Node.js and NPM on Windows

Since you are going to use JavaScript for your React development on Windows, there is no way around Node.js and NPM. Node.js is a JavaScript runtime which makes it possible to run JavaScript outside of the browser. The underlying node package manager (NPM) is used to install libraries, such as React.js, to your project on the command line. You will see later on how this works out on the command line for Windows users.

In order to install Node.js for Windows, you need to head over to the [Node.js website](https://nodejs.org/en/) and download the Windows version which is recommended for most users. Afterward, you should install it. NPM comes automatically with Node.js.

Note: An alternative way to install and manage Node.js on your Windows machine is [nvm-windows](https://github.com/coreybutler/nvm-windows" \t "_blank). If you are just getting started with web development, I wouldn’t recommend it in the beginning. But it can be a powerful asset later on. nvm-windows is the equivalent to the popular [nvm](https://github.com/creationix/nvm" \t "_blank) package for Unix systems. It’s called Node Version Manager. It helps you to upgrade Node.js version and to manage multiple Node.js versions on your PC. I didn’t try it myself yet, but I hope it just works similar to the MacOS version.

Visual Studio Code for React on Windows

There are plenty of editors and IDEs out there to develop web applications. Depending on your personal, project or company requirements, you can choose from a range of lightweight editors to full-blown IDEs. Visual Studio Code is a solution in between. Many developers, operating on Windows but also MacOS, enjoy using it. That’s why my recommendation would be Visual Studio Code to start developing React applications on Windows. The installation on a Windows machine is simple: Navigate to the official [Visual Studio Code](https://code.visualstudio.com/) website and download the recent version of VS Code in 32 or 64 bit. Afterward, install it and check if everything is working when you open it. That’s it for the installation of an editor/IDE on your machine.

If you want to use another editor/IDE, it is up to you. But keep in mind that a couple of the following steps are building up on top of Visual Studio Code. However, it should be possible to substitute these steps for your own editor/IDE.

NPM on the Command Line for create-react-app

If you are just getting started with web development, you should create a dedicated folder on your machine to manage all your web development projects. It’s up to you to divide the folder into subfolders. For instance, there could be a folder for React applications and another one for plain Node.js applications. Once you have your folder for your projects, open this folder in Visual Studio Code.

In Visual Studio Code you should be able to open a tab which is called “Terminal” at the bottom. That’s your integrated command line in Visual Studio Code to install node packages with NPM or to start/test your project. It’s up to you to use the integrated terminal or another command line interface for your Windows machine.

Now, you should check whether the Node.js installation for Windows was successful. On the command line, type the following commands. They should output the versions for Node.js and NPM. The following versions may vary from your versions.

node --version

\*v8.9.4

npm --version

\*v5.6.0

In most cases, it should just work for you. If node or npm are no valid commands, you need to check your environment variables in Windows. But I didn’t run into this issue anymore in the past when I worked together with Windows users on React applications.

Now you can install your first node package with npm on the command line. You will install it globally with a -g flag. Because of installing it globally, you will always have access to it on the command line. Later on, when you install a node package for your project without the -g flag, you will only have access to the node package (node module) in your project. On the command line (in Visual Studio Code), type the following command to install [create-react-app](https://github.com/facebook/create-react-app):

npm install -g create-react-app

This package allows you to bootstrap React applications with zero-configuration. There is no need to get involved too early in toolings with [Webpack and Babel](https://github.com/rwieruch/minimal-react-webpack-babel-setup). If you are going to read “The Road to learn React”, you will use create-react-app as well. It’s the simplest approach to learn plain React without worrying about all the tooling with Webpack and Babel around it. After installing it, you can check its version again on the command line:

create-react-app --version

\*v1.5.1

Finally, you can bootstrap your first React.js application on Windows. You can use create-react-app by passing the name of your application to it on the command line:

create-react-app my-app

Afterward, you can navigate into the project on the command line and start it with npm:

cd my-app

npm start

The command line should give you an output where you can find the application in the browser. The default should be localhost:8080. If you are only using IE or Edge on your Windows machine, I can recommend you to install Chrome as well to access the developer environment and the React Developer Tools which are available as Chrome extension. On the other side, now you have an advantage over the MacOS developers, because you can debug your web applications in the Internet Explorer and Edge too.