Chapter 02 Igniting Our App

recap by @jx

1 🏛️ What is a build?

In web development, a build is the **process** of converting source code into standalone **software artifact(s)** that can be run on a client/target/browser.

> 💡 This process and the resultant software artifact are interchangeably called a build.

Some commonly known builds in a project are Nightly, **Development**, Beta and **Production** builds.

Note 1. Dev Build has debugging and often times hot reloading capabilities built-in and is usually run on developers/QA's machine. It is used for testing purposes.

Note 2. Production Build, on the other hand is meant to face your client. The production build contains uglified code built from your source files into one or multiple minimized files.

2 🛠️ Build Tools *a.k.a Bundler a.k.a Beast*

Build tools manage, organize, and optimize your builds. Few popular build tools are Webpack, Parcel, Vite.

2.1 Why do I need Parcel?

React is a UI library. It does NOT help you build a production-ready App. You need help from other libraries to help you build an App that is optimized & production-ready. *Parcel is one such tool.*

2.2 👹 What makes Parcel a beast?

Because Parcel orchestrates many things to aid your frontend development & building experience,

1. It takes pieces of JavaScript and it's dependencies to bundles them into a single file, for use in the browser.
2. It comes with a dev server which serves on your *localhost* at port *1234*.
3. It granularly rebuilds files that have only changed since the last time it ran. It uses *.parcel-cache* for this. This saves time.
4. It includes features such as tree-shaking, minifying your code, resizing & optimizing image assets, content hashing, automatic code splitting etc.
5. It performs HMR(Hot Module Replacement), which is, updating modules in the browser at runtime without needing a whole page refresh.

💡 It is more accurate to note that Parcel leverages other packages under the hood to achieve all of the above. Your project is **transitively dependent** on those packages through Parcel.

2.3 What is bundling?

Bundling is the process of following imported files and merging them into a single file: a bundle. This bundle can then be included on a webpage to load an entire app at once.

2.4 What is minification?

Minification is the process of removing comments, extra spaces & shortening long variable names so that the resulting minified code provides the same functionality as source meanwhile reducing the bandwidth of network requests.

2.5 What is code-splitting?

As your app grows, your bundle will grow too, making your App take a long time to load. To get ahead of this, Parcel can help you create multiple bundles that can be dynamically loaded(lazy-load) at runtime.

This dramatically improves the performance of your app. While you have not reduced the overall amount of code in your app, you have avoided loading code that the user may never need, and reduced the amount of code needed during the initial load.

2.6 What is tree-shaking?

Tree-shaking is a concept in frontend development that involves the elimination of dead code or unused code.

3 📦 Adding Parcel to a project

To add Parcel in a project it must be initialized as an npm project,

```
$ npm init
```

This will create a package.json file in your project directory. Now to add Parcel use either of the following command. Both are the same.

```
$ npm install -D parcel
$ npm install –save-dev parcel
```

Note 3. 'npm install' saves any specified packages into 'dependencies' by default.

Note 4. '-D' or '–save-dev' results in the package to be added as a 'devDependencies'.

3.1 🤔 Difference between dependencies & devDependencies

devDependencies*(such as Parcel, Babel)* are modules which are only required during development whereas dependencies*(such as React, ReactDOM)* are required at runtime.

If you are deploying your application, dependencies has to be installed, or else your app simply will not work.

4 🚀 3..2..1.. IGNITION!

$ **npm** parcel index.html *generates dev build, does HMR, caching etc.*
$ **npm** parcel build index.html *generates production build after minifying etc.*

⚠️ Importing modules cannot be done in a normal script tag. Set script tag type to 'module'.

<script **type = 'module'** src = 'App.js'></script>

5 🌐 Building for the Web

Build is a process of converting source into standalone software artefacts that can be run on a **client/target/browser.**

5.1 How to specify what target/browser to build for?

The answer is **browserlist**.

It allows you to describe which browsers your site needs to support. You can describe this using an array of strings. Then, when Parcel is building your App it can read this information from *package.json* under "browserlist" key.

```
  "browserslist": [
  "cover 90%"  //this is a string that Parcel(Babel) will read during build
]
```

5.2 Why browserlist?

Browserslist helps you keep the right balance between browser compatibility and bundle size. With Browserslist, you will cover wider audience and have smaller bundle size.

6 🔗 References

1. Different types of Builds used in an Open-Source project here.
2. Parcel Development Build Docs here.
3. Parcel Production Build Docs here.