Three.js journey

three官方文档地址：

https://threejs.org/docs/

**L2 what is webGL and why three.js**

Three.js可与SVG和CSS一起使用但是可能会受性能限制

所以用3js webGL

先学习Node.js，built tools（前称bundlers）vite、webpack；react。。

**L3. First three.js project**

推荐使用vs code中的终端

cd 当前目录

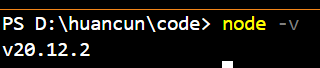
ls list

pwd 当前路径

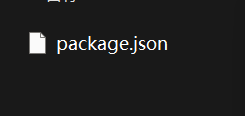
一些简单操作

安装node.js NTL版https://nodejs.org/en

安装后终端输入检查（重启vscode或windows）



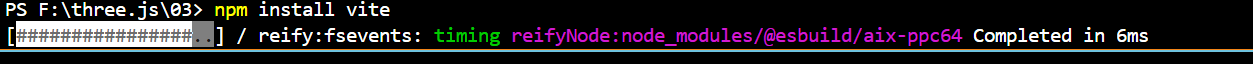
输入npm init -y可以新建文件 （node package manager）

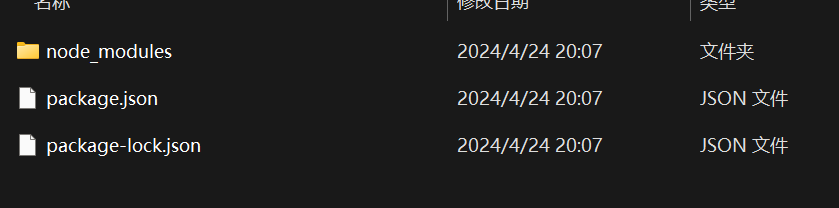


库：

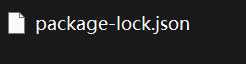
npmjs.com/package/vite

npmjs.com/package/three

安装vite



共享文件时可以去除

文件

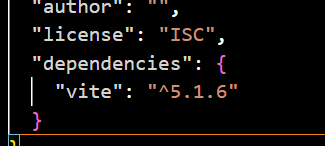
在package.json中修改

"scripts": {

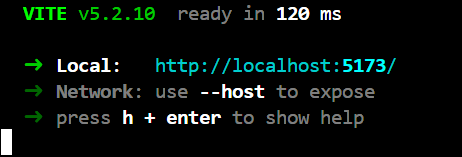
"dev":"vite",

"build":"vite build"

},

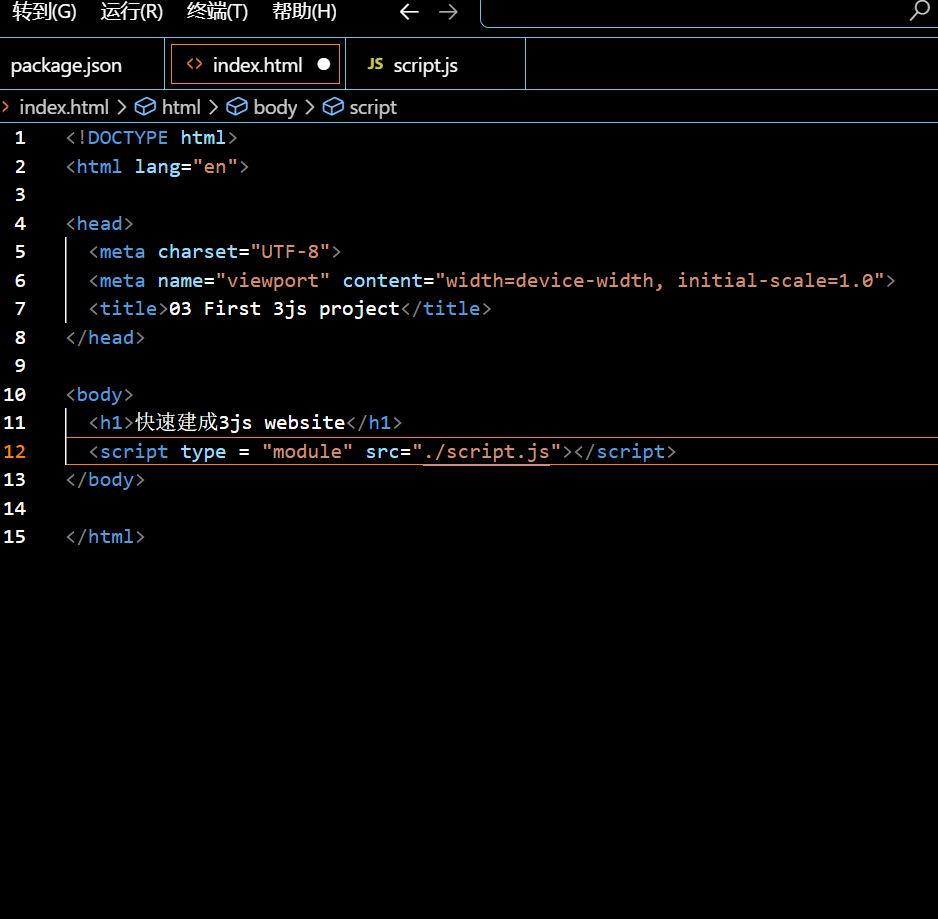
加上“de。。”

接下来就可以用npm run dev来运行vite

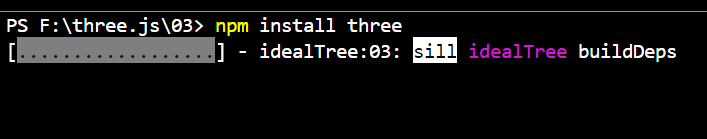


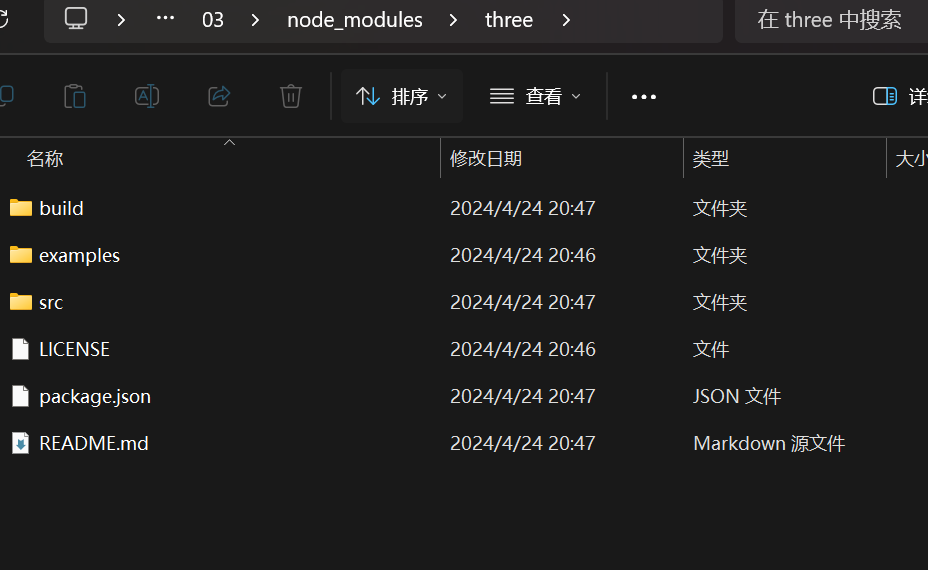
（停止快捷键：ctl+c）

在html中加入



安装3js（需要新建终端或关闭vite）





在script.js中import

import \* as THREE from 'three'

一个基本场景需要四个element：

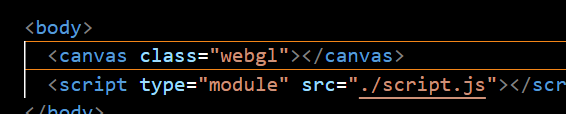
Scene

Objects

Camera

Renderer 渲染器

为了添加渲染器，需要在html文件中添加canvas：



Script.js代码：

import \* as THREE from 'three'

//canvas

const canvas = document.querySelector('canvas.webgl')     //呈现画布

console.log(canvas)

//scene

const scene = new THREE.Scene()

//object

const geometry = new THREE.BoxGeometry(1, 1, 1)

const material = new THREE.MeshBasicMaterial({ color: '#ff0000' })//color, wireframe: true 可看到网格

const mesh = new THREE.Mesh(geometry, material)

scene.add(mesh)

//sizes

const sizes = {

  width: 800,

  height: 600

}

//camera

const camera = new THREE.PerspectiveCamera(75, sizes.width / sizes.height)

camera.position.z = 3  //camera位置

scene.add(camera)

//renderer

const renderer = new THREE.WebGLRenderer({

  canvas: canvas

})

renderer.setSize(sizes.width, sizes.height)  //编辑画布大小

renderer.render(scene, camera)

问题总结：1：10：05

因为是全英文的教程看得好累！一个多小时的教程看了一晚上，不过可能也有配置环境的原因。

L4 Transform objects