Siddharth Singha Roy

Winter 19-20

Modern JavaScript

Babel is used for converting ESNext (ES6 and above) to ES5 so that it can be supported by all the browsers.

Webpack is used for converting the JavaScript modules into a single bundle. This allows us to write files in different locations and then combine them into one using Webpack. Without Webpack we cannot do this. Webpack is a module bundler.

Dev-Dependencies

We have two types node packages, one is the library development tools like React, Angular etc and other are development tools. Dev-dependencies are for the development tools. They are used only by the developers. The client will never know these were used.

For saving dev-dependency

npm install package name —save-dev

For normal dependencies

npm install package name —save

To install our packages for all other users, we use npm install

To delete a package npm uninstall package name —save

To save a package globally

npm install package name —global
npm install package name -g

Webpack

Webpack technically needs 0 configuration. It needs only a src folder and an index.js inside it. However we will be writing our own config file. To config our webpack we need a file known webpack.config.js.

We will export an object known as module.exports

The first property in this object is the entry point ie. where the webpack should look to start bundling.

The second property is the output. The output is an object with 2 parameters path and filename. For filename we can give any filename we want. For path we need to give the absolute path. This is why we use the node package path.

We have two modes in which we can bundle our file

1. Production mode
2. Development mode

In the development mode, bundling is done as quickly as possible. However in production mode the bundles JS file is minified.

For our webpack to work we will need to add a npm script which will call the webpack.

```
const path = require('path');

module.exports = {
   entry: './src/index.js',
   output: {
```

```
        path:path.resolve(__dirname, "dist"),
        filename:'js/bundle.js',

    },
    devServer: {
        contentBase: './dist',
    }
};
```

ES6 Modules

There are two ways we can export something in ES6

1. export default abc;
2. export const pqr;

If we use the second one we can use

import something from ‘./index’;

If we use the second we can use

import pqr from ‘./index’

OR

import pqr as something from ‘./index’

If we import like this

import * as something from ‘./index’

To access pqr we use something.pqr

LocalStorage

Local Storage is used for persisting our data inside the web browser for future use. The localStorage is an object.

To set something inside the local storage use

```
localStorage.setItem('key', 'value');

localStorage,getItem('key'); // Returns value

localStorage.removeItem('key')

localStorage.length() // Returns the length of localStorage ie. number of items inside it
```

We can only save strings inside the localStorage. To convert an array to string we use `JSON.stringify(arr).`

To convert back to array

`JSON.parse(string_name)`

However if it was empty then null will be returned. So check for null to know if it was stored in the local storage or not.