CSCI 4140 – Tutorial 7

Introduction to npm

Matt YIU, Man Tung (mtyiu@cse)
SHB 118
Office Hour: Wednesday, 3-5 pm

2016.03.10

Outline

- What is npm?
- Installing and updating npm
- (Un)Installing packages [locally|globally] with npm
- Rebuilding an application with `npm install`
- `node_modules` and `.gitignore`
- What is `package.json`?
- Creating `package.json` with `npm init`
- Updating `package.json`
- Running scripts with `package.json`
- References

What is npm?

- According to the documentation,
  - *"npm makes it easy for JavaScript developers to share and reuse code, and it makes it easy to update the code that you're sharing."*
- npm is a **package manager** for JavaScript
  - Developers share their source code in **packages**
  - npm faciliates **open-source** project development
  - It is almost a must for **Node.js** applications

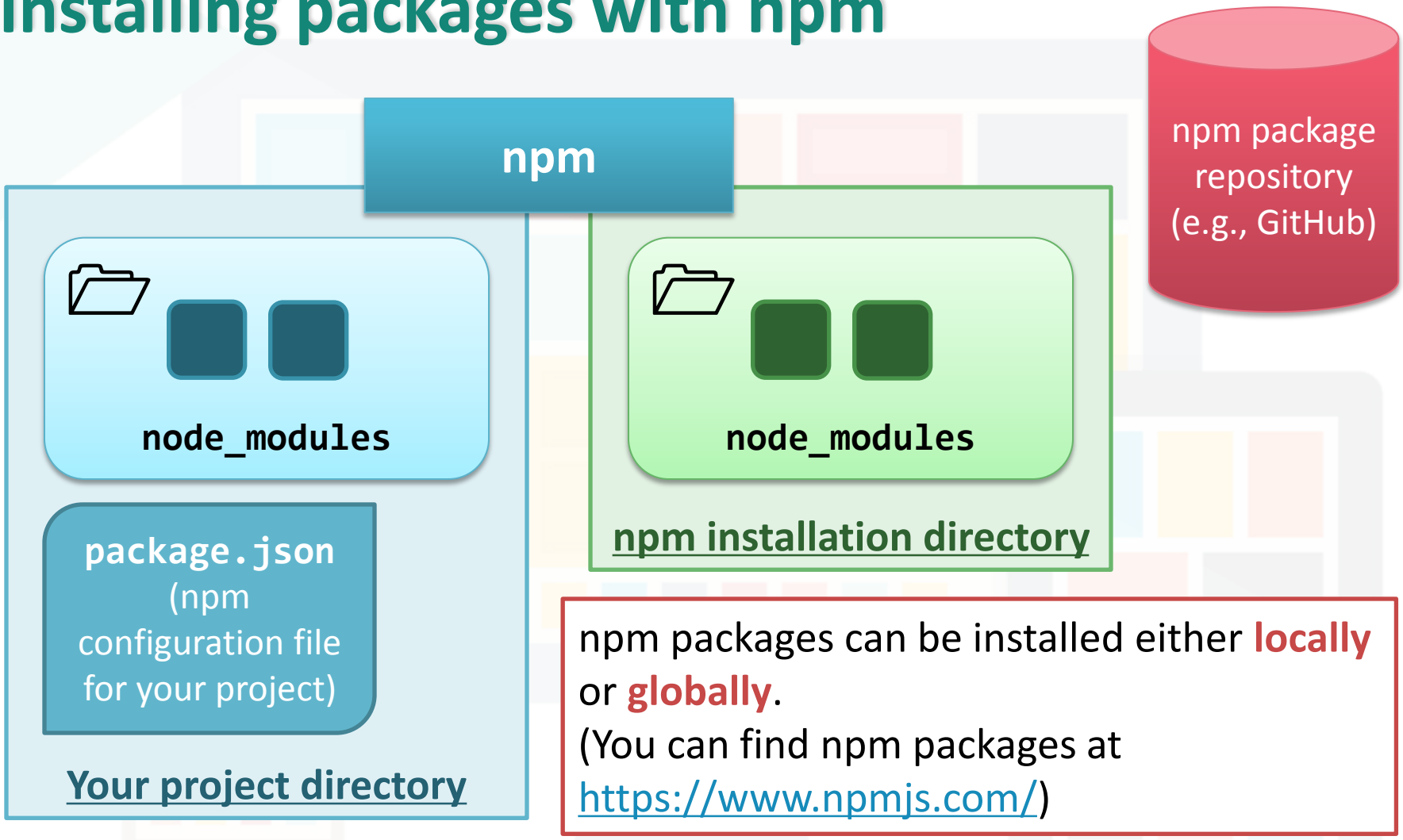
Installing and updating npm

- npm is installed for you as you install Node.js
- Before using npm, let's **update** npm to its latest version:

```
npm install npm -g
```

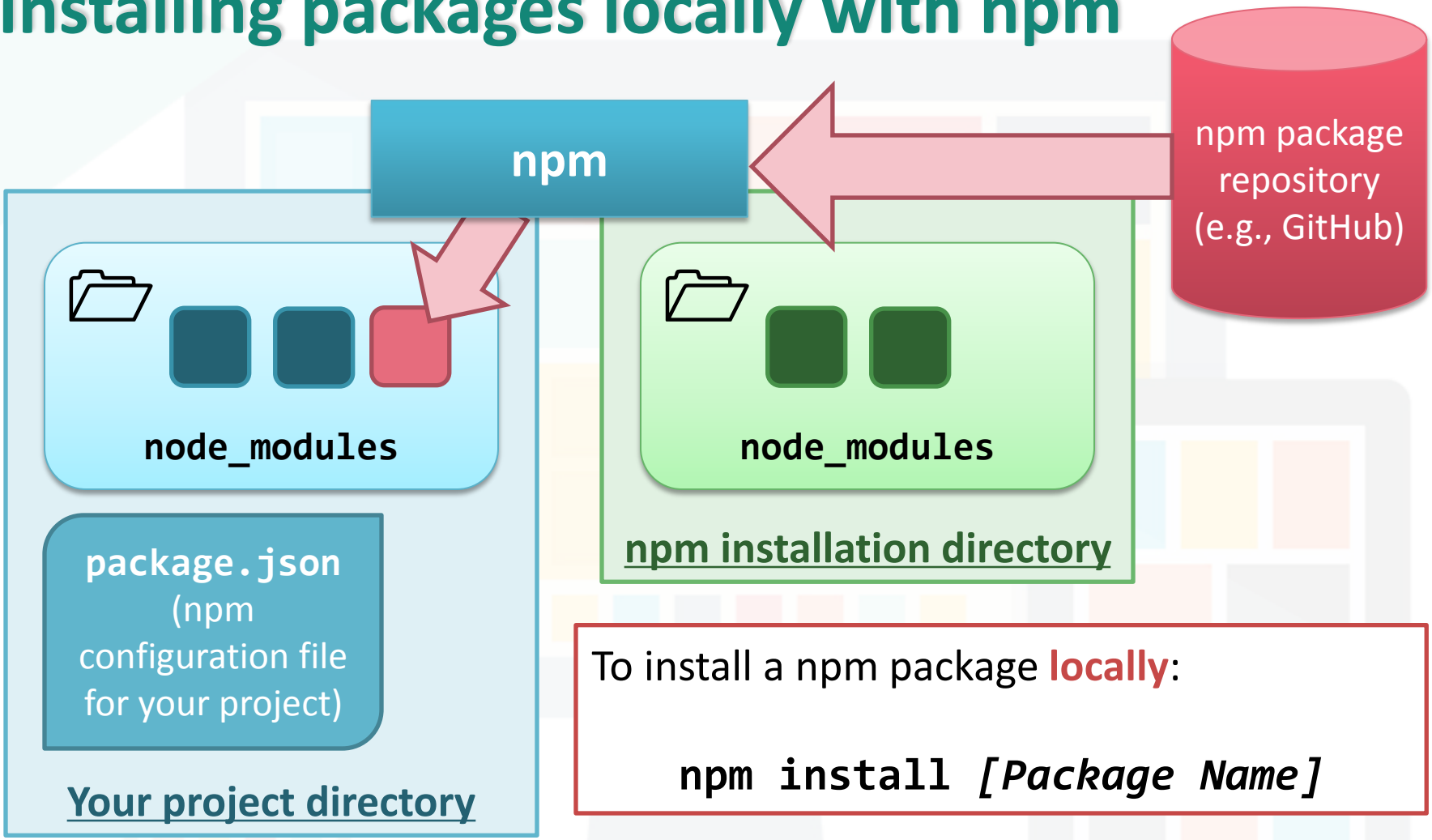
- If you receive "**Permission denied**" error when you install a package globally, then you may either
  - Change the permission to npm's default directory
  - Change npm's default directory to another directory
  - Follow the instructions if you encounter this error: https://docs.npmjs.com/getting-started/fixing-npm-permissions

Installing packages with npm



npm packages can be installed either **locally** or **globally**.
(You can find npm packages at https://www.npmjs.com/)

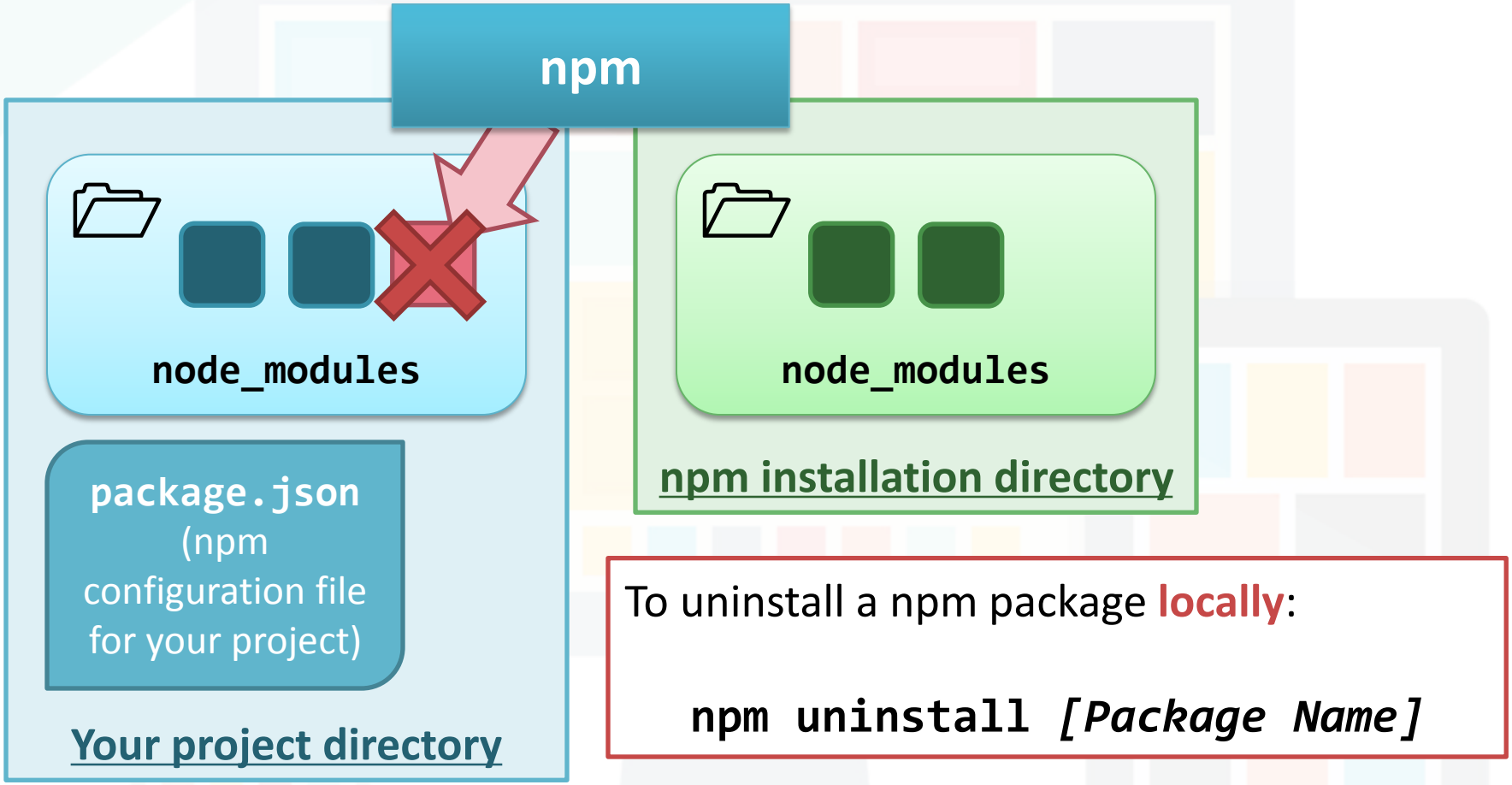
Installing packages locally with npm



To install a npm package **locally**:

```
npm install [Package Name]
```

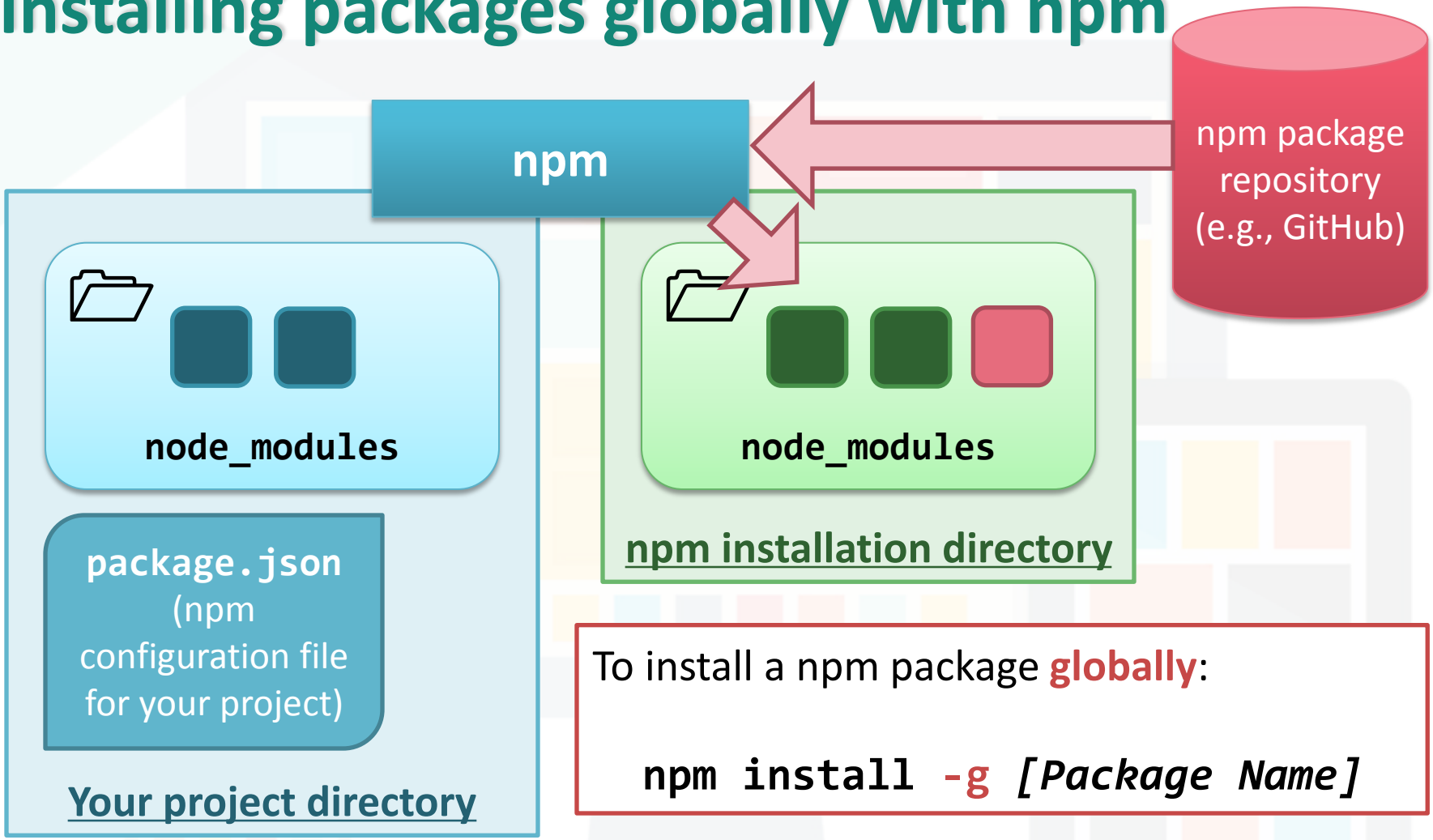
Uninstalling packages locally with npm



To uninstall a npm package **locally**:

```
npm uninstall [Package Name]
```

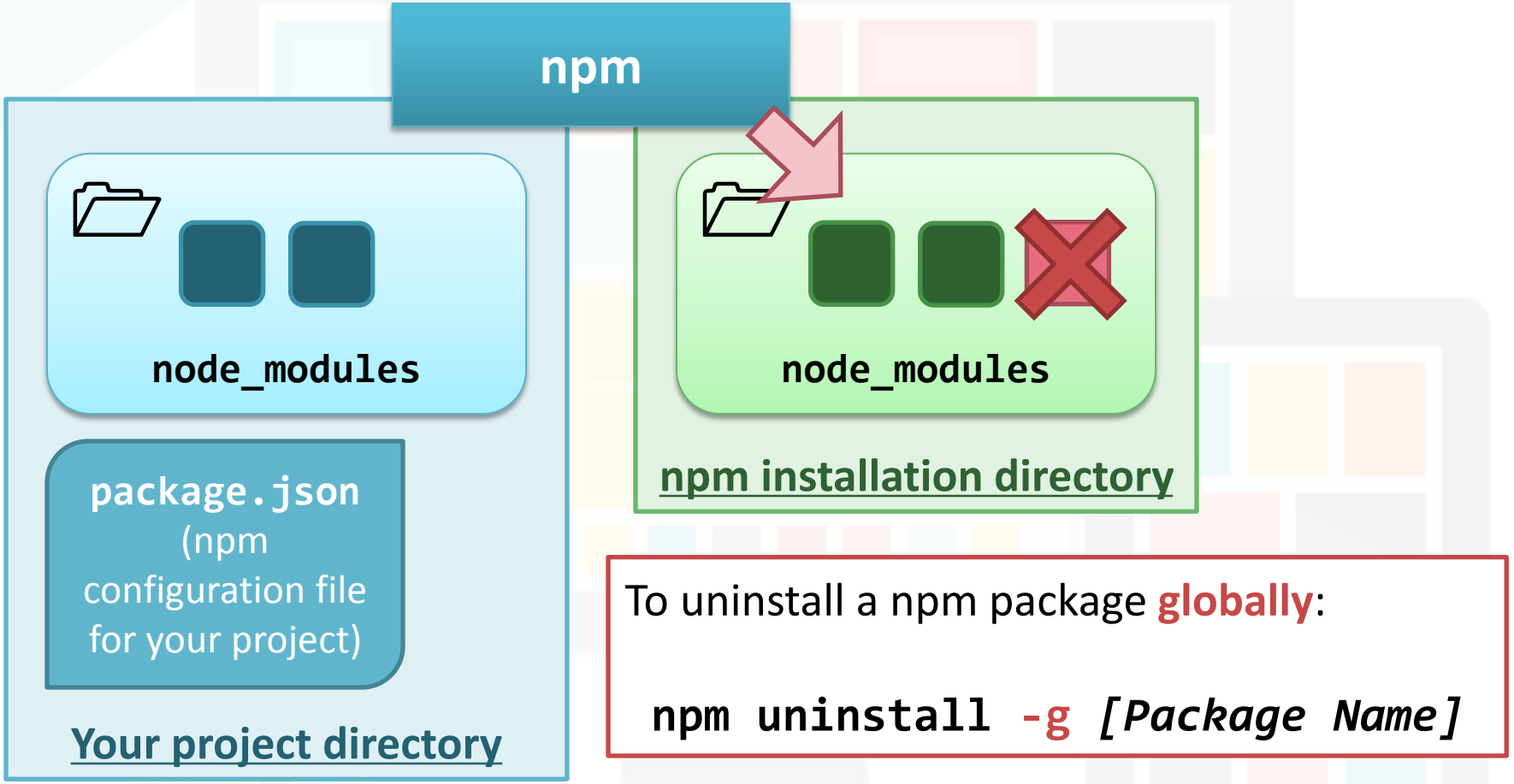
Installing packages globally with npm



To install a npm package **globally**:

```
npm install -g [Package Name]
```

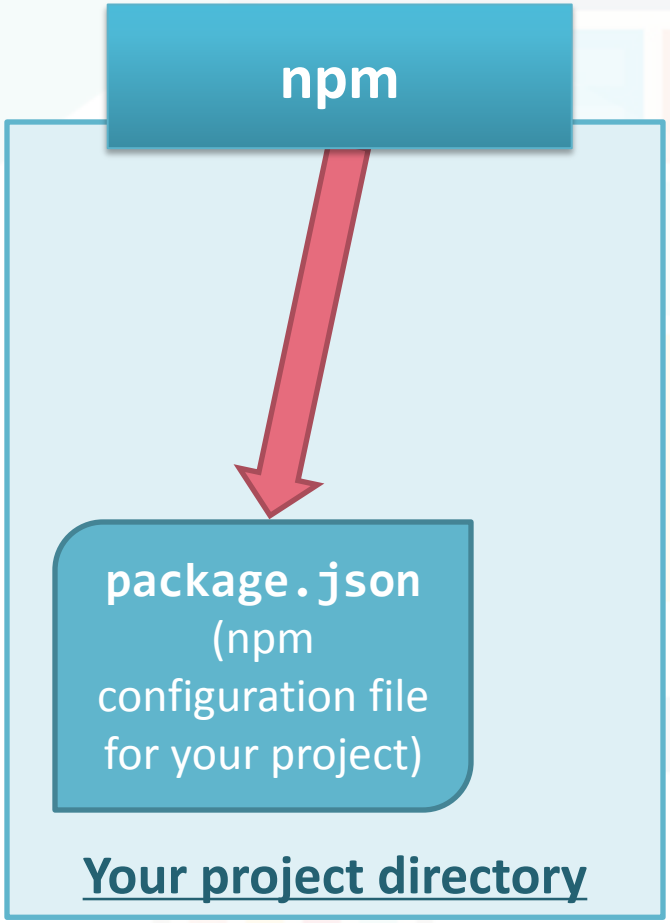
Uninstalling packages globally with npm



To uninstall a npm package **globally**:

```
npm uninstall -g [Package Name]
```

Rebuilding an application with `npm install`



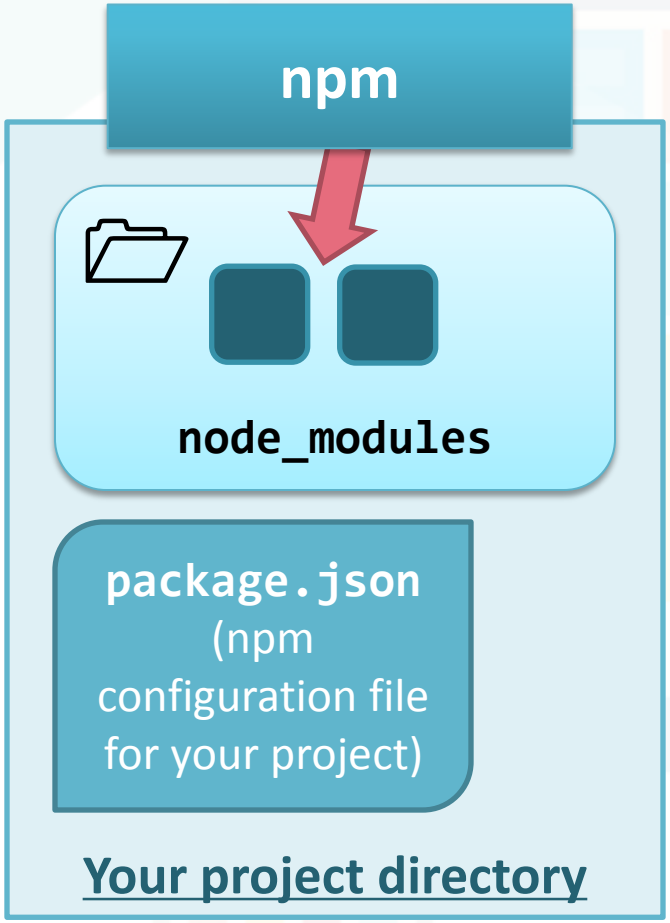
If you downloaded a Node.js application, you can use the following command to install all packages required by this application:

```
npm install
```

npm will:

1. Read `package.json` in the project directory
2. Install all **dependencies** specified in `package.json`

Rebuilding an application with `npm install`



If you downloaded a Node.js application, you can use the following command to install all packages required by this application:

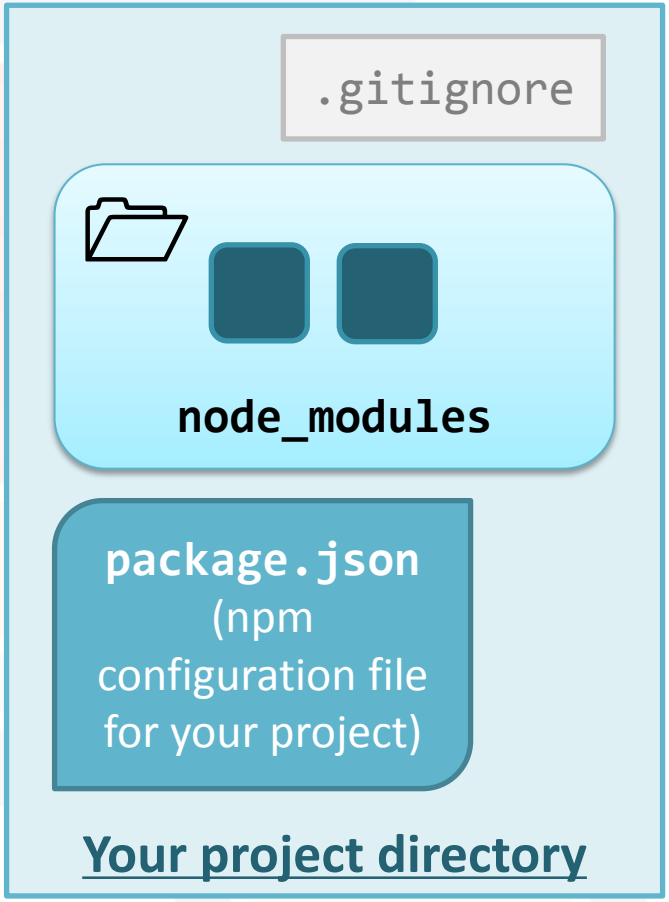
```
npm install
```

npm will:

1. Read `package.json` in the project directory
2. Install all **dependencies** specified in `package.json`

node_modules and .gitignore

- Adding the directory "**node_modules**" into your **git** repository is not a good practice
  - Let npm manage the dependencies for you!
- Use "**.gitignore**" to exclude the directory
  - Visit https://www.gitignore.io/ and search "node" and put the file (name it as "**.gitignore**" to your project directory)



What is `package.json`?

- To manage locally installed npm packages, create a **`package.json`** file
- Why `package.json`?
  - It serves as **documentation** for what packages your project depends on
  - It allows you to specify the **versions of a package** that your project can use using semantic versioning rules
  - Makes your build **reproducible** which means that its way easier to share with other developers
- `package.json` is in **JSON format**

What is package.json?

- package.json has at least two fields: **name** and **version**
  - For example,

```
{
  "name": "my-awesome-package",
  "version": "1.0.0"
}
```

  - "**name**": all lowercase; one word, no spaces; dashes and underscores allowed
- Alternatively, let npm create a package.json for you:

```
npm init
```

Creating package.json with npm init

- "**npm init**" will ask you the following questions (or use the default value by pressing Enter directly):

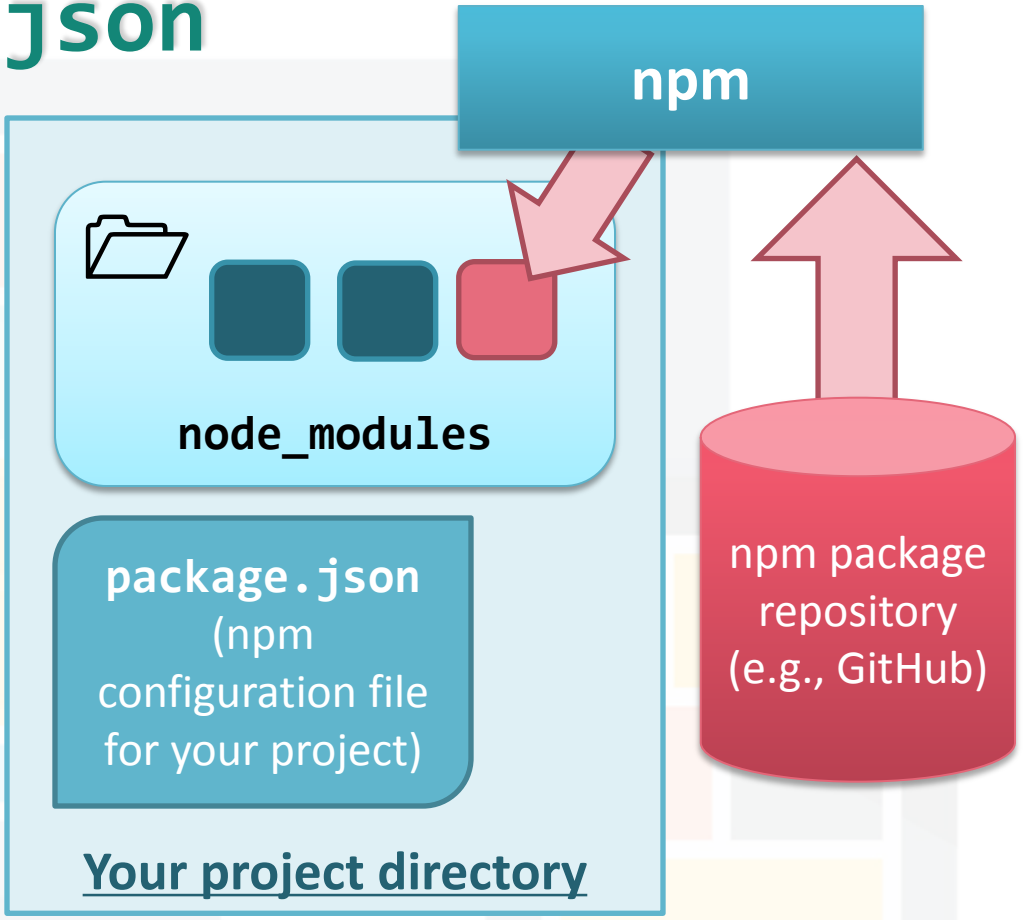
```
name: (csci4140)
version: (1.0.0)
description:
entry point: (index.js)
test command:
git repository:
keywords:
author:
license: (ISC)
```

- If you want to keep all default values, use

```
npm init --yes
```

Updating package.json

- Let's go back to this figure:
  - As you use "**npm install**" to install a package, the package is download to "**node_modules**"
  - Yet, the file "package.json" is not updated!
  - To force a certain package to be installed, update the dependencies of your project by




```
npm install --save [Package Name]
```

Running scripts with `package.json`

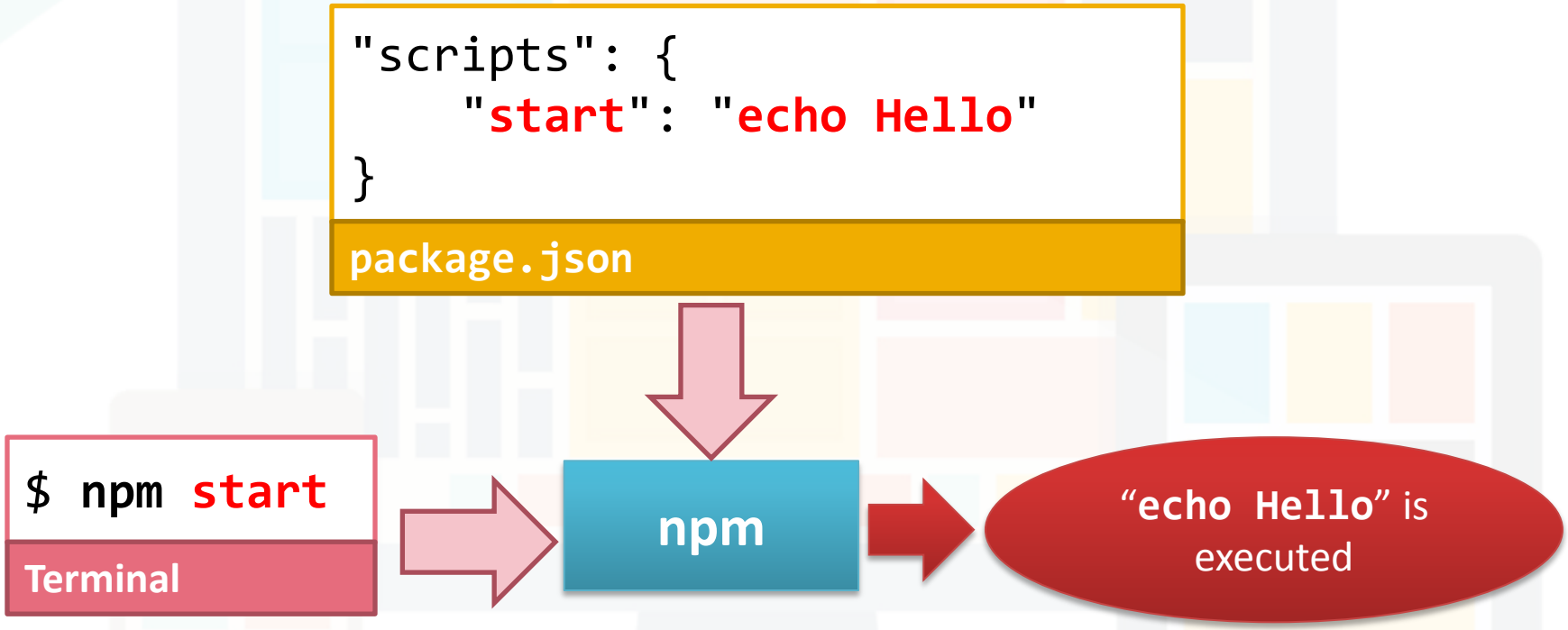
- package.json has a "**`scripts`**" section
- For example, in an Express application, the "`scripts`" section in `package.json` is

```
"scripts": {
    "start": "node ./bin/www"
}
```

- It defines the **scripts** executed by some "special" npm command

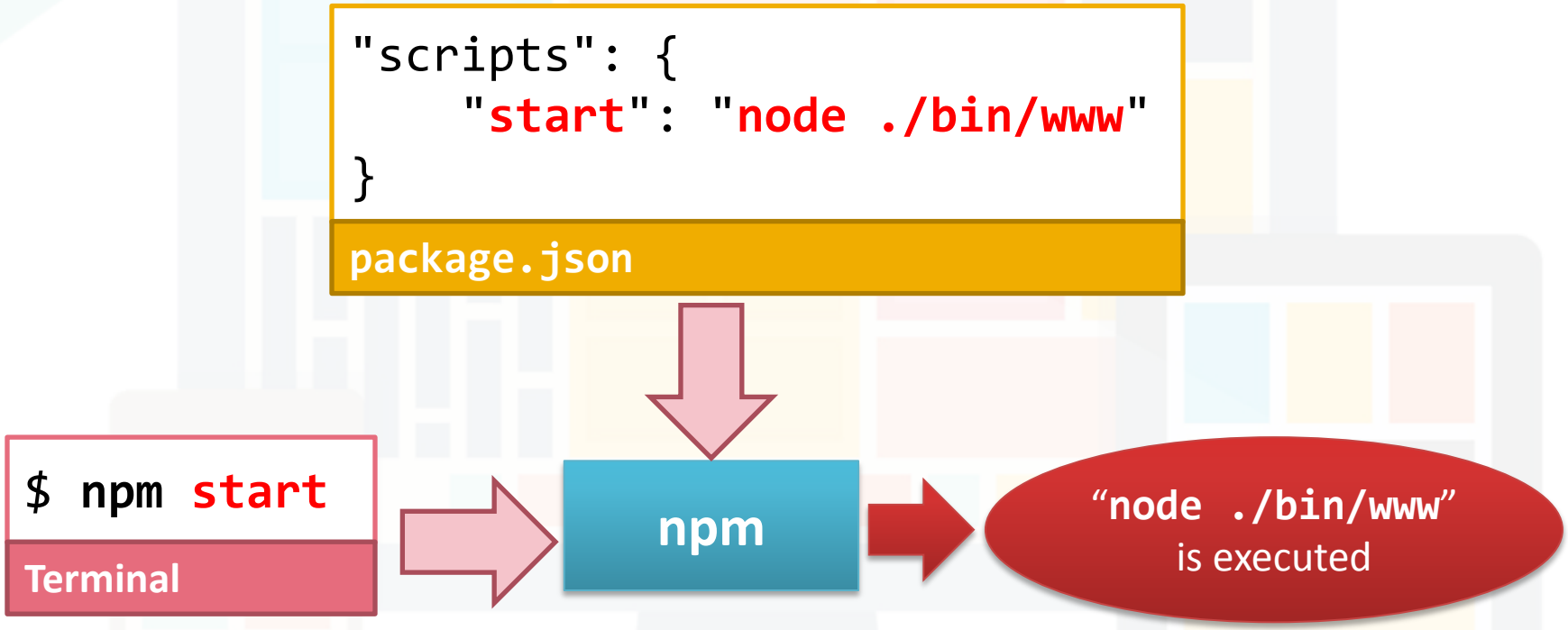
Running scripts with package.json

- Example 1



Running scripts with package.json

- Example 2



References

- This set of slides provides the things you need in the React JS lab (in lecture) and Assignment 2
- There are much more to explore!
- Read the following references for more details
  - npm Documentation: https://docs.npmjs.com/
  - `package.json`: https://docs.npmjs.com/files/package.json
  - `npm-scripts`: https://docs.npmjs.com/misc/scripts

– End –