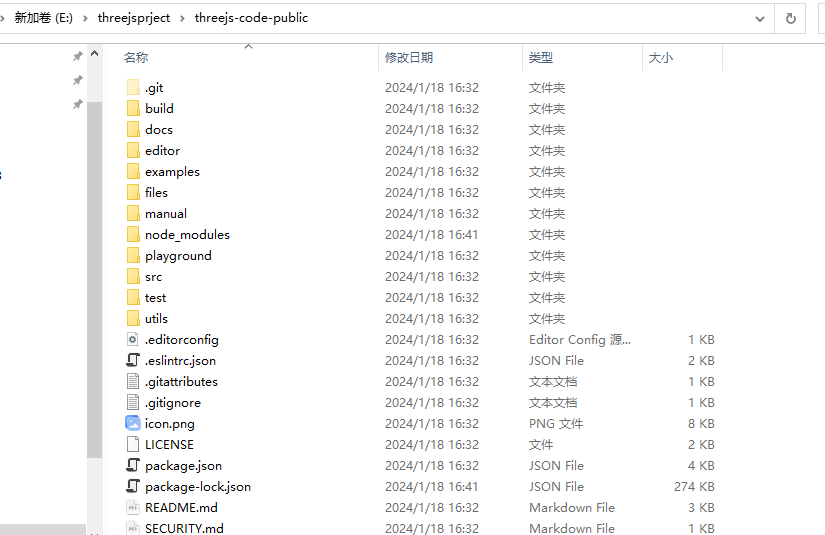
# 准备

## 部署文档到本地

在<https://gitee.com/xin_hu199/threejs-code-public/> git下载到本地

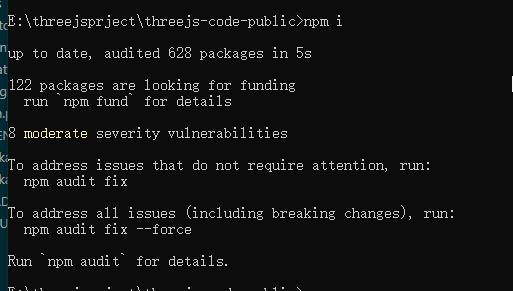
git clone [git@gitee.com:xin\_hu199/threejs-code-public.git](mailto:git@gitee.com:xin_hu199/threejs-code-public.git)

下载后如下图所示：



然后在VCCODE部署

安装依赖 npm i

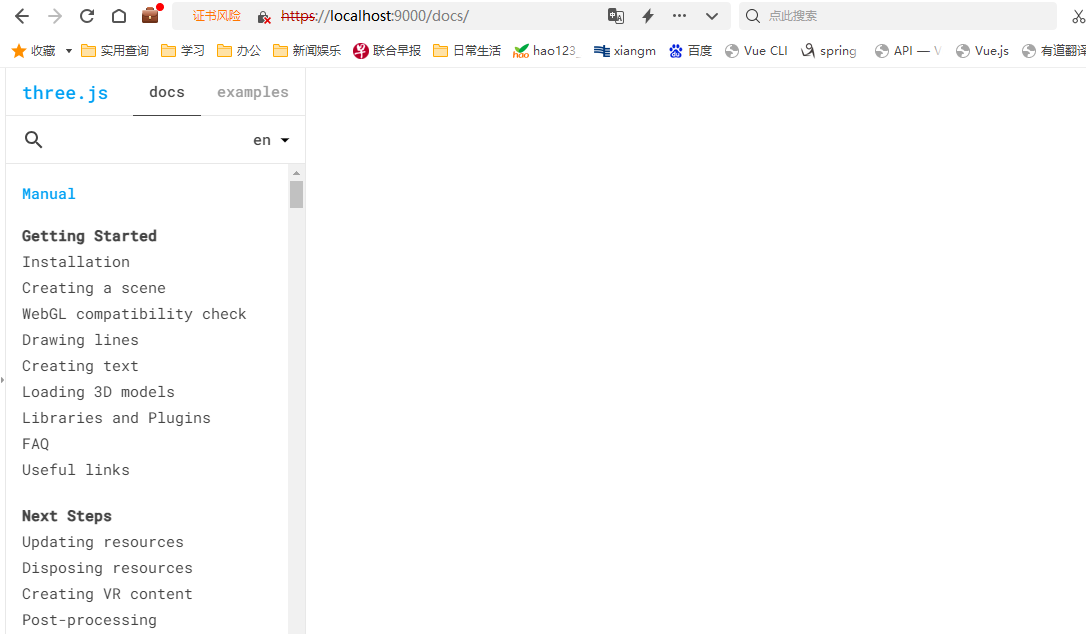
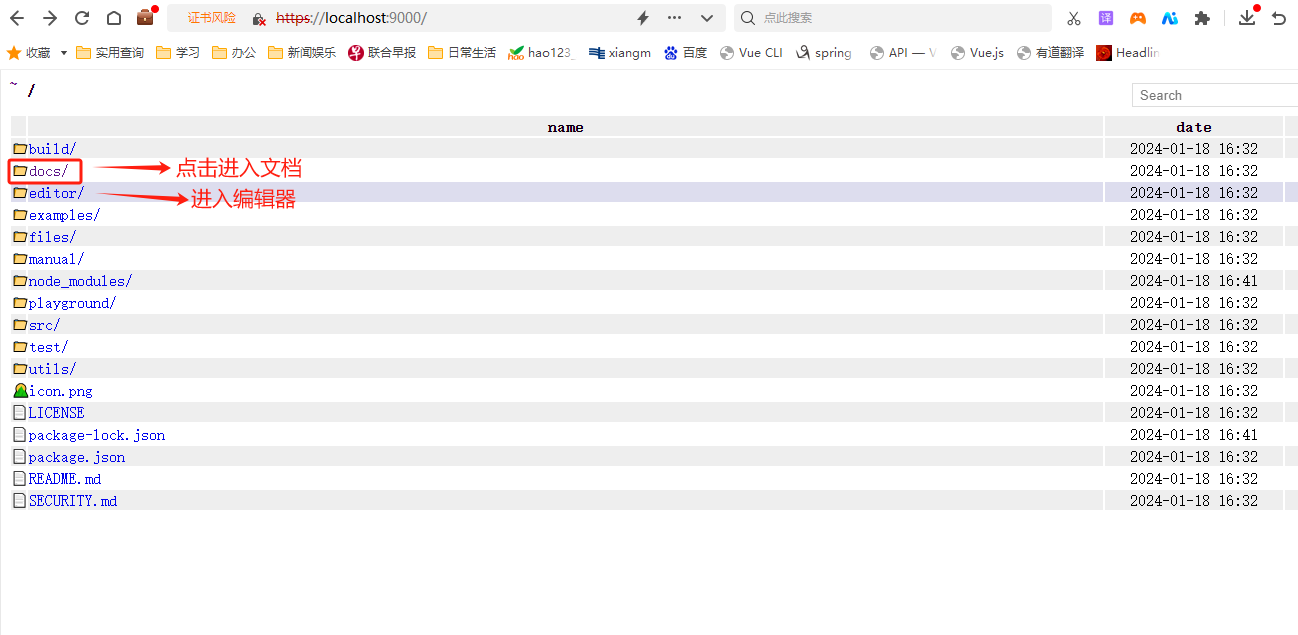


启动

npm run dev



启动后在浏览器输入



## 官方中文网

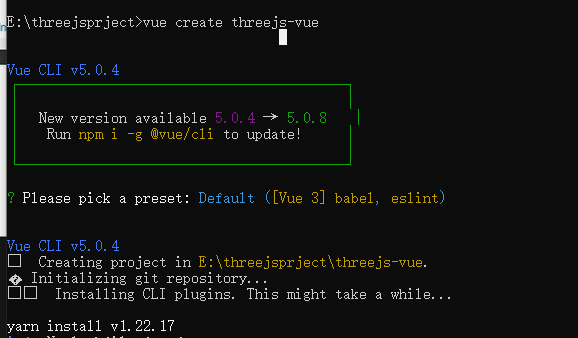
http://www.webgl3d.cn/

# 创建vue项目

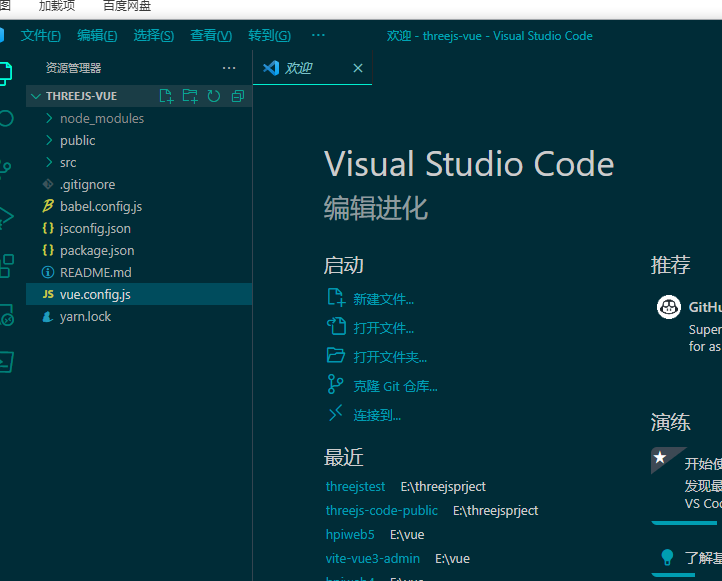
## 创建vue项目

在工程目录中以脚手架的模式创建vue3项目

vue create threejs-vue

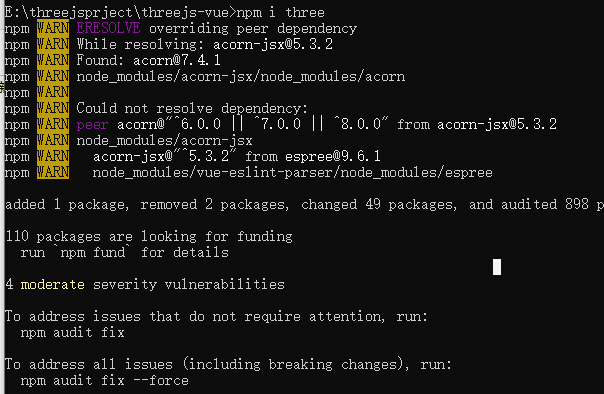


创建成功后，在vccode打开



## 添加threejs依赖

npm i three



## 运行threejs中的实例

项目创建成功后

在App.vue中编写如下代码

<template>

  <div></div>

</template>

<script setup>

*//import HelloWorld from './components/HelloWorld.vue'*

*/\* export default {*

*name: 'App',*

*components: {*

*HelloWorld*

*}*

*} \*/*

import \* as THREE from 'three';

const width = window.innerWidth, height = window.innerHeight;

*// init*

const camera = new THREE.PerspectiveCamera( 70, width / height, 0.01, 10 );

camera.position.z = 1;

const scene = new THREE.Scene();

const geometry = new THREE.BoxGeometry( 0.2, 0.2, 0.2 );

const material = new THREE.MeshNormalMaterial();

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

scene.add( mesh );

const renderer = new THREE.WebGLRenderer( { antialias: true } );

renderer.setSize( width, height );

renderer.setAnimationLoop( animation );

document.body.appendChild( renderer.domElement );

*// animation*

function animation( time ) {

  mesh.rotation.x = time / 2000;

  mesh.rotation.y = time / 1000;

  renderer.render( scene, camera );

}

</script>

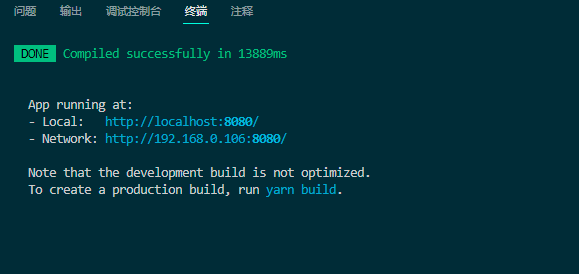
<style>

</style>

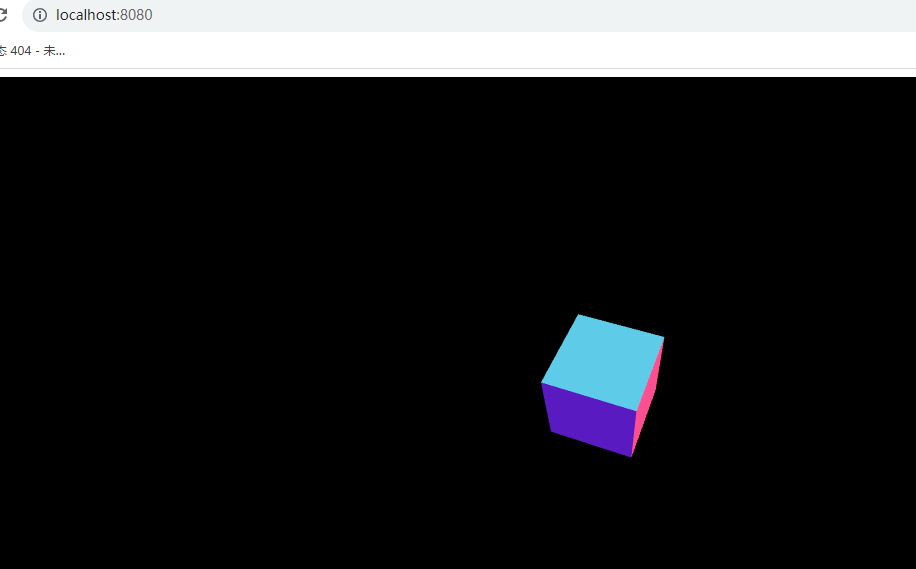


## 启动

npm run serve



启动成功后运行如下：



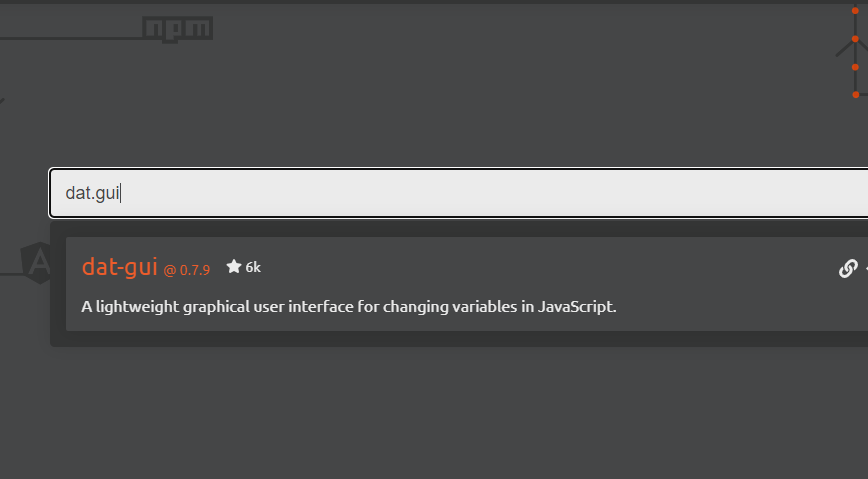
# 结合dat.gui实现界面可视化修改及调试详解

dat.GUI 是一个轻量级的图形用户界面库（GUI 组件），使用这个库可以很容易地创建出能够改变代码变量的界面组件。

一个轻量型的JavaScript库，他能提供便捷的方式，让你去修改或者操作js中的变量和函数

## 引入js

打开<https://cdnjs.com/，搜索> dat.gui,如下图所示：



然后再html中引入

<script src="https://cdnjs.cloudflare.com/ajax/libs/dat-gui/0.7.9/dat.gui.min.js"></script>

## Html测试效果

<!DOCTYPE html>

<html>

    <head>

        <meta charset="utf-8">

        <title>My first three.js app</title>

        <style>

            body { margin: 0; }

        </style>

    </head>

    <body>

        <script type="text/javascript" src="https://cdnjs.cloudflare.com/ajax/libs/dat-gui/0.7.9/dat.gui.min.js"></script>

        <script>

*//创建控制对象，该对象将保存希望通过 dat.GUI 改变的属性。*

            const controlData={

                rotationSpeed: 0.01,*//物体旋转的速度，默认为0.01*

                color:'#66ccff',*//颜色*

                wireframe:false,*//显示的一个线框*

                wireframe1:false,*//显示的一个线框*

                envMap:'无'*//材质选项*

            }

*//创建gui实例*

            const gui = new dat.GUI();

            const f = gui.addFolder('配置')*//添加文件夹*

*//f.add(controlData, "rotationSpeed",0.01,0.1,0.01 );//最小值，最大值、步长*

            f.add(controlData, "rotationSpeed").min(0.01).max(0.1).step(0.01)

*//颜色选择器*

            f.addColor(controlData,"color")

*//下拉列表*

            f.add(controlData,"envMap",['无','全反射','漫反射'])

*//checkbox*

            f.add(controlData,"wireframe")

            f.add(controlData,"wireframe1")

            f.open()

            f.open()

        </script>

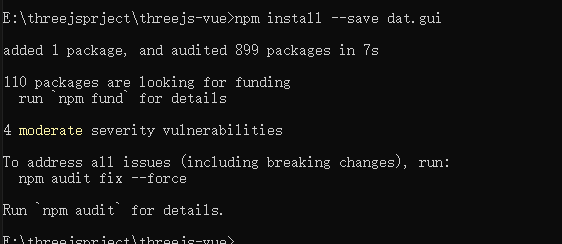
    </body>

</html>

## 在vue和threejs集成

先安装dat.gui

npm install --save dat.gui



<template>

  <div id="container">

*<!-- <button @click="moveCamera">移动相机</button>*

*<button @click="moveCube">移动物体</button> -->*

  </div>

</template>

*<!--*

*dat.gui实现界面可视化修改及调试*

*通过dat.gui组件调整材质运行速度、颜色、线框属性*

*-->*

<script setup>

import { onMounted } from "vue";

import \* as THREE from "three";

*//OrbitControls 轨道控制器是一个附加组件，必须显式导入。*

import { OrbitControls } from "three/addons/controls/OrbitControls.js";

*/\*\*\*\*\*\*\*\*\*\*\*dat.gui 集成start\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \*/*

import dat from "dat.gui"

*//创建控制对象，该对象将保存希望通过 dat.GUI 改变的属性。*

const controlData = {

  rotationSpeed: 0.01, *//物体旋转的速度，默认为0.01*

  color: "#66ccff", *//颜色*

  wireframe: false, *//显示的一个线框*

  wireframe1: false, *//显示的一个线框*

  envMap: "无", *//材质选项*

};

*//创建gui实例*

const gui = new dat.GUI();

const f = gui.addFolder("配置"); *//添加文件夹*

*//f.add(controlData, "rotationSpeed",0.01,0.1,0.01 );//最小值，最大值、步长*

f.add(controlData, "rotationSpeed").min(0.01).max(0.1).step(0.01);

*//颜色选择器*

f.addColor(controlData, "color");

*//下拉列表*

f.add(controlData, "envMap", ["无", "全反射", "漫反射"]);

*//checkbox*

f.add(controlData, "wireframe");

f.add(controlData, "wireframe1");

*//绑定dom元素，给dat.gui定义id为gui,需要通过css调整样式*

f.domElement.id= "gui"

f.open();

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*dat.gui 集成 end\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*8888888\*/*

const width = 4;

const height = 4;

*//创建一个场景*

const scene = new THREE.Scene();

*//const camera = new THREE.PerspectiveCamera();*

*//创建透视相机*

*/\*\**

*PerspectiveCamera( fov : Number, aspect : Number, near : Number, far : Number )*

*fov — 摄像机视锥体垂直视野角度*

*aspect — 摄像机视锥体长宽比*

*near — 摄像机视锥体近端面*

*far — 摄像机视锥体远端面*

*\*/*

const camera = new THREE.PerspectiveCamera(45, width / height, 1, 1000);

camera.position.z = 10;

camera.position.y = 2;

*//创建一个立方体*

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

*//const geometry = new THREE.BoxGeometry();*

*//创建一个基础网格材质*

const material = new THREE.MeshBasicMaterial({ color: 0x00ff00 });

*//创建网格*

const cube = new THREE.Mesh(geometry, material);

cube.position.set(0, 2, 0); *//设置x、y、z 轴，设置方块的位置*

scene.add(cube);

*//添加一个网格地面*

const size = 10;

const divisions = 10;

const gridHelper = new THREE.GridHelper(size, divisions);

scene.add(gridHelper); *//网格地面添加到场景*

*/\* const moveCamera = () => {*

*camera.position.y = 15; //向上移动15各单位*

*camera.position.z = 10;*

*camera.lookAt(0, 2, 0);*

*};*

*const moveCube = () => {*

*cube.position.set(3, 3, 0); //x、y移动3*

*camera.lookAt(cube.position); //相机看向物体*

*}; \*/*

*//*

onMounted(() => {

*//创建渲染器*

*//const renderer = new THREE.WebGLRenderer();*

  const renderer = new THREE.WebGLRenderer({ antialias: true, alpha: true });

  renderer.setClearAlpha(0.2);

  renderer.setSize(window.innerWidth, window.innerHeight);

*//document.body.appendChild(renderer.domElement);*

*//将dat.gui组件加入到场景页面中*

  document.getElementById("container").appendChild(f.domElement)

*//vue热加载，不用涮新浏览器，结合onMounted钩子函数*

  document.getElementById("container").appendChild(renderer.domElement);

*//增加一个轨道控制器，控制地面三维效果,当鼠标左键按住地面时旋转*

  const controls = new OrbitControls(camera, renderer.domElement); *// eslint-disable-line no-unused-vars*

*/\* 给场景设置背景 \*/*

  controls.addEventListener("change", function () {

    console.log("轨道控制器改变发生改变");

  });

*//给控制器添加阻尼效果*

  controls.enableDamping = true;

*//当enableDamping设置为true的时候，阻尼惯性有多大。 Default is 0.05.*

  controls.dampingFactor = 0.01;

*//将其设为true，以自动围绕目标旋转。*

  controls.autoRotate = true;

*//当.autoRotate为true时，围绕目标旋转的速度将有多快，默认值为2.0，相当于在60fps时每旋转一周需要30秒。*

  controls.autoRotateSpeed = 1;

*//设置雾*

*//Fog( color : Integer, near : Float, far : Float )*

*//颜色参数传入Color构造函数中，来设置颜色属性。颜色可以是一个十六进制的整型数，或者是CSS风格的字符串。*

*//效果是当推动鼠标的滚轮的时候，盒子颜色变淡，产生雾的感觉*

*// scene.fog = new THREE.Fog( 0xcccccc, 10, 15 );*

*//创建一个相机*

*//const camera = new THREE.PerspectiveCamera( 75, window.innerWidth / window.innerHeight, 0.1, 1000 );*

*//创建一个动画函数,函渲，让方块转动*

  function animate() {

    requestAnimationFrame(animate); *//动画函数*

*//cube.rotation.x += 0.01; //x轴方向*

*//cube.rotation.y += 0.01;*

*//替换成通过dat.gui组件控制旋转速度*

    cube.rotation.x +=controlData.rotationSpeed

    cube.rotation.y +=controlData.rotationSpeed

*//cube.rotation.z += 0.01;*

*//通过dat.gui组件color属性控制材质颜色*

    cube.material.color = new THREE.Color(controlData.color)

*//通过线框改变材质*

    material.wireframe=controlData.wireframe

*//轨道控制器更新*

    controls.update();

    renderer.render(scene, camera);

  }

*//添加网格地面*

*//进行渲染*

*//renderer.render( scene, camera );*

*//调用函数渲染*

  animate();

});

</script>

<style>

*/\* body {*

*margin: 0;*

*overflow: hidden;*

*background: url("E://1116/32.jpg") center no-repeat;*

*background-size:cover;*

*} \*/*

*/\* 通过样式控制dat.gui位置和大小 \*/*

#gui {

  position: absolute;

  right: 0;

  width: 300px;

}

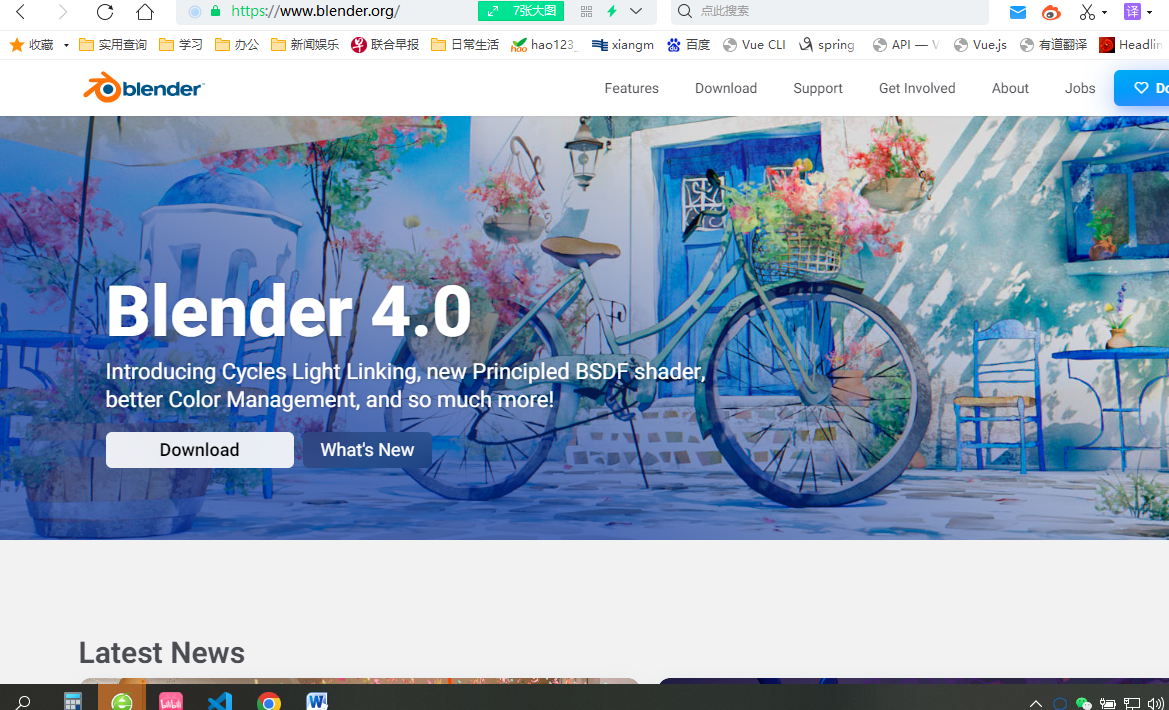
</style>

# 模型建模工具blender

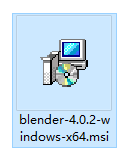
Blender是一款免费开源三维图形图像软件，提供从建模、动画、材质、渲染、到音频处理、视频剪辑等一系列动画短片制作解决方案。

## 下载安装

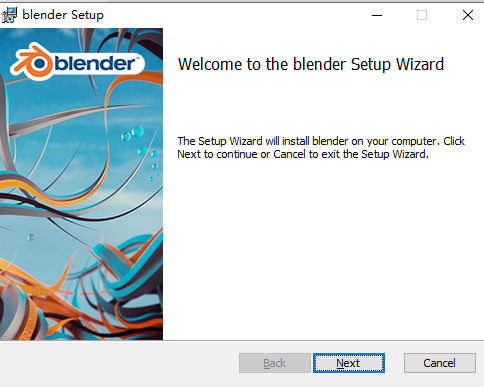
进入官网<https://www.blender.org/>，点击



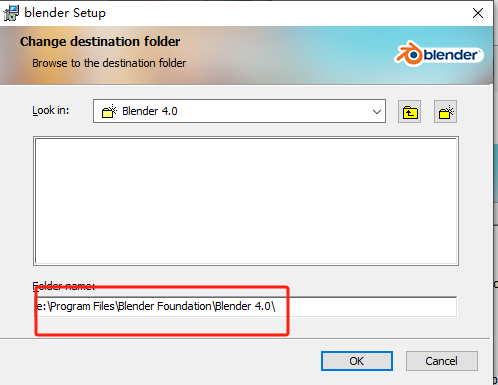
下载成功后，如下图所示：

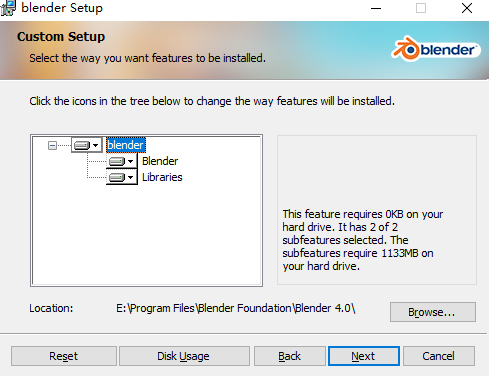


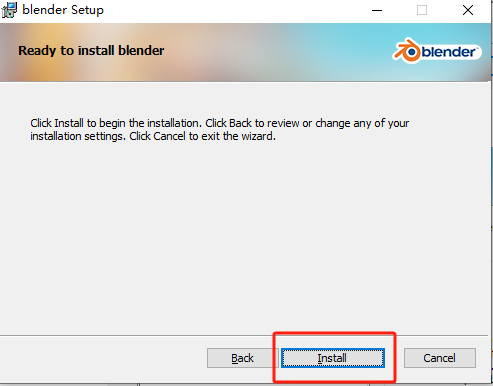
下载完成后，双击下载的文件，一步一步安装

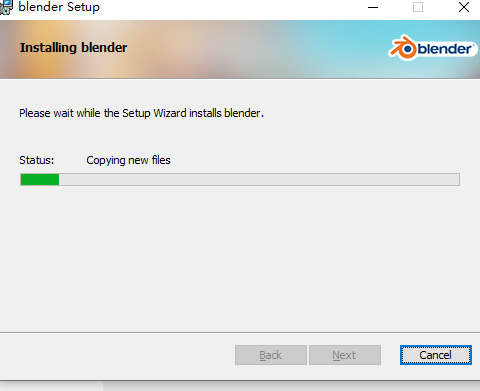




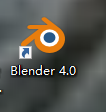








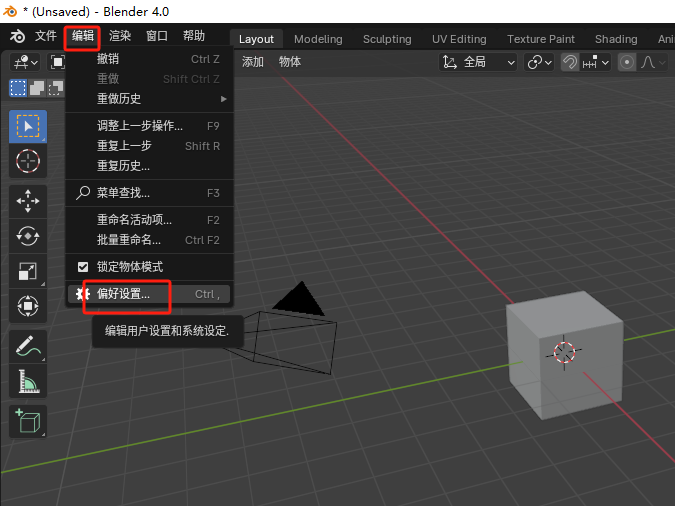
安装成功后，如下图

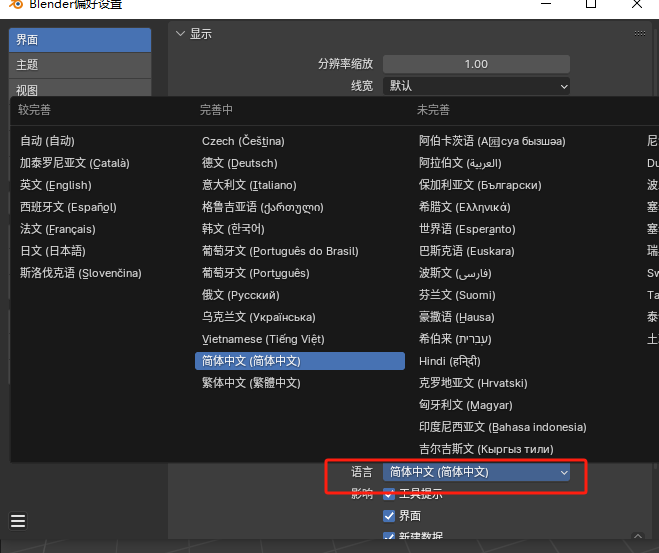


## 使用

### 2.1、 基本设置

设置中文：

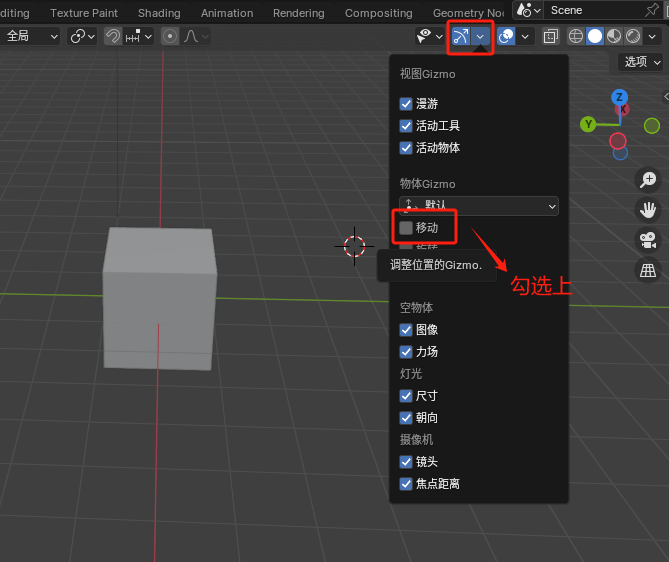




### 2.2、 基本操作

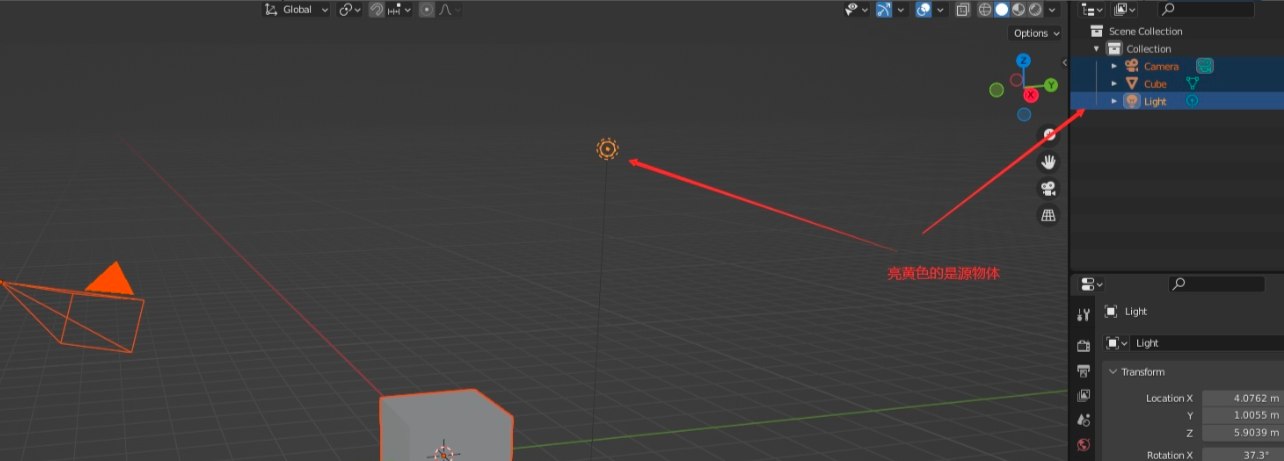
参考https://zhuanlan.zhihu.com/p/476659431?utm\_id=0

#### 2.2.1、 视觉操作

* 旋转视角————中键（按住滚轮）
* 缩放视角————滚轮滚动
* 平移视角————SHIFT + 中键
* 添加物体————SHIFT + A
* 移动物体————如下图
* 切换视角————小键盘5正交/透视；7正Z轴；1正Y轴；3正X轴；9反转视角
* 聚焦视角————~号键，或者小键盘点键
* 切换为相机视角————小键盘0键
  + 保持相机视角————在N窗的View Lock下，勾选Camera to View
  + （使相机快速聚焦某物体————Lock Camera，选则某物体，小键盘点键聚焦）

#### 2.2.2、 选择操作

* 套索选择————CTRL+右键
* 加选、选择源物体、去除源物体————SHIFT + 左键选择
  + 源物体：选中后为亮黄色的物体



* + 切换模式————CTRL+ TAB

#### 2.2.3、 显示操作

* 单独显示某物体————小键盘/（再按一次返回）
* 隐藏物体————H
* 显示所有物体————ALT + H
* 隐藏其他物体————SHIFT + H
* 某物体总是显示在最前面(绑骨时一般用到)——选择一个物体->右侧属性栏->物体属性->视图显示->在前面
* 线框、物体、材质、渲染显示切换——Z

#### 2.2.4、 复制操作

* 快速复制（new新建）————SHIFT + D
* 快速复制（instance不新建）————ALT + D
* instance：共享实例，编辑模式下一个改变，另一个也会改变

#### 2.2.5、 Transform操作

* 移动时开启吸附————G + 按住CTRL（会暂时开启吸附模式）
* 应用变换Apply————CTRL + A
* 同时相对改变许多物体参数————多选物体+ALT+改变参数
* 对齐操作——打开M3插件(Machine3tools)，按ALT+A
* 翻转物体——CTRL+M+X/Y/Z

#### 2.2.6、 其他操作

* 重复上一步————SHIFT + R

### 2.3、 建模操作

#### 2.3.1、 选择

* 加选————SHIFT
* 减选————CTRL
* 圈选————ALT
* 环选————CTRL + ALT
* 切换点线面模式保留选则————CTRL + 123切换
* 阔选、缩选————CTRL + 小键盘加减号
* 连接在一起的所有————CTRL + L
* 选择锐边（一般倒角常用）————上方“选择”-> 选择锐边

#### 2.3.2、 建模操作

* 点在线上滑动 or 线在面上滑动————G+G
* 挤出————E
* 倒角————CTRL + B
  + 倒角边数————CTRL + B +滚轮
* 环切（原理：穿过对边）————CTRL + R
* 内插面————I
* 连线————J
* 填充————F
* 栅格填充————CTRL + F（对偶数个点有效）
* 桥接循环边————CTRL + E
* 拆分————右键

## 导出GLTF

打开软件 ，文件—导出—选择导出gltf

