1. Webpack takes care of bundling and also the **order** in which the files are to be included in html.

# Setting up webpack for a project

* Go to the root folder and open CLI and run ***npm init –y***  which creates **package.json**
* To install new packages in npm 🡪 *npm install (name of all packages needed separated by a space)*  ***npm install webpack webpack-cli –save-dev***

# Webpack.config.js

This is the file which has all the details about how the webpack has to deal with different kinds of files and where to start the bundling and what tools to use to achieve the goals

It directs webpack to make use of the settings configured to achieve goals like bundling, importing images,css into JS files and minification as well

This file has below

|  |  |  |
| --- | --- | --- |
| Property | What it does | Type |
| entry | Specifies the file for webpack to start bundling from here | String | [String] |
| Output | Specifies where to place the bundled JS file with some other properties | Object |
| Loaders | Specifies various rules and what loaders to use if the rule return true for imported files in JavaScript | Object |
|  | Array of Rules  And each rule is an Object |  |

# Loader

* It helps all the dependencies to be built into single file
* Webpack only understands **js** and **JSON** files. All other files like css, scss, img are imported using loaders
* In the ***use*** property of rules , we should specify the loaders in reverse i.e, the order in which we want, first is to be placed last in the array of loaders. This is the way webpack is designed. Webpack process the loaders from **right to left**

{

            test: /\.(scss)$/,

            use: [

                 'style-loader','css-loader','sass-loader'

            ]

         }

Above rule shows the use of 3 loaders to be used on files ending with .scss extension. Its clear that **right to left** takes place during loaders processing in webpack

1. First the files with ***.scss***  are to be converted to css 🡪 hence we use ***sass-loader***
2. Next the converted css files have to be loaded into the JavaScript files 🡪 hence the ***css-loader*** comes next
3. Then the css styles have to be loaded into the html after creating a <style> tag which is taken care by ***style-loader***

So the order is 'style-loader','css-loader','sass-loader'

# Babel Loader

As of webpack basic installation, only ES5 is usable. To use features > ES6, ***babel loader***  is to be used for transpiling latest JS code back to browser understandable JS code i.e., ES5 code

{

                test: /\.(js)$/,

                exclude: /node\_modules/,

                use: {

                    loader: 'babel-loader',

                    options: {

                        presets: ['@babel/env'],

                        plugins: ['transform-class-properties']

                    }

                }

            }

# Plugins

A JavsScript library which provides some extra functionality such as minification of bundled js files.

# Clearing out dist folder before every build

Use ‘Clean-webpack-plugin ’ to clear out dist folder before generating new bundles

# Generate HTML files during build process

Use ‘html-webpack-plugin’

# Webpack-dev server

Used to instantly deploy the bundled changes and see the changes we made to our app on a designated server

<https://webpack.js.org/configuration/dev-server/>

# when nothing works out even after having clear webpack.config.js

just run the command ***npm cache clean –force***

<https://github.com/webpack/webpack.js.org/issues/2035>