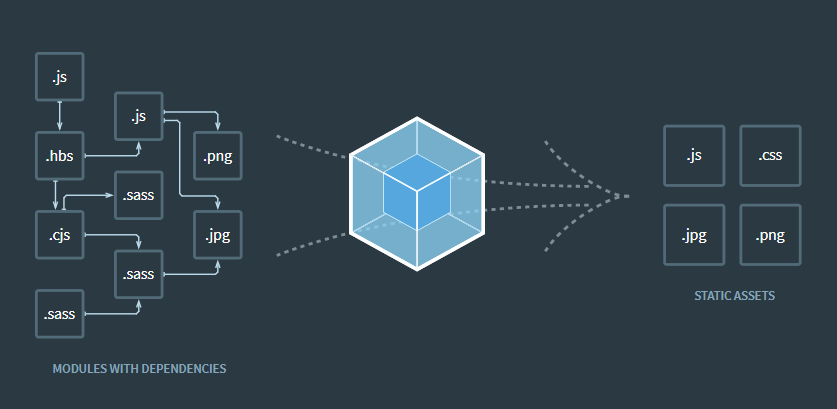
# Complete Webpack Starter Guide

1. *This article is a beginners guide to webpack. It contains a list of things you should know when thinking of starting a new javascript application. It is basically a list of things I wish I knew before starting with webpack.*
2. *You can find the finished code for this article on my github page* [*@Dsaint109*](https://github.com/Dsaint109) [*https://github.com/Dsaint109/webpack-project*](https://github.com/Dsaint109/webpack-project)



In modern web development, there is a swarm of cool tools and packages (modules) you can include into your project to have access to some advanced features without having to reinvent the wheel. With **npm (Node Package Manager)**,you have the power to include any package into your project.

With great power comes great responsibility. When including multiple packages and modules, it is very important to have a tool that handles the bundling, minification and other really important logics of packages and modules.

1. *So what is webpack? Webpack is a module bundler.*

# Table of Content

1. Requirements and Installation
2. Hello World!
3. Webpack Dev Server
4. Style Loaders
5. Babel + ES2015
6. File + Image Loaders
7. Add Bootstrap
8. Extras

# Requirements and Installation

So we begin our adventure into module bundling with webpack, but first things first we need to do the basic installation.

To get started, you should have these:

* + Node js (you can download it from the official site [here](https://nodejs.org/en/))
  + Basic Javascript knowledge
  + Any text editor ([Sublime](https://www.sublimetext.com/3) Rules!!)

**Create a new project**

We start by creating a new javascript project (obviously). Using npm we can initialise the present working directory for our project as a javascript project. This is done by running this command in your terminal:

npm init

1. *Note: This should be done when your terminal is cd’ into the directory of your project*

Then follow the instructions based on your preference.

**Installing Webpack Globally**

Since we would be using webpack alot, it would be preferable to install the necessary packages globally. The packages we would be installing globally are webpack, webpack-dev-server and webpack-cli. This packages are required to ensure we get the most out of webpack.

To do this we just need to run this command:

npm install -g webpack webpack-dev-server webpack-cli

# Hello, World! in webpack

Don’t you just love “Hello, world”s? It makes everything so much better. We are going code the most basic of builds in this section.

In the root directory of our project we are going to create a new file webpack.config.js, It is from this file we get to configure webpack. We are also going to create some extra folders and files…

$ webpack-project-folder

      | dist

        | index.html

      | src

        | index.js

      package.json

      webpack.config.js

Our project should look like the structure displayed above.

1. *So we configure the basics of how we want our modules to be bundled in the webpack.config.js file.*

 We add this snippet of code to our webpack.config.js to configure the behaviour of the module bundler.

const path = require('path')

const config = {

  entry: './src/index.js',

  output: {

    path: path.resolve(\_\_dirname, 'dist'),

    filename: 'app.bundle.js'

  }

}

module.exports = config

So what we are doing here is,

* **We specify the entry point of our application *(line 4)*:** This is where most of our javascript would be written.
* **We set the directory of our out put to the dist folder *(line 6)***
* **We set the name of the output file *(line 7)***
* **We export our configurations *(line 11)***

In our dist/index.html file we should also add this lines of code.

<!DOCTYPE html>

<html>

<head>

  <meta charset="UTF-8">

  <title>Hello, World fam</title>

</head>

<body>

  <div id="root">

    <h1>Hello, World</h1>

  </div>

<script type="text/javascript" src="app.bundle.js"></script>

</body>

</html>

Then in our src/index.js

console.log('Hello, World!')

And finally, in our package.json we are going to add a script to the scripts object.

"scripts": {

  "start:dev": "webpack-dev-server",

  "prod": "webpack -p"

}

What this does is, it specifies the command we are going to use to in the terminal to launch our project in development mode or in production mode.

So to run our beautiful “Hello , World” webpack application, we go to the terminal and type this command.

$ npm run start:dev

Then you should be  able to access you site on localhost:8080