Namaste React (Episode 02 Igniting our App)

npm

npm != node package manager (does not have a full form)

Definition
package manager for JS runtime environment Node.js

npm init : Initialize npm in project
creates a package.json file

package.json

- (configuration file for npm)
- Tracks project dependencies and approx. versioning
- Contains ~ and ^

Packages = dependencies.

package-lock.json

- Keeps track of exact version of all dependencies installed

Best practice: Always include package.json and package-lock.json in git (track information about what all dependencies project needs with versioning)

node_modules folder

Contains all node modules with code (contains actual data)

Best practice: put node_modules folder inside .gitignore (bulky folder not required in production)

If you have package.json and package-lock.json, you can get node_modules.
npm install

parcel

Definition
Bundler.

HMR (Hot Module Replacement)
Captures changes in project and reload instantly. Done using File Watcher Algorithm (in C++).

.parcel-cache (caching files so subsequent builds are delivered faster)

How to install parcel
npm install -D parcel (npm package for parcel)
-D means Dev Dependency

How to execute parcel
npx parcel index.html (source is index.html)
Creates development ready build, puts files in dist folder, host app on localhost.

npx parcel build index.html (source is index.html)
Create a production ready build, put files in dist folder, host app on localhost.

Best practice: put dist folder and .parcel-cache inside .gitignore (can be regenerated using parcel)

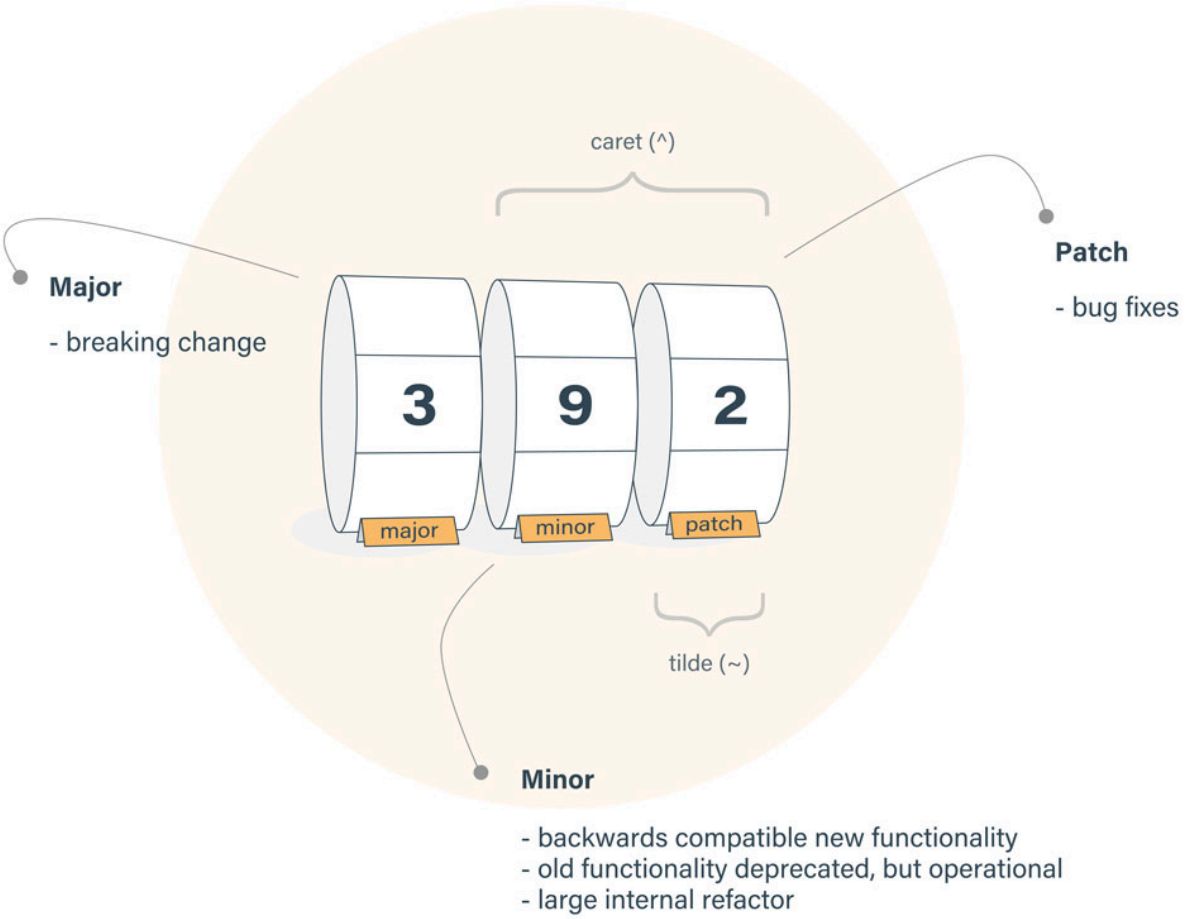
Parcel Tasks (One of reason why React app is fast)

- Bundling
- Minification for production build
- Compressing (remove whitespaces)
- HMR = Hot Module Replacement
- File Watching Algorithm (in C++)
- Caching Files for lesser build time for subsequent builds
- Hosting on local server
- Image Optimization (most expensive task to render image)

Two types of dependencies

1. Dev dependency (required for development only)
2. Normal dependency (required for dev + production)

"parcel": "^2.11.0"



Code status	Stage	Rule	Example version
First release	New product	Start with 1.0.0	1.0.0
Backward compatible bug fixes	Patch release	Increment the third digit	1.0.1
Backward compatible new features	Minor release	Increment the middle digit and reset last digit to zero	1.1.0
Changes that break backward compatibility	Major release	Increment the first digit and reset middle and last digits to zero	2.0.0

As long as the first number ("major") is at least 1:

~ (Tilde)

- locks major and minor numbers.
- Ready to accept only bug-fixes (increments in the third number), but don't want any other changes, not even minor upgrades that add features.
- The tilde matches the **most recent minor version** (the middle number).

^ (caret)

- locks the major number only.
- Ready to accept bug fixes (increments in the third number) and minor upgrades that add backward compatible features (increments in the second number). However you do not want changes that break existing code (increments in the first number).
- Update you to the **most recent major version** (the first number).

Semver explained - why is there a caret (^) in my package.json?

Why did the parcel install node_modules ?

Parcel has some dependencies, those dependencies have other dependencies (Transitive dependency) and this continues. Therefore it installed them in the node_modules folder.

Each package in node_modules has a package.json that tells about its dependencies.

CDN is not a good way to include React in a project.

Reason:

1. Overhead of calling a request
2. Difficult to maintain React versioning

Solution (using npm to install React as dependency)

- npm install react
- npm install react-dom

Note: included as normal dependency since we need them for both production and development

import React from “react”
import ReactDOM from “react-dom/client”
(use the objects from react and react-dom modules)

Reason for Error:
import React from “react”. import is not a part of Javascript.

Solution
Add module attribute to script tag.

browserslist

Determine list of browsers and environments compatible with web application

Create project specific bundles to work for specific browser versions.

(Inside package.json)
“browserslist” : [
“last 2 versions”
]

“browserslist” : [
“last 2 Chrome versions”
]