

git clone <https://github.com/ron2015schmitt/project-web-dev-workflow.git>

now clone his files

git clone <https://github.com/LearnWebCode/travel-site-files.git>

now copy his files, excluding his .git/ files

mkdir project-web-dev-workflow/travel-site-files/

cp -r travel-site-files/app/ project-web-dev-workflow/travel-site-files/

cd project-web-dev-workflow/

ls travel-site-files/

verify that we are still pointed at our own repo, not his

git remote -v

npm install lodash

$ ls

app/ node\_modules/ package-lock.json

now create the initial package.json file

$ npm init -y

Wrote to F:\development-environments\rs2015\project-web-dev-workflow\travel-site-files\package.json:

{

"name": "travel-site-files",

"version": "1.0.0",

"description": "",

"main": "index.js",

"dependencies": {

"lodash": "^4.17.21"

},

"devDependencies": {},

"scripts": {

"test": "echo \"Error: no test specified\" && exit 1"

},

"keywords": [],

"author": "",

"license": "ISC"

}

$ ls

app/ node\_modules/ package-lock.json package.js

If package.json has changed on git, you can update the project using npm install. Likewise if you clone the repo from git.

He recommends the following although I haven’t performed this yet

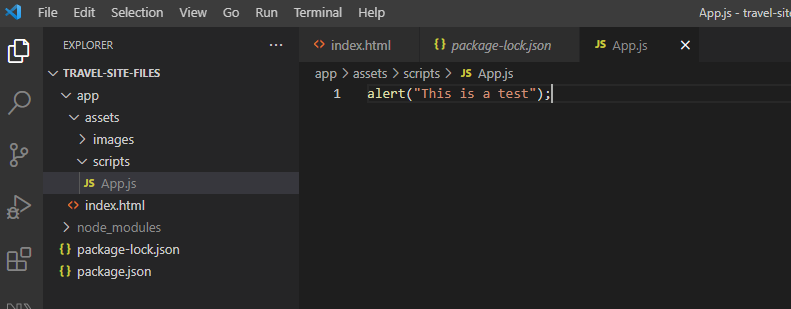
|  |
| --- |
|  |
| Important Note About Package Versions To Save You Frustration  Hello everyone,  In the real world you'll usually want to use the newest version of a package that is available on NPM. However, to make it easier to follow along with the video lessons in this course I strongly encourage you to use the same versions of packages that I'm using. Here's how you can achieve that:   1. Within your project folder **delete** the node\_modules folder and also **delete** the package-lock.json file. 2. You'll see a downloadable file associated with this lesson that is named package.json. Download this file and replace the package.json file that is currently in your project folder with this newly downloaded file.  If you've already made it to the later chapters of this course and are just now circling back to this lesson you will have a **"scripts"** area in your existing package.json file; copy your existing **scripts** object code into your clipboard before replacing your package.json file so that you can paste your existing **scripts** area back into the file you download from this article. This way you don't lose the npm run dev and npm run build tasks you've already setup. 3. In your command line run npm install and be aware that it could take a few minutes because it is installing all of the packages that we'll need for the entirety of our Complex App.   That's it! This gives you two huge benefits:   1. You avoid the frustration of code working in my video lessons for me, but it not working for you. The creators of NPM packages update their packages weekly or monthly and this will change the syntax and options that they demand in order to work properly. Obviously, it's not feasible for me to re-film and recreate the entire course every week as new versions of packages are rolled out. You can now rest assured that you're using the same package versions as I am in the video lessons and this can save us a lot of frustration. 2. Now while we are building the application together you don't need to worry about jumping to your command-line to install the packages that I mention. You already have **ALL** of the packages you'll need for this project / application.   Enjoy! Brad |

|  |  |
| --- | --- |
| node  node script.js | This runs the Javascript runtime. It has different functions from the Chrome/browser runtime because it has access to the file system and not a DOM. |
| npm install [package] | The package loader for node. By default the package is loaded in the current working directory. If –g is present it is loaded globally for the user account (in windows this is in C:\Users\rs2015\AppData\Roaming\npm\node\_modules\) |

Finished Through Chap 12

**Chap 13 Webpack Introduction**

Add scripts folder and App.js files to project as shown below

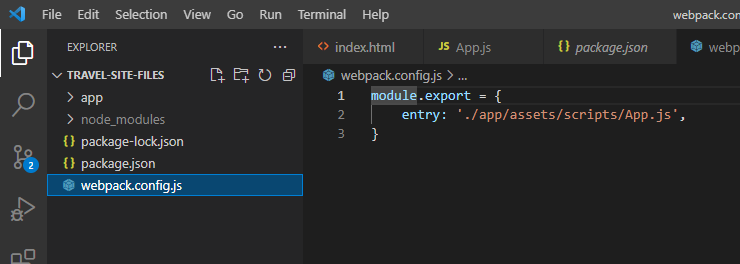


Install webpack

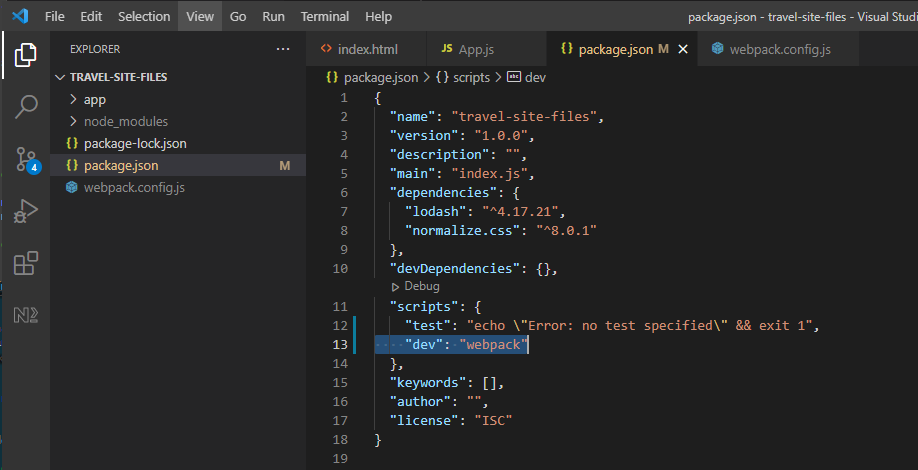
cd ~/project-web-dev-workflow/travel-site-files

npm install webpack webpack-cli --save-dev

create webpack config file as shown below. I wrote module.export it should be module.exports



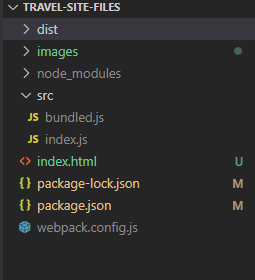
modify the package.json file, adding the webpack line



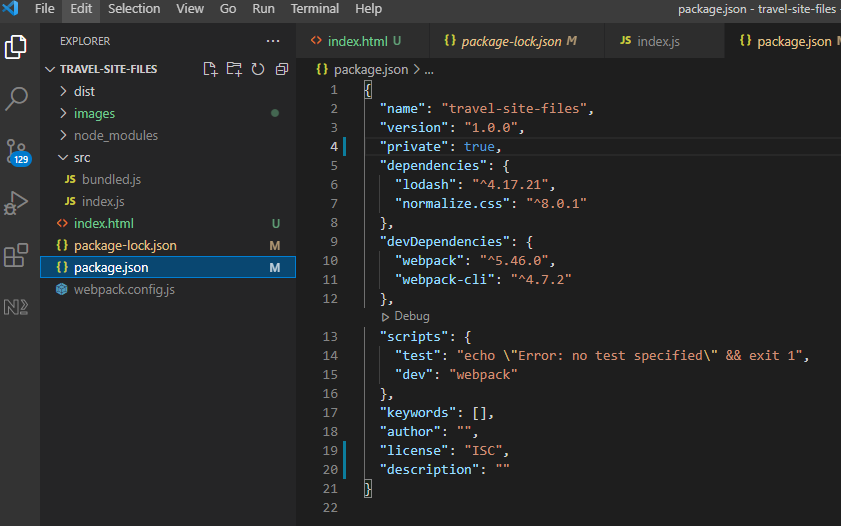
Class instructions are out of date. I was guided by the following instead

|  |
| --- |
| <https://webpack.js.org/guides/getting-started/> |
|  |

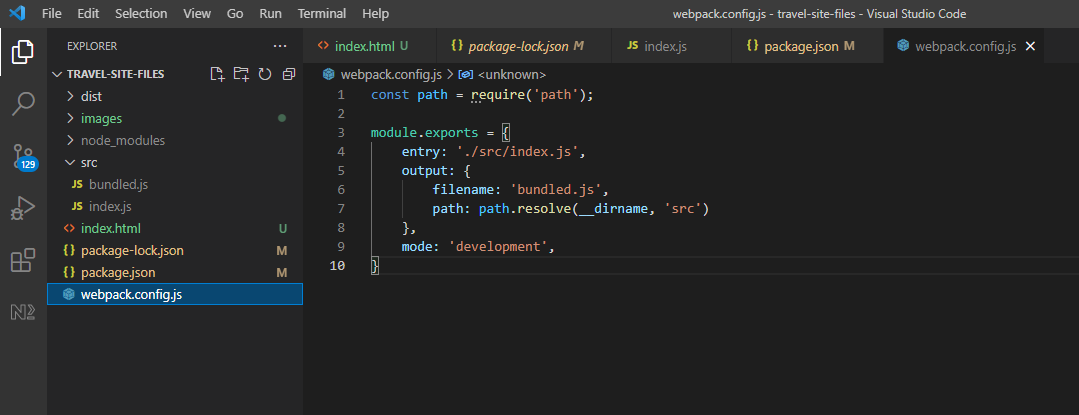
I changed the directory structure to:



package.json is now



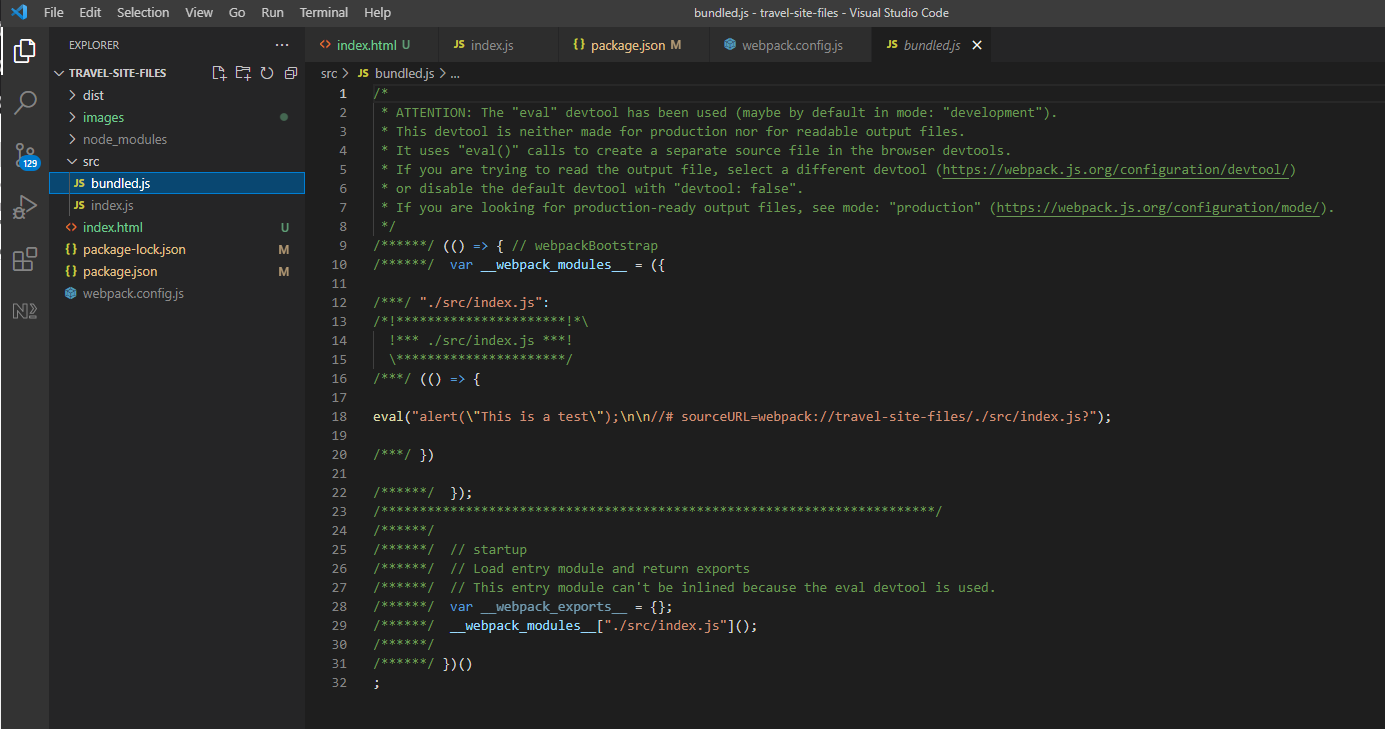
Changed webpack.config.js to



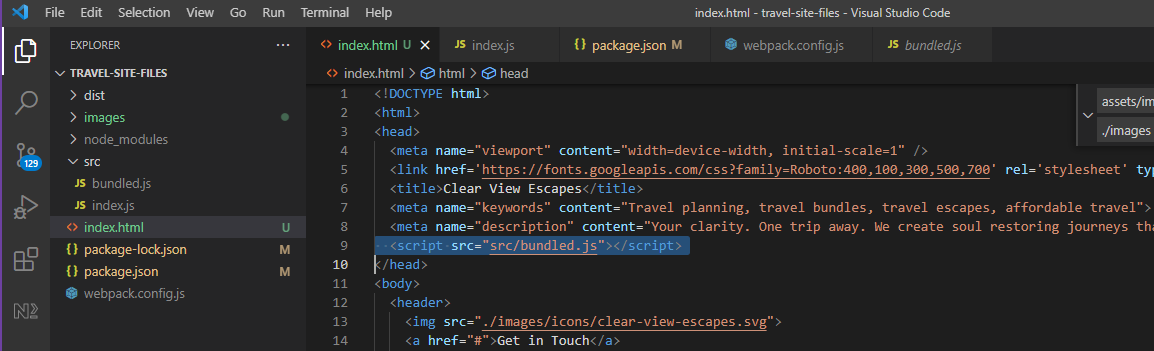
Now run the dev script (which runs webpack)

npm run dev

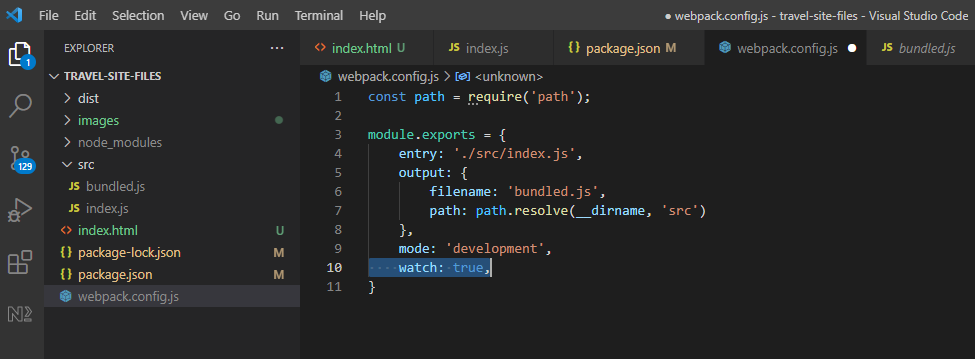
This creates the following file



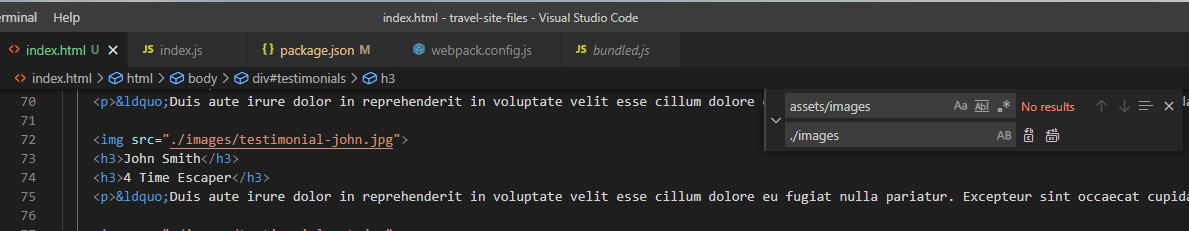
Now add the following script line at the end of the <head> section in index.html. (Instructor prefer body section, but head makes more sense)



Next add the following line to make webpack stay running and automatically update the webpage on any source file change



Now since we changed the directory structure, we need to search and replace for the image urls in the index.html file



Now re-run the dev script

npm run dev

notice that it doesn’t return to the prompt anymore

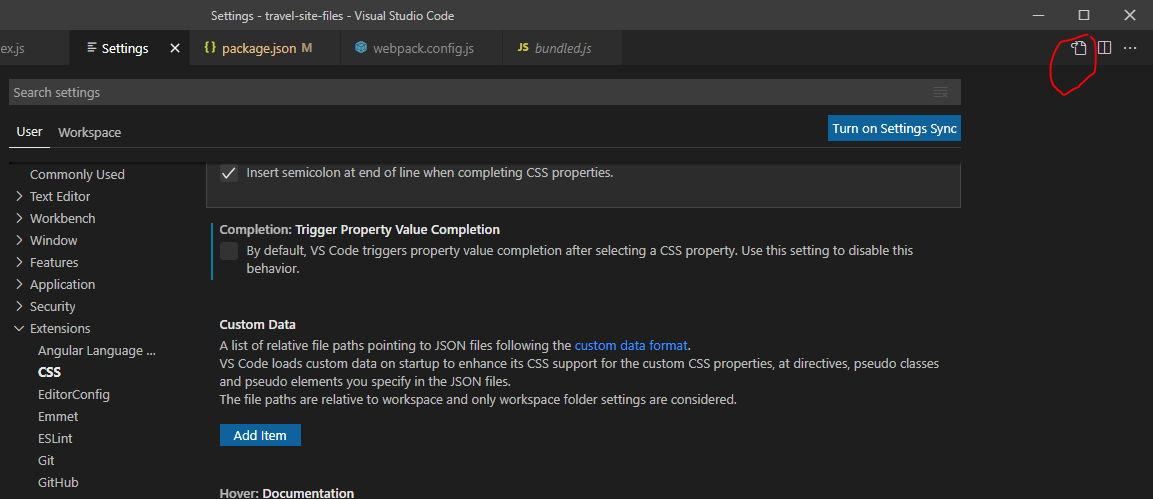
open the webpage at

<file:///F:/development-environments/rs2015/project-web-dev-workflow/travel-site-files/index.html>

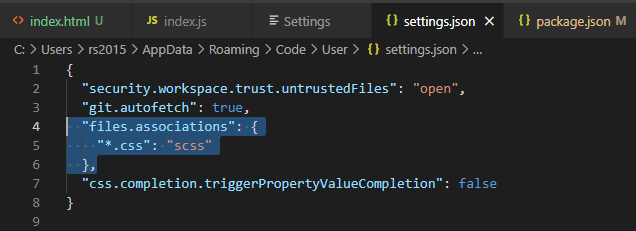
Now if we change the index.js file, webpack recompiles. We still have to hit F5 in the browser though to pick up the new changes

**Chap 14 Set up your VSCode to understand PostCSS syntax**

In VSCode, type Ctrl-, to bring up the settings



Click on the icon as shown above and add the lines shown below



Chap 14 done