Getting Started With React Project

Monday, March 21, 2022 2:55 PM

- One Min Application is good only for Hello World.
- Any decent React Project will use
  - Import/export
  - ES2015+ features
  - JSX
- We will need
  - React
  - ReactDOM
  - Babel
    - ES2015+ features
    - JSX
  - WebPack
    - To invoke Babel
    - Create Deployment.

Step #1 Create A new Project

```
c:\workspace> cd my-new-react-app
c:\workspace\my-new-react-app> npm init --y
c:\workspace\my-new-react-app> code .
```

- **Create your 'src' folder**

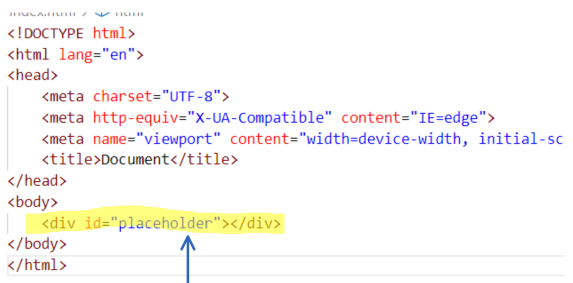
Step #2 Get React libraries using npm

```
c:\workspace\my-new-react-app> npm i react react-dom
```

Step #3 Create a Simple React Project (with react.CreateElement)

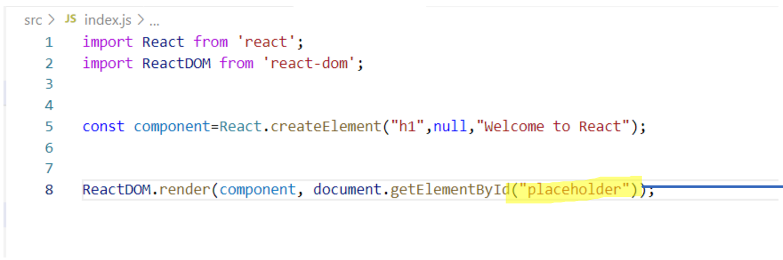
Step #3.1 Create Index.html

- **Leave a placeholder div**



```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-sc
    <title>Document</title>
</head>
<body>
    <div id="placeholder"></div>
</body>
</html>
```

Step #3.2 Create Index.js with required React Code



src > JS index.js > ...

```
import React from 'react';
import ReactDOM from 'react-dom';


const component=React.createElement("h1",null,"Welcome to React");


ReactDOM.render(component, document.getElementById("placeholder"));
```

- **Note**, we are using import/export here.
- We need webpack to resolve and run our code

Step #4 Configure basic webpack for simple use case

Step #4.1 install webpack essentials

- Install the common plugins

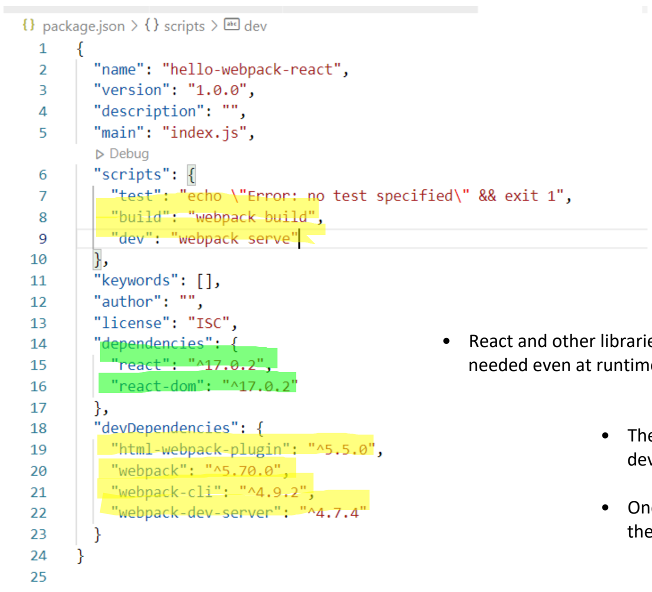
```
$ npm i --save-dev webpack webpack-cli webpack-dev-server html-webpack-plugin
```

Step 4.2 configure the webpack using webpack.config.js

```
const path=require('path');
const HtmlWebPackPlugin=require('html-webpack-plugin');
```

```
module.exports={
    entry:path.join(__dirname, 'src', 'index.js'),
    output:{
        path:path.join(__dirname, 'dist')
    } ,
    plugins:[
        new HtmlWebPackPlugin({
            template: path.join(__dirname,'src','index.html'),
        })
    ],
    mode: 'production'
}
```

Step 4.3 configure package.json scripts



```
package.json > scripts > dev
{
  "name": "hello-webpack-react",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1",
    "build": "webpack build",
    "dev": "webpack serve"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "dependencies": {
    "react": "^17.0.2",
    "react-dom": "^17.0.2"
  },
  "devDependencies": {
    "html-webpack-plugin": "^5.5.0",
    "webpack": "^5.70.0",
    "webpack-cli": "^4.9.2",
    "webpack-dev-server": "^4.7.4"
  }
}
```

- We will run our code using "dev" script while we are developing
- We build script when our development is complete and we need to send the file to the deployement
- By default only "dist" folder will be needed for deployment

- React and other libraries are needed even at runtime.

- These dependencies are needed only for devlopment.
- Once our code is build we don't need them at runtime

*

Phase 1 complete

- Now our application should run without any problem using

```
c:\workspace\hello-webpack-react> npm run dev
```

- The current application supports
  - Import/export
  - Packing

- The Current application DOESN'T SUPPORT JSX

Phase 2 — Enable JSX and ES2015+ support

Phase #2.1 Install babel and babel webpack plugins

- We need to install
  - Babel
  - Babel webpackplugins
  - Babel jsx plugins

```
c:\workspace\react-webpack-app> npm i --save-dev @babel/core @babel/preset-env
@babel/preset-react babel-loader
```

Phase #2.2 Configure webpack.confg.js

const path = require('path');
const HtmlWebPackPlugin = require('html-webpack-plugin');

module.exports = {

```
    entry: path.join(__dirname, 'src', 'index.js'),

    output: {
      path: path.join(__dirname, 'dist')
    },
    plugins: [
      new HtmlWebPackPlugin({
        template: path.join(__dirname, 'src', 'index.html'),
      })
    ],
    module: {
      rules: [
        {
          test: /\.?js$/,
          exclude: /node_modules/,
          use: {
            loader: "babel-loader",
            options: {
              presets: ['@babel/preset-env', '@babel/preset-react']
            }
          }
        },
      ]
    },

    mode: 'development'

}
```

- babel-loader should supply all the JS files needed by webpack
- It should look for all .js and .jsx file
- It should ignore files in node_modules that are already compiled.
- Use @babel/preset-env to convert ES2015+ code to ES5+ code
- Use @babel/preset-react to convert JSX code

GET SET GO

```
c:\workspace\react-webpack-project> npm run dev
```