NPM AND BOWER

JAMES MUELLER

What are they?

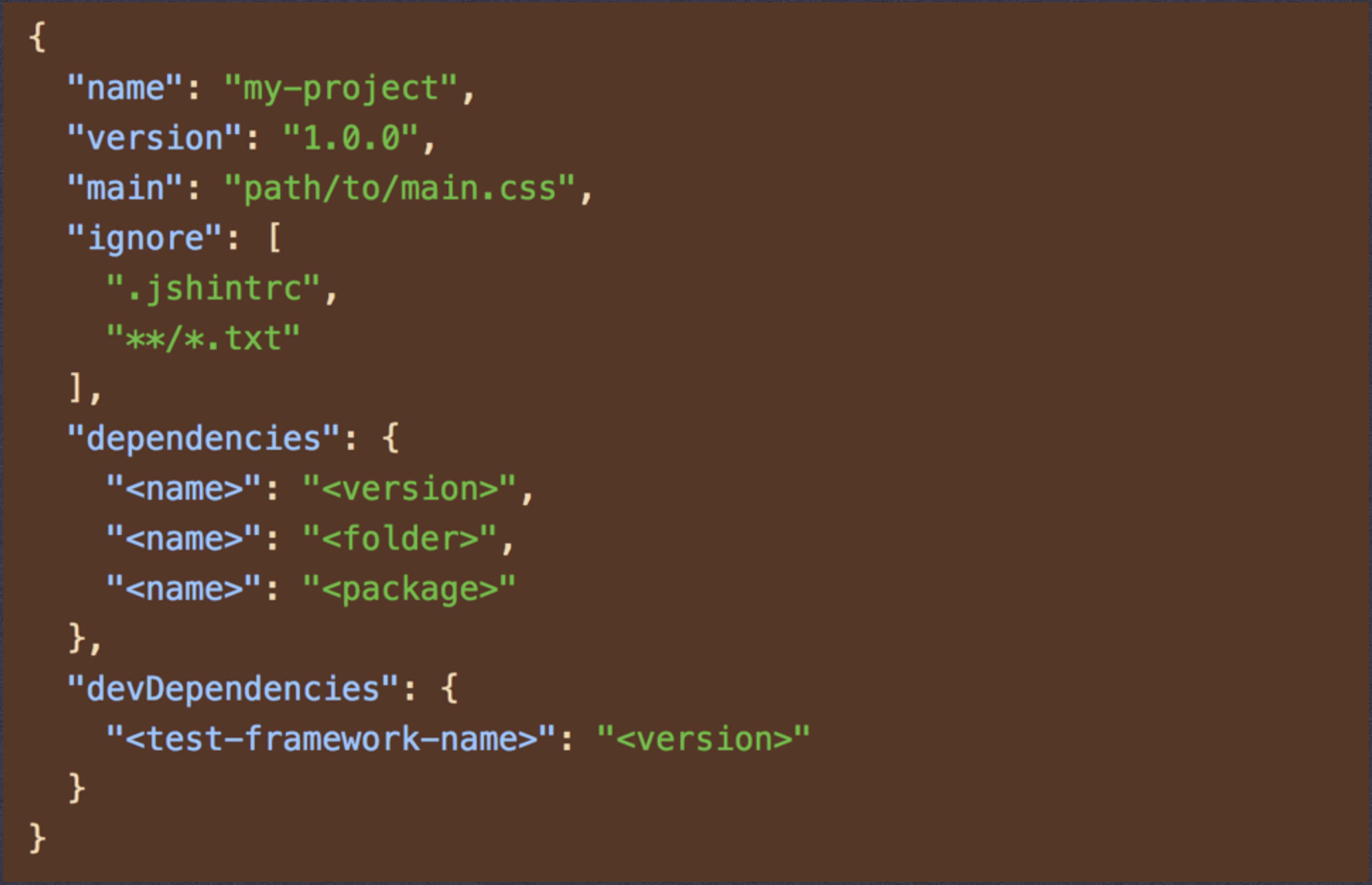
- Package Managers
- Dependency Managers
- NPM = Back End
- Bower = Front End

How do they Work?

- Bower uses a registry that associates keys with repositories
- The NPM also uses a registry but we have an Artifactory cache that sits in front
- Bower can hit random URIs (prefers git repos)
- NPM likes tarballs and files with package.son

Details: Bower

- Basically a Git Wrapper
- If the key is in a registry it will download that package
- If the value is a git project or url it will download that, otherwise the value specifies a version
- Use “ignore”: to remove unnecessary files



```
{
  "name": "my-project",
  "version": "1.0.0",
  "main": "path/to/main.css",
  "ignore": [
    ".jshintrc",
    "**/*.txt"
  ],
  "dependencies": {
    "<name>": "<version>",
    "<name>": "<folder>",
    "<name>": "<package>"
  },
  "devDependencies": {
    "<test-framework-name>": "<version>"
  }
}
```

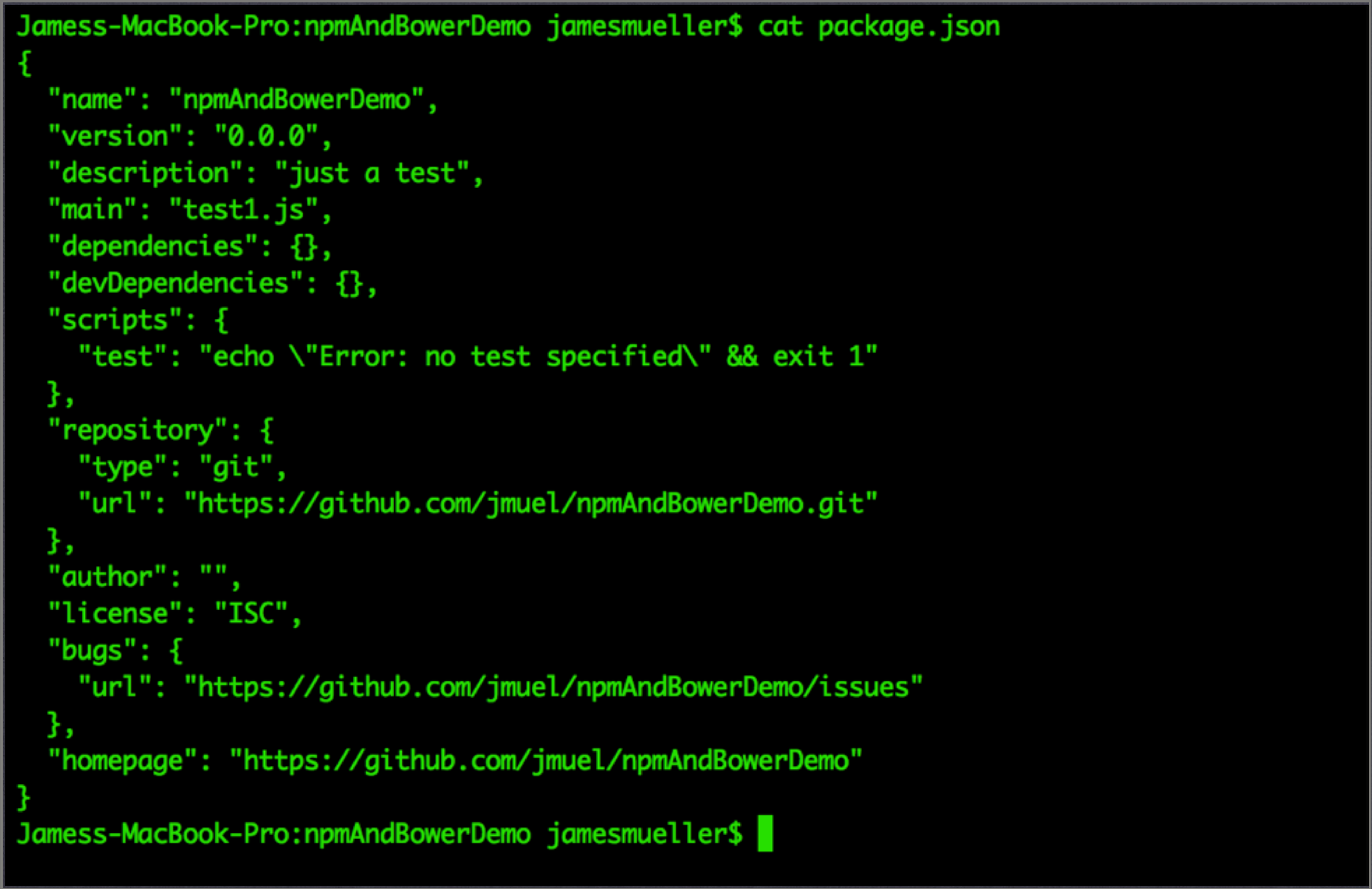
BOWER.JSON

Registering With Bower

- Have a valid bower.json (bower init helps)
- bower register mything git://mything.git
- Use git tags for versioning

Details: NPM

- Package Manager, not just Dependencies
- -g is for command line utilities, local installs are for dependencies
- —save is cool, so is —save-dev



```
Jamess-MacBook-Pro:npmAndBowerDemo jamesmueller$ cat package.json
{
  "name": "npmAndBowerDemo",
  "version": "0.0.0",
  "description": "just a test",
  "main": "test1.js",
  "dependencies": {},
  "devDependencies": {},
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "repository": {
    "type": "git",
    "url": "https://github.com/jmuel/npmAndBowerDemo.git"
  },
  "author": "",
  "license": "ISC",
  "bugs": {
    "url": "https://github.com/jmuel/npmAndBowerDemo/issues"
  },
  "homepage": "https://github.com/jmuel/npmAndBowerDemo"
}
Jamess-MacBook-Pro:npmAndBowerDemo jamesmueller$
```

NPM PACKAGE.JSON

Publishing with NPM

- First make sure you’re set to publish to our private Artifactory repository (publishing Carfax proprietary code to the world is bad). This is done through the global settings or environment variables - .npmrc
- npm publish ./

Building an NPM Module in Node

- Use a package.json with a “main” entry point
- Or have an index.js
- Use it with require(‘foo’) from a Node JS program
- Or build a command line app and have people install it with -g
- Example in stash: OAS-CLI

.npmrc

- Three levels: Root, User, Project
- User is on the K: Drive
- strict-ssl=false, ca=‘’

.bowerrc

- Copying your ssh folder from the K: drive to the C: Drive userland can help
- strict-ssl=false
- registry=fcfm01p.carfax.cfx:5678
- analytics: false

Questions?