# 46 配置选项详解 – Node.js集成

周爱民 (Aimingoo)



# 目录

- 在 Node.js 中的基本逻辑
- 在 VSCode 中的一般配置
- 3 总结

## 包在NPM视角下的四种身份



包是一个有单个或多个入口的模块 const xxx = require(...) import xxx ...

包是一个类型库 "types": ...

包是一个可运行服务 > npm start # or stop, restart

包是一个工具或可执行程序(1) > npx xxx

1) npx 会优先从当前package.json的bin中执行,然后resolve

https://docs.npmjs.com/cli/v9/using-npm/scripts#life-cycle-operation-order

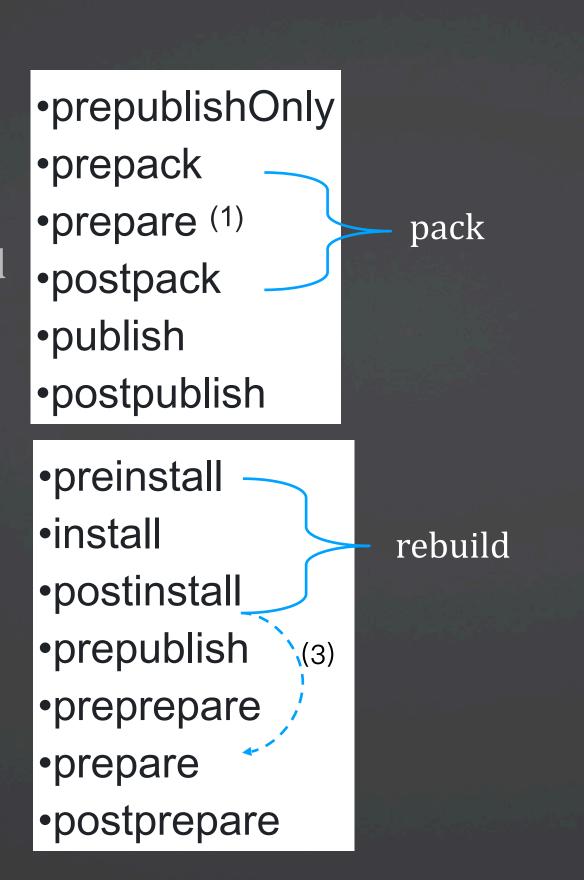


# 包在NPM视角下的第五种身份

### 包(XXX)是可交付、发布和安装的组件

- > npm publish # or unpublish, non-lifecycled
- > npm pack

- > npm install # or uninstall, non-lifecycled
- > npm install XXX
- > npm rebuild



## 其它行为

- > npm ci (2)
- > npm test
- > npm version
- > npm diff
- > npm cache add (4)
- 1) 如果指定了--dry-run,则 prepare 不会执行
- 2) 与 install 过程相同,但先完成全部模块的集成
- 3) 仅当当前目录是符号链接执行 prepare (例如 linked 包)
- 4) 仅执行 prepare

https://docs.npmjs.com/cli/v9/using-npm/scripts#life-cycle-operation-order https://docs.npmjs.com/cli/v9/commands/npm-publish#files-included-in-package

# 两种映射

```
#子路径(subpath) in package.json
# - 将替换默认main/module/types的行为
# - exports有三种不同的形式(并且也支持通配符)
"exports": {
      "types": "./dist/index.d.ts",
      "import": "./dist/index.js",
      "require": "./dist/index.cjs"
  },
  "./aaa": {
        "types": "./dist/index.d.ts",
 定义包的多个导出 {".": ... // 由type来决定模块类型
• 定义包在不同语法下的导出{"require": ... }
 定义包的多个导出,且指明它们在不同语法下的导出{".": {"require": ... }
```



#映射(module mapping) in tsconfig.json # - 将映射本项目内的代码按"包名"导入时的路径

#### @See:

https://nodejs.org/api/packages.html#subpath-exports
https://nodejs.org/api/packages.html#subpath-imports
https://webpro.nl/articles/using-subpath-imports-and-path-aliases



# 总结

- NPM包在Node.js中的一般逻辑
- 在 Node.js / NPM 环境中编译与发布 TypeScript 项目
  - ▶ 我们面临的生产环境通常是没有 VSCode 的
  - ▶ 尽量脚本化(scripts 配置或 package-scripts.js)
- 映射
  - ▶ mapRoot的映射与使用
  - ▶ imports / exports 的使用
- 在 VSCode 环境中使用 Node.js 运行和调试 TypeScript 项目
  - @see https://code.visualstudio.com/docs/editor/variables-reference



## 作业

## 1: 重现一个在VSCode中开发Node.js项目的全部步骤

- ▶ 使用Node.js运行和调试
- ▶ 使用两种方法(缺省和exports)指定Node.js项目的导出
- ▶ 在tsc生成时正确的输出文件(与上述导出位置相关)

## 2:简单了解声明文件和map文件

- · .d.ts
- · .js.map
- · .d.ts.map

## 3:(可选)在package.json中声明exports,使46/index.ts中的代码可用。

- ▶ 是 my-typescript-project/package.json
- · 在46目录中,my-typescript-project包是作为本地文件包来使用的(file: ...)
- ▸ 注意修改完 my-typescript-project 后,需要在46中npm install重新安装(你可以使用link来规避)



# THANKS