Modules have been available in JavaScript through third-party libraries. ECMAScript 6 adds native support for modules to JavaScript. When you compile a modular ECMAScript 6 application to ECMASCript 5, the compiler relies on a third party library to implement modules in ECMAScript 5. [Webpack](http://webpack.github.io/) and [Browserify](http://browserify.org/) are two popular options, and Babel supports both (and others). We use Webpack in this tutorial.

In this unit, you add Webpack to your development environment.

1. On the command line, make sure you are in the es6-fundamentals-student directory and install the **babel-loader** and **webpack** modules:

npm install babel-loader webpack --save-dev

1. Open **package.json** in your code editor, and add a **webpack** script (right after the **babel** script). The scripts section should now look like this:

"scripts": {

"babel": "babel --presets es2015 js/main.js -o build/main.bundle.js",

"start": "http-server",

"webpack": "webpack"

},

1. In the es6-fundamentals-student directory, create a new file named webpack.config.js defined as follows:

## 

## Build Using Webpack

1. On the command line, make sure you are in the **es6-fundamentals-student** directory and type the following command:

npm run webpack

Webpack uses Babel behind the scenes to compile your application. You can build an application using Webpack even if that application is not using ECMAScript 6 modules. In other words, the **babel** script in package.json is not needed anymore.

1. Open a browser, access [http://localhost:8080](http://localhost:8080/), and click the **Calculate** button.