# 第四节 脚手架工作原理



## 1. 原理

- 询问一些项目的问题
- 根据这些问题,结合一些模板文件生成一些项目结构
- 脚手架工具其实是一个node cli应用

## 2.步骤

1. 创建目录sample-scaffloding
2. yarn init -y 初始化一个package.json
3. 修改package.json,添加一个bin字段,指定cli的入口文件

```
{
  "name": "02-07-sample-scaffolding",
  "version": "1.0.0",
  "main": "index.js",
  "license": "MIT",
  "bin": "cli.js"
}
```

4. 项目中创建cli.js文件
5. 通过yarn link把模块关联到全局
6. 安装inquirer模块,来询问用户输入

```
yarn add inquirer
```

7. 安装模板引擎ejs

```
yarn add ejs
```

8. 编写cli.js文件

```
#!/usr/bin/env node

// Node CLI 应用入口文件必须要有这样的文件头
// 如果是 Linux 或者 macOS 系统下还需要修改此文件的读写权限为 755
// 具体就是通过 chmod 755 cli.js 实现修改

// 脚手架的工作过程:
// 1. 通过命令行交互询问用户问题
// 2. 根据用户回答的结果生成文件
const inquirer  = require('inquirer')
const ejs = require('ejs')
const fs = require('fs')
const path = require('path')

inquirer.prompt([
    {
        type: 'input',
        name: 'name',
        message: 'Project name?',
        default: 'demo'
    }
]).then(answers => {
  // 根据用户回答的结果生成文件

  // 模板目录
  const tempDir = path.join(__dirname,'templates')
  const destDir = process.cwd()
  // 将模板下的文件全部转换到目标目录
  fs.readFile(tempDir, (err,files) => {
      if(err) return
      files.forEach(file  => {
          // 通过模板引擎渲染文件
          ejs.renderFile(path.join(tempDir,file),answers,(err,result
) => {
              if(err) return
              // 将结果写入目标文件路径
              fs.writeFileSync(path.join(destDir,file),result)
```

```
            })
        })
    })
})
```