What is NPM and Node.js? Why do we need it for this project?

Node is a server that runs on JavaScript. Remember, a server is a computer that has a lot of programs on it that can interact with another computer. A web server is a computer that runs web apps through the browser. We will eventually use a web server so users can access your project when you’re ready to put it on the internet.

NPM stands for Node Package Manager. NPM is automatically installed when you download Node.js. NPM packages extend the capabilities of JavaScript because they are libraries built by other developers that can be used in various project types. A benefit of using NPM is that you don’t necessarily have to write a new library with useful functions for your project because there might be a node package already that has the capabilities that you need. By using NPM, you will save development time and many of the packages available have been tested by many other developers. You can also easily update the versions of packages that you are using. Many different web developer tools have been written in Node. Some examples of NPM packages are jQuery (which you are familiar with), moment (which is a package that does a lot of date/time manipulation for you), etc. The names of some Node packages that we’re going to use today are gulp, yo, and browserify.

What is a dependency?

A dependency is a piece of software that relies on another. For example, in your project, jQuery will be a dependency because your code will rely on jQuery to work properly.

What is bower?

Bower is a package manager. It will install the correct versions of the dependencies that you are using.

What is gulp? What is a bundler? Why is a bundler important?

Gulp is a task manager. There are lots of tasks that you can create as a developer, including running units tests, compiling your styling, and bundling your scripts. When you use Gulp, it takes care of running all of those tasks for you so you don’t have to write a ton of configuration for your project. You just set up your gulpfile.js to run all of the tasks that you specify for certain files.

One thing that we specifically want to use Gulp for is to bundle together our JavaScript files. Bundling is an important step in the development process. When you bundle together your JavaScript files (or other files like CSS or HTML), those individual files are taken and stitched together into one large file in the correct order. Otherwise, if you have a bunch of scripts that are getting called from your index.html file, it can take your browser a long time to load so many files because it has to keep making requests to your code base. By bundling your scripts, the browser only has to make one request to get your code, so your code is served up faster for the end user!

What is a CLI?

CLI stands for command line interface. A CLI allows you to interact with a program (like Gulp) through your command line or terminal.

What is yo/yeoman? Why is scaffolding important?

Yeoman (in your command line, it is called yo) is a scaffolding tool for new applications. When you scaffold a project, you build out the structure of the files you want to use based on a specific type of programming pattern, for example the Model-View-Controller pattern. Another way of saying that, is a scaffold is a way of architecting your site code base. With Yeoman, the developer is walked through the scaffolding process by answer questions about additional packages that they would like to use in their site build. For example, Yeoman asks if we want to use the ReactJS library. Using scaffolding, while not always necessary, can be a way to standardize your code so other developers can easily understand it. If you use a tool that generates the scaffolding for you, you can save a lot of time programming and quickly get your project up and running.

What is generator-webapp?

Generator-webapp is a plugin, or component that extends a piece of software’s capabilities, that is used with yeoman. It specifically uses Gulp for your build process.

What does ‘gulp serve’ do?

This command spins up a web server so that you can see what your project will look like in the browser. Gulp uses port 9000 on localhost.

What does ‘gulp build’ do?

This command runs through the tasks that are configured in your gulpfile. For example, it can bundle your JavaScript files and put them into bundle.js. All of the files that have been bundled together or that have been modified by running your build will be housed in your /dist folder. Running this command can also make changes to your build if you made modifications to your gulpfile.

What is Browserify for? What are JavaScript modules? Why is it important to separate your concerns?

Browserify allows you to export variables and functions that are declared in one file and import them into another file so they can be used in various locations. For example, if you wanted to keep the variable for your API endpoint in one file, separate from the others, you can export that variable and import it into another file so you don’t have to keep re-typing the domain in all other files. The process of keeping different layers of your code in separate files and exporting/importing them is called modularity. We will strive to make our code as modular as possible, meaning we will separate out the different functionalities of our application into different parts. For example, we will keep all of the events, like click events, in one file. That will be in a separate file from the API calls that we make to retrieve data from our database. Another way of saying this is that we are ‘separating our concerns’, or breaking our code up by category. By maintaining our code in this way, we can easily update one part of our code, again back to the API endpoint variable, and the change will affect the rest of our code base but will only have to be updated in one single spot. If we weren’t modular in our approach, we would have to search through our code and update the API endpoint everywhere it appeared in our code.

How are we going to make our code modular:

* Keep the API endpoint variable in a separate file (app.js)
* Keeping the events that occur in our program in one file (events.js)
* Keep the actual calls to the database in one file (api.js)
* Keep the program response, or User Interface, to the API calls or user events in one file (ui.js)
* Go through HTML file
* Add app.js file and discuss modules

Discuss synchronous vs asynchronous code

<https://www.pluralsight.com/guides/front-end-javascript/introduction-to-asynchronous-javascript>

Synchronous code is where one line of code is read by the interpreter, then the interpreter does what is indicated in that line, and then moves on to the next line and interprets that and so on. The second line cannot be run before the first line is finished running. An example of this kind of behavior would be when you’re waiting in line to get tickets for the movies. You can’t cut ahead of anyone before you.

Now, JavaScript is actually asynchronous. This means that when you have one line of code, the interpreter could still be working on accomplishing what’s written in that line, but it moves on to the next line of code. So, you don’t have the lines interpreted and executed one by one, they are interpreted and then you don’t know when their execution finishes. For example, it’s like eating at a restaurant. You can order your food, and so can other tables. If another table ordered before you, you don’t have to wait to receive or eat your food before they are finished eating theirs, right?

One particular place we will see the importance of programming in a way to force our code to be synchronous is with our AJAX calls to the API/database. When we have a click event, then it calls the API, then we either get a success or error and the program has to respond accordingly (like display data or an error message), we want the lines to be executed in the way we set them up to be executed. We don’t want the interpreter to try to run the succeed function before an API call is even completed. So, one way we can do that is by using Promises in our code.

Promises force the code to be synchronous by using callback functions in a more standard way. What’s a callback? It’s a function that calls another function. So, when we use Promises, we can have one function run, and then when it is finished, it returns either a success or failure and then calls another function.

What is AJAX?  
It stands for Asynchronous JavaScript and XML. Don’t worry about the XML part, jQuery and other libraries can take care of that for you. What AJAX does is it requests data from the server, receives data from the server, sends data to the server, and it doesn’t update the webpage by reloading it. You won’t see the page refresh when we make an AJAX call.