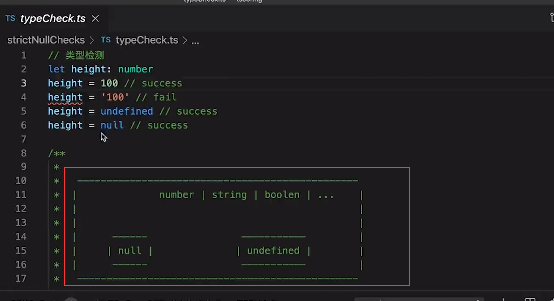


tsconfig.json配置介绍

"strictNullChecks": true,



"strictNullChecks": true,让number类型不包含null, undefined

启动之后，如果let width:number这样写的话，ts会做处理，不代表它的值就是跟js一样的undefined，具体是什么值呢，不知道，不过后面使用的时候,width必须先赋值，再使用。

如果可能是多种类型，

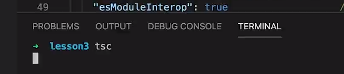
let weight: number | undefined

weight = undefined

查找文件的路径不一样

"moduleResolution": "node",            /\* Specify module resolution strategy: 'node' (Node.js) or 'classic' (TypeScript pre-1.6). \*/

进行tsc会利用tsconfig.json进行编译



"moduleResolution": "node", 会

import \_ from ‘redux’，去node\_modulse里找redux

而不是找 redux.ts

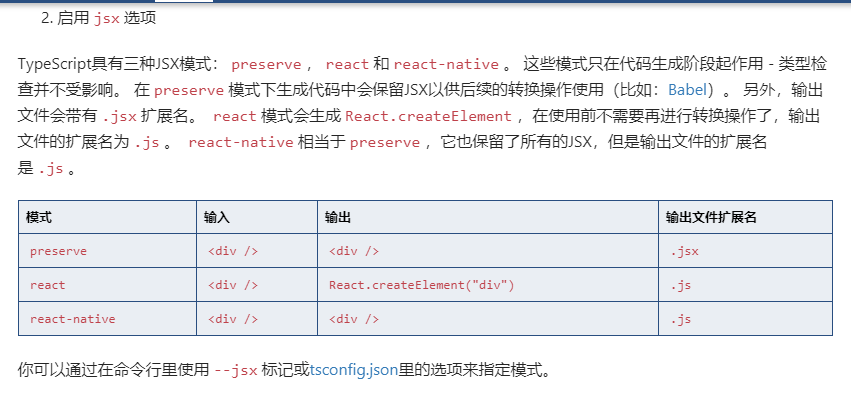
"moduleResolution": "classic", 会

import \_ from 'redux'，先找当前目录下的redux.ts, 如果没有，再找node\_modules里的redux模块。如果没有，就往父目录，一直找到根目录。

2-3 jsx配置

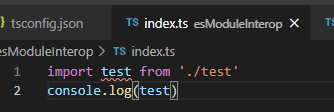
Ts允许我们写jsx, 文件名后缀名是.tsx

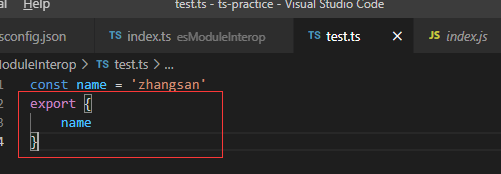
"jsx": "preserve",



2-4

"esModuleInterop": true





报错



就是说这种导入方式import test from './test'要有的导出方式为export default xxx

但是这种方式怎么处理



a.js Module.exports = 12

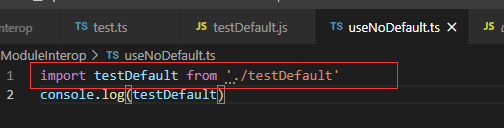
B.js

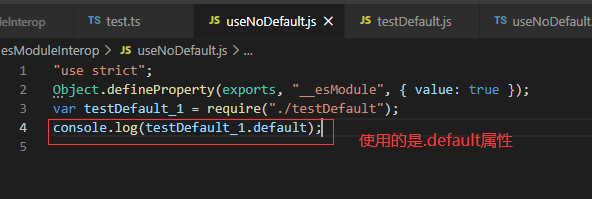
import a from ‘./a’

Console.log(a) //undefined 不是12

应为import a from ‘./a’ 读取的是a.js 里的.default

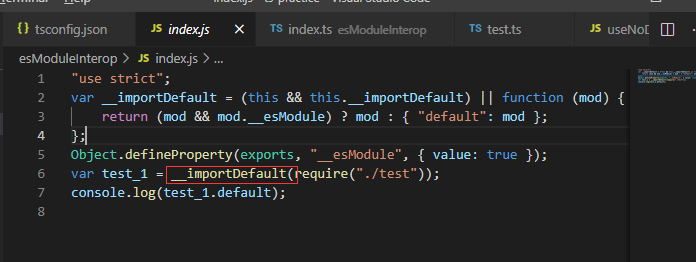
像





"esModuleInterop": true 这个配置项会给没有.default的到处模块加上.default

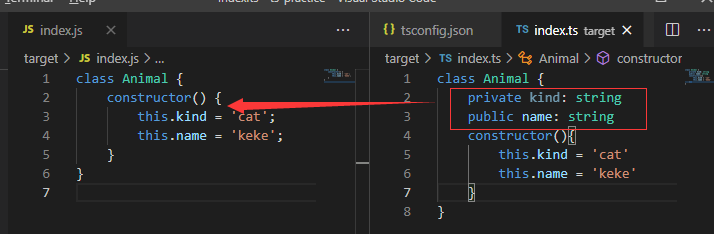
打包后的代码如下



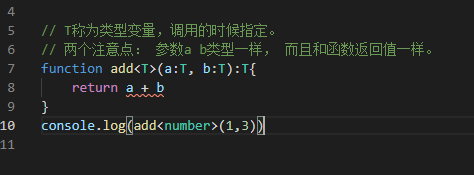
2-5 "noImplicitAny": true, 不能有隐式的any类型，比如const studeny ={}, key的类型没有指定就被认为是any，就报错了

2-6 "target": "es5", 代码编译成js的哪个版本

Public private是类型声明，js里面是没有的，编译后会被删除。

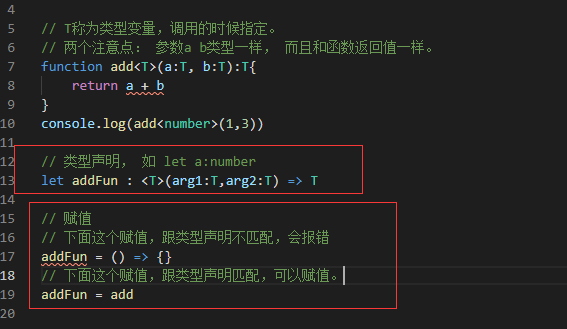


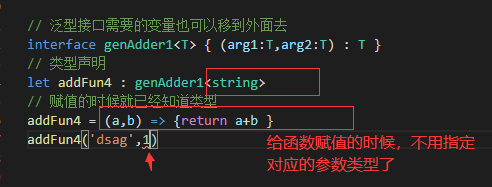
3-1 泛型



3-2 泛型接口

泛型函数

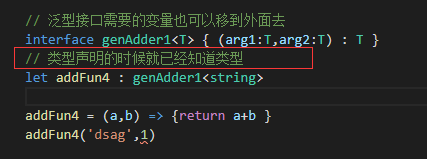




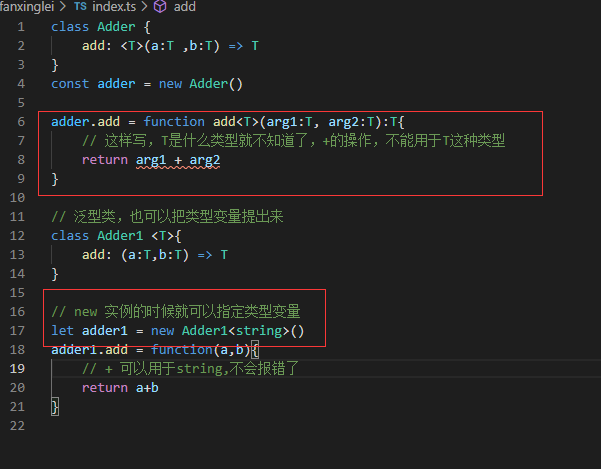
泛型的两种写法

// 调用的时候指定类型

addFun<number>(2,3)



泛型类可以两种写法，实例的时候确定或者实例后在赋值去确定



3-4 泛型约束



