Coding & Cocktails Session 13:
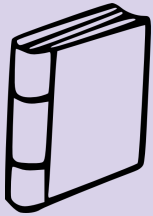## Intro to Front-End Architecture

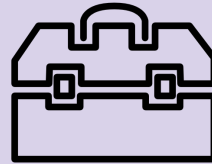## Overview

Basics of how to structure your websites and applications for more efficient and sustainable management.

Don't know a term?	**First time at C&C ?**
Check out our glossary bit.ly/CnCgloss	**Install recommended tools** bit.ly/CnCTools

Project

Front-end architecture is the organization of code and project files. Though there can be customizations specific to a team, the goal of good front-end architecture is to provide an efficient workflow that is maintainable throughout the project's life.

In this session, we will become familiar with best practices for front-end architecture by organizing a pack of assets in a way that will make it easy to maintain the website in a sustainable way.

Part 0: Verify NodeJS is installed

In order to use NPM later in this lesson, you have to install NodeJS. (We won't be writing a NodeJS application, but the two are essentially installed together.)

1. Open your terminal / command line

> **Helpful tip:**
>
> The "terminal" and "command line" (aka CLI, command line interface) are the same thing.
>
> - On Windows, this is **Git Bash**
> - On Mac, this is **iTerm2**

2. In Git Bash (windows) or iTerm2 (macs), type: **node --version && npm --version**
3. If you get version numbers, then NodeJS and NPM are already installed. Proceed to Part I.

Command not found

If your system doesn't recognize the node command, it's probably not installed. You can get it from **http://nodejs.org**

## Part I: Manual Practice

First, let's take a look at how to manually organize your project files.

1. Download the assets at http://bit.ly/2bVHBAi and unzip the file.
2. Create a root directory named **app** in your CodingAndCocktails folder.

Helpful tip:

● The CREDITS, LICENSE, and index.html files will live here for the website we'll be previewing. These contain information on copyright information and credits for any licensed objects. The template downloaded is licensed by templated.co, a great resource for website templates.

3. Let's get organizing! Create the necessary folders and add your project files like in the following diagram.
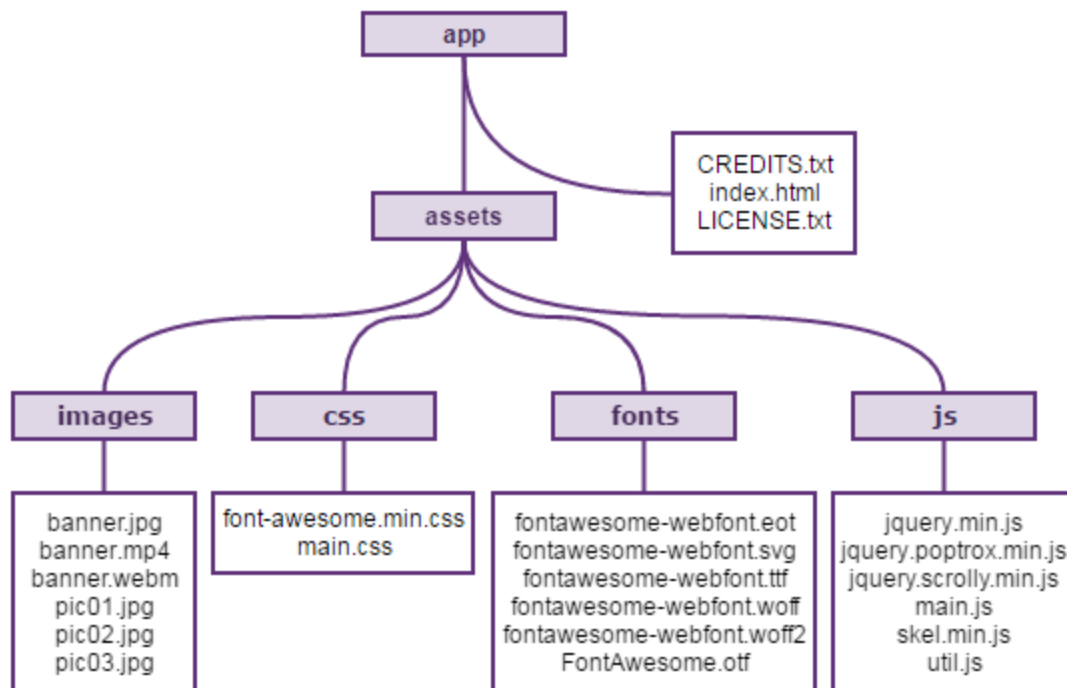
4. Now open the *index.html* file in Google Chrome to preview your site. Oh no! Looks like the site is broken. We're missing the styling and some images specifically
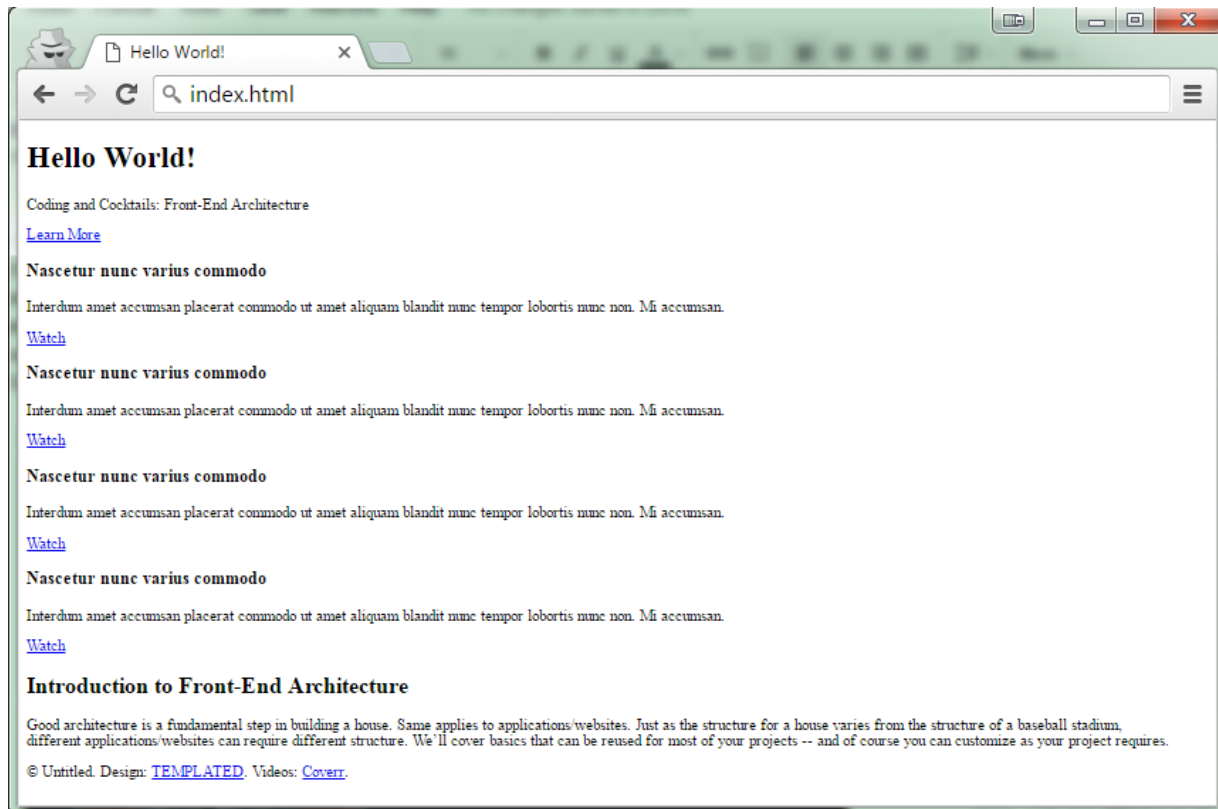
5. This is because our HTML page doesn't know how we've organized our project so we'll need to tell it where to find the files it needs. Open *index.html* in SublimeText.
6. Remove the comment marks `<!-- REPLACE DIR PATH` where applicable. Also remove the closing comment mark, `-->` at the end of the same lines.
7. You'll want to replace the `DIR` in those lines with the path of the file referenced. In the example below, that would be `assets/css/main.css`.

   a. Before:

   ```
   <html>
     <head>
       <title>Hello World!</title>
       <meta charset="utf-8" />
       <meta name="viewport" content="width=device-width, initial-scale=1" />
       <!-- REPLACE DIR PATH <link rel="stylesheet" href="DIR" />-->
     </head>
     <body id="top">
   ```

   b. After:

```
<html>
  <head>
    <title>Hello World!</title>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1" />
    <link rel="stylesheet" href="assets/css/main.css" />
  </head>
  <body id="top">
```
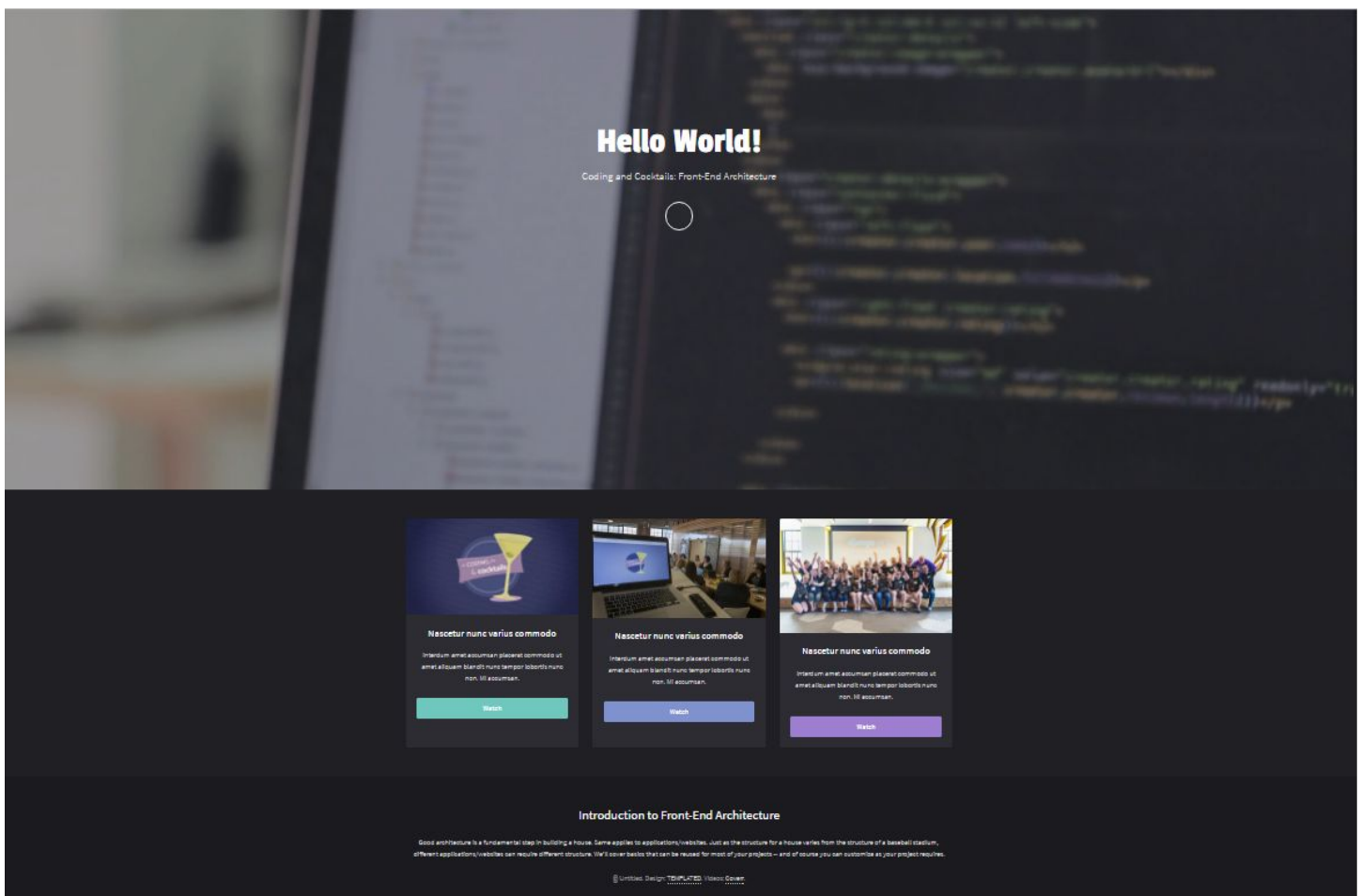
8. Go through the rest of the *index.html* file and make the same necessary changes for the other lines beginning with `<!-- REPLACE DIR PATH`. Once done, save *index.html* and reload it in your browser.

Your site should now be working! We've organized all of your files AND *index.html* knows where they are located.

Nice work!

Take a break and grab another drink.

You've earned it!

Part II: Scaffolding with Yeoman

You just did a great job with manually applying good front-end architecture practices! There are also tools which will help you automate this process of scaffolding your app. The term 'scaffolding' refers to creating a predefined structure for your app. Let's go over using Yeoman, one such toolset. We'll be installing the [TodoMVC](#) app.

Part 1: Create a project folder

When you're starting from scratch, an empty project is simply an empty folder. You could do that visually (using your file explorer) but let's do it in Git Bash (windows) or iTerm2 (macs).

> **Command line woes?**
>
> **Revisit the command line worksheet from March:**
> bit.ly/cmdln

1. Navigate to your CodingAndCocktails folder: **cd <your home directory>/CodingAndCocktails**

> **Helpful tips:**
>
> - Your home directory is:
>   - Mac: /users/<yourUsername>
>   - Windows: C:/Users/<yourUsername>
> - Command to change folders: cd <folderToGoTo>
> - You can also use **~** as a shortcut on your Mac Terminal or in Git Bash for your home directory: cd ~
> - Command to make a folder: mkdir <newFolder>
> - Most command line applications are not case sensitive, but a few of them are!

2. Make a new folder called mytodo: **mkdir mytodo**
3. Move into that new folder: **cd mytodo**

> **Helpful tip:**
>
> - If you start typing the name of a folder or file, hit tab and it will autocomplete
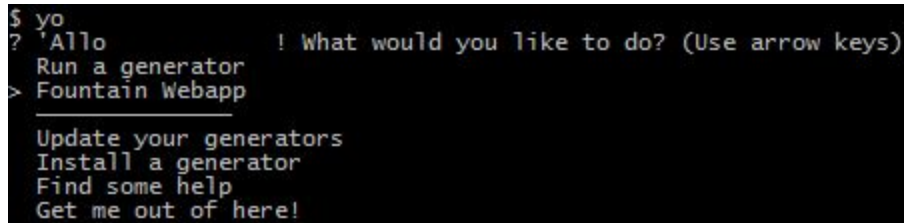
Part 2: Install Yeoman

Yeoman is a tool for setting up your project structure which can be used throughout a variety of projects. You can run these commands from anywhere, since we'll be installing Yeoman globally.

1. In Git Bash (windows) or iTerm2 (macs), type: **npm install -g yo**
2. You can confirm the installation by typing: **yo --version**

Part 3: Install a generator

There are over [3500+ generators](#) available for Yeoman. A generator is a plugin that can be used to scaffold your app. A plugin is software that complements and integrates into an existing application.

1. Let's install the [Fountain Webapp](#) generator using in Git Bash (windows) or iTerm2 (macs), by typing: **npm install -g generator-fountain-webapp**
2. Run **yo** to see your generators.

```
$ yo
? 'Allo          ! What would you like to do? (Use arrow keys)
  Run a generator
> Fountain Webapp
  ──────────────
  Update your generators
  Install a generator
  Find some help
  Get me out of here!
  ──────────────
```

> **Helpful tip:**
>
> ● You can run these commands for extra help if you run into trouble...
>   ○ **yo --help**
>   *for help with yeoman*
>   ○ **yo fountain-webapp --help**
>   *for information on the fountain-webapp generator*
>   ○ **yo doctor**
>   *detect potential issues*

Part 4: Configure your generator

There are custom options available for the fountain-webapp generator. Let's make our configuration selections now.

1. Select the 'Fountain Webapp' option after running the **yo** command.

> **Helpful tip:**
>
> ● Make sure you are in the folder you want to scaffold into before continuing.
> ● You can use **pwd** to see your current directory and **cd** if you need to change directories.

2. Make the selections in the following screenshot.

3. Once Yeoman is done, explore your directory structure using SublimeText or explorer to see what scaffolding has been created. You'll notice that the structure is different than what we saw with the manual practice on this worksheet. This is a good example of two different application types that use two different types of directory structures.

Part 5: Preview your app

Now, let's run a NodeJS web server to preview our app.

1. In Git Bash (windows) or iTerm2 (macs), type: **node run serve**
2. Go to localhost:3000 in your browser.

> **Helpful tip:**
>
> ● When you run the **node run serve** command, you are telling node to start a server that uses the http protocol to listen on port 3000.
> ● Ports are communication endpoints that exist between operating systems.
> ● Port 3000 is a local port, so a web server listening on that port will only be available on your local system. You'd typically use ports 80 or 443 if you wanted to publish a site to the internet.
> ● Once you're ready to stop your http server, you can enter **Ctrl+C** to cancel out of your command.

Homework

The more you practice, the better you'll get. Reinforce what you've learned tonight with the following tutorial.

Hey Slacker!

**Remember, we're here to help.
Join the KCWiT codingandcocktails
Slack Channel:
kcwit.slack.com**

Part 1: Another Yeoman tutorial

If you need some more practice, you can follow the tutorial at http://bit.ly/2ciNXfC. You'll be setting up a directory structure using Yeoman with another generator. The tutorial also introduces you to Grunt, a task runner, and Bower, a package manager.

Part 2: Git

Yeoman initializes a git repository for you in your source folder. Check git by running **git status**. You can also stage and commit your files to your own Github repository if you'd like more practice.

Don't remember git?

**See the git version control
worksheetfrom April:
bit.ly/CnCvers**

Part 3: Advanced Topic: Creating your own generators

You can also create your own Yeoman generators for any custom projects you're working on. Just follow along with the tutorial at http://yeoman.io/authoring/.

Part 4: Advanced Topic: More Yeoman

You can go further with Yeoman by configuring testing for your app, getting your app ready for production, or making your todos persistent with TodoMVC . If you'd like to learn more, follow the tutorials at:

Step 6: Test with Karma and Jasmine
Step 7: Make TODOs Persistent with Local Storage
Step 8: Get Ready for Production