**环境搭建和调试**

1. 安装 vscode 和node
2. 在工作目录下

建立目录 testroot

在其下 建立 src 和 dist 子目录

1. 初始化 npm

在testroot 目录下 执行 npm init -y

此时会在 testroot 目录下 建立 package.json 文件

1. 安装 typescript

在testroot 目录下 执行 npm install -D typescript

1. 创建和配置 tsconfig.json

在testroot 目录下 执行 tsc --init

此时会在 testroot 目录下 建立 tsconfig.json 文件

修改 tsconfig.json 文件，取消相关注释，添加以下两条

"outDir": "./dist",

"rootDir": "./src",

1. 写入测试代码

在 src 子目录中新建 index.ts 文件 添加如下测试代码

const hello: string = 'hello world';

console.log(hello);

1. 手工编译和执行

在testroot 目录下执行如下命令

tsc (编译 ts 到 js)

node ./dist/index.js (执行)

1. 自动编译

在vscode中 Ctrl + Shift + B 弹出构建任务窗口

选择 tsc watch tsconfig.json ,此时ts的任何修改都会在自动编译到js中

1. 代码调试

npm install -D ts-node 安装依赖

在testroot目录下新建.vscode目录，在.vscode目录下新建launch.json文件，内容如下

{

    "version": "0.2.0",

    "configurations": [

        {

            "type": "node",

            "request": "launch", // 请求配置类型，可以为launch（启动）或attach（附加）

            "name": "Launch Program",

            "runtimeArgs":[

                "-r",

                "ts-node/register"

            ],

            "args": [

                "${workspaceFolder}/src/index.ts"

            ]

        }

    ]

}

设置断点，通过 F5调试，

**类的定义和引用**

定义test1.ts

export class test1 {

    constructor(parm1: string) {

        console.log('test1:.' + parm1);

    }

    static statictest() :string  {

        console.log('statictest');

        return 'bbb';

    }

     funtest() :string  {

        console.log('funtest');

        return 'ccc';

    }

}

在index.ts中通过 import {test1} from '../src/test1'; 引用和调用

import {test1}   from '../src/test1';

let  mm:test1 = new test1('mm');

test1.statictest();

mm.funtest();

ESLINT配置

搭建http服务器，搭建restful接口