[Babel](https://babeljs.io/)是一个广泛使用的ES6转码器，可以将ES6代码转为ES5代码，从而在现有环境执行。这意味着，你可以用ES6的方式编写程序，又不用担心现有环境是否支持。

Babel的配置文件是.babelrc，存放在项目的根目录下。使用Babel的第一步，就是配置这个文件。

该文件用来设置转码规则和插件，基本格式如下。

{

"presets": [],

"plugins": []

}

presets字段设定转码规则，官方提供以下的规则集，你可以根据需要安装。

# ES2015转码规则

$ npm install --save-dev babel-preset-es2015

npm install --global babel-preset-es2015

# react转码规则

$ npm install --save-dev babel-preset-react

# ES7不同阶段语法提案的转码规则（共有4个阶段），选装一个

$ npm install --save-dev babel-preset-stage-0

$ npm install --save-dev babel-preset-stage-1

$ npm install --save-dev babel-preset-stage-2

$ npm install --save-dev babel-preset-stage-3

然后，将这些规则加入.babelrc。

{

"presets": [

"es2015",

"react",

"stage-2"

],

"plugins": []

}

注意，以下所有Babel工具和模块的使用，都必须先写好.babelrc。

Babel提供babel-cli工具，用于命令行转码。

它的安装命令如下。

$ npm install --global babel-cli

基本用法如下。

# 转码结果输出到标准输出

$ babel example.js

# 转码结果写入一个文件

# --out-file 或 -o 参数指定输出文件

$ babel example.js --out-file compiled.js

# 或者

$ babel example.js -o compiled.js

# 整个目录转码

# --out-dir 或 -d 参数指定输出目录

$ babel src --out-dir lib

# 或者

$ babel src -d lib

# -s 参数生成source map文件

$ babel src -d lib -s

上面代码是在全局环境下，进行Babel转码。这意味着，如果项目要运行，全局环境必须有Babel，也就是说项目产生了对环境的依赖。另一方面，这样做也无法支持不同项目使用不同版本的Babel。

一个解决办法是将babel-cli安装在项目之中。

# 安装

$ npm install --save-dev babel-cli

然后，改写package.json。

{

// ...

"devDependencies": {

"babel-cli": "^6.0.0"

},

"scripts": {

"build": "babel src -d lib"

},

}

转码的时候，就执行下面的命令。

$ npm run build