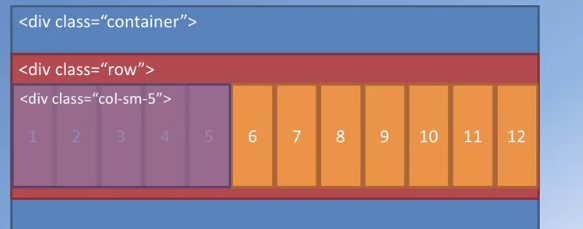
* Default which targets all screen sizes from extra small to extra large
* sm for small
* md for medium
* lg for large
* xl for extra large screen size

#### Columns (within the Bootstrap grind system)

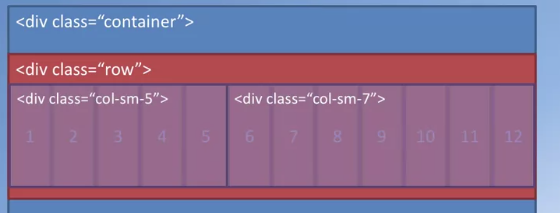
Each row in Bootstrap grid system is divided into 12 columns. Using the following classes will allow you to define the layout for various screen sizes:

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

Where the \* represents the number of columns the row will occupy

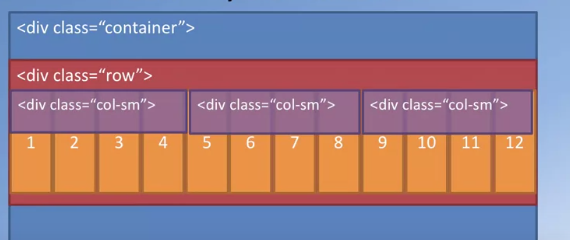


This figures show how you can divide the row



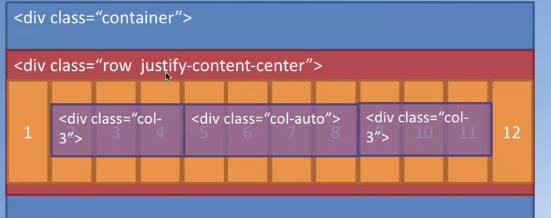
##### Bootstrap’s Auto-layout

This figure show Bootstrap’s Auto-layout Columns features

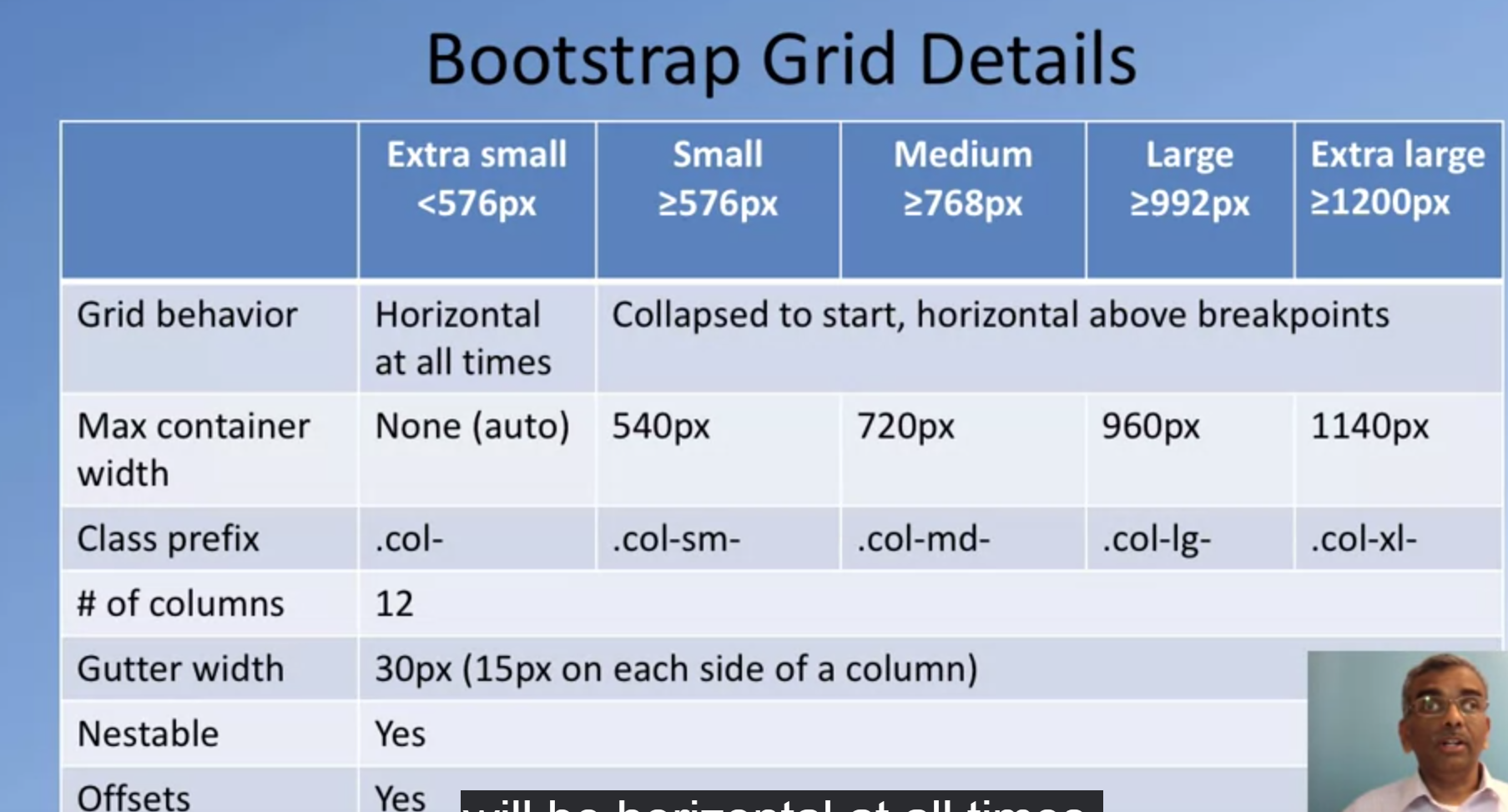


Divides the row equal ( thirds of the row )

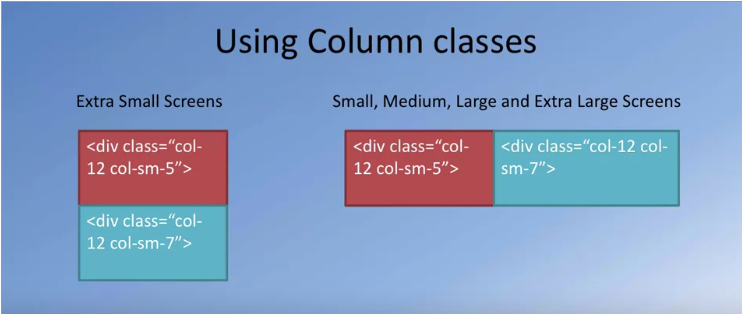
* Use the div class = “col-auto” to have the content dynamically set the column span



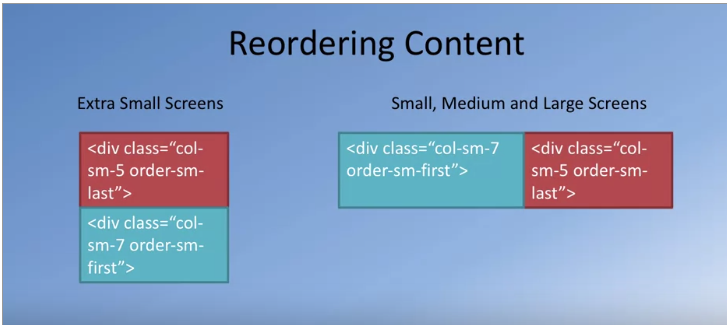
#### Bootstrap Grid Details



This figure provides an example of how content will be laid out

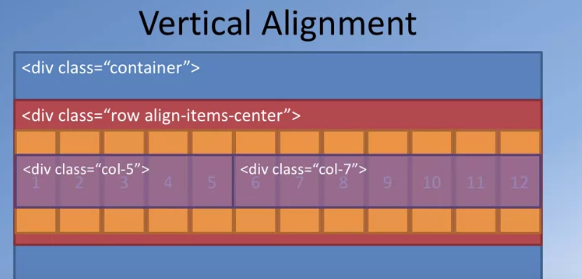


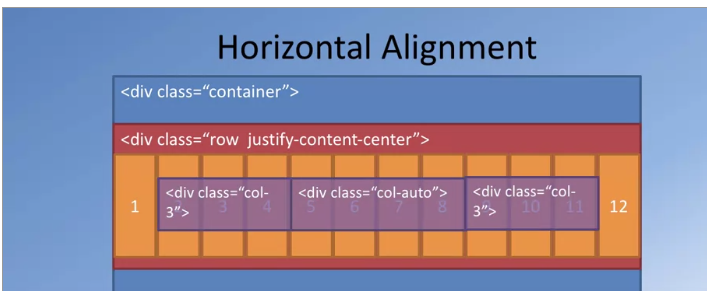
Reordering columns with class order-\* ( using default classes e.g., sm, lg or column numbers 1-12 )



##### Vertical Alignment and Horizontal Alignment

Class align-<alignment behavior> class justify-<justified behavior>





##### Column Offsets

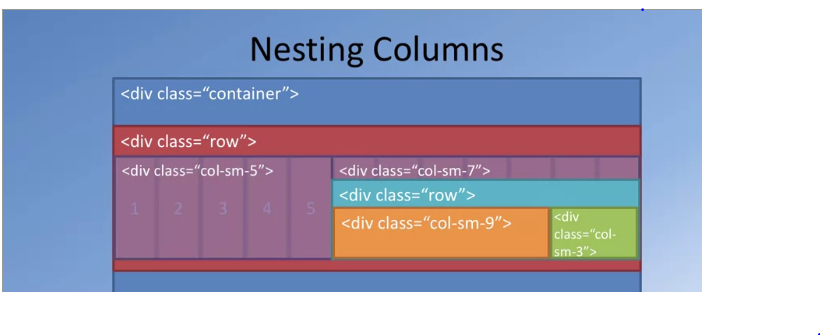
This figure shows the class offset feature



Offsets the column span by one column

##### Nesting

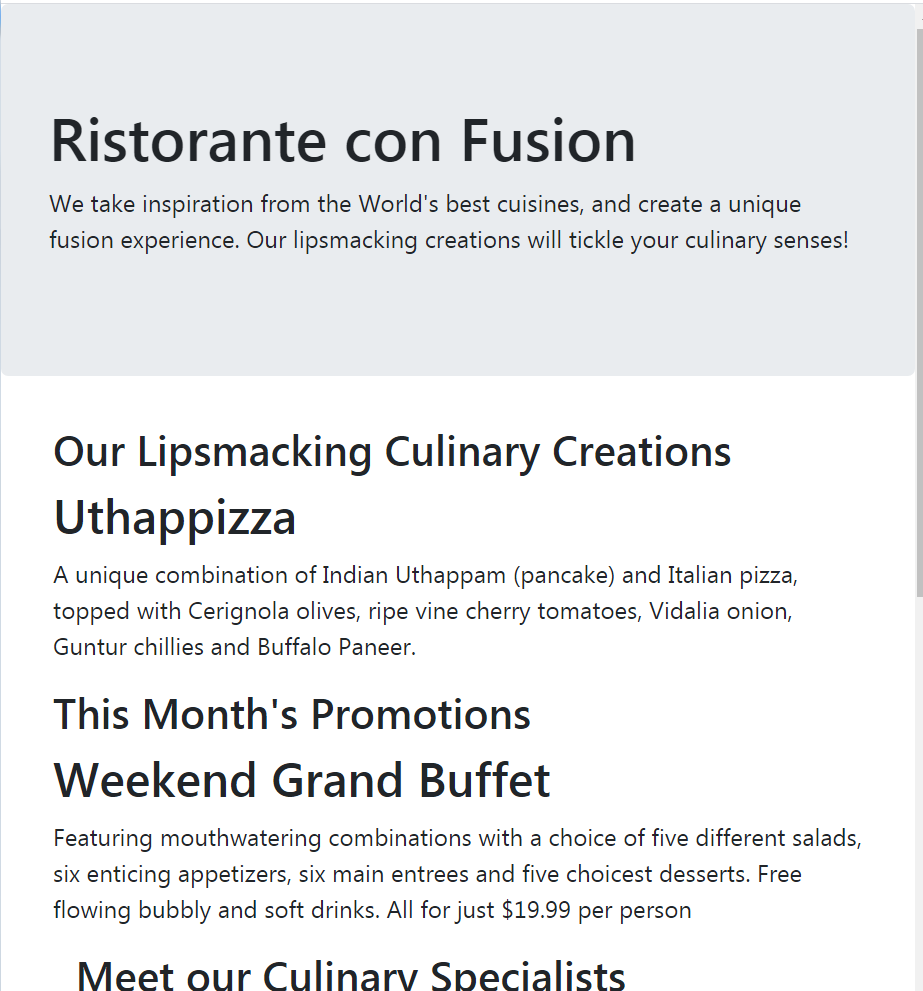
This figures shows Bootstraps nesting capabilities



# Working with Bootstrap

##### Class jumbotron

This class sets the specific content apart from the rest of the content with a distinct look

Header tag header tag with jumbotron class applied

Links

### Useful Links

* [What is a Full Stack developer?](http://www.laurencegellert.com/2012/08/what-is-a-full-stack-developer/)
* [Wait, Wait… What is a Full-stack Web Developer After All?](http://edward-designer.com/web/full-stack-web-developer/)
* [The Myth of the Full-stack Developer](http://andyshora.com/full-stack-developers.html)
* [Multi-tier Architecture](https://en.wikipedia.org/wiki/Multitier_architecture)
* [What is the 3-Tier Architecture?](http://www.tonymarston.net/php-mysql/3-tier-architecture.html)

### Additional Resources (Git)

* Git site [http://git-scm.com](http://git-scm.com/).
* [Installing Git](https://git-scm.com/book/en/v2/Getting-Started-Installing-Git) chapter from Pro Git
* [Git reference manual](https://git-scm.com/docs)
* Quick reference guides: [GitHub Cheat Sheet](https://services.github.com/on-demand/downloads/github-git-cheat-sheet.pdf) (PDF) | [Visual Git Cheat Sheet](http://ndpsoftware.com/git-cheatsheet.html) (SVG | PNG)
* <https://docs.npmjs.com/about-npm/> (everything node package manager)
* [Atlassian comprehensive Git tutorial](https://www.atlassian.com/git/tutorials/)

### Additional Resources (Node.js and NPM)

* [Nodejs.org](https://nodejs.org/)
* [Npmjs.com](https://www.npmjs.com/)
* [Node API Documentation](https://nodejs.org/api/)
* [NPM Documentation](https://docs.npmjs.com/)
* [lite-server](https://github.com/johnpapa/lite-server)

### Responsive Design and Bootstrap Grid Resources

* [CSS Flexible Box Layout Module Level 1](https://www.w3.org/TR/css-flexbox/) (W3C Documentation)
* [A Complete Guide to Flexbox](https://css-tricks.com/snippets/css/a-guide-to-flexbox/)
* [A Visual Guide to CSS3 Flexbox Properties](https://scotch.io/tutorials/a-visual-guide-to-css3-flexbox-properties)
* [The Bootstrap 4 Grid: What's New?](http://tutorialzine.com/2016/11/boostrap-4-regular-vs-flex-grid/)
* [How the Bootstrap Grid Really Works](http://blog.codeply.com/2016/04/06/how-the-bootstrap-grid-really-works/)
* [The Subtle Magic Behind Why the Bootstrap 3 Grid Works](http://www.helloerik.com/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?](http://johnpolacek.github.io/scrolldeck.js/decks/responsive/) (a short presentation that introduces responsive web design)
* [Beginner’s Guide to Responsive Web Design](http://blog.teamtreehouse.com/beginners-guide-to-responsive-web-design) (simple introduction to responsive web design)
* [The 2014 Guide to Responsive Web Design](http://blog.teamtreehouse.com/modern-field-guide-responsive-web-design) (an updated guide to responsive design)

Closing Thoughts

Week 2

# Navigation and Navigation Bar: Objectives and Outcomes

In this lesson, you will be given an overview of navigation design and the importance of providing appropriate navigation support within your website. You will learn about support for navigation design elements available in Bootstrap, including the Navbar and Breadcrumbs. Other navigation aids will be covered in subsequent modules. In addition, the use of icon fonts in web page design will be covered. The exercises will concentrate on adding a responsive navigation bar to the website. At the end of this lesson, you will be able to:

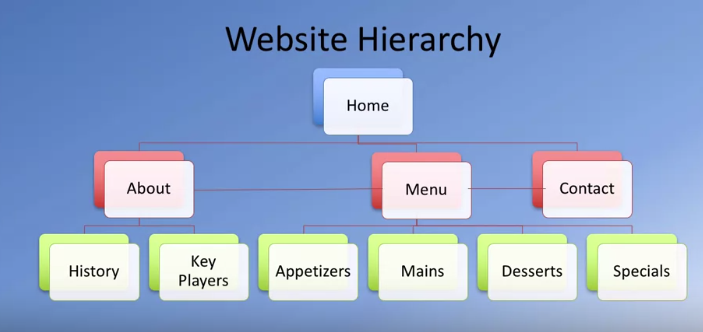
* Understand the need for navigation support in a web project
* Use the Bootstrap navigation features including the Navbar and breadcrumbs in providing navigation support in websites
* Use icon fonts for decorating your website with meaningful graphical elements

## Navigation and Navigation Bar

### Information Architecture

Structure of a system with respect to the way the information is: organized, labeled and navigation methods provided to access the information. This framework is a field of study own its own.

Model of Website basic hierarchy

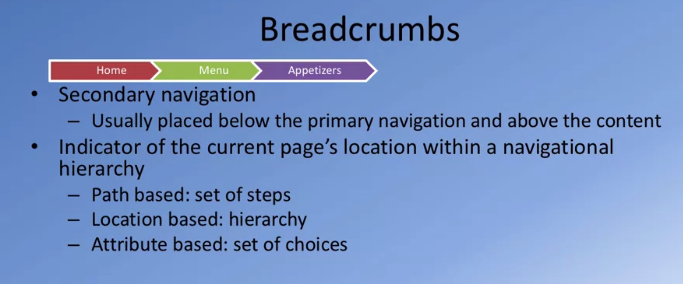


Navigation bar dos and don’ts:

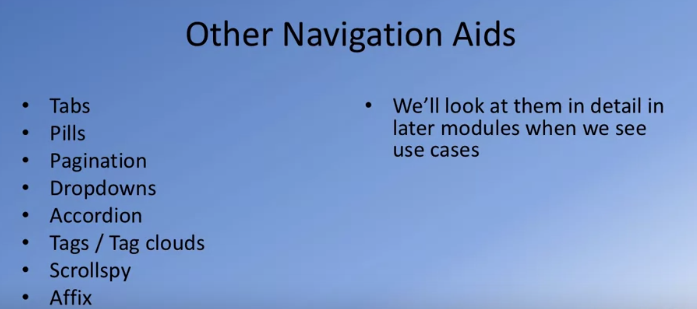
* Use simple, user-friendly terms
* Standardize navigation
* Provide indication of the location within the navigation hierarchy
* Use standard web conventions
  + Clicking on a organization logo takes you back to home page
* Try not to have too many items
* Don’t use generic labels be specific and concise

#### Breadcrumbs

Breadcrumbs refer to the navigational hierarchy the user is currently in (e.g., the use is currently viewing a page called appetizers which is accessed through home/menu/appetizers the breadcrumbs would detail this access tree)



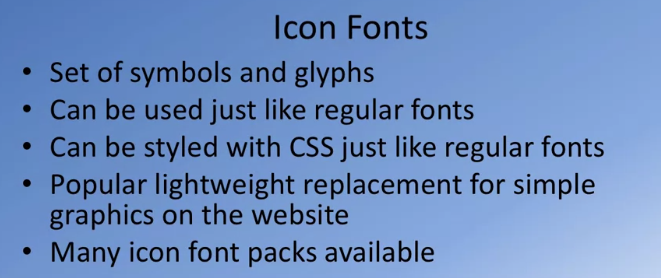
#### Other Navigational Aids



### Navbar and Breadcrumbs

### Icon Fronts

Icon fonts provide a very flexible way of including tiny images into our web pages that can be styled just like text.



Recommend using Font Awesome for icon fonts



You may use <span> or <i> for font awesome icons

#### Bootstrap-social



### Using Icon Fonts and Other CSS classes

* One of the most popular icon font toolkit is Font Awesome. Go to its website <http://fontawesome.io/> to check out more details about this icon font. You can get Font Awesome using npm by typing the following at the prompt:

npm install font-awesome@<current version> --save

* Another module that we install is Bootstrap Social that enables the addition of Social buttons to our site. You can find more information about it at <https://lipis.github.io/bootstrap-social/>. To install it using npm, type the following at the prompt:

npm install bootstrap-social@<current version> --save

We now need to include the CSS files for font awesome and bootstrap-social in the index.html file. Add the following code to the head of the file after the links for importing Bootstrap CSS classes. Do the same change to aboutus.html file:

<link rel="stylesheet" href="node\_modules/font-awesome/css/font-awesome.min.css">

<link rel="stylesheet" href="node\_modules/bootstrap-social/bootstrap-social.css">

# User Input: Buttons and Forms: Objectives and Outcomes

In this lesson we review the support for user input through the use of buttons and forms in a web page. We review Bootstrap button classes and Forms classes. At the end of this lesson you will be able to:

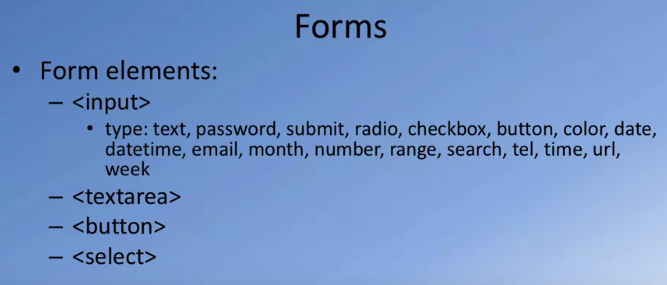
* Create and style buttons on a web page using Bootstrap button classes
* Create and style forms on a web page using Bootstrap form classes

## User Input

User input to a web page can be facilitated through three approaches:

* <a> tags to provide hyperlinks
* <button> tags to create buttons
* <form> tag to create forms
  + <input> tag to create elements in forms

#### Forms



### Buttons

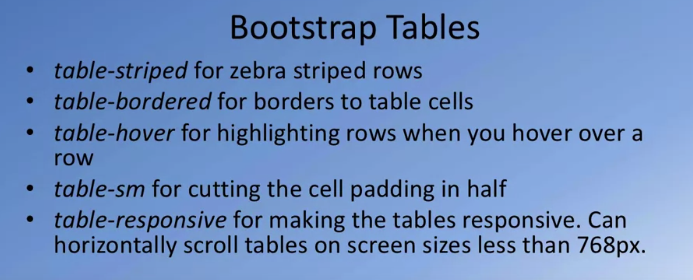
# Displaying Content: Tables and Cards: Objectives and Outcomes

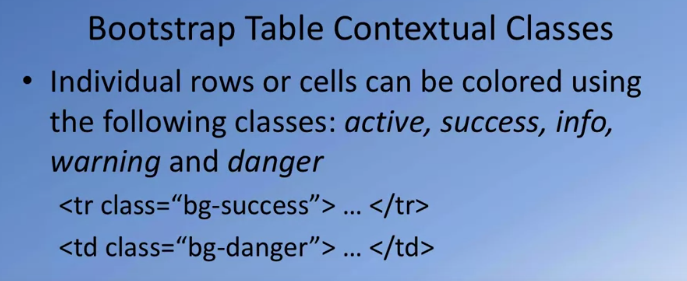
In this lesson we will be reviewing the support for tables in Bootstrap. In addition we will look at a versatile component called card that enables the display of content in myriad ways. At the end of this lesson you will be able to:

* Present and style tabular data in a table form using Bootstrap support for tables
* Display content using a card on a web page.

## Bootstrap Tables and Cards

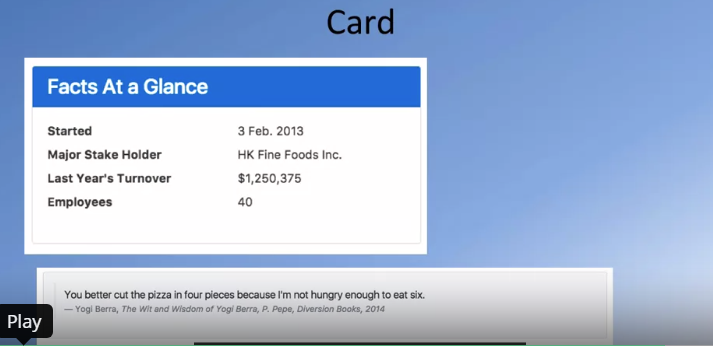
Bootstrap tables with class=”table”





A Bootstrap card is designed to display content in myriad ways

Figure shows some examples of cards

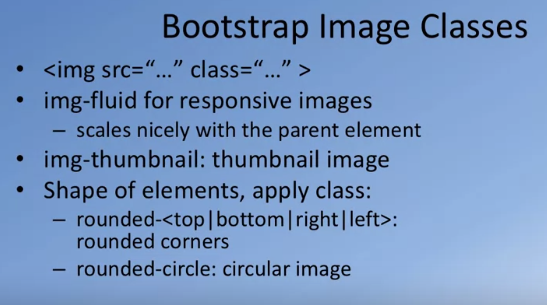


# Images and Media: Objectives and Outcomes

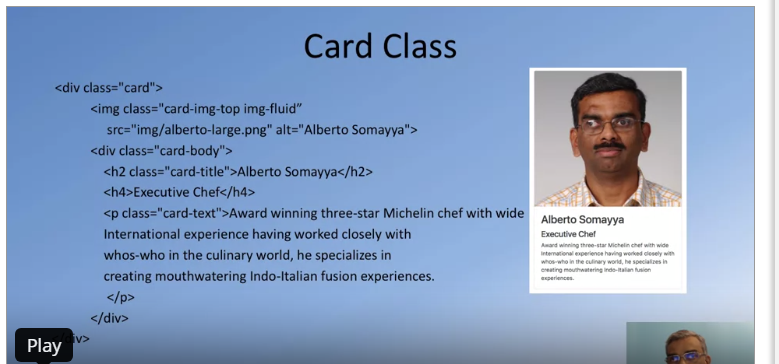
In this lesson we will look at the use of images and media on websites. In particular we will review the Bootstrap classes to support the inclusion of images and media, supporting responsiveness of images and media, and the use of these as thumbnails and part of other components, in particular the media component. At the end of this lesson you will be able to:

* Use images and media and include them in your website
* Support responsive images and media using responsive Bootstrap classes for images and media
* Use thumbnails and media components using Bootstrap classes

## Images and Media



Card class with image example



Note if the image is too small to occupy all the upper most area ( in this example) then it will not render like the card in the example shown above

### Media Object

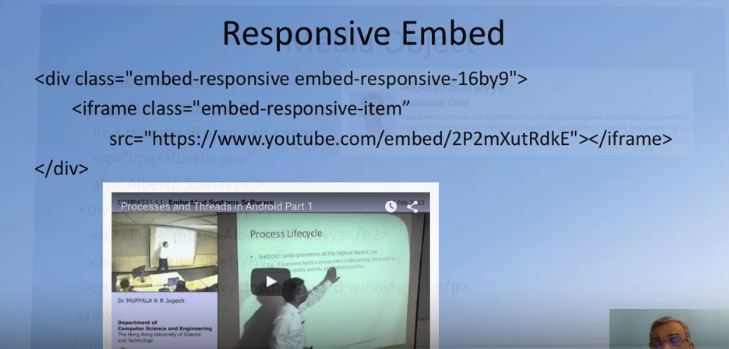
Another way of including media in your web pages are media objects



### Responsive Embed

Embedding media content, e.g., video, and making it responsive

* Apply embed-responsive-item to <embed>, <iframe>, <video>, <object>
* Enclose in <div> and apply embed-responsive and embed-responsive-4by3 or embed-responsive-16by9



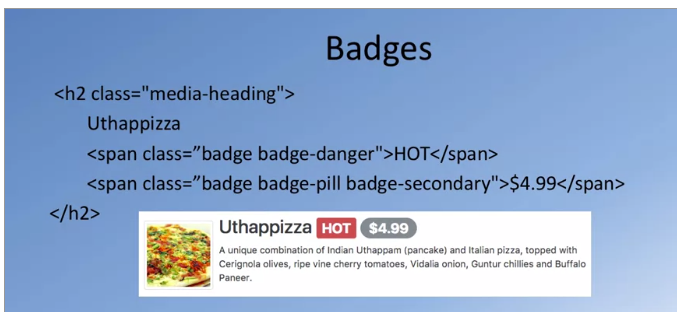
# Alerting Users: Objectives and Outcomes

In this lesson we examine various ways of delivering alert information to users. We examine labels, badges, alerts and progress bars. At the end of this lesson, you will be able to:

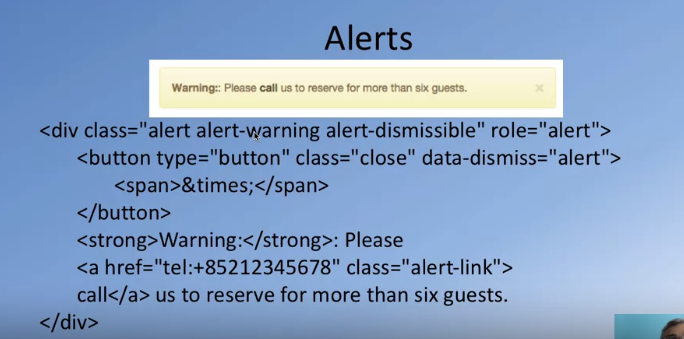
* Include labels and badges in your web page
* Create, style and include alerts in your web page
* Appreciate the use of progress bars and controlling the state of the progress bars

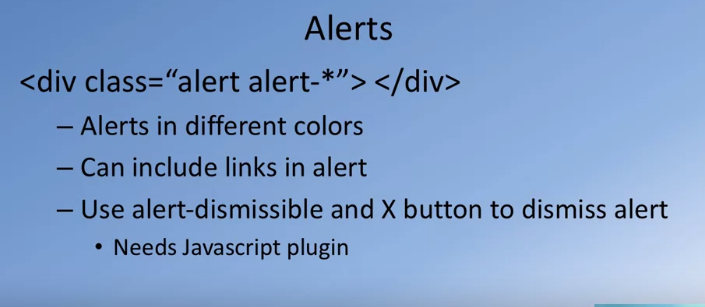
## Alerting Users

#### Badges

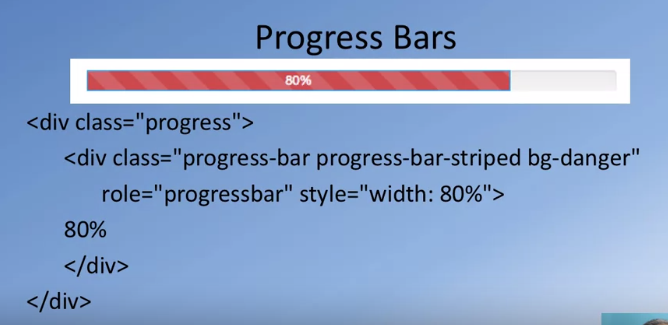


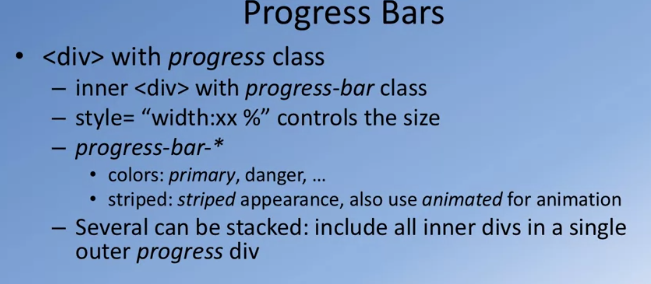
#### Alerts





#### Progress Bars





Notes

Links

### General

* [Accessible Rich Internet Applications (ARIA)](https://developer.mozilla.org/en-US/docs/Web/Accessibility/ARIA) (Accessibility support and screen reader support)

### Information Architecture

* [Information Architecture 101: Techniques and Best Practices](http://sixrevisions.com/usabilityaccessibility/information-architecture-101-techniques-and-best-practices/) (Quick introduction to Information architecture with respect to website design)
* [Web Site Information Architecture models](http://webdesignfromscratch.com/website-architecture/ia-models/) (Another good resource on information architecture)
* [What is information architecture?](http://www.steptwo.com.au/papers/kmc_whatisinfoarch/) (Good definition and explanation about the topic)
* [Information Architecture Tutorial](http://www.webmonkey.com/2010/02/Information_Architecture_Tutorial/) (Comprehensive look from a website design perspective)

### Navigation Bar Design

* [Designing A Winning Navigation Menu: Ideas and Inspirations](http://www.hongkiat.com/blog/navigation-design-ideas-inspiration/) (Good suggestions on how to design navigation for a website)
* [Are You Making These Common Website Navigation Mistakes?](https://blog.kissmetrics.com/common-website-navigation-mistakes/) (Worth reading at least to learn what not to do)
* [3 Reasons We Should Stop Using Navigation Bars](http://www.webdesignerdepot.com/2014/01/3-reasons-we-should-stop-using-navigation-bars/) (A provocative view on navigation bars)

### Breadcrumbs

* [Breadcrumb Navigation Examined: Best Practices & Examples](http://www.hongkiat.com/blog/breadcrumb-navigation-examined-best-practices-examples/) (Great suggestions on using breadcrumbs for navigation)
* [Breadcrumb Navigation: A Guide On Types, Benefits And Best Practices](http://blog.woorank.com/2014/11/breadcrumb-navigation-guide/) (Another great resource on types and usage of breadcrumbs)

### Icon Fonts

* [Why And How To Use Icon Fonts](http://vanseodesign.com/web-design/icon-fonts/) (a good overview of icon fonts)
* [Icon Fonts are Awesome](https://css-tricks.com/examples/IconFont/) (another good introduction to icon fonts)
* [Font Awesome](http://fontawesome.io/) (one of the most popular icon fonts)
* [Get started with FontAwesome](http://fontawesome.io/get-started/) (good official help)
* [Bootstrap-Social](http://lipis.github.io/bootstrap-social/)
* [The Final Nail in the Icon Fonts Coffin?](http://www.sitepoint.com/final-nail-icon-fonts-coffin/) (a controversial opinion piece on icon fonts)
* [Using SVGs](http://gomakethings.com/using-svgs/) (alternative to icon fonts)

### Other Useful Resources

* [The Difference Between Anchors, Inputs and Buttons](http://davidwalsh.name/html5-buttons) (Semantic differences in the usage)
* [When To Use The Button Element](https://css-tricks.com/use-button-element/) (The multifaceted button element)

Closing Thoughts

Week 3

Notes

Links

Closing Thoughts

Week 4

Notes

Links

Closing Thoughts

# Closing Thoughts