vue3

ES6模块化与异步编程

什么是ES6模块化规范

ES6模块化规范是浏览器端与服务器端通用的模块化开发规范。它的出现极大的降低了前端开发者的模块化学

习成本，开发者不需再额外学习AMD、CMD或CommonJS等模块化规范。

ES6模块化规范中定义：

* 每个js 文件都是一个独立的模块
* 导入其它模块成员使用import关键字
* 向外共享模块成员使用export关键字

在node.js中体验ES6模块化

node.js中默认仅支持CommonJS 模块化规范，若想基于node.js体验与学习ES6的模块化语法，可以按照

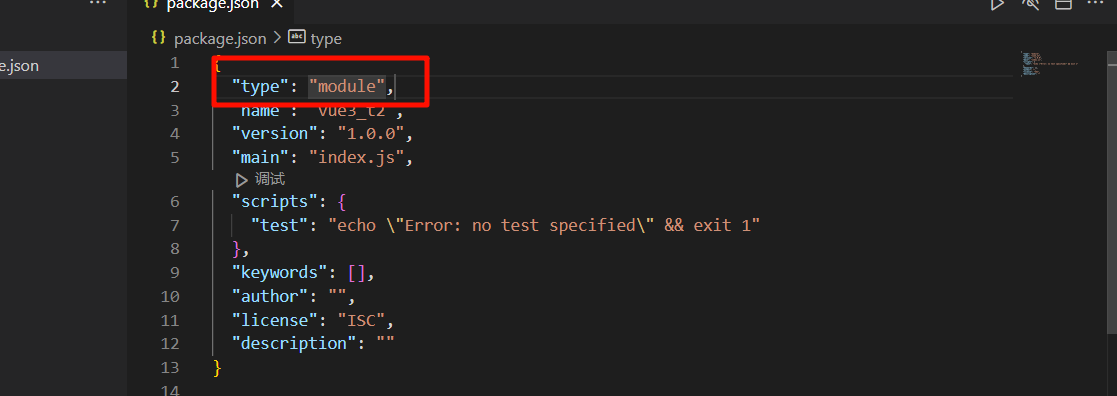
如下两个步骤进行配置：

①确保安装了v14.15.1或更高版本的node.js

②在package.json的根节点中添加"type":"module"节点

PS E:\Desktop\Example\VUE3\vue3\_t2> npm init -y #快速创建一个package.json

增加"type":"module"节点



ES6模块化的基本语法

ES6的模块化主要包含如下3种用法：

默认导出与默认导入

默认导出

每个模块中，只允许使用唯一的一次exportdefault，否则会报错！



默认导入

接收名称命名规范和c语言相同，字母数字下划线，不能数字开头



按需导出与按需导入

注意：

* 每个模块中可以使用多次按需导出
* 按需导入的成员名称必须和按需导出的名称保持一致
* 按需导入时，可以使用as关键字进行重命名

import { s1,s2 as s123,show} from "./3.js";

console.log(s1);

console.log(s123);

show(1,2);

* 按需导入可以和默认导入一起使用，默认导出用变量名

import info, { s1,s2 as s123,show} from "./3.js";

console.log(s1);

console.log(s123);

show(1, 2);

console.log(info); // { qwe: 123 }



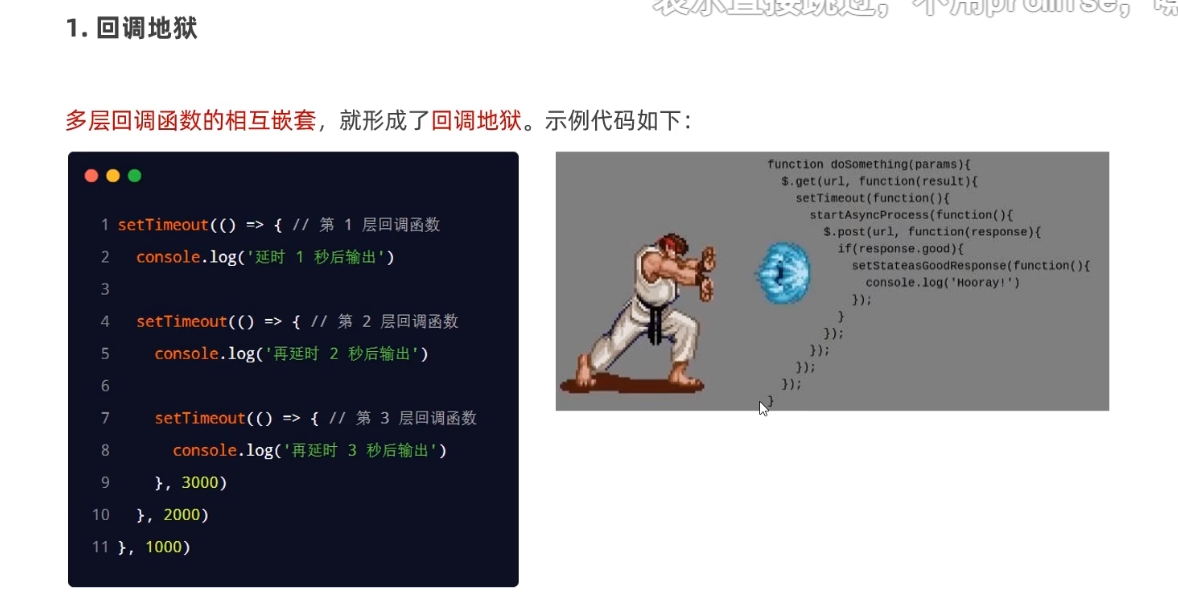


直接导入并执行模块中的代码



Promise

回调地狱



Promise的基本概念

1. Promise是一个构造函数

我们可以创建Promise的实例constp=newPromise()

new出来的Promise实例对象，代表一个异步操作

2. Promise.prototype上包含一个.then()方法

每一次new Promise（）构造函数得到的实例对象，

都可以通过原型链的方式访问到.then(）方法，例如p.then()

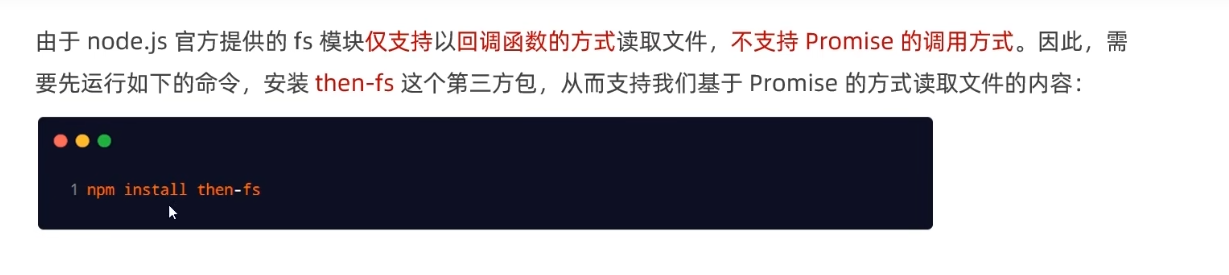
3.then(）方法用来预先指定成功和失败的回调函数

p.then(成功的回调函数，失败的回调函数)

p.then(result => {}, error =>{l)

调用.then()方法时，成功的回调函数是必选的、失败的回调函数是可选的

then-fs的基本使用





基于Promise按顺序读取文件的内容



通过.catch捕获错误



