**tsconfig.json**

tsconfig.json是ts编译时的配置文件，如下配置示例

{

  "compilerOptions": {

// 编译目标语言，如下：es5

    "target": "ES2015",

    "module": "esnext",

    "jsx": "react",

    "strict": true,

    "esModuleInterop": true,

"forceConsistentCasingInFileNames": true,

// 基路径

"baseUrl": "./",

// 编译文件输出路径 obj

    "outDir": "obj",

    "lib": [

      "dom",

      "dom.iterable",

      "esnext"

    ],

    "allowJs": true,

    "skipLibCheck": true,

    "allowSyntheticDefaultImports": true,

    "moduleResolution": "node",

    "resolveJsonModule": true,

    "isolatedModules": true,

"noEmit": true,

"paths": {

// 将 @/ 开的的 替换为 src/ 来查找文件

      "@/\*": ["src/\*"]

    }

  },

// 编译的文件，如下为 src 下所有文件

  "include": [

    "src/\*\*/\*"

  ]

}

详细配置

{

"compilerOptions": {

// "incremental": true, // 增置编译

// "tsBuildlnfoFile": "./buildFile", // 增置编译文件的存储位置  
　　// "diagnostics": true,　　　　　　　　　　　　　　// 打印诊断倍息

// "target": "es5", // 目标语言的版本

// "module": "commonjs", // 生成代码的模块标准

// "outFile": "./app-js", // 将多个相互依赖的文件生成一个文件，可以用在AMD模块中

// "lib": [], // TS 需要引用的库，即声明文件，es5 默认"dom", "es5", "scripthost"

// "allowJs": true, // 允许编译：IS 文件（js、jsx)

// "checkJs": true, // 允许在JS文件中报错，通常与allowJS \_起使用

// "outDir": "./dist", // 指定输出目录

// "rootDir": "./", // 指定输入文件目录（用于输出）

// "declaration": true, // 生成声明文件

// "declarationDir": "./d", // 声明文件的路径

// "emitDeclarationOnly": true, // 只生成声明文件

// "sourceMap": true, // 生成目标文件的 sourceMap

// "inlineSourceMap": true, // 生成目标文件的 inline sourceMap

// "declarationMap": true, // 生成声明文件的 sourceMap

// "typeRoots":[], // 声明文件目录，默认node\_modules/@types

// "types":[], // 声明文件包

// "removeComments":true, // 删除注释

// "noEmit": true, // 不输出文件

// "noEmitOnError": true, // 发生错误时不输出文件

// "noEmitHelpers": true, // 不生成helper 函数，需要额外安装ts-helpers

// "importHelpers": true, // 通过tslib 引入 helper 函数，文件必须是模块

// unoEmitOnErrorM: true, // 发生错误时不输出文件

// "noEmitHelpers": true, // 不生成 helper 函数，需额外安装 ts-helpers

// "importHelpers": true, // 通过tslib引入helper函数，文件必须是模块

// "downlevellteration" : true, // 降级遍历器的实现（es3/5)

// "strict": true, // 开启所有严格的类型检查

// "alwaysStrict": false, // 在代码中注入"use strict";

// "noImplicitAny": false, // 不允许隐式的any类型

// "strictNullChecksilj false, // 不允许把null、undefined赋值给其他类型变置

// "strictFunctionTypes": false, // 不允许函数参数双向协变

// "strictPropertyInitialization": false, // 类的实例属性必须初始化

// strictBindCallApply: false, // 严格的 bind/call/apply 检査

// "noImplicitThis": false, // 不允许this有隐式的any类型

// "noUnusedLocals": true, // 检査只声明，未使用的局部变置

// "nollnusedParameters": true, // 检查未使用的函数参数

// "noFallthroughCasesInSwitch": true, // 防止switch语句贯穿

// "noImplicitReturns": true, // 每个分支都要有返回值

// "esModulelnterop": true, // 允许export = 导出，由import from导入

// "allowUrndGlobalAccess": true, // 允许在模块中访问UMD全局变置

// "moduleResolution": "node", // 模块解析策略

// "baseUrl": "./", // 解析非相对模块的基地址

// "paths": {

// "jquery": ["node-modules/jquery/dist/jquery.slim.min.js"]

// }

// "rootDirs": ["src", "out"], // 将多个目录放在一个虚拟目录下，用于运行时

// "listEmittedFiles": true, // 打印输出的文件

// "listFiles": true, // 打印编译的文件（包括引用的声明文件） },

// "include": [ // 指定一个匹配列表（属于自动指定该路径下的所有ts相关文件）

// "src/\*\*/\*.ts",

// "src/\*\*/\*.tsx",

// "src/\*\*/\*.vue",

// "tests/\*\*/\*.ts",

// "tests/\*\*/\*.tsx"

// ],

// "exclude": [ // 指定一个排除列表（include的反向操作）

// "node\_modules"

// ],

// "files": [ // 指定哪些文件使用该配置（属于手动一个个指定文件）

// "demo.ts"

// ]

}