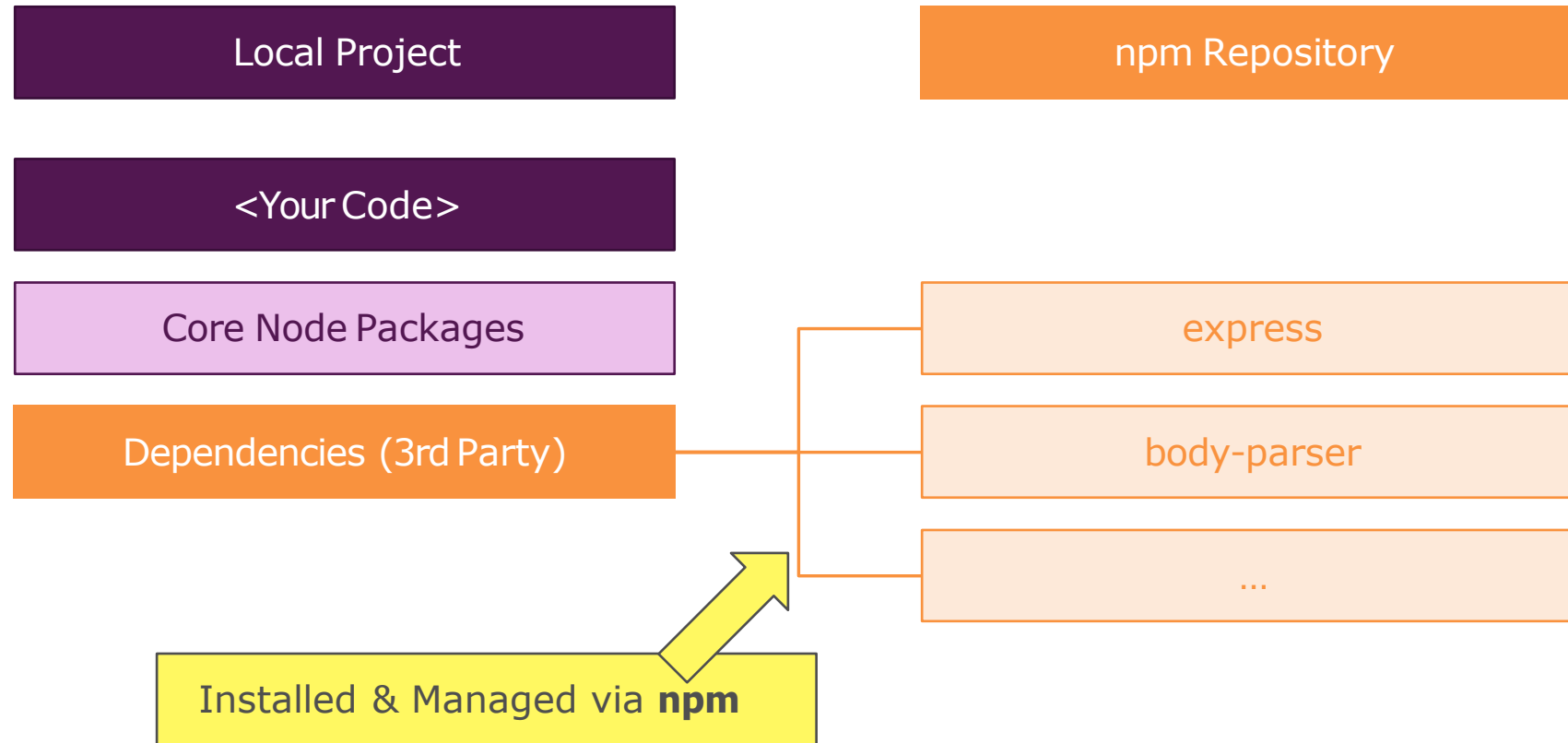




NPM



npm & packages Intro



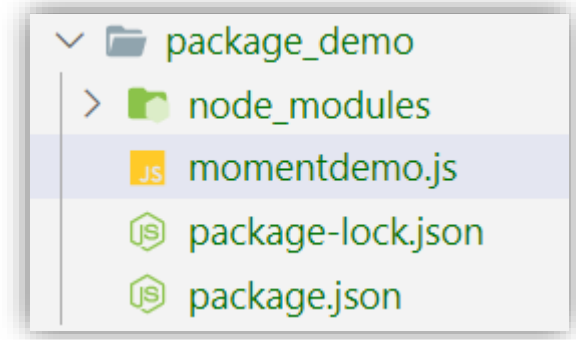
What is npm?

- **npm** is the standard package manager for Node.js. It also manages downloads of dependencies of your project.
- www.npmjs.com hosts thousands of free packages to download and use.
- The NPM program is installed on your computer when you install Node.js.
 - `npm -v` // will print npm version
- What is a package?
 - A package in Node.js contains all the files you need for a module.
 - Modules are JavaScript libraries you can include in your project.
- A package contains:
 - JS files
 - `package.json` (manifest)
 - `package-lock.json` (maybe)

Create & use a new package

```
npm init // will create package.json
```

- When we install a package:
 - Notice dependencies changes in `package.json`
 - notice folder: `node_modules`
 - This structure separate our app code to the dependencies. Later when we share/deploy our application, there's no need to copy `node_modules`, run: `npm install` will read all dependencies and install them locally.



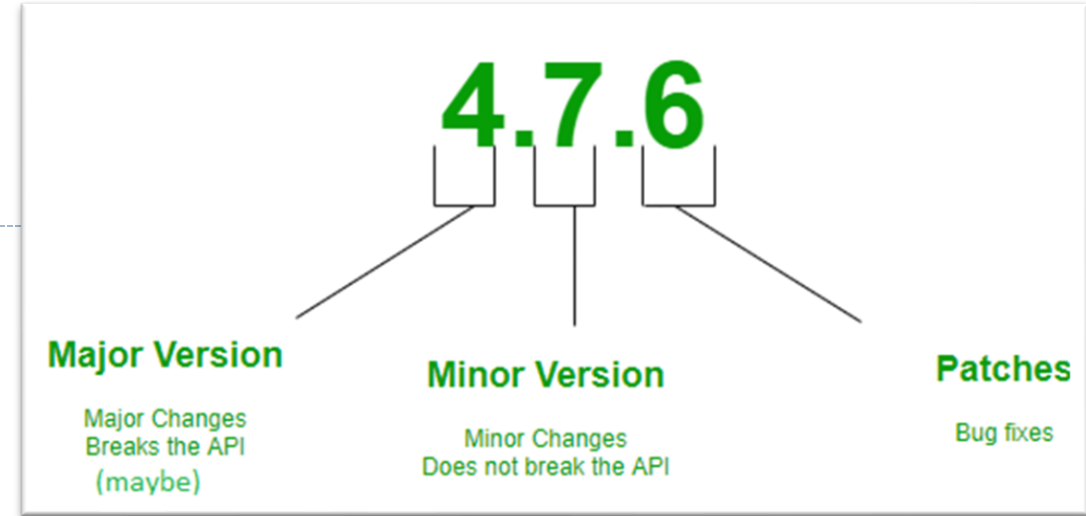
package.json Manifest

- The `package.json` file is kind of a manifest for your project.
 - It can do a lot of things, completely unrelated.
 - It's a central repository of configuration for installed packages.
 - The only requirement is that it respects the JSON format.
-
- `version`: indicates the current version
 - `name`: the application/package name
 - `description`: a brief description of the app/package
 - `main`: the entry point for the application
 - `scripts`: defines a set of node scripts you can run
 - `dependencies`: sets a list of npm packages installed as dependencies
 - `devDependencies`: sets a list of npm packages installed as development dependencies

```
{  
  "name": "package_demo",  
  "version": "1.0.0",  
  "description": "",  
  "main": "index.js",  
  "scripts": {  
    "start": "node momentdemo.js"  
  },  
  "author": "Anna Smith",  
  "license": "ISC",  
  "dependencies": {  
    "moment": "^2.29.1"  
  },  
  "devDependencies": {  
    "eslint": "^7.28.0"  
  }  
}
```

Semantic Versioning

- The Semantic Versioning concept is simple: all versions have 3 digits: $x.y.z$.
 - the first digit is the major version
 - the second digit is the minor version
 - the third digit is the patch version
- When you make a new release, you don't just up a number as you please, but you have rules:
 - you up the **major** version when you make incompatible API changes
 - you up the **minor** version when you add functionality in a backward-compatible manner
 - you up the **patch** version when you make backward-compatible bug fixes



More details about Semantic Versioning

- Why is that so important?
 - Because `npm` set some rules we can use in the `package.json` file to choose which versions it can update our packages to, when we run `npm update`.
- The rules use those symbols:
 - `^`: it's ok to automatically update to anything within this major release. If you write `^0.13.0`, when running `npm update`, it can update to `0.13.1`, `0.14.2`, and so on, but not to `1.14.0` or above.
 - `~`: if you write `~0.13.0` when running `npm update` it can update to patch releases: `0.13.1` is ok, but `0.14.0` is not.
 - `>`: you accept any version higher than the one you specify

package-lock.json

- Introduced by NPM version 5 to capture the exact dependency tree installed at any point in time.
- Describes the exact tree
- Guarantee the dependencies on all environments.
- Don't modify this file manually.
- Always use npm CLI to change dependencies, it'll automatically update package-lock.json

```
{
  "name": "lesson03-demo",
  "version": "1.0.0",
  "lockfileVersion": 1,
  "requires": true,
  "dependencies": {
    "moment": {
      "version": "2.24.0",
      "resolved": "https://registry.npmjs.org/moment/-/moment-2.24.0.tgz",
      "integrity": "sha512-bV7f+6l2QigeBBZSM/6yTNq4P2fNpSWj/0e7jQcy87A8e7o2nAfP/34/2ky5Vw4B9S446EtIhodAzkFCcR4dQg=="
    }
  }
}
```


More About Packages

- **Development Dependencies:** Needed only while I'm developing the app. It's not needed for running the app.
 - `npm install mocha --save-dev`
`// notice devDependencies entry now in package.json`
- **Global Dependencies:** Available to all applications
 - `npm install -g nodemon`
 - `nodemon app.js` `//auto detects changes and restarts your project`