Jacob A. Nelson

COMP.4610 GUI Programming I

Summer 2023

Prof. Wenjin Zhou

Readings Summary for Assignment #1

This is a summary of the assigned readings for Assignment #1 for GUI Programming. This summary will mostly focus on one subject in particular: npm, which stands for “Node Package Manager.” In the full-stack developer roadmap, there is an excellent article called “Modern JavaScript for Dinosaurs” which explains in detail the benefits of npm and how to use it. I chose to discuss this reading in particular because I have never used npm, and it seems like a critical tool for programming in JavaScript. I learned a lot from this article, and I will share what I have learned from it.

Node Package Manager (npm), is a command-line package manager for JavaScript. Before package managers like npm, if one wanted to use an external library in their code, you would have to explicitly refer to that particular package, either in the JavaScript or in a script tag in your HTML. This was easy enough to do, but was really inconvenient to keep track of library updates. The idea behind a package manager is to take care of all of the library management for you.

To use npm, you must install Node.js, and once you have done that, you can use the npm command on your system’s command line. To include npm in your JavaScript/web project, navigate to the project folder and enter the command **npm init** to initialize npm for this project. You will be asked a bunch of questions about the configuration, and once that is completed, a new file, **package.json,** is created, which is a JSON file containing the information about your project and what libraries and dependencies it is configured for.

To install a particular package, you enter the command **install [package] --save.** In the article, the author uses a date/time library called ‘moment’ so the command entered was **install moment –save.** This command downloads all code from the moment.js package into a folder called node\_modules. It also modifies package.json to include “moment” as a dependency. Now, you can use the functions and modules within moment.js in your code. However, you still need to reference the node module in your code. There is a process to automate this task as well, and that involves using a JavaScript module bundler (a.k.a, webpack).

Using a module bundler saves you the headache of having to dig through your node\_modules folder to find the exact path of the library you are trying to use. The module bundler, once installed and configured, handles all of the file management across all libraries into a single bundle your program and use. To install a module bundler into your project, run the command **npm install webpack webpack-cli –save-dev.** This installs both webpack and the CLI version of webpack. Now, to use the exampled ‘moment.js’ package, instead of having to use require(‘./node\_modules/moment/[..the particular module.js]), you can instead simply use require(‘moment’) and you’re all set to use it.

Using a package manager in combination with a module bundler is a great way to manage your libraries and packages used in your project. It is easy to see why it is imperative to use a package manager and module bundler for large scale team projects

References:

Full-Stack Developer Roadmap: <https://roadmap.sh/full-stack>

Modern Javascript for Dinosaurs: https://peterxjang.com/blog/modern-javascript-explained-for-dinosaurs.html