TypeScript 项目使用 npm 模块

目录

简介
TS 模块转 npm 模块
如何写 TypeScript 模块
参考链接

简介
npm 模块都是 JavaScript 代码。即使模块是用 TypeScript 写的，还是必须编译成 JavaScript 再发布，保证模块可以在没有 TypeScript 的环境运行。

问题就来了， TypeScript 项目开发时，加载外部 npm
模块，如果拿不到该模块的类型信息，就会导致无法开发。所以，必须有一个方法，可以拿到模块的类型信息
。

有些 npm 模块本身可能包含 .d.ts文件甚至完整的 TypeScript 代码。它的 package.json文件里面有一个 types字段，指向一个 .d.ts文件，这就是它的类型声明文件。

{
"name": "left-pad",
"version": "1.3.0",
"description": "String left pad",
"main": "index.js",
"types": "index.d.ts",
// ...
}

如果某个模块没有 .d.ts文件， TypeScript 官方和社区就自发为常用模块添加类型描述，可以去 官方网站搜索，然后安装网站给出的 npm 类型模块，通常是 @types/[模块名]。

```
$ npm install --save lodash
$ npm install --save @types/lodash
```

lodash 的类型描述就是 `@types/lodash` 的文件 `index.d.ts`。

TS 模块转 npm 模块

TS 代码放在 `ts` 子目录，编译出来的 CommonJS 代码放在 `dist` 子目录。

如何写 TypeScript 模块

首先，创建模块目录，然后在该目录里面新建一个 `tsconfig.json`。

```
{
  "compilerOptions": {
    "module": "commonjs",
    "target": "es2015",
    "declaration": true,
    "outDir": "./dist"
  },
  "include": [
    "src/**/*"
  ]
}
```

- `"declaration": true`：生成 .d.ts 文件，方便其他使用 TypeScript 的开发者加载你的库。
- `"module": "commonjs"`：编译后的模块格式为 `commonjs`，表示该模块供 Node.js 使用。如果供浏览器使用，则要写成 `"module": "esnext"`。
- `"target": "es2015"`：生成的 JavaScript 代码版本为 ES2015，需要 Node.js 8 以上版本。
- `"outDir": "./dist"`：编译后的文件放在 `./dist` 目录。
- `include`：指定需要编译的文件。

然后，使用 TypeScript 编写仓库代码。可以在 `src` 子目录里面，编写一个入口文件 `index.ts`。

最后，编写 `package.json`。

```
{
  "name": "hwrld",
  "version": "1.0.0",
```

```
  "description": "Can log \"hello world\" and \"goodbye world\" to the console!",
  "main": "dist/index.js",
  "types": "dist/index.d.ts",
  "files": [
    "/dist"
  ]
}
```

里面的 `"types": "dist/index.d.ts"` 字段指定类型声明文件，否则使用这个库的 TypeScript 开发者找不到类型声明文件。 `files` 属性指定打包进入 npm 模块的文件。

然后，就是编译和发布。

```
$ tsc
$ npm publish
```

参考链接

- How to Write a TypeScript Library, by tsmean

版本号 #2
由 壳壳虫 创建于 2023-07-09 12:07:22 UTC
由 壳壳虫 更新于 2023-08-01 01:15:39 UTC