📙

Ep2: Igniting our App

Our last code was not production ready code because it has comments, console.log and many more.

In order an app to be in production, it need to consider a lot of things. Such as optimizing images, minification of code, etc.

What its take to build production ready app?

We need many other packages which will make our app production ready. But what is package? package is some code written by other developers which can be used by others.

NPM : () It is everything but not Node Package Manager. Means don't have full form. Its a standard repository for all the packages. All packages are hosted on NPM. NPM manages all this packages.

Package.json is the configuration file for the npm.

Why do we need? it will track and manages all the packages that we are using in our app. It will also able to download all the required packages mentioned in the package.json file.

Just by running npm install. Packages are also called as Dependencies.

The most important dependency will be bundler → Webpack, parcel , vite → are bundler which packages our app properly which ships to production.

Webpack is used by create-react-app

vite is also becoming popular.

and parcel we are going to see now.

npm install -D parcel

Why -D ?

There are two types of dependency that an app can have:

1. Dev Dependency: It means the dependency is required in a development phase.
2. Normal Dependency: this dependency can be used in production also.

Parcel: ^2.10.0:

^ Carat symbol: Suppose if there is 2.10.1 or 2.11.2 get released then it will automatically download that dependency but not version 3. something. Means it will do minor upgrades automatically and ones major upgrade happens it will not go ahead.

~ tilde symbol: If tilde is used than it will install major version also such 3.0.0

Always to use ^

Package.json : Its configuration for npm which keep track of the version of that package that we are using.

Package-lock.json is also created why?: package lock file keep track of exact version that is being installed. And also keep track of the sub dependencies used.

integrity: “sha512…………………………..”

It working for me but failing in production.

So to avoid this condition package-lock.json keep the hash to verify whatever is present in my machine should also be present in production machine.

It keeps track of all the version of all the dependency.

node_modules → this contains all the code that we fetched from npm for parcel

We install parcel and parcel has its own dependency such type of dependencies are called as transitive dependency.

package-lock.json

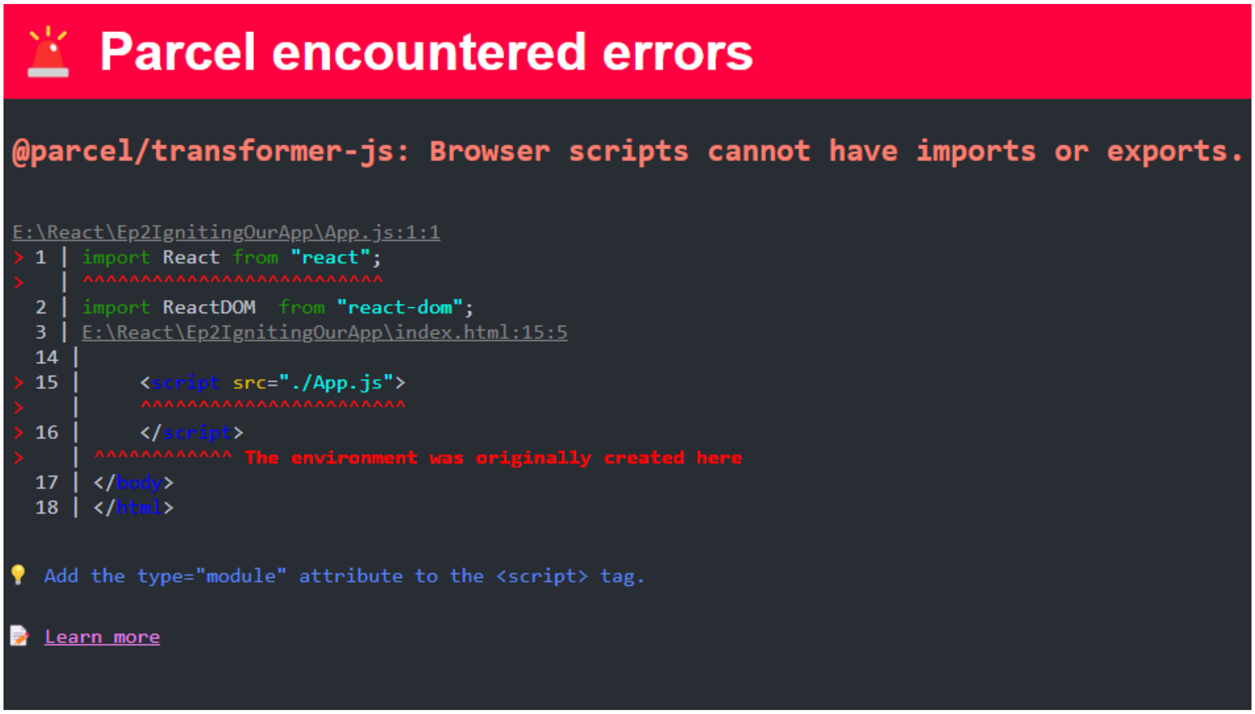
package.json help us generate node_modules any time we want

npx parcel index.html (npx means executing package)

parcel created a server and hosted our app on localhost:1234

CDN links are not a preferred way. We should use NPM instead.

We get error after import react and reactdom in app.js file



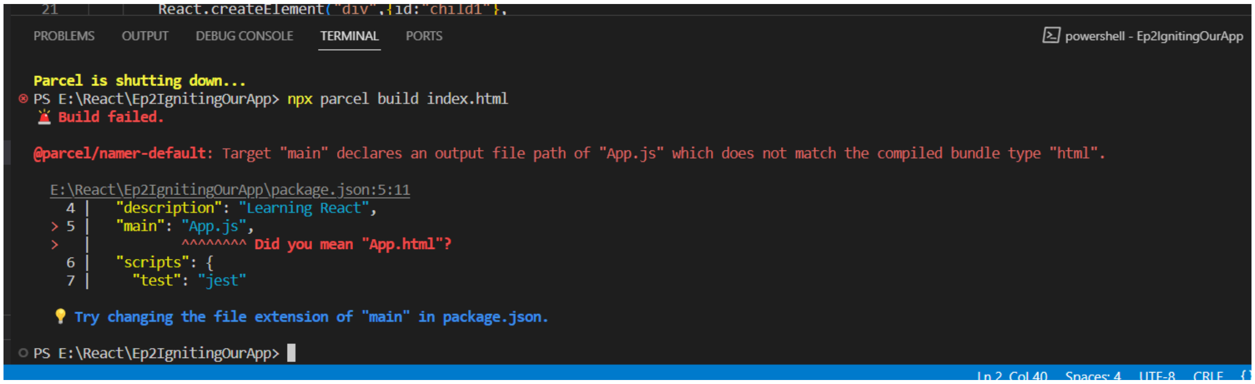
```
🚨 Parcel encountered errors

@parcel/transformer-js: Browser scripts cannot have imports or exports.

E:\React\Ep2IgnitingOurApp\App.js:1:1
> 1 | import React from "react";
>   | ^^^^^^^^^^^^^^^^^^^^^^^^^^
  2 | import ReactDOM  from "react-dom";
  3 | E:\React\Ep2IgnitingOurApp\index.html:15:5
  14 |
> 15 |     <script src="./App.js">
>    |     ^^^^^^^^^^^^^^^^^^^^^^^
> 16 |     </script>
>    | ^^^^^^^^^^^^ The environment was originally created here
  17 | </body>
  18 | </html>

💡 Add the type="module" attribute to the <script> tag.

📝 Learn more
```

As normal JS doesn’t have import statement we are getting error because its treating App.js as a normal JS file and not as reactjs file.

We resolve this by below code changes in html: (type module will treat this file as a module)

<script type="module" src="./App.js">

We will get into one more error when we try to create production build of our app:



```
  21   |     React.createElement("div",{id:"child1"},

  Parcel is shutting down...
⊗ PS E:\React\Ep2IgnitingOurApp> npx parcel build index.html
  🚨 Build failed.

  @parcel/namer-default: Target "main" declares an output file path of "App.js" which does not match the compiled bundle type "html".

    E:\React\Ep2IgnitingOurApp\package.json:5:11
      4 |   "description": "Learning React",
    > 5 |   "main": "App.js",
    >   |           ^^^^^^^^ Did you mean "App.html"?
      6 |   "scripts": {
      7 |     "test": "jest"

    💡 Try changing the file extension of "main" in package.json.

○ PS E:\React\Ep2IgnitingOurApp> █
```

this is because when we npm init we have taken App.js as default app but now we are providing index.html. So just removing it from package.json file will help.

#Parcel Features

- Dev Build
- Local Server
- HMR - Hot Module Replacement. (By its File Watching Algorithm- written in C++)
- Caching - Faster Builds
- Image Optimization
- Minification
- Bundling- Core Job
- Compress
- Consistent Hashing. (Read) Content hashing.
- Code Splitting.
- Differential Bundling - When app is hosted and opened on different browser like internet explorer, chrome ,edge and older version of this browser also All of this is managed by parcel. - support older browsers.
- Diagnostic
- Error Handling.
- HTTPs hosting also available.
- Tree shaking. - remove unused code from files (example 100 of functions are present and we are using just 4 or 5 then it will remove other things).
- Different dev and prod bundles.

Homework

▼ What is NPM?

NPM is the central JS repository which manages package. We can download this package from npm and used it in our project. And yes it is **not** called “Node Package Manager”

▼ What is Parcel/Webpack? Why do we need it?

Parcel/Webpack/Vite are all bundler. Bundler helps us to make a production ready app and does a lot of work behind the scene.

It does a lot of thing when we create production build such as caching, optimizing code and image, minification, compress, content hashing and error handling.

▼ What is .parcel-cache?

Whenever we run our build for the first time then it will .parcel-cache which contains most used files and whenever a we rerun the build then it will take the code from the cache which greatly reduces time for building.

Production build in real world are huge it takes time to build because at the backend personal does lot of work such as minification and compression of code, optimization of image, and tree shaking.

▼ What is npx?

Node Package eXecute is a package runner. It allows us to execute any JS package available on npm registry. It comes with npm.

▼ What is difference between dependencies vs devDependencies?

When we want to just use/test a package in development environment then we can use devDependencies which will store that package for development only and will not be moved to production.

"dependencies" : Packages required by your application in production. "devDependencies" : Packages that are only needed for local development and testing

▼ What is Tree Shaking?

Tree shaking is a feature in which we remove unwanted code. That means in parcel the code that is not reachable is automatically removed. Its bascially a removal of dead code.

▼ What is Hot Module Replacement?

Its a file watching algorithm which provides Auto Refresh feature whenever it detects the code change in our project.

▼ List down your favourite 5 superpowers of Parcel and describe any 3 of them in your own words?

1. HMR: Hot Module Replacement which provides us Auto Refresh feature. It even works with CSS also as soon as we save any file its automatically make the changes and behind the scenes it uses file watching algorithm.
2. Caching - It does caching to make our next build fast.
3. Differential Bundling - which allows the build to be compatible with older browser. We use BrowsersList feature.

▼ What is gitignore? What should we add and not add into it?

When we don’t want a file to be push on a central repo we add that file in .gitignore. .gitingore is a file in which mention file or directories that we want to exclude.

We generally exclude things that are not necessary and can be regenerated any time in-order to save time and space. For eg: node_modules file we don’t need to store node_modules as we can regenerate it any time we want. And also node_modules are heavy package it takes lot of storage.

▼ What is the difference between package.json and package-lock.json?

Package.json: Its configuration for npm which keep track of the version of that package that we are using.

Package-lock.json: package lock file keep track of exact version that is being installed. And also keep track of the sub dependencies used.

It is used to lock dependencies to a specific version number.

▼ Why should I not modify package-lock.json?

Lock file should not be modify unless we are actively updating the package.

Because if we are modifying some times it could break the synchronization between package.sjon and package-lock.json.

▼ What is ‘node_modules’? Is it good idea to push that on git?

Node_modules is contains the actual code of the package that we have downloaded from npm. Its not a good idea to push node_modules to git. As the package takes a lot of storage and it can be generated any time we want. So it is not a good way.

▼ What is the dist folder?

When we build our app for development or production. It will store that files that are required to run on web are stored in dist folder.

dist folder stores the latest run copy of the build. And the app that opens up uses dist folder to open our app and not our normal folder. dist folder contains the optimized version of our code.

▼ What is browserlists?

When we want to ensure that our app runs on different browser and older chrome version we want than browserlists is used.