## babel的使用

### 基本使用

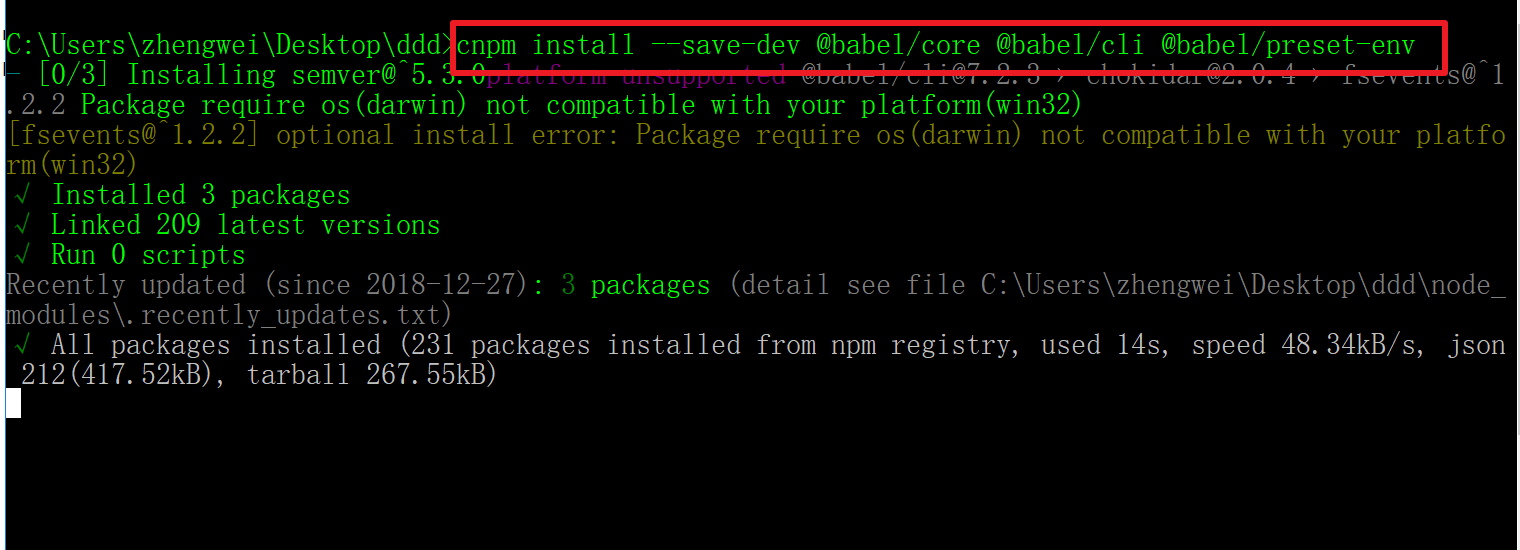
第一步、创建一个目录，初始化项目

npm init -y

第二步、安装babel相关的东西

不推荐大家安装全局,而是像下面这样本地安装，因为如果安装在全局的话，则我们以后想在不同的项目当中使用不同的版本实现不了了

cnpm install --save-dev @babel/core @babel/cli @babel/preset-env



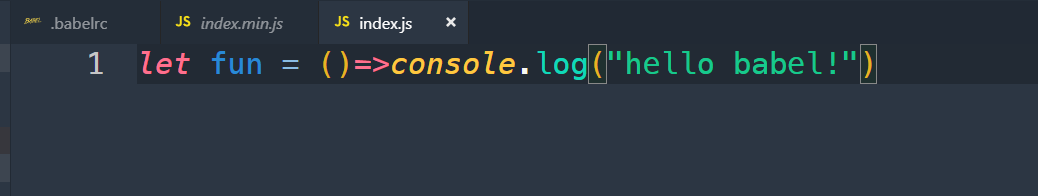
第三步、编写.babelrc文件

如果你想让你的代码在node当中运行，配置如下：

|  |
| --- |
| {  "presets": [  [  "@babel/preset-env",  {  "targets": {  "node": "current"  },  "useBuiltIns": "usage"  }  ]  ]  } |

## 配置package.json文件

第一步、写一个es6代码（测试用的）



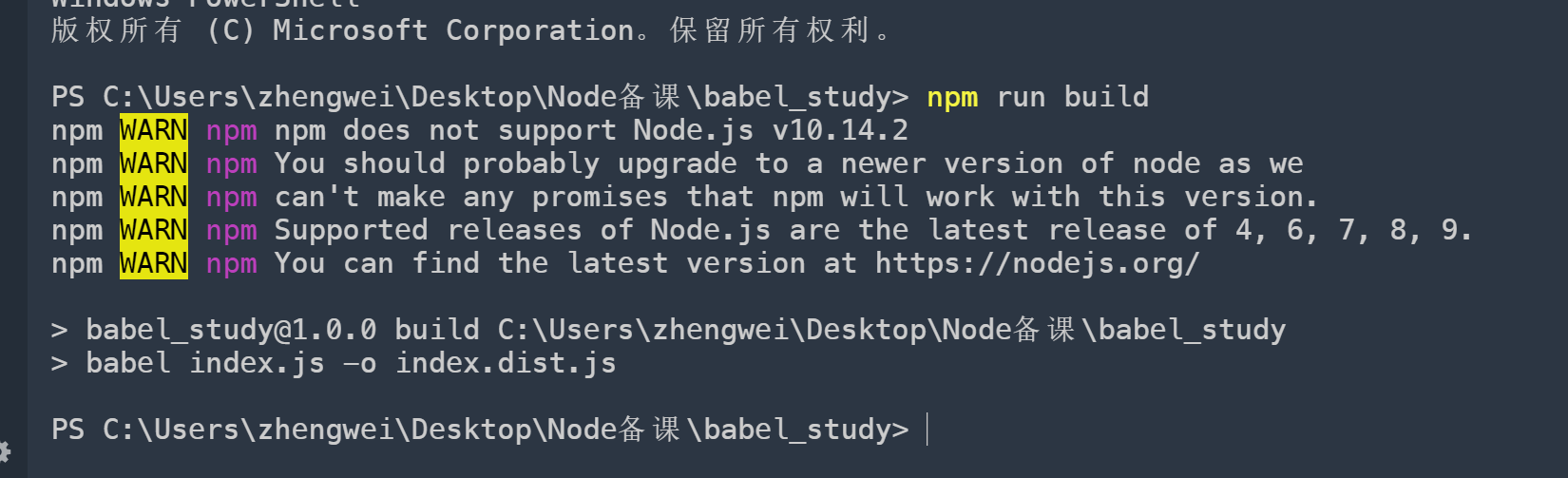
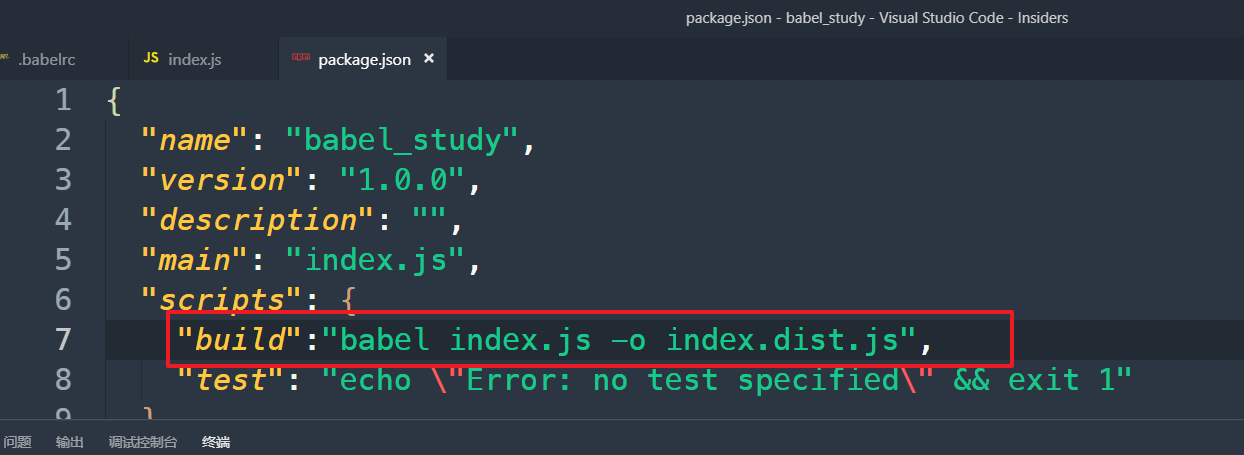
第二步、编译

.\node\_modules\.bin\babel index.js -o index.dist.js

第三步、简化命令

当然，如果觉得这个命令太长了，可以如下的方式运行

首先，配置好package.json

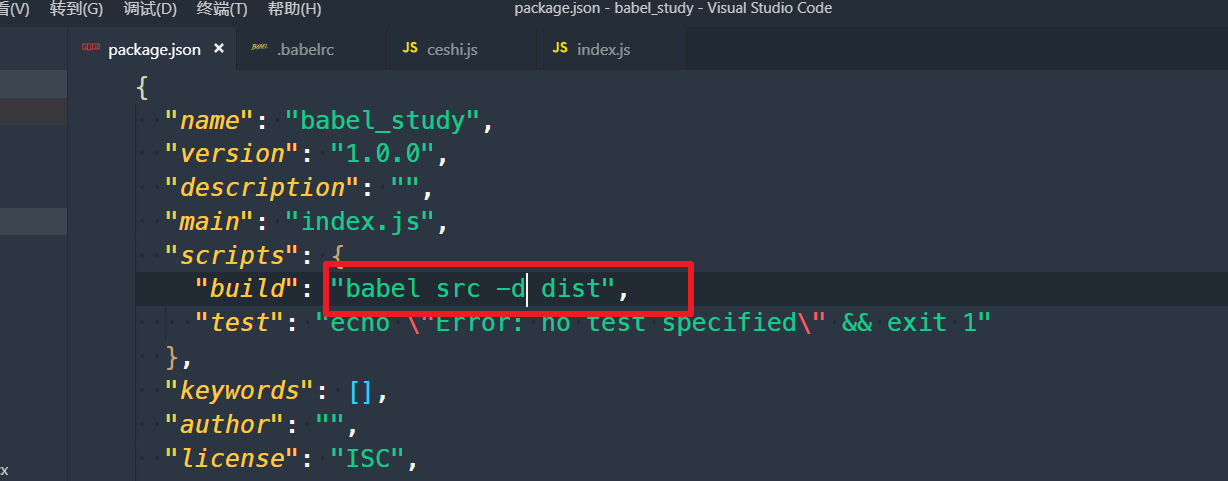


## 配置目录编译

第一步、创建src目录（我们写代码的地方）

第二步、创建dist目录（编译好代码的地方）

第三步、修改package.json



## ES6模块化的理解

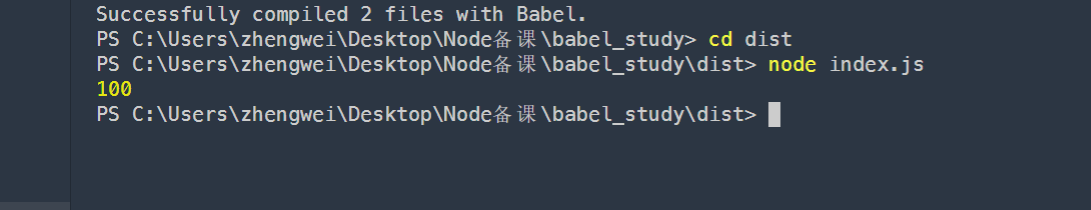
### 导出变量



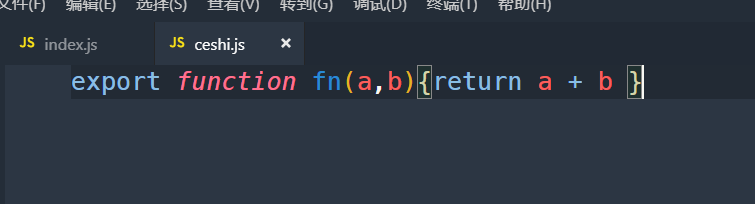


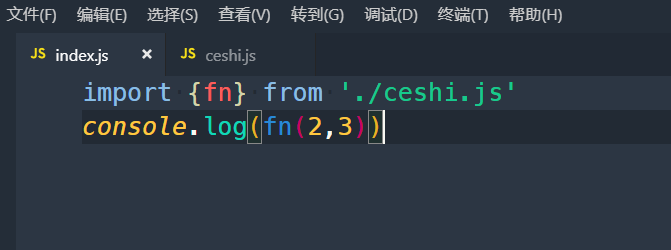
执行npm run build

切换到dist目录中，测试代码

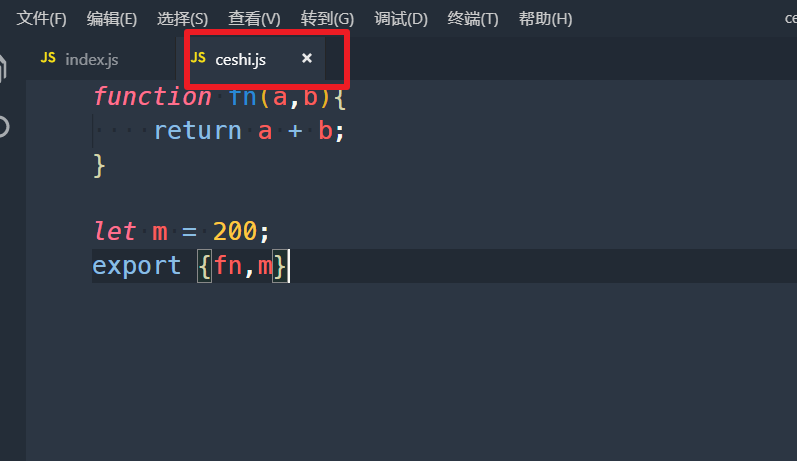


### 导出函数



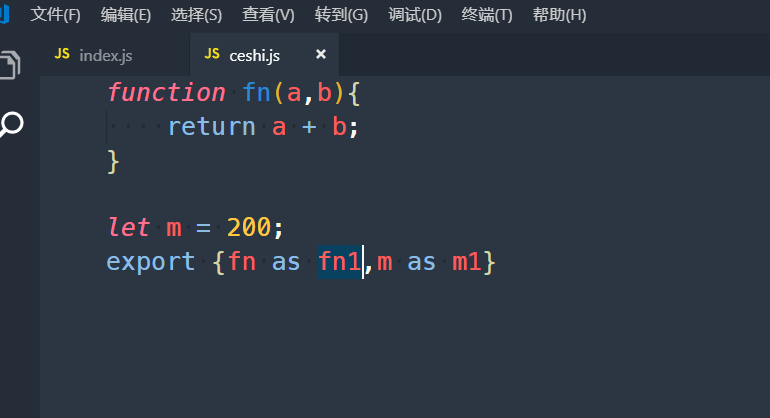


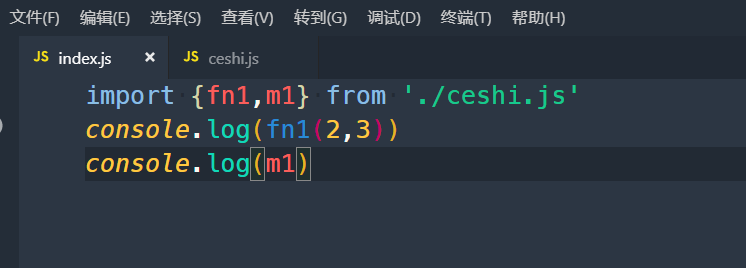
### 以export{}的形式



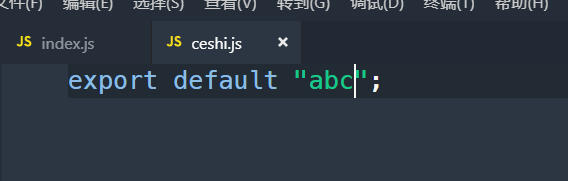


### 别名

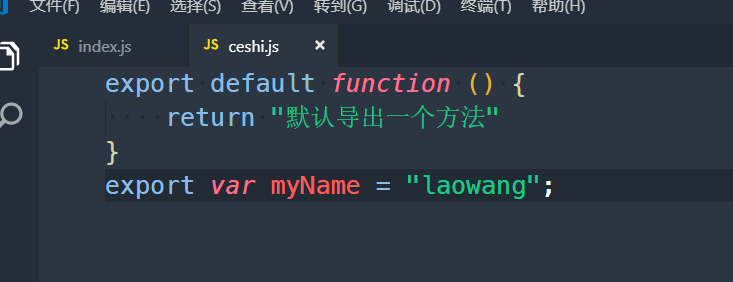




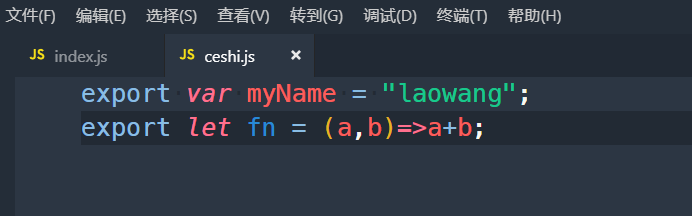
### 默认值

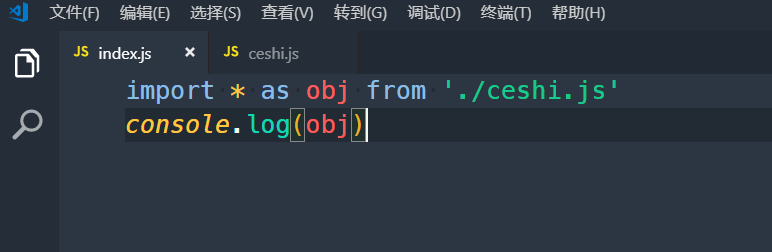


### 混合导出



### 通配符

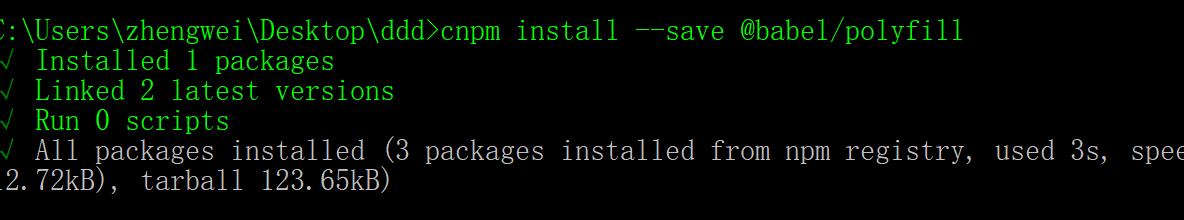




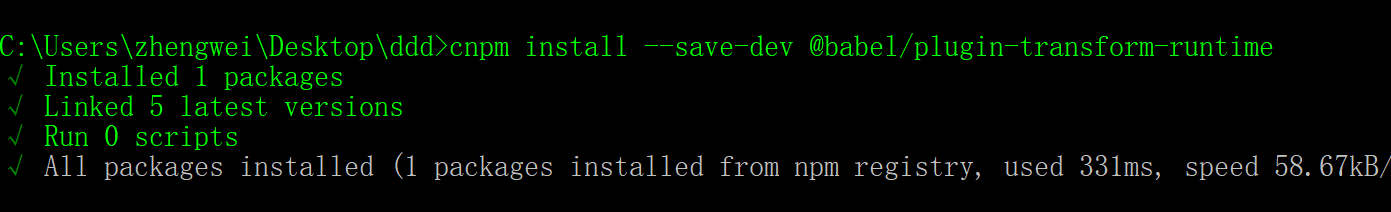
## 其他

如果你想让你的代码在浏览器当中运行

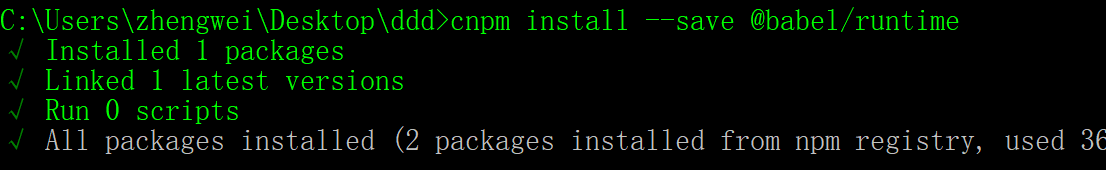
第一步、安装@babel/polyfill



第二步、安装 @babel/plugin-transform-runtime



第三步、安装@babel/runtime



|  |
| --- |
| {  "presets": [  [  "@babel/preset-env",  {  "targets": {  "ie": "7"  },  "useBuiltIns": "usage"  }  ]  ],  "plugins": [    "@babel/plugin-transform-runtime",  "@babel/plugin-transform-modules-amd"  ]  } |

我这里的设定是，如果是浏览器端，我们会编译成amd模块化，需要在用的时候引入require.js