ThingJS 离线开发网络版

用户手册

（Rev.2.1.9）

北京优锘科技有限公司

2025 年 02 月 20 日

目 录

1 概述

“ThingJS 离线开发网络版”是 ThingJS 平台（https://www.thingjs.com）推出的可在独立局域网环境运行的离线开发 ThingJS 3D 可视化（或数字孪生）项目的开发服务器产品。“ThingJS 离线开发网络版”专为企业内有较多 ThingJS 开发人员需协同完成 ThingJS 项目开发的企业打造。

本文档是说明“ThingJS 离线开发网络版”产品安装、使用、迁移等各方面技术问题的用户手册。

1.1 说明

ThingJS 3D 可视化开发平台提供在线开发、离线开发两种开发方式。其中离线开发又分离线开发 SDK 版（坐席版）和离线开发网络版。

ThingJS 离线开发 SDK 版提供一个 U 盘形式的 Key，插入特定计算机，该机就可以通过离线开发 SDK 软件包进行 ThingJS 离线开发了。离线开发 SDK 版从形式上是单坐席授权，每个 Key 对应一台计算机可进行开发。比较适合企业内仅投入 1 个或少数人员进行 ThingJS 开发的情况。而对于企业有多位 ThingJS 开发人员要投入开发，如果购买多个离线开发 SDK 版授权来支持，这在成本、便捷度两方面都不是特别合适。

ThingJS 离线开发网络版是一个可在局域网环境部署的开发服务器。它通过和开发版本管理工具 Git 结合，理论上可支持不限人数的 ThingJS 开发人员共同进行项目开发。适合于企业内有较多开发人员会参与 ThingJS 开发的情况。离线开发网络版所支持的本机开发 IDE（VSCode）+Git 方式，在开发习惯跟大部分开发人员的日常工作习惯相同，可让开发人员在开发方式上完成无缝切换。

若您希望咨询、了解、购买“ThingJS 离线开发网络版”，可到 ThingJS 网站（https://www.thingjs.com）联系网站客服人员，或致电 ThingJS 400 电话：400-666-9832。

1.2 硬件配置

1.2.1 安装服务器推荐配置

	配置
CPU	2G 主频 4 核
内存	8G
硬盘	80G，推荐 SSD
带宽	推荐 100Mbps
操作系统	Linux x64， Windows x64

（注：(1)不支持在 CPU 为 ARM 的服务器上进行部署；(2)不支持在 Docker 中进行部署）

1.2.2 客户端推荐配置

	配置
CPU	Intel i7 或同档 CPU 以上
内存	16G
硬盘	80G，推荐 SSD
显卡	独显 GTX1070 以上，2G 显存以上

1.3 软件依赖

1.3.1 服务器端

“ThingJS 离线开发网络版”基于 Git 对开发项目进行管理，因此需要在部署安装“ThingJS 离线开发网络版”的环境中安装 Git。

Git 是一个开源的分布式版本控制系统，用于敏捷高效地处理任何或小或大的项目。可在 https://mirrors.edge.kernel.org/pub/software/scm/git/ 中下载安装包。

详细的安装步骤可参考第三章中关于 Git 安装的内容。

1.3.2 客户端

1.3.2.1 CampusBuilder 客户端

CampusBuilder 在园区级 3D 可视化场景的搭建方面，功能强大，不仅可以搭建园区场景，在建筑楼层和室内搭建方面也表现卓越。

最新版本可在 https://store.thingjs.com/tools 中下载。

1.3.2.2 Visual Studio Code 编辑器（推荐）

Visual Studio Code 是一款针对于编写现代 Web 和云应用的跨平台源代码编辑器，推荐使用 Visual Studio Code 进行项目开发。

最新版本可在 https://code.visualstudio.com/中下载。

1.3.2.3 Git

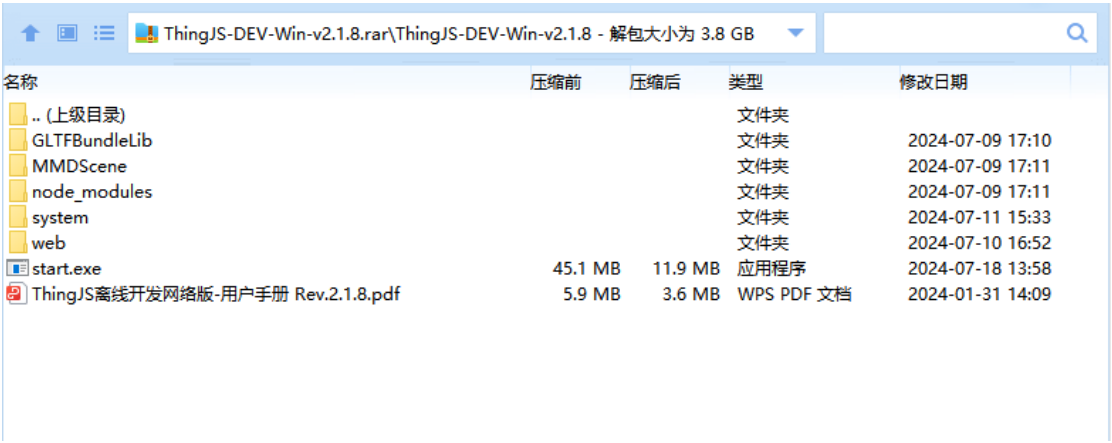
“ThingJS 离线开发网络版”基于 Git 对开发项目进行管理，因此需要在进行开发的客户端环境中安装 Git。

可在 https://git-scm.com/download 中下载安装包。

详细的安装步骤可参考第三章中关于 Git 安装的内容。

2 获取

同 ThingJS 商务人员购买“ThingJS 离线开发网络版”后，告知商务人员将要部署安装“ThingJS 离线开发网络版”的环境（Linux/Windows），将收到商务人员发送的对应环境的“ThingJS 离线开发网络版”压缩安装包。如下图所示：



3 安装

3.1 Windows 服务器

3.1.1 安装 Git

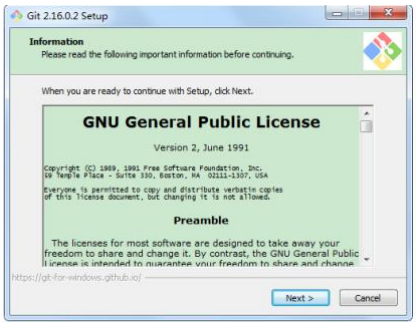
“ThingJS 离线开发网络版”基于 Git 对开发项目进行管理，因此需要在部署安装“ThingJS 离线开发网络版”的环境中安装 Git。

3.1.1.1 下载安装包

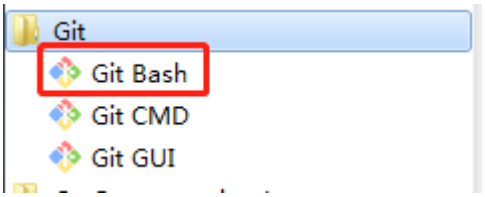
访问网址 https://gitforwindows.org/ 下载安装包（官网速度慢，可使用国内的镜像地址进行下载 https://npm.taobao.org/mirrors/git-for-windows/，推荐使用稳定版本 v2.16.0.windows.2）。

3.1.1.2 安装

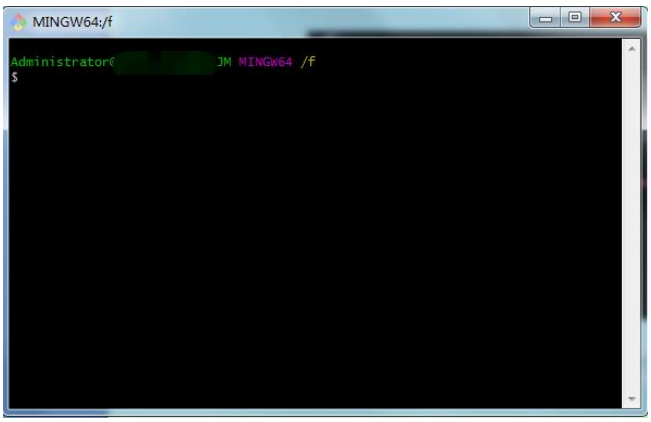
运行安装文件，根据提示内容完成安装。



安装完成后，运行“Git Bash”。

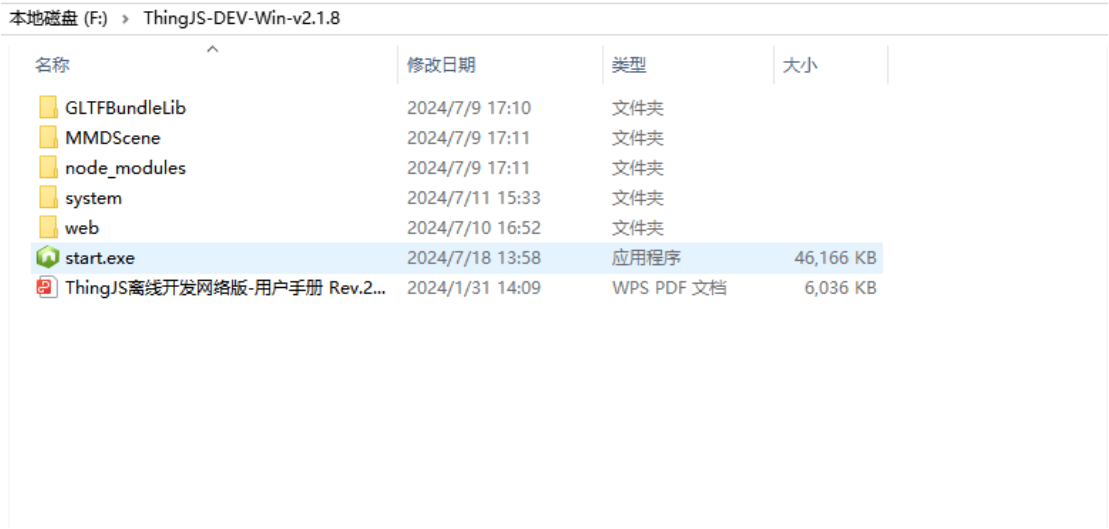


出现下图所示界面，即说明 Git 安装成功。



3.1.2 解压安装包

将获取到的 Windows 版本的“ThingJS 离线开发网络版”压缩安装包解压至需要部署的目录下。解压后目录结构如下图所示：



3.1.3 启动服务

打开“ThingJS 离线开发网络版”服务的部署目录，双击运行 start.exe 即可启动，服务默认的端口号为 9100，如需修改请参考第 5.1 节的内容。服务使用期间，请勿关闭服务控制台窗口。

3.2 Linux 服务器

3.2.1 安装 Git

3.2.1.1 安装依赖包

安装 Linux 系统对应的依赖包，命令如下：

Centos/RedHat：

```
$ yum install curl-devel expat-devel gettext-devel \
  openssl-devel zlib-devel
```

Debian/Ubuntu：

```
$ apt-get install libcurl4-gnutls-dev libexpat1-dev gettext \
  libz-dev libssl-dev
```

3.2.1.2 下载源码包

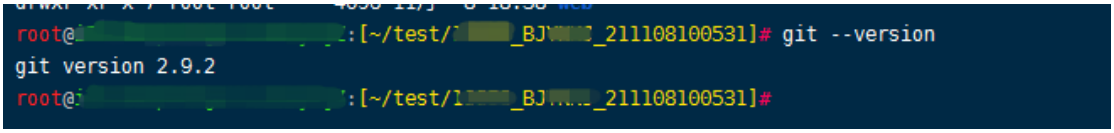
访问网址 https://git-scm.com/download 下载源码包。

3.2.1.3 安装

解压并安装下载的源码包，命令如下：

```
$ tar -zxf git-x.x.x.tar.gz
$ cd git-x.x.x
$ make prefix=/usr/local all
$ sudo make prefix=/usr/local install
```

安装完成，查看 Git 版本号，如下图所示，即说明 Git 安装成功。



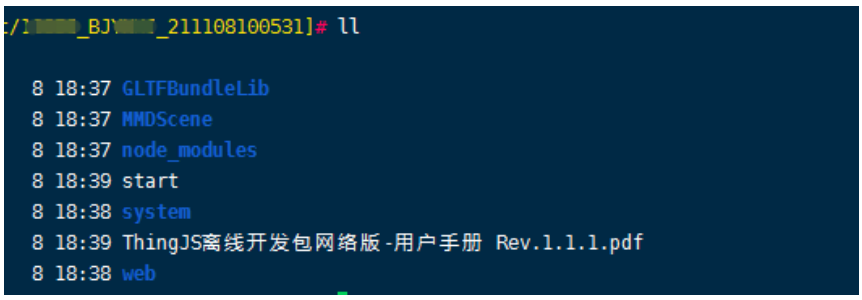
3.2.1.4 配置

成功安装 Git 后，依次执行以下命令，存储凭证。

```
$ git config --global credential.helper store
$ git config --global user.name "username"
$ git config --global user.email "email"
```

3.2.2 解压安装包

将获取到的 Linux 版本的“ThingJS 离线开发网络版”压缩安装包解压至需要部署的目录下。解压后目录结构如下图所示：



3.2.3 启动服务

通过命令 chmod u+x start 为 start 文件增加执行权限。

执行 start 启动服务，服务默认的端口号为 9100，如需修改请参考第 5.1 节的内容。

为保证进程能够保持在后台运行，可在 screen（多重视窗管理程序）中执行，或使用其他后台执行的方法。

可通过命令 netstat –napl | less 或 ps axu | grep start | less 查看端口情况。

```
21858  2.2  1.6 945572 34544 pts/0    Sl   17:58   0:00 ./start
```

若要关闭服务，可通过 kill -9 PID 命令结束相应进程。

```
h22101280]# kill -9 21858
```

3.3 安装更新包文件

3.3.1 Windows 服务器

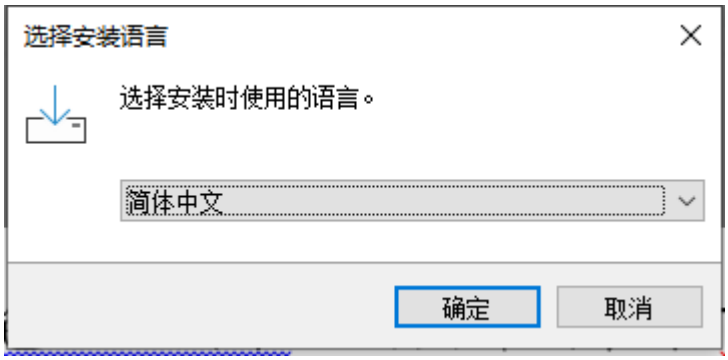
3.3.1.1 下载更新包

打开低代码在线开发-平台产品页面（https://www.thingjs.com/guide/offline/），选择 Windows 版更新包文件，如下图所示：

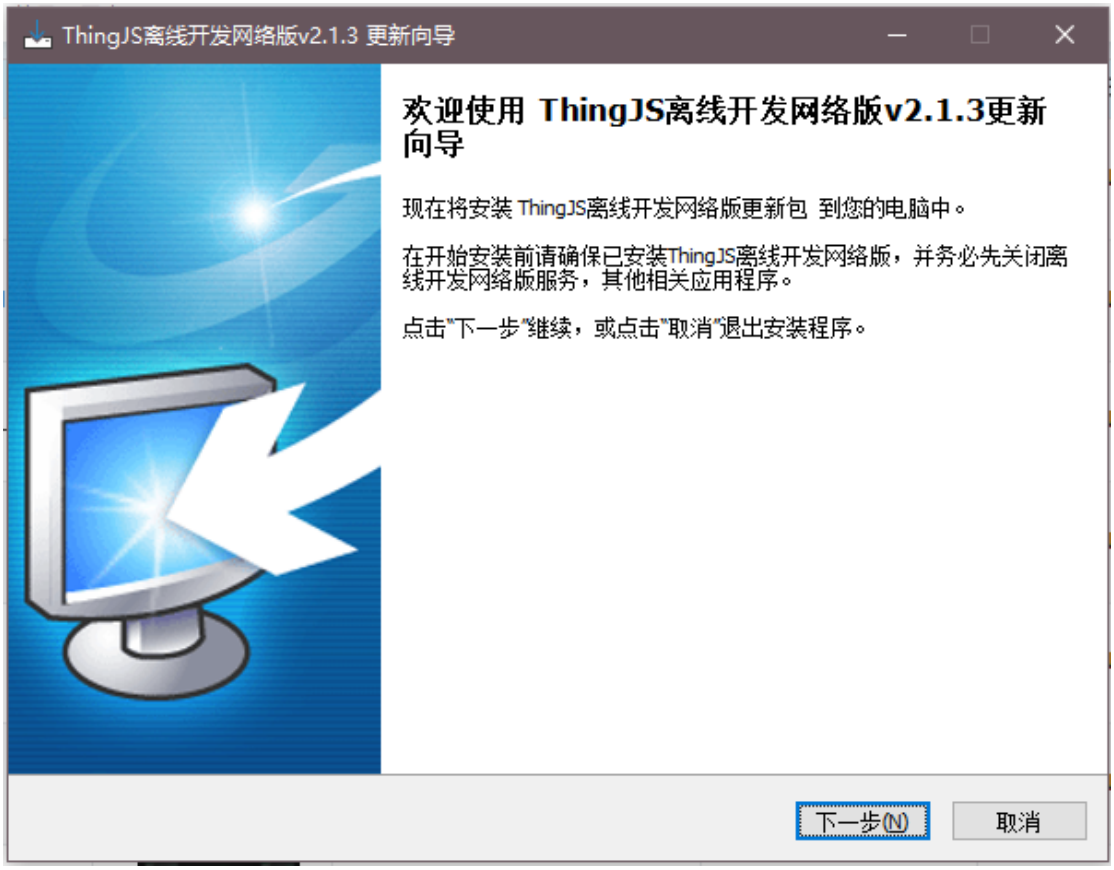
注：该更新包仅适用于已购买并安装了 ThingJS 离线开发网络版软件，且安装软件包版本在 2.0.1 以上（含 2.0.1）的用户。如您的版本低于 2.0.1，请联系商务人员获取最新版全量包。

3.3.1.2 安装更新包

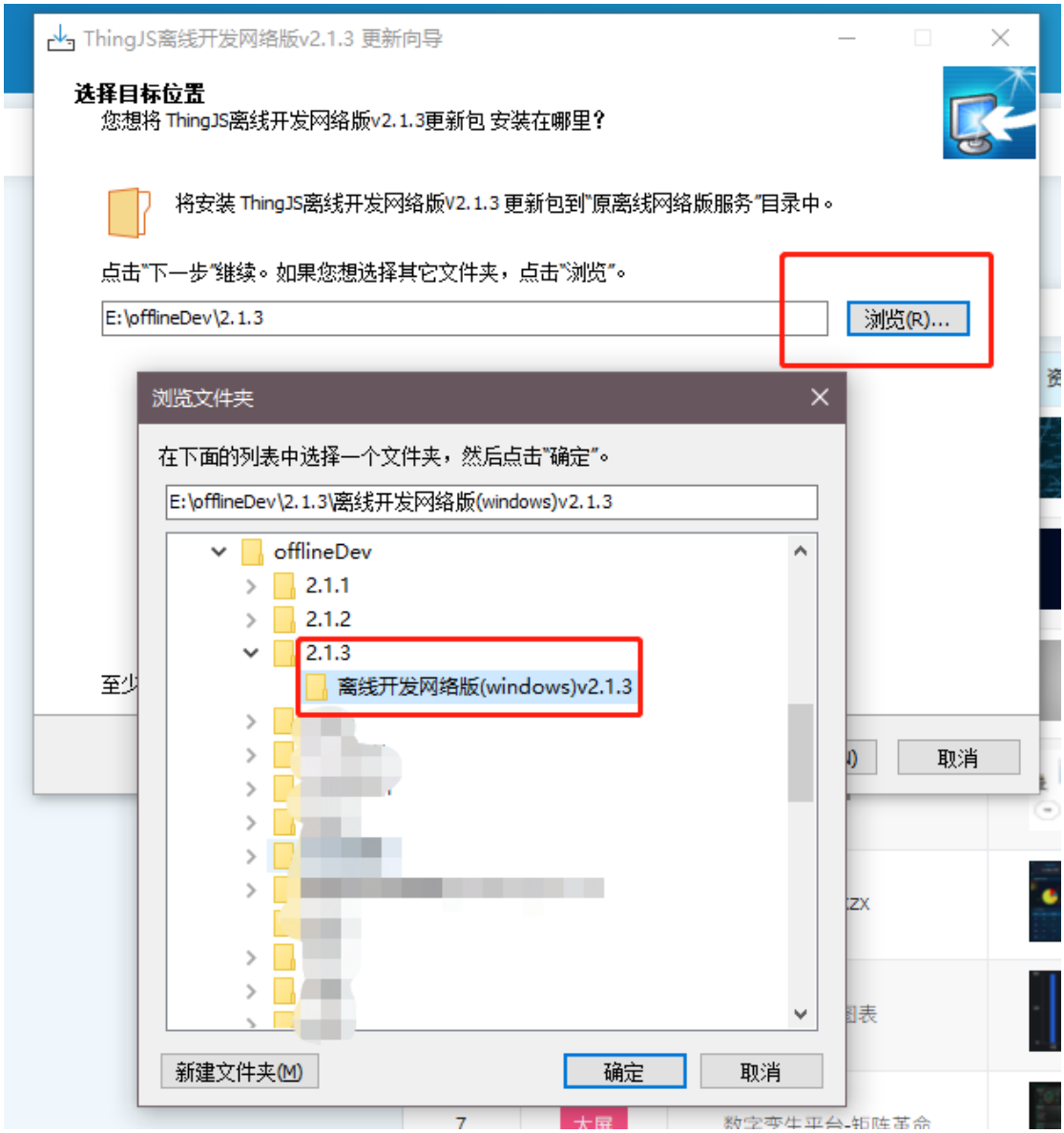
双击 3.3.1.1 中获得的 mysetup.exe 更新包，进入更新页面，默认安装语言为中文，如下图所示：



点击“确定”后，进入安装界面，按照界面操作提示进入下一步，如下图所示：



进入下一步后，提示选择更新包安装位置，需要手动点击“浏览”，选择原离线开发网络版软件包安装的根目录，如下图所示：



选择完安装目录后，点击“确定”进入安装步骤，直至提示更新完成，如下图所示。

3.3.2 Linux 服务器

3.3.2.1 下载更新包

打开低代码在线开发-平台产品页面（https://www.thingjs.com/guide/offline/），选择 Linux 版更新包文件，如下图所示：

注：该更新包仅适用于已购买并安装了 ThingJS 离线开发网络版软件，且安装软件包版本在 2.0.1 以上的用户。

3.3.2.2 安装更新包

将 3.3.2.1 中获得的 mysetup.sh 更新包放入部署的 linux 服务器中，执行安装命令，如示例中“./mysetup.sh --target /root/test/pkg/installjammer/makeself-2.1.5/离线开发网络版 v2.0.1/”，在 mysetup.sh 文件存放的目录执行时，通过“--target”找到原离线开发网络版部署的根目录，执行命令即可开始安装更新包，如下图所示：

```
Creating directory /root/test/pkg/installjammer/makeself-2.1.5/离线开发网络版v2.0.1
============================================================================================
                        离线开发网络版 v2.1.3 更新包 For Linux
============================================================================================
ThingJS 离线开发网络版 v2.1.3 更新包 Linux版&Win版同步上线。
ThingJS Offline Development v2.1.3 update pakge will be available for Linux and Win simultaneously。

更多信息请访问  https://cdn.thingjs.com/doc/ThingJS离线开发网络版-用户手册 Rev.2.1.3.pdf
============================================================================================

正在验证离线开发网络版 v2.1.3 更新包的完整性...
Verifying archive integrity of the ThingJS Offline Development update pakge...

该离线开发网络版 v2.1.3 更新包是完整的。
The ThingJS Offline Development update pakge is fine.

> 更新前请确定 当前的离线开发网络版本在 v2.0.1 及以上？ [Y/N]
> To confirm that your offline development version is v2.0.1 or higher ? [Y/n]
您的选择是:y
> 更新前请确定 离线开发网络版本服务 已经关闭 ！！！ [Y/N]
> Before updating, ensure that the offline development service has been stopped! ! ! [Y/n]
您的选择是:y

安装中请耐心等待..... Please Waitting.....

============================================================================================
离线开发网络版更新完成！！！
重新启动离线开发网络版服务即可。
```

注：安装过程中会询问两个问题，第一个是确保原离线开发网络版版本在 v2.0.1 及以上，若是则输入“y”即可进入下一步，第二个是确保原离线开发网络版服务已经关闭，若是则输入“y”即可进入安装过程，直至更新安装完成。

4 用户

本章将介绍网络版用户管理功能，网络版中管理员分为系统管理员（admin）和应用管理员。仅系统管理员（admin）具有添加应用管理员权限。

注：该章节内容在离线开发网络版 v2.0.1 版本及以上中存在。

4.1 系统管理员权限

系统管理员可查看“离线开发网络版”管理页面的激活授权、项目列表、场景列表、用户列表和日志列表。

注：该文档中的 5、6、7、8、9、10 章节均以系统管理员（admin）身份操作。

4.1.1 添加应管理员

admin（系统管理员）在“用户列表”中点击“添加管理员”，如下图所示：

输入自定义应用管理员名称，示例添加名称为“thingjs”的应用管理员，如下图所示：

系统管理员可通过开关按钮来控制应用管理员账号状态，操作该应用管理员账号为停用，或者激活，停用状态下无法登录管理页面。

4.2 应用管理员权限

应用管理员可查看“离线开发网络版”管理页面的激活授权、项目列表和场景列表。

4.2.1 修改密码

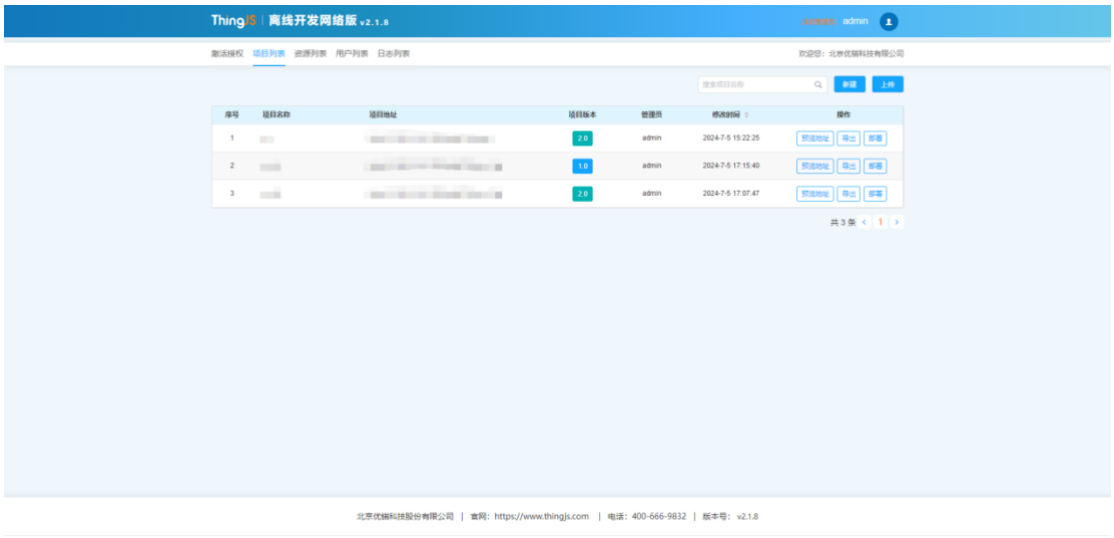
应用管理员首次登录默认密码为 admin，可在“离线开发网络版”管理界面修改自定义密码，详细操作请见 5.2 章节修改密码。

4.2.2 授权状态

应用管理员无权限查看和下载机器码，无权限对“离线开发网络版”部署进行迁移，无权限查看和下载授权码，如下图所示：

4.2.3 上传项目和场景

应用管理员可上传场景和添加项目，详细步骤请见“7.1.3”章节和“7.2”章节，其他应用管理员无权限查看到该应用管理员账号下项目列表和资源列表。以上传项目为例，如下图所示：

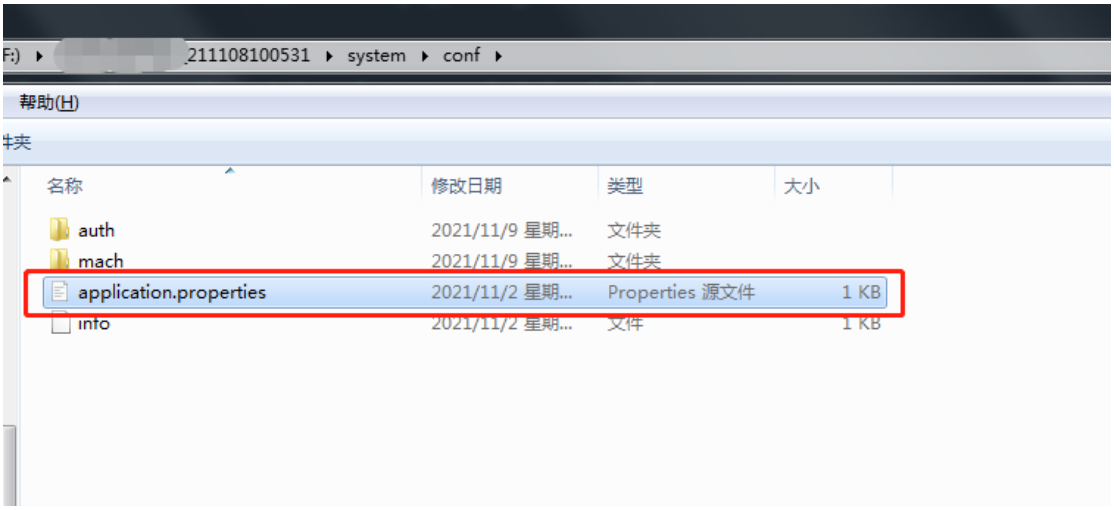


5 配置

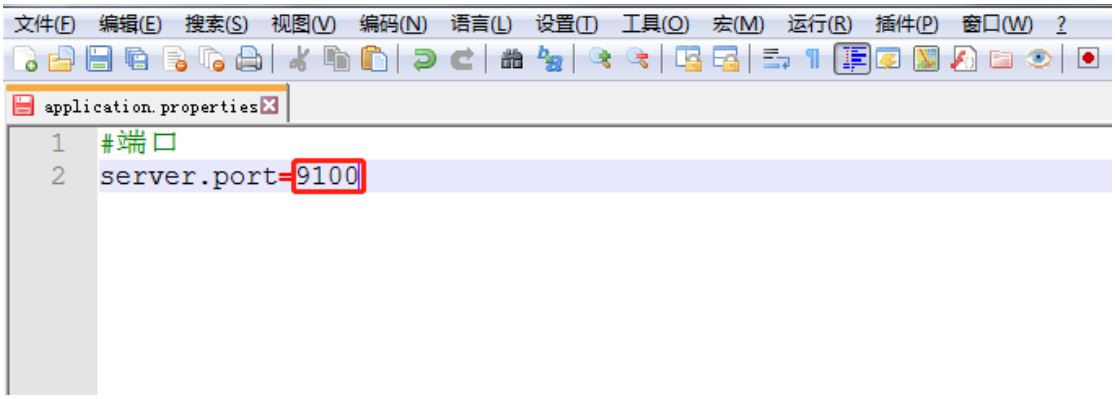
本章介绍了几种常见的“ThingJS 离线开发网络版”支持的配置方法，请勿自行更改未说明的其他文件内容，以免发生不可预期的问题。

5.1 修改服务启动端口

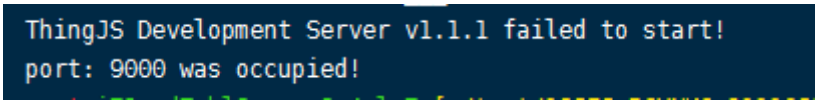
“ThingJS 离线开发网络版”服务默认端口号为 9100，可以通过配置服务目录中 system\conf 目录下的 application.properties 文件来修改端口号。



用文本编辑器打开 application.properties，修改 server.port 为新端口号。



注意不要和原来的服务端口冲突，若端口冲突会无法启动服务，如下图所示：

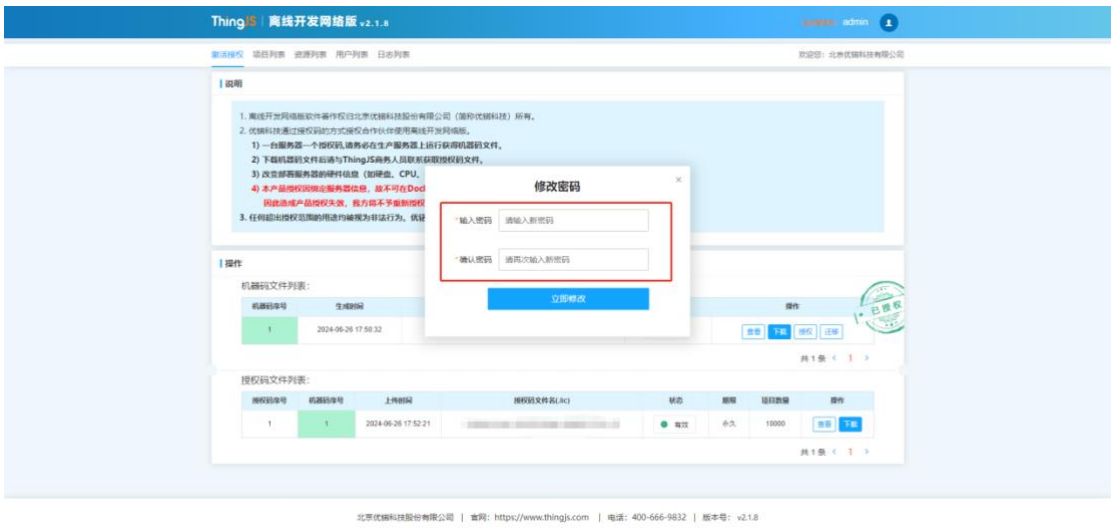


5.2 修改登录密码

为确保“ThingJS 离线开发网络版”中项目的安全，请在首次使用默认密码登录后尽快修改登录密码。

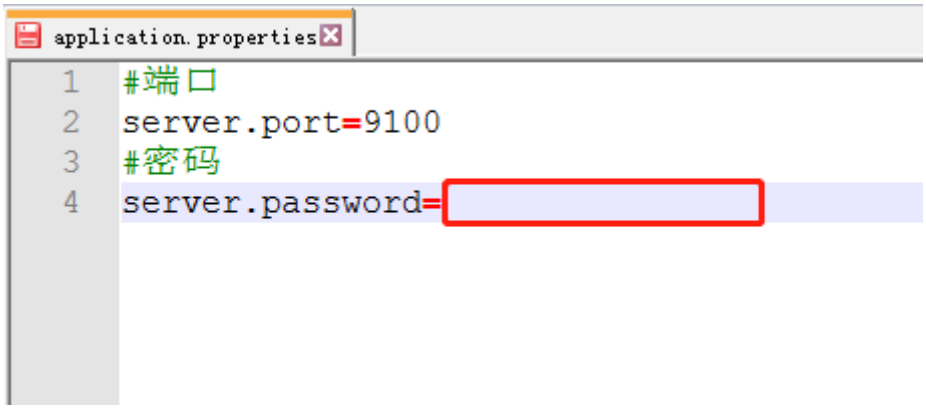
成功登录“离线开发网络版”管理界面后，如图所示，点击“修改密码”，

在弹出的对话框中输入需要修改的新密码，即可完成登录密码的修改。



密码将以“密文”的形式保存在 system\conf 目录下的 application.properties 文件中，如忘记密码，可自行“删除”配置文件中的密码设置项，即可自动恢复默认密码（admin），如下图所示。

（注：请勿擅自填写密码设置项的内容。）



6 授权

6.1 访问“离线开发网络版”管理界面

根据服务启动信息，访问“离线开发网络版”管理界面（http://ip:port/admin）。

访问“离线开发网络版”管理界面，需要进行登录操作，用户名和密码初始默认都为“admin”（为确保项目的安全，请在首次登录后尽快修改密码，参考第5.2节内容），登录界面如下图所示：

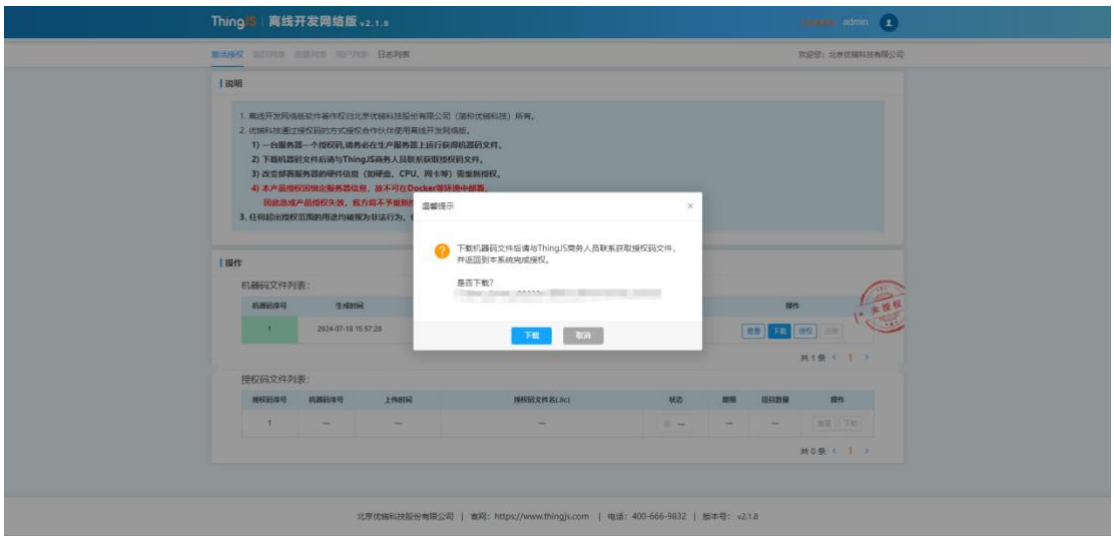
登录成功后即可进入到“离线开发网络版”管理界面，如下图所示：

6.2 获取授权

6.2.1 下载机器码文件

首先在机器码文件列表中，下载当前有效的机器码文件：

将该机器码文件下载至本地：



6.2.2 申请授权码文件

将 6.2.1 节获取的机器码文件发送给 ThingJS 商务人员，获取对应的授权码文件。

6.2.3 完成授权

返回到“离线开发网络版”管理界面，在当前有效的机器码文件一栏中选择“授权”按钮，上传所获取的对应的授权码文件，即可完成授权。

7 开发

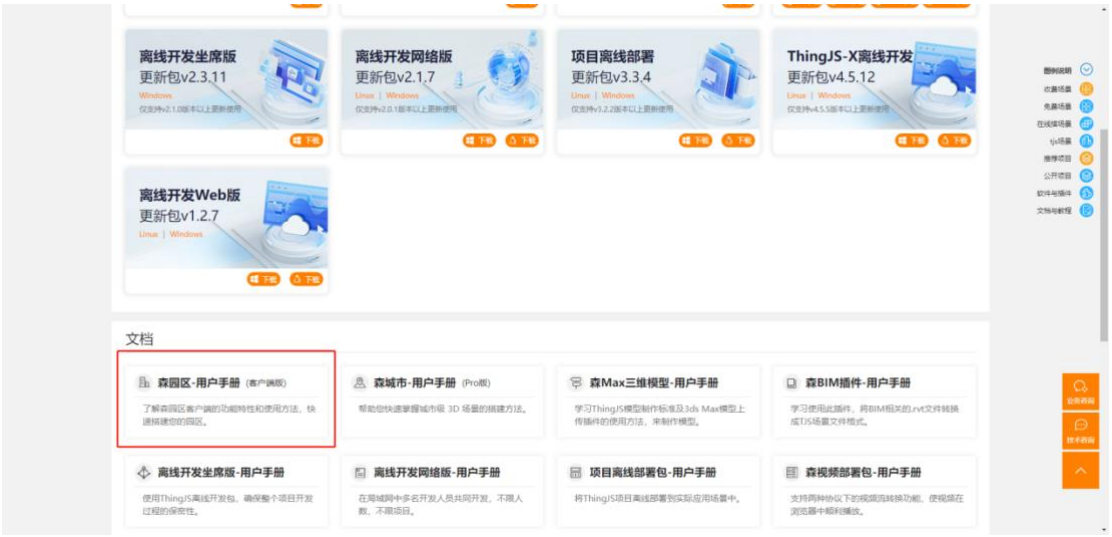
本章将详细的介绍如何通过“ThingJS 离线开发网络版”进行项目开发，包括搭建场景、开发应用，到最后发布项目。

7.1 搭建场景

7.1.1 安装“CampusBuilder”

CampusBuilder 在园区级 3D 可视化场景的搭建方面，功能强大，不仅可以搭建园区场景，在建筑楼层和室内搭建方面也表现卓越。

可在网站 https://store.thingjs.com/tools 中下载最新版本，并查看“CampusBuilder 用户手册”进行后续的安装使用。



7.1.2 搭建场景

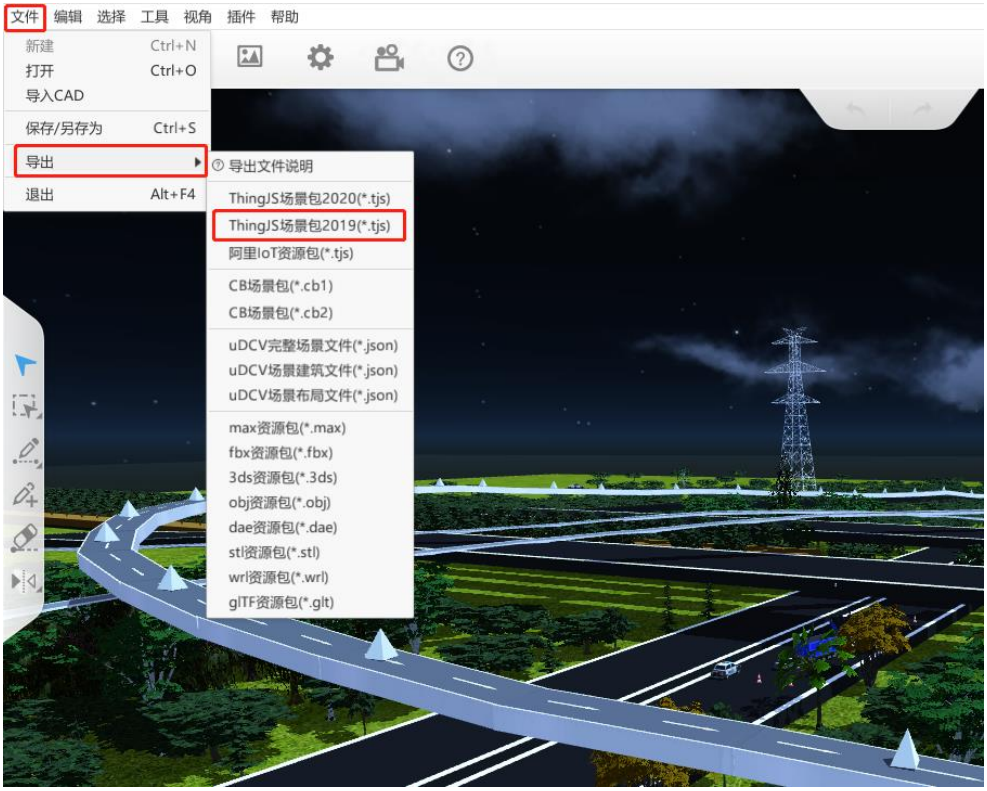
可在网站 https://store.thingjs.com/tools 中查看“CampusBuilder 用户手册”学习搭建场景。

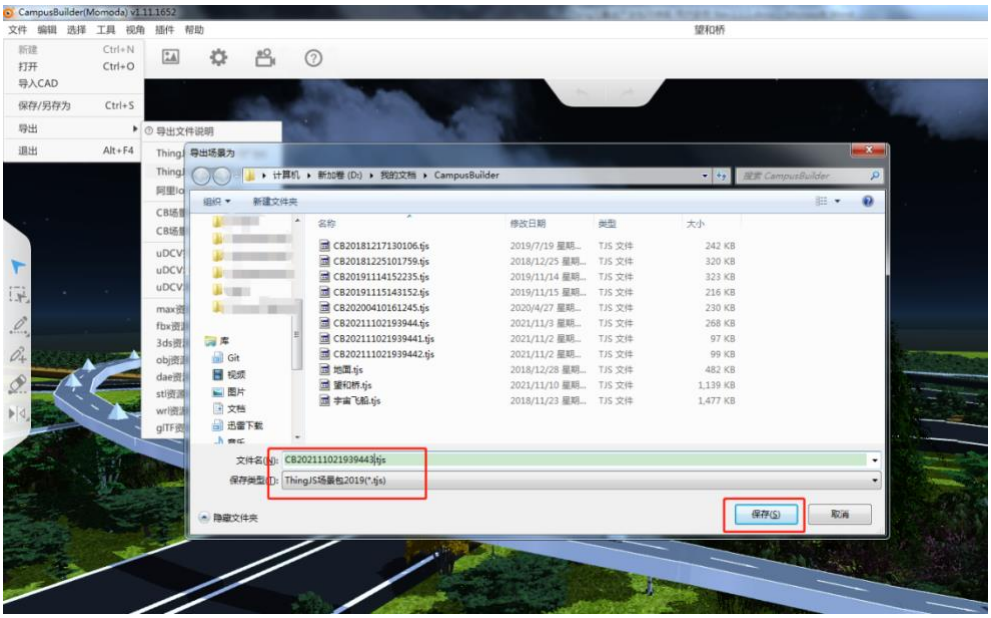
7.1.3 上传场景

为后续使用场景进行应用开发，需将“CampusBuilder”中搭建好的场景上传到“ThingJS 离线开发网络版”中。

7.1.3.1 客户端导出场景

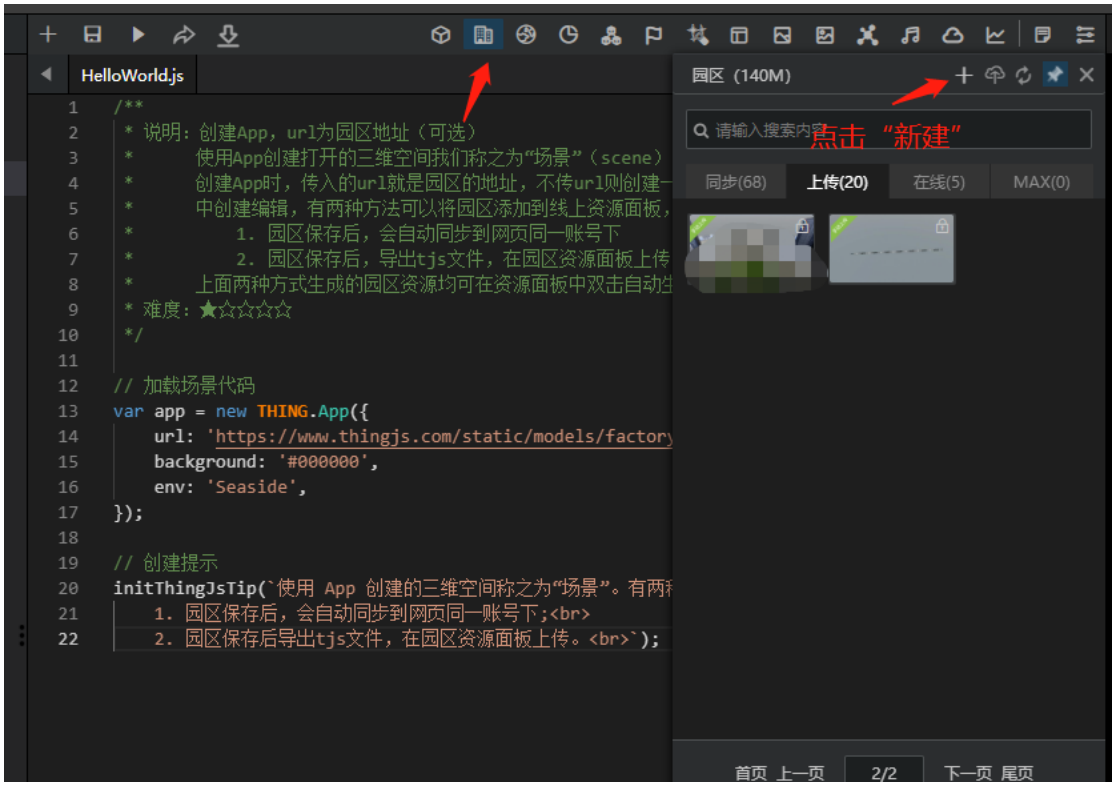
在“CampusBuilder”中搭建好场景后，点击“文件-导出-ThingJS 场景包 2019(*.tjs)”，如下图所示。

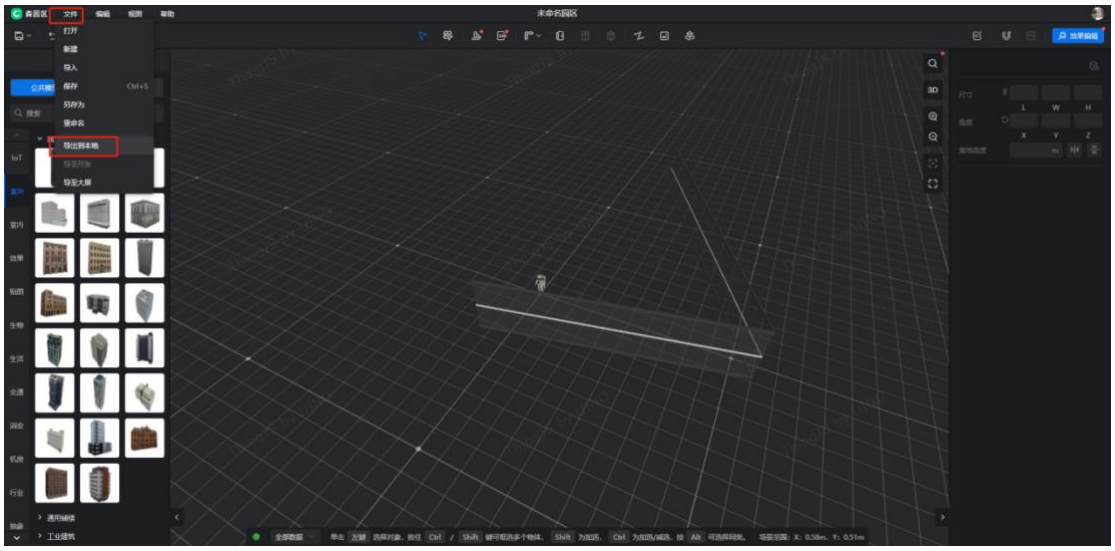


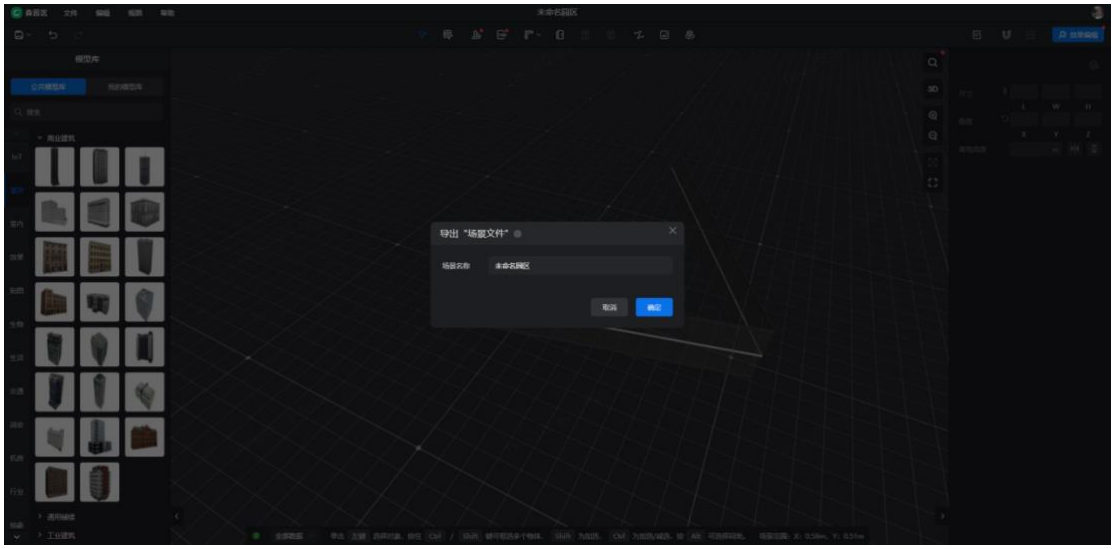


7.1.3.2 森园区导出场景

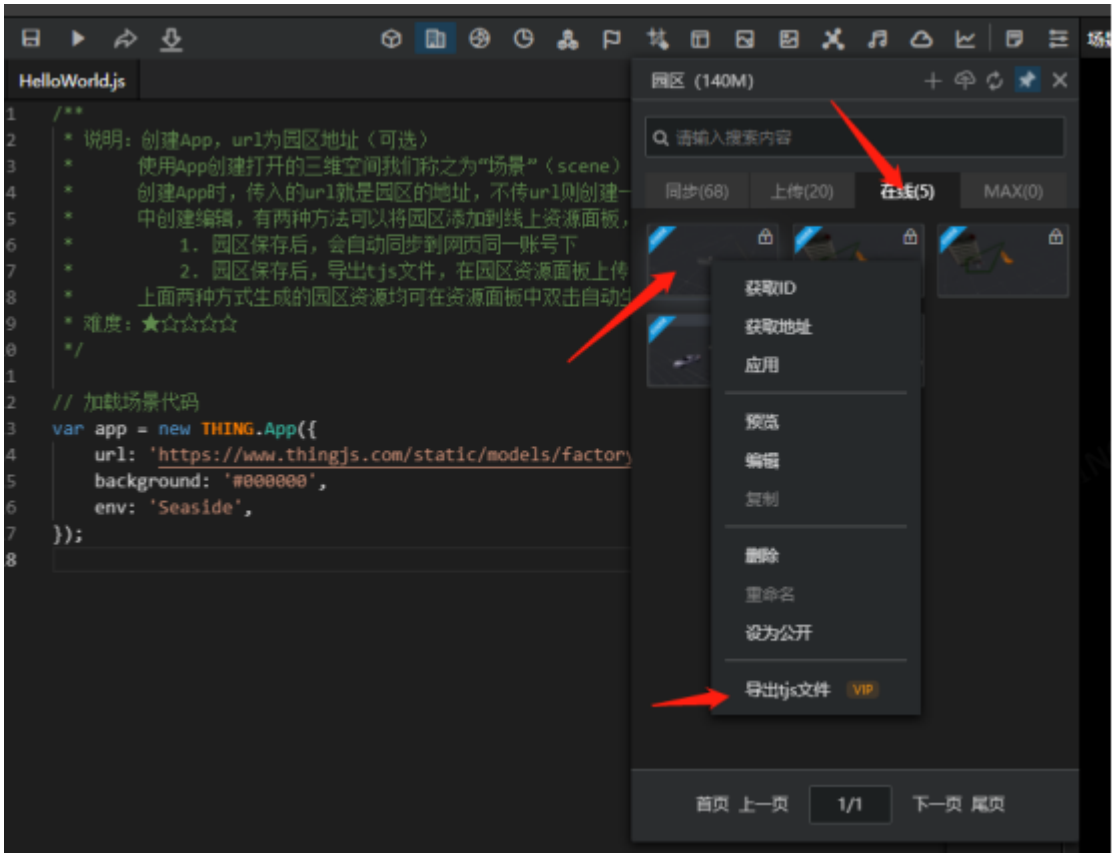
在“在线开发-园区-新建”进入森园区创建场景，在森园区中搭建好场景后，点击“文件-导出到本地-确定”，下载到本地。如下图所示：





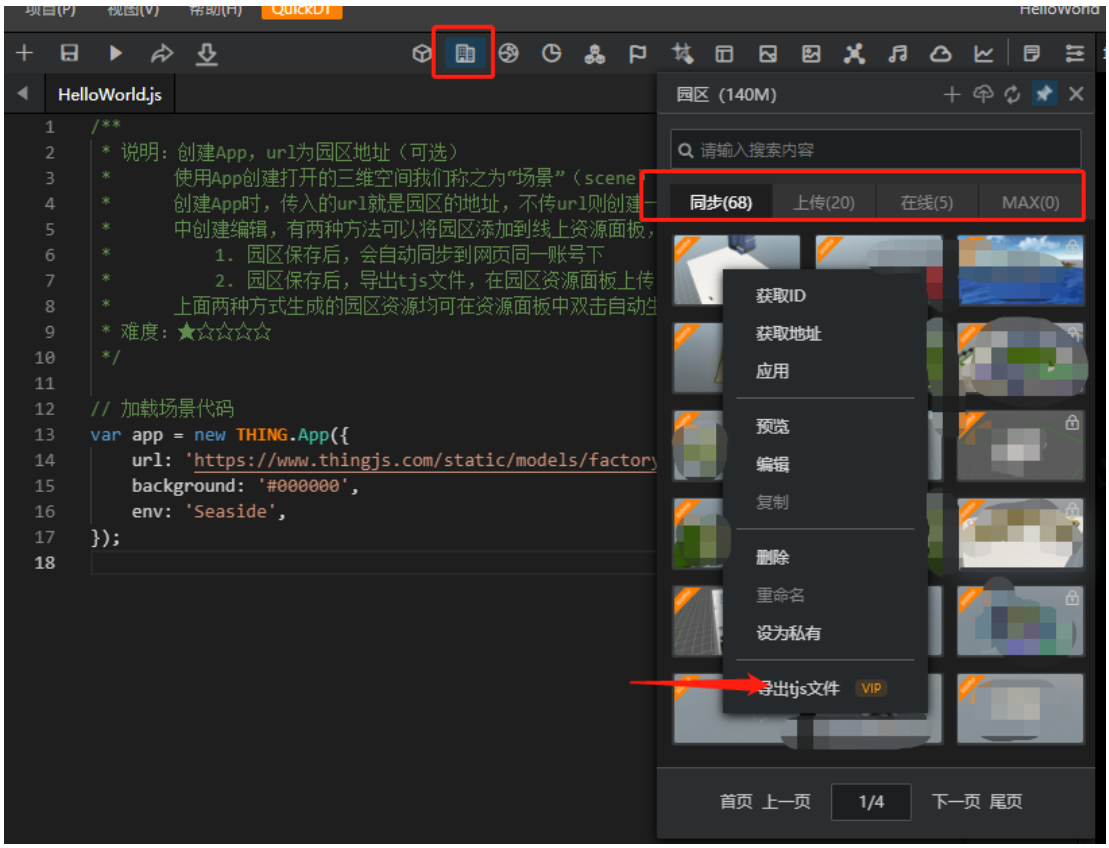


也可以在森园区内直接保存，到在线开发”园区-在线”里找到刚保存的场景，右键点击“导出 tjs 文件”下载至本地。如下图所示：



7.1.3.3 在线开发可导出场景

在线开发“园区”里的场景，右键点击“导出 tjs 文件”下载至本地。如下图所示：



7.1.3.4 场景上传

网络版中上传场景，可参考 8.2 章节，园区资源在离线开发网络版中的使用。

上传成功后，可对该场景进行“复制地址”和“场景预览”操作。

7.2 应用开发

7.2.1 安装 Git

“ThingJS 离线开发网络版”基于 Git 对开发项目进行管理，因此需要在进行开发的客户端环境中安装 Git。

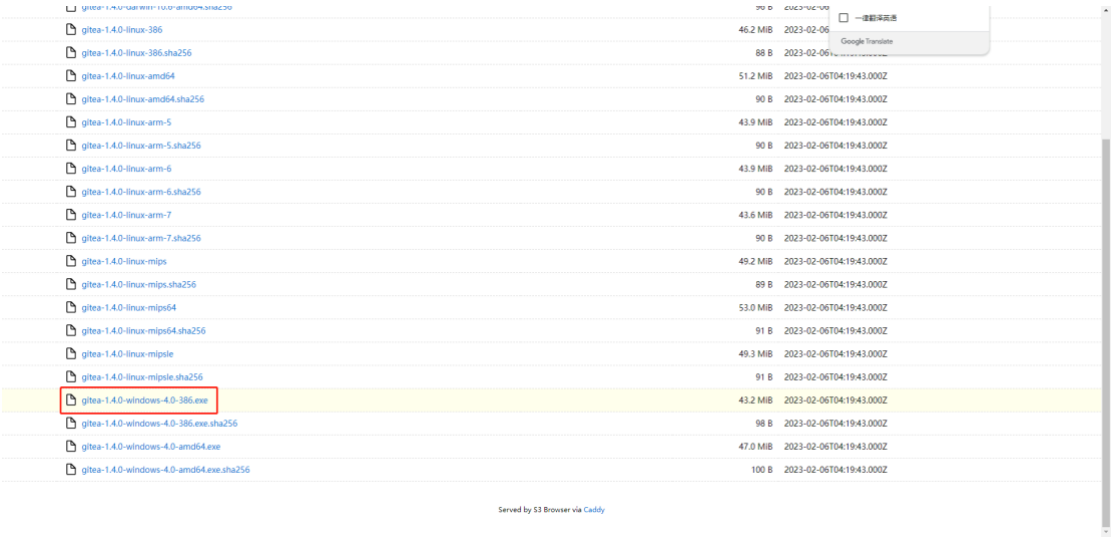
详细的安装步骤可参考第三章中关于 Git 安装的内容。

7.2.2 搭建 Git 服务器

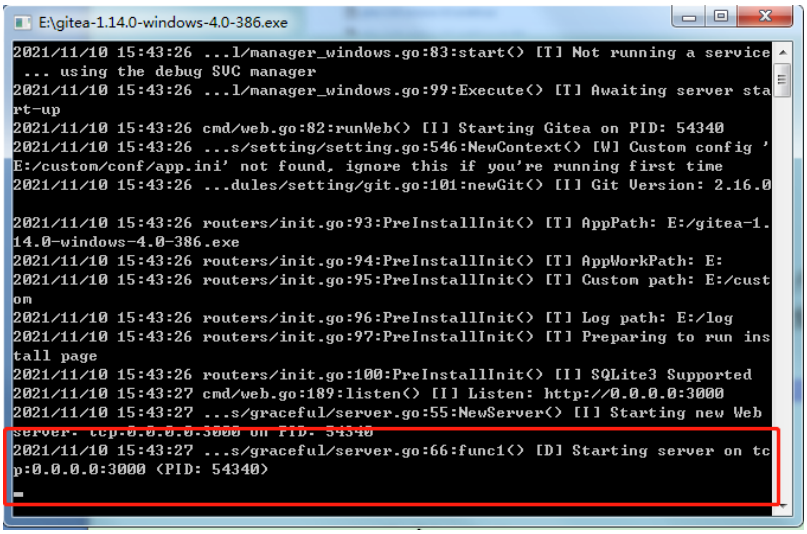
互联网上有许多免费托管开源代码的远程仓库，例如 GitHub，如果项目需要保密，也可以自行搭建 Git 服务器用于代码托管。

Git 服务器的程序有很多，比如 GoGs、Gitea、Gitblit、Gitlab 等等，都是较为成熟的 Git 服务器程序，这里仅对 Gitea 的安装进行简单的介绍，如需要搭建其它 Git 服务器，可自行查阅相关资料进行安装搭建。

访问 https://dl.gitea.io/gitea/1.4.0 下载对应系统的安装包，这里示例安装 Windows 版本，如下图所示，



下载完成，双击运行安装程序。



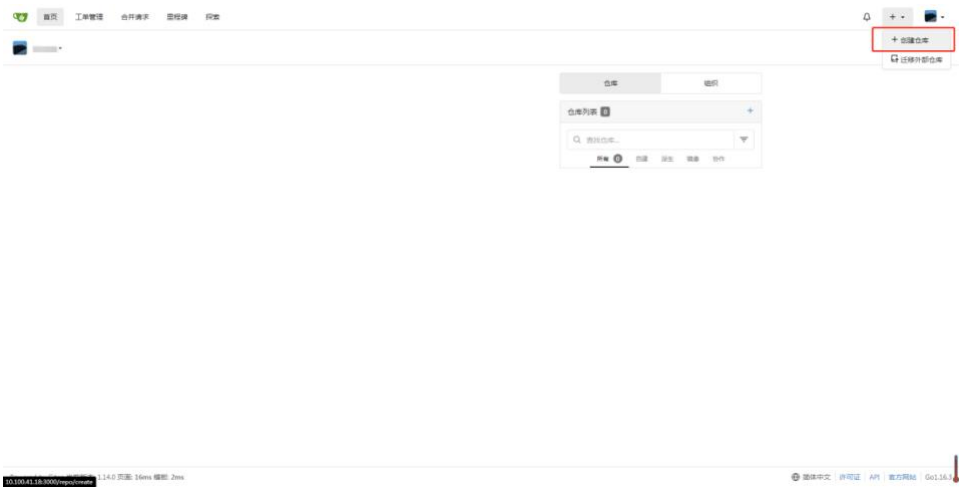
根据提示，访问服务（默认端口号为 3000），如下图所示，

根据页面内容填写初始配置，填写完成后，点击“立即安装”即可。

更多关于 Gitea 的使用说明，请查阅文档 https://docs.gitea.io/zh-cn/ 。

7.2.3 创建仓库

以 Gitea 为例，如下图所示，点击“创建仓库”，



按需填写仓库配置内容，

成功创建仓库后，即可得到一个仓库地址，如下图所示：

7.2.4 安装 Visual Studio Code

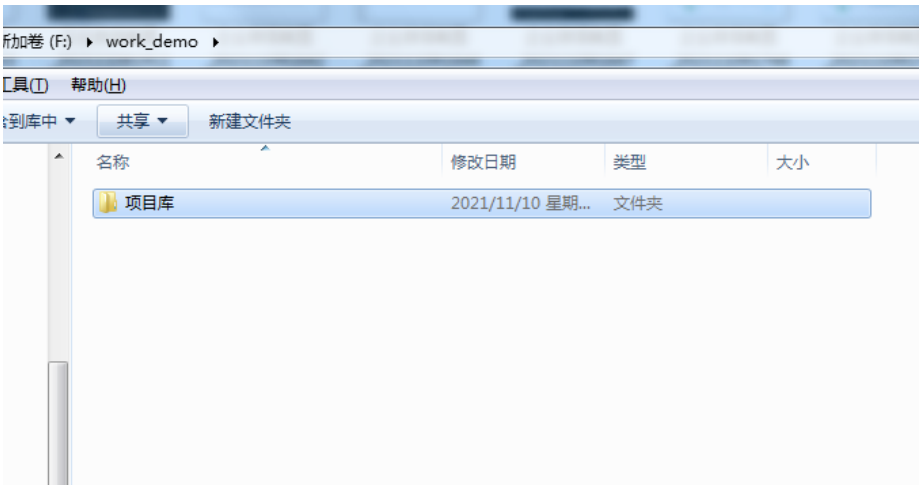
Visual Studio Code 是一款针对于编写现代 Web 和云应用的跨平台源代码编辑器，推荐使用 Visual Studio Code 进行项目开发。

最新版本可在 https://code.visualstudio.com/ 中下载。

7.2.5 本地开发

7.2.5.1 克隆仓库

在本地开发环境中的工作目录下，创建一个名为“项目库”的文件夹，如下图所示：



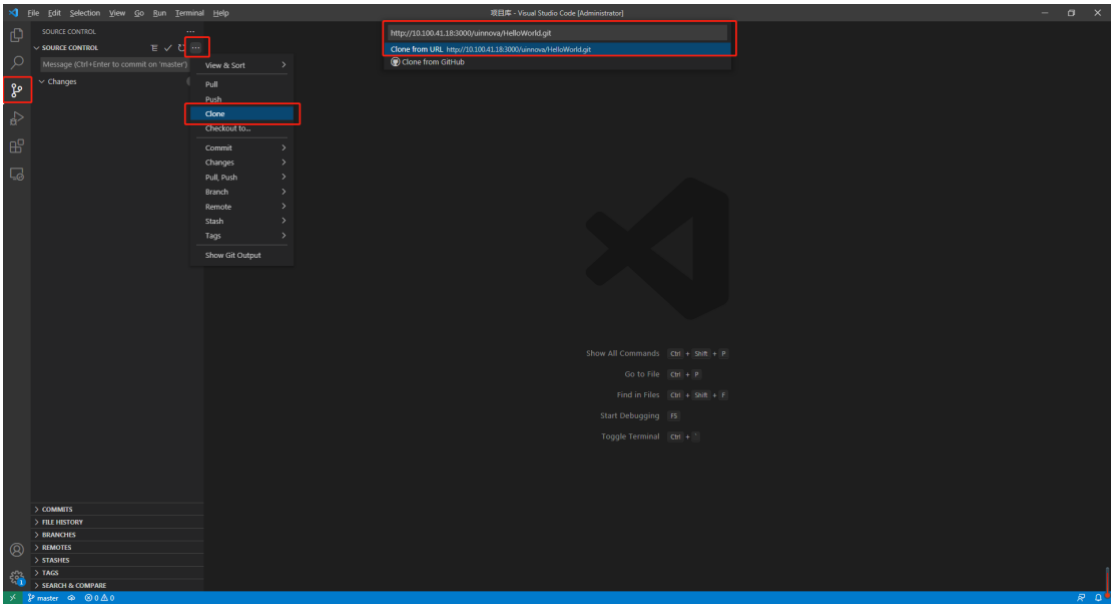
7.2.5.2 应用开发

进行应用开发时，可选择“新建项目”或者可通过“ThingJS 离线开发网络版”提供的“示例开发包”快速创建项目，也可以从“ThingJS 在线开发平台”获取一个已有项目的“离线开发包”进行“上传项目”开发。

（1） 新建项目

访问“ThingJS 离线开发网络版”，切换至“项目列表”页签，点击“新建”按钮，在“新建项目引导”中输入 7.2.3 节中得到的仓库地址，点击“创建”，创建运行项目基础文件，如下图所示：

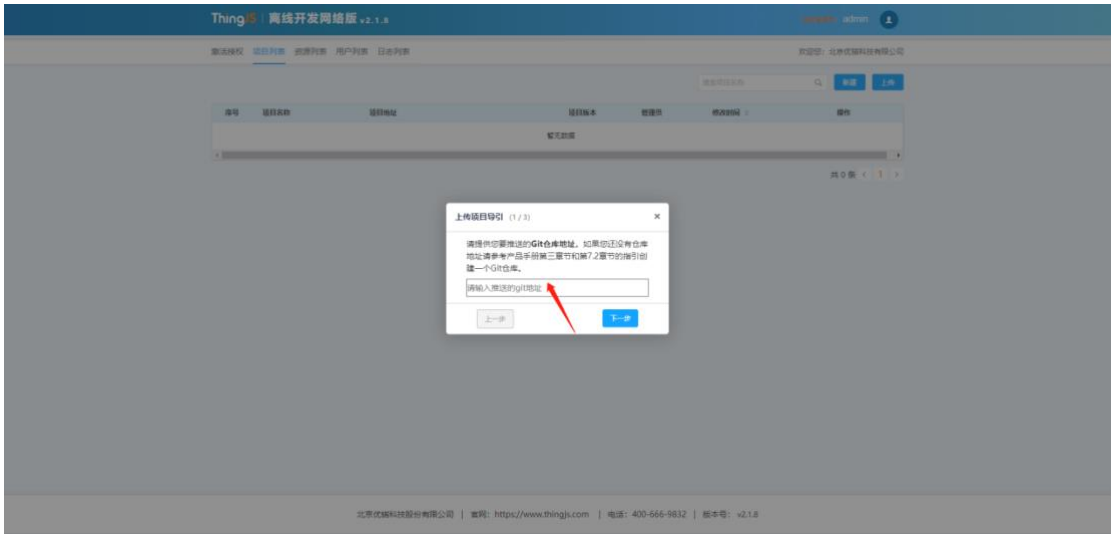
项目创建完成后，使用 Visual Studio Code 打开项目库文件夹，将 7.2.3 节中得到的仓库地址，克隆（clone）到项目库目录下（如有其它 Git 版本控制系统，例如 TortoiseGit，也可直接进行仓库克隆操作，本处以 Visual Studio Code 进行示例），如下图所示：

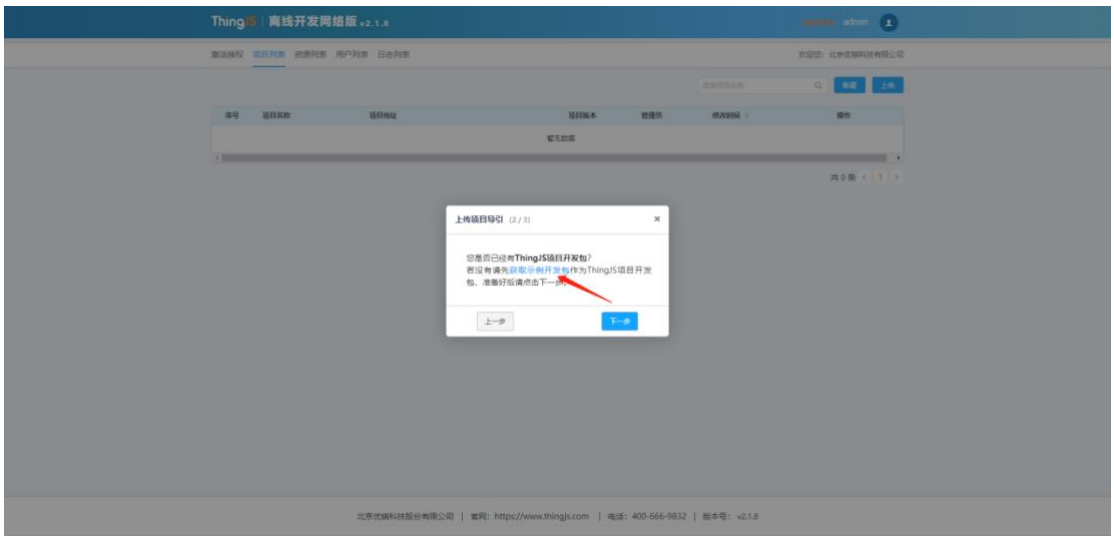


可在此基础文件上继续进行开发。

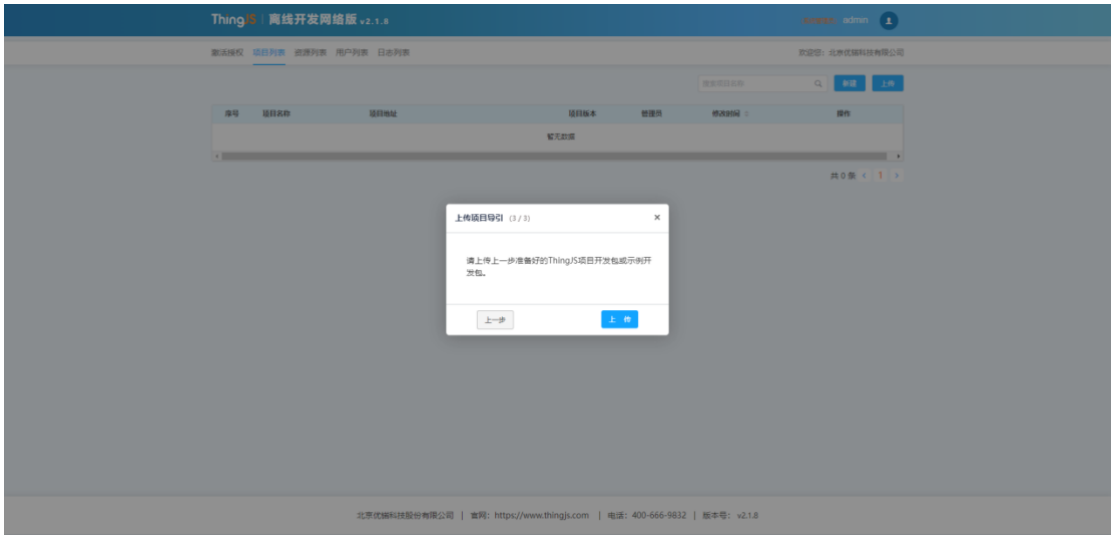
（2） 上传项目-获取“示例开发包”

访问“ThingJS 离线开发网络版”，切换至“项目列表”页签，点击“上传”按钮，在“上传项目引导”中先输入 7.2.3 章节的 Git 仓库地址，再下一步点击“获取示例开发包”，如下图所示：



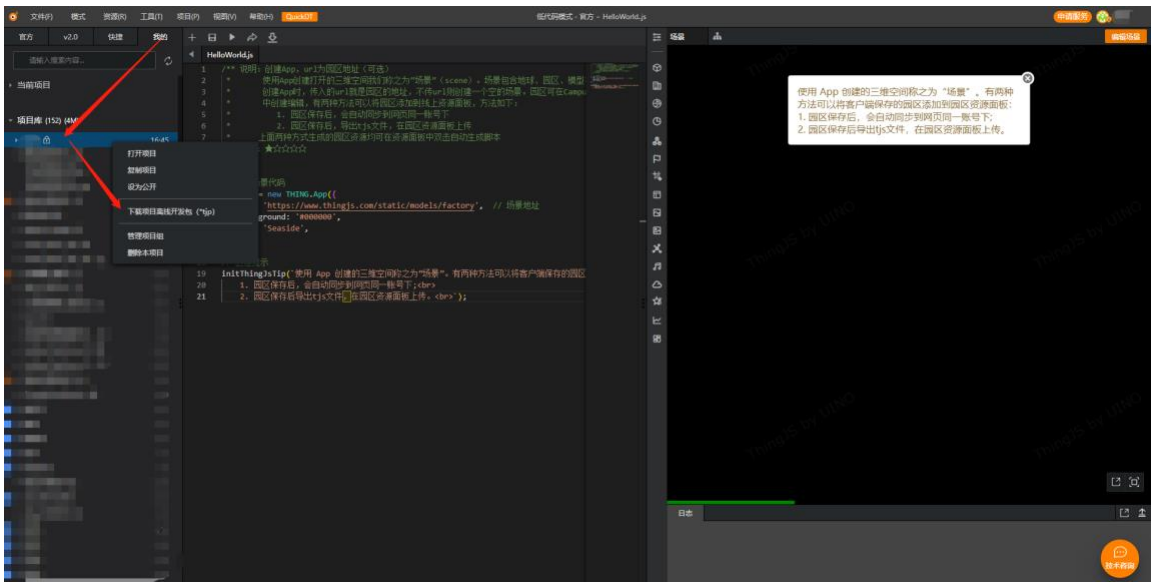


将下载好的“示例开发包”上传，等待项目上传成功后即可在 vscode 上继续进行开发（详细操作参考 7.2.5.2（1）章节关于 vscode 部分内容）。

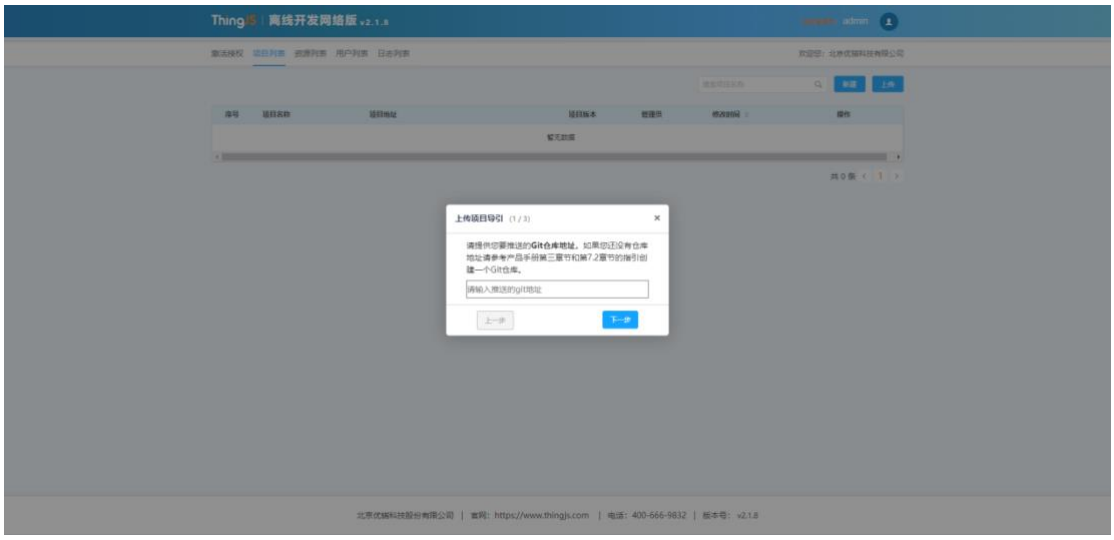


（3） 上传项目-获取“离线开发包”

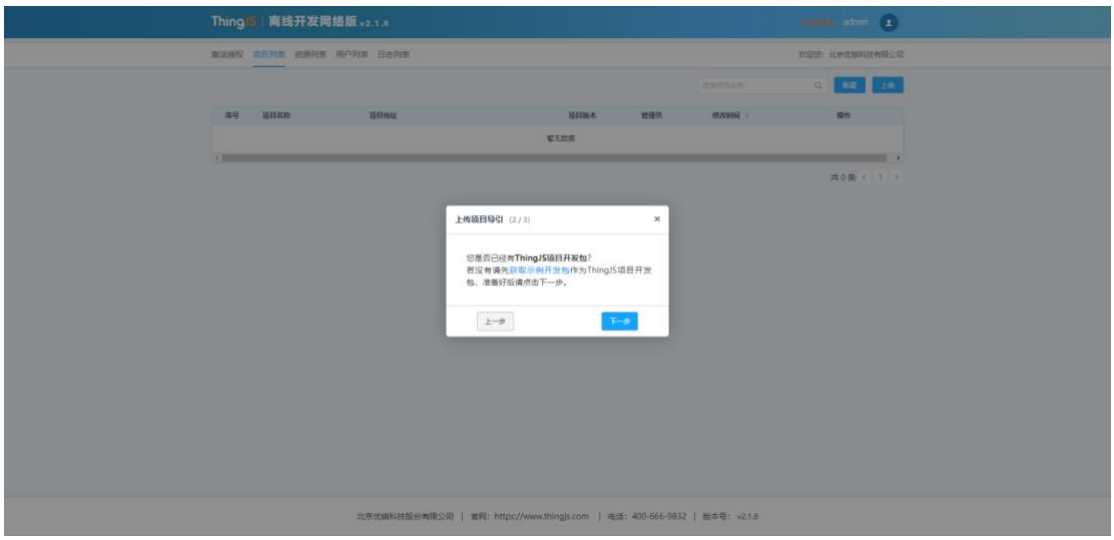
在“ThingJS 在线开发平台”中，打开一个已有项目，右键该项目，点击“下载项目离线开发包”，如下图所示：



访问“ThingJS 离线开发网络版”管理页面，切换至“项目列表”页签，点击“上传”按钮，在“上传项目引导”中先输入 7.2.3 章节的 Git 仓库地址，如下图所示：



将下载好的“离线开发包”上传，等待项目上传成功后即可在 vscode 上继续进行开发（详细操作参考 7.2.5.2（1）章节关于 vscode 部分内容）。



7.2.5.3 引用资源

（1） 场景地址为第 7.1.3.4 节中上传场景后所获取到的场景地址；

（2） 如果想用代码创建模型，可以去 CampusBuilder 客户端右侧模型列表找到要加载的模型 id，例如模型 id 为“7bfb3321557a40fead822d7285ac5324”，则该模型地址为：“/api/models/7bfb3321557a40fead822d7285ac5324/0/gltf/”，其中的 7bfb3321557a40fead822d7285ac5324 为模型 id，如下图所示。

```
var app = new THING.App({
    // 引用场景
    url: '/api/scene/20210616095248956300069',  // 场景地址
    skyBox: 'BlueSky', // 天空盒
    resourceLibraryUrl: "./"
});

app.on('load', function () {
    // 创建模型
    let obj = app.create({
        type: 'Thing',
        name: '宇航员',
        url: '/api/models/7bfb3321557a40fead822d7285ac5324/0/gltf/',
        position: [0, 0, 0],
        angle: 45
    });

    obj.playAnimation({
        name: '_defaultAnim_',
        loopType: THING.LoopType.Repeat,
      });
});
```

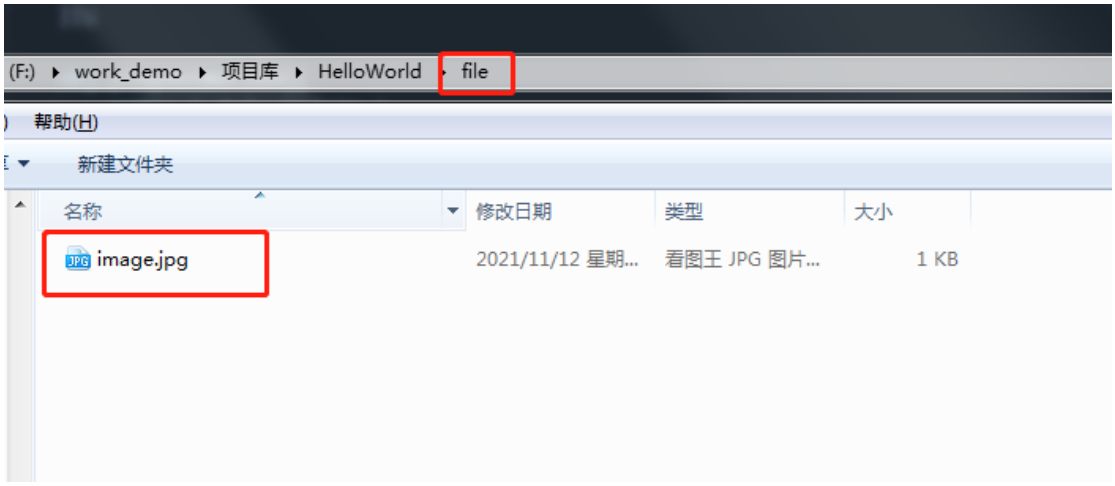
（3） 地图离线开发组件引用地址为“/source/uearth.min.js”。

（4） 地图离线开发森城市场景资源路径为“./cityBuilder/1/map.bundle.json”，标准版和专业版场景资源路径为“./cityBuilder/1/map. json”,配置

resourceConfig 参数，如下图所示：

```
// 加载地图
var app = new THING.App();
// 设置app背景为黑色
app.background = [0, 0, 0];
// 引用地图组件脚本
THING.Utils.dynamicLoad(['./source/uearth.min.js'], function () {
    app.create({
        type: 'Map',
        // 森城市
        url: './cityBuilder/1/map.bundle.json',
        // // 标准版、专业版
        // url: './cityBuilder/1/map.json',
        resourceConfig:{
            resourcePrefix: './'
        },
        complete: function (event) {
            console.log(event.object.userLayers.length);
        }
    });
});
```

（5） 项目中其他的文件资源引用地址需使用相对地址进行加载，例如：需要引用项目中/file 目录下的文件 image.jpg，目录结构如下图所示：

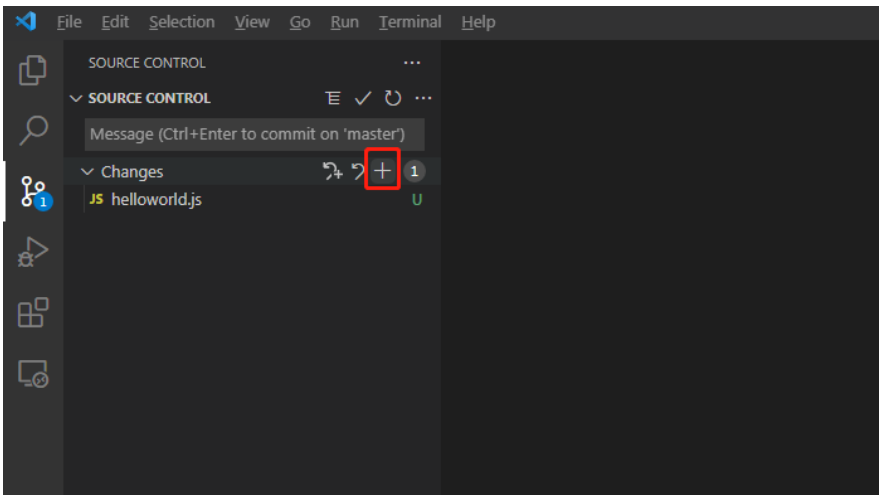


则引用地址为“./file/image.jpg”，如下图所示：

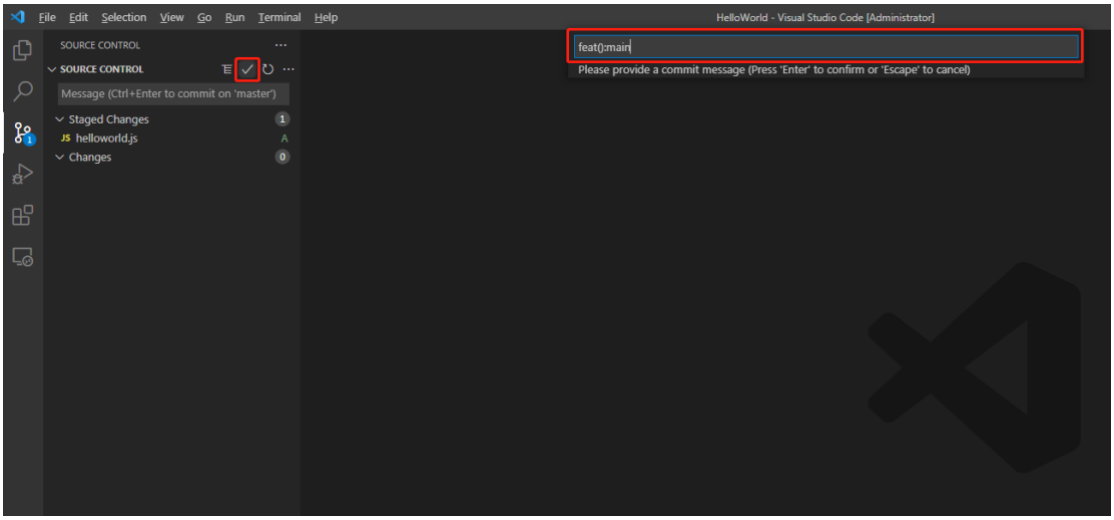
```
23
24  new THING.widget.Button('设置背景图片', function () {
25      app.background = './file/image.jpg';
26  })
27
```

7.2.5.4 推送项目

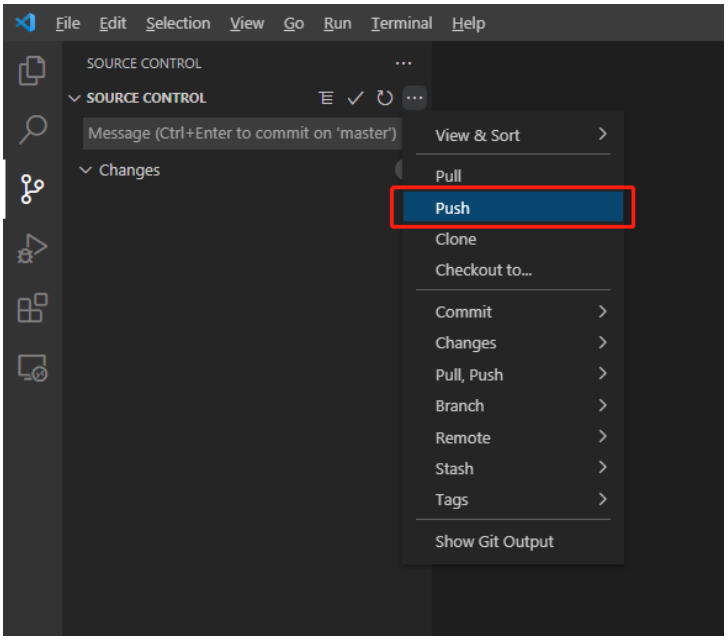
将项目添加到克隆好的仓库目录中后，Visual Studio Code 的 Git 版本控制面板中将展示新添加的文件，点击 + 将文件添加到 Git 中，如下图所示：



添加后点击 ✓，将更改内容提交（commit）到 Git 中，如下图所示：



提交完成后，选择“推送”，如下图所示：



注：为能预览到最新的项目，每次都需将修改内容推送到仓库中。

7.2.6 添加项目

项目推送到仓库中后，访问“ThingJS 离线开发网络版”管理界面，切换到“项目列表”页签，点击“新建”按钮，如下图所示：

根据“添加项目引导”提示，填写第 7.2.5.1 节中项目的“仓库地址”，如下图所示：

点击“创建”按钮，即可完成项目的添加，添加成功后，可对项目进行预览和离线部署操作。

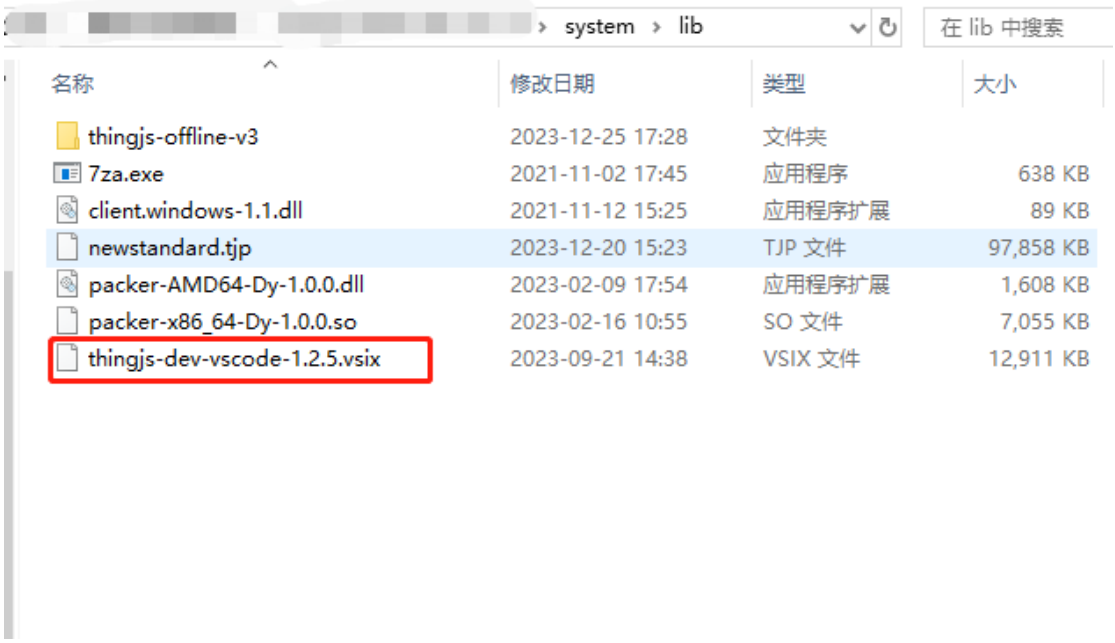
7.2.7 预览项目

7.2.7.1 远端项目预览

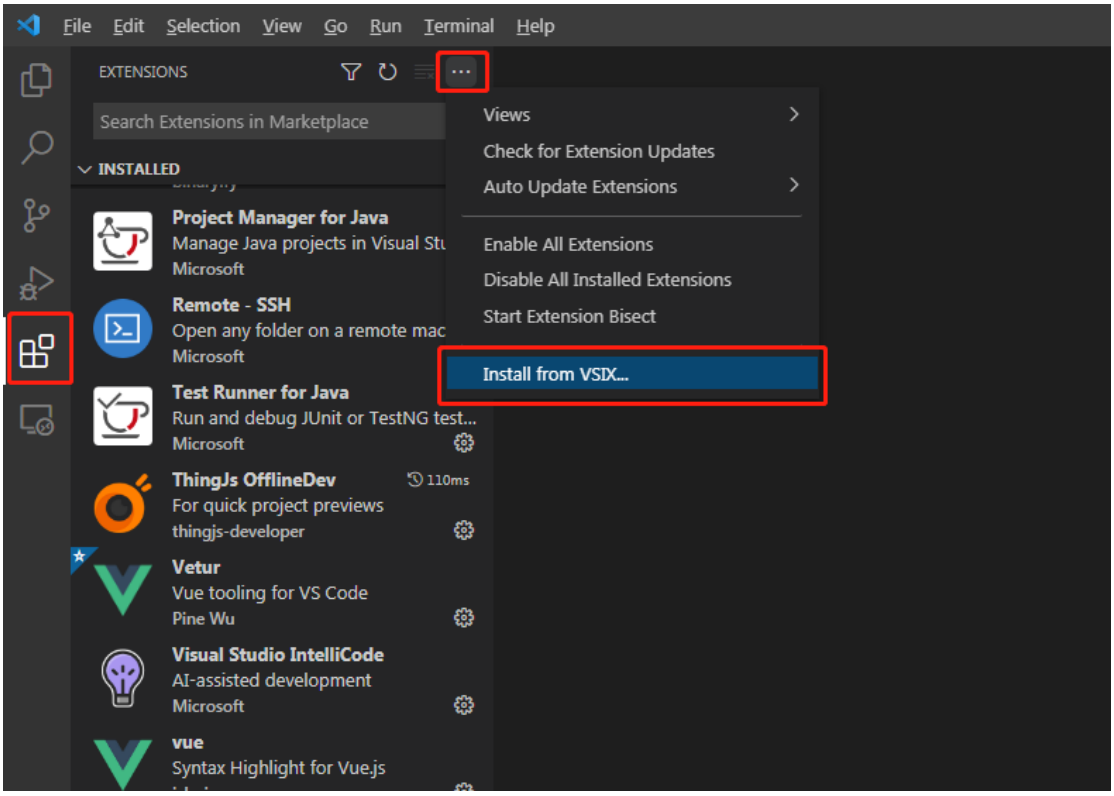
在 ThingJS 离线开发网络版管理页面，打开“项目列表”，选择要预览的项目，点击“预览地址”复制预览地址跳转进行项目预览，如下图所示：

7.2.7.2 本地开发预览

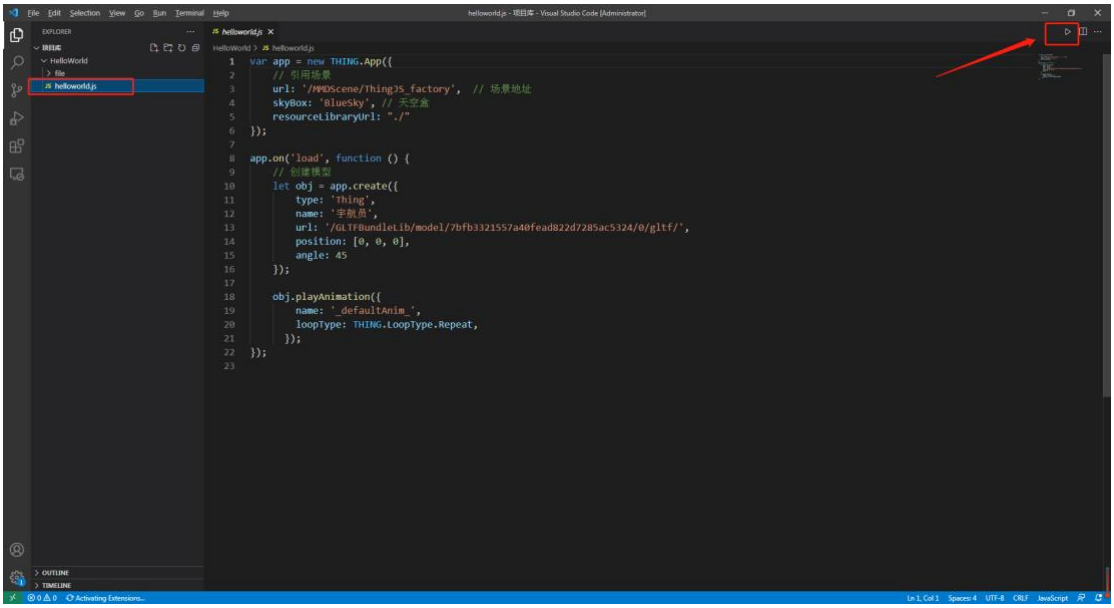
在本地开发环境中，使用提供的 Visual Studio Code 添加插件“thingjs-vscode-1.2.5.vsix”（注： Visual Studio Code 版本需在 1.50.0 以上），该插件可在“服务安装包”中“system/lib”目录下获取，



在 Visual Studio Code 中添加插件，如下图所示：

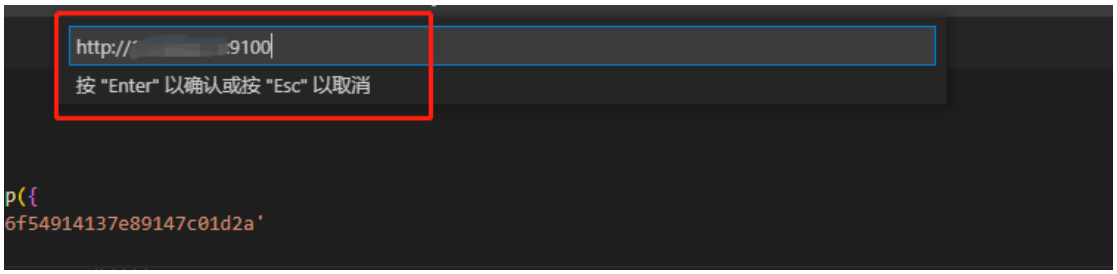


添加成功后，返回到项目主文件中，点击右上方的“项目预览”按钮，如下图所示：

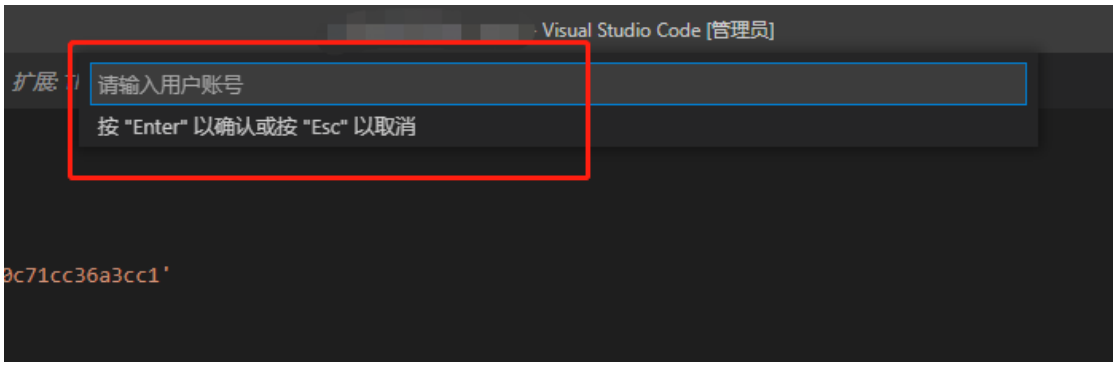


注：请确保在项目主文件页面进行项目预览。

首次预览需在弹出的提示框中需完整填写“ThingJS 离线开发网络版”部署的服务地址，如下图所示：

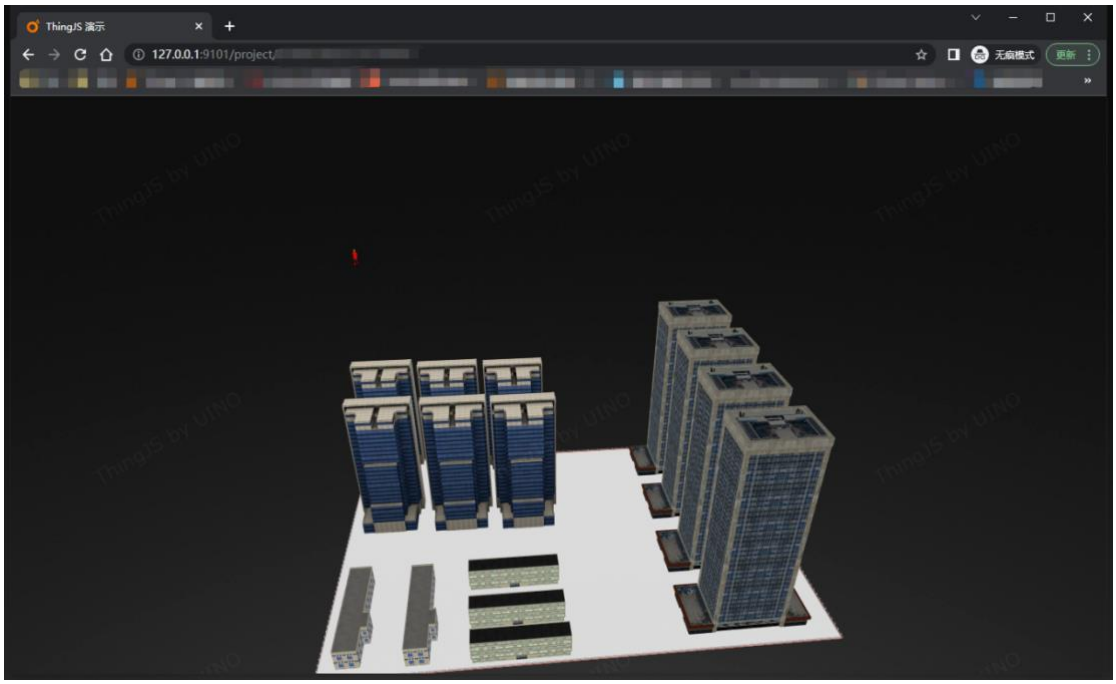


添加完“ThingJS 离线开发网络版”服务地址后，在弹出“请输入用户账号”提示框中，输入离线开发网络版中当前项目所属的管理员名称即可，如下图所示：



确定后即可默认跳转到浏览器进行项目本地开发预览，每次修改后，刷新预览链接即可看到最新的项目内容。

注：本地开发预览服务默认启动在 9101 端口。



8 资源

本章节介绍如何在离线开发中上传加载模型、园区、地图、图表、拓扑等单资源，用户可以根据需求预览对应章节。

8.1 资源兼容表

ThingJS 在线/离线开发最新版能够支持的资源情况如下：

ThingJS在线/离线开发-资源兼容表

序号	资源	分类	在线开发v1.0		离线开发坐席版v2.3.10			离线开发网络版v2.1.5		
			资源上传/使用	下载离线开发	加载离线开发	资源上传使用	产出离线部署	加载离线开发	资源上传使用	产出离线部署
1	园区	手动上传	✓	✓	✓	✓	✓	✓	✓	✓
		模模搭自动同步	✓	✓	✓	✓	✓	✓	✓	✓
		森园区在线搭建	✓	✓	✓	✓	✓	✓	✓	✓
		max场景	✓	✓	✓	✓	✓	✓	✓	✓
		BIM场景	✓	✓	✓	✓	✓	✓	✓	✓
2	地图	标准版地图	✓	✓	✓	✓	✓	✓	✓	✓
		专业版地图	✓	✓	✓	✓	✓	✓	✓	✓
		森城市地图	✓	✓	✓	✓	✓	✓	✓	✓
3	模型	公共模型	✓	✓	✓	-	✓	✓	-	✓
		已购买模型	✓	✓	✓	✓	✓	✓	✓	✓
		CB上传模型 （含obj）	✓	✓	✓	✓	✓	✓	✓	✓
		场景中的模型（含obj）	✓	✓	✓	✓	✓	✓	✓	✓
		max上传模型	✓	✓	✓	✓	✓	✓	✓	✓
4	图表	QuickChart（代码块）	✓	✓	✓	✓	✓	✓	✓	✓
		森大屏	✓	✓	✓	✓	✓	✓	✓	✓
		森图表	✓	✓	✓	✓	✓	✓	✓	✓
5	动态背景	动态背景	✓	✓	✓	-	✓	✓	-	✓
6	贴图	贴图资源	✓	✓	✓	-	✓	✓	-	✓
7	粒子模型	官方粒子	✓	✓	✓	-	✓	✓	-	✓
		个人粒子	✓	✓	✓	-	✓	✓	-	✓
8	音乐资源	音乐资源	✓	✓	✓	-	✓	✓	-	✓
9	效果模板	园区效果（官方）	✓	✓	✓	✓	✓	✓	✓	✓
		地图效果（官方）	✓	✓	✓	✓	✓	✓	✓	✓
		个人园区效果	✓	✓	✓	✓	✓	✓	✓	✓
10	配饰	配饰标识	✓	✓	✓	✓	✓	✓	✓	✓
11	拓扑图	拓扑图	✓	✓	✓	✓	✓	✓	✓	✓
12	天空盒	天空盒	✓	✓	✓	-	✓	✓	-	✓
13	全景图	全景图	✓	✓	✓	-	✗	✓	-	✗
14	快捷代码	快捷代码	✓	✓	✓	-	✓	✓	-	✓
15	官方示例	各示例引用资源	✓	✓	✓	-	✓	✓	-	✓
		官方场景	✓	✓	✓	-	✓	✓	-	✓

8.2 模型

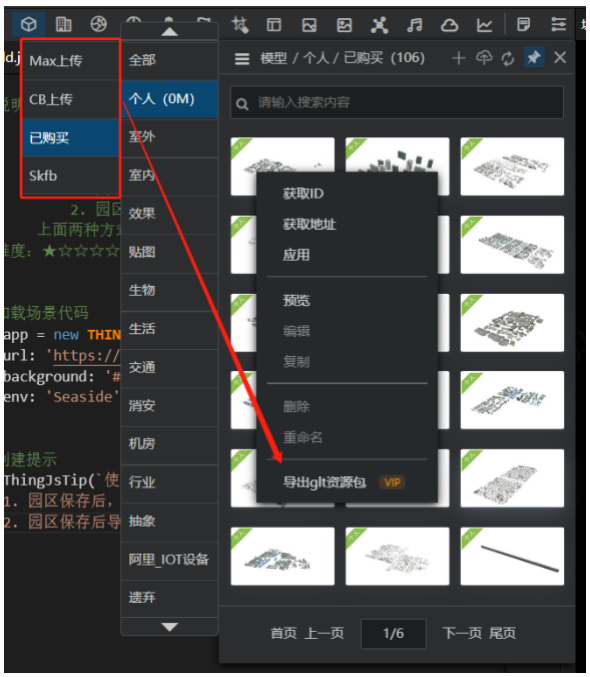
本节依次介绍模型资源的获取渠道、使用方式和更新方法。

官方支持获取模型资源的途径有以下四种，分别为 Max 插件上传模型（Max 模型）、模模搭中手动上传的模型（CB 上传）、已购买模型、skfb 模型。

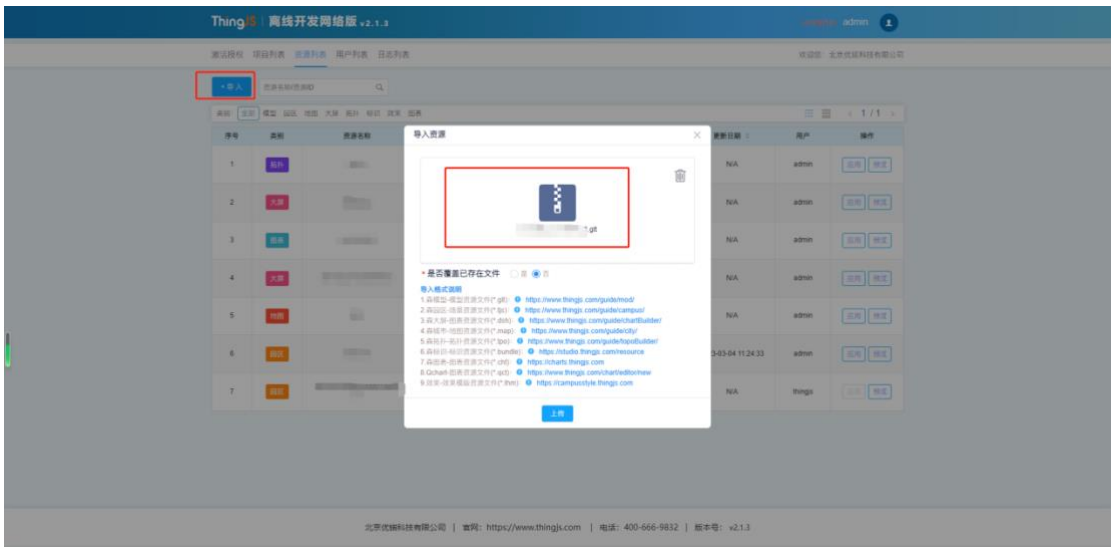
如果您想将上述模型资源引用到离线开发的项目中，或是将项目中已有的模型资源进行更新，您都可以参照本节。

①若您想将在线环境中的园区资源在离线环境中使用，具体操作步骤如下：

首先，在“在线开发”找到所需要的模型资源，右键点击导出.gltf 文件。如下图所示：



然后，将导出文件上传到离线开发中。点击“导入”按钮，将资源文件拖拽至矩形框中，点击“上传”按钮。



上传成功后，提示框弹出“XXX.gltf 包上传成功！”，如下图所示：

上传成功后，界面切换到显示“模型”的列表下，该资源可“应用”“预览”。

注：系统管理员可以“预览”应用管理员资源，不可“应用”其资源。

②如您想要使用该资源：在资源列表中找到目标资源，点击右侧“应用”，此时代码块已复制到您的剪切板中，打开或创建项目后，在您 vscode 中“ctrl+v”即可完成复制。

③如您需要更新本资源：点击“导入”按钮，上传需要更新的资源文件，并在是否覆盖文件处，选择“是”，点击上传即可。

8.3 园区

本节依次介绍园区资源的获取渠道、使用方式和更新方法。

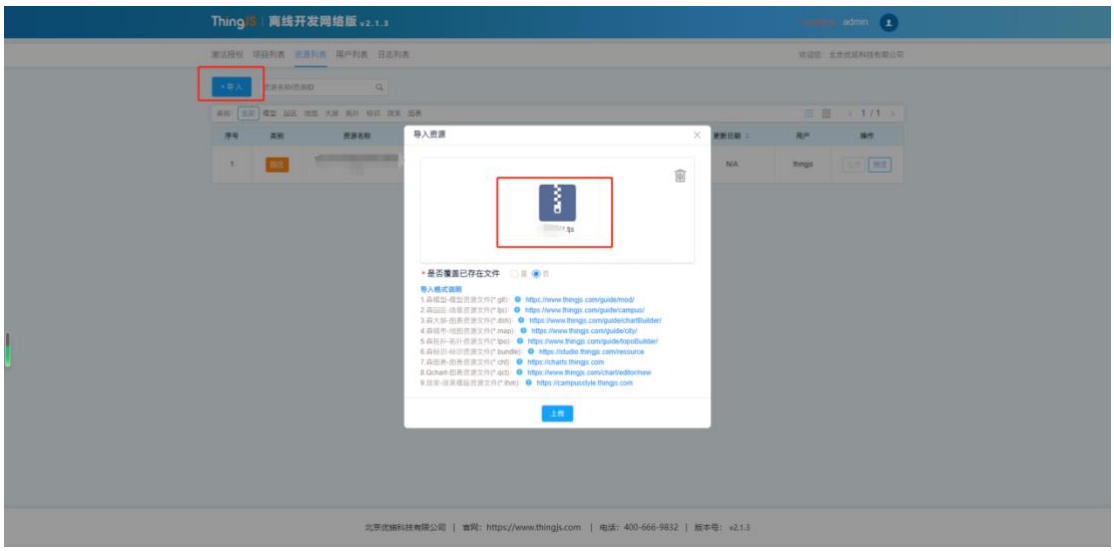
官方支持获取园区资源的途径有以下四种，分别为模模搭同步场景（同步）、手动上传的园区（上传）、森园区搭建的园区（在线）、MAX 插件导入的园区（MAX）。

如果您想将上述园区资源引用到离线开发的项目中，或是将项目中已有的园区资源进行更新，您都可以参照本节。

①若想将在线环境中的园区资源在离线环境中使用，具体操作步骤如下：

首先，在“在线开发”找到所需要的园区场景，右键点击导出 tjs 文件。如下图所示：

然后，将导出的 tjs 包上传到离线开发网络版资源列表中，点击导入按钮上传对应 tjs 文件。如下图所示：

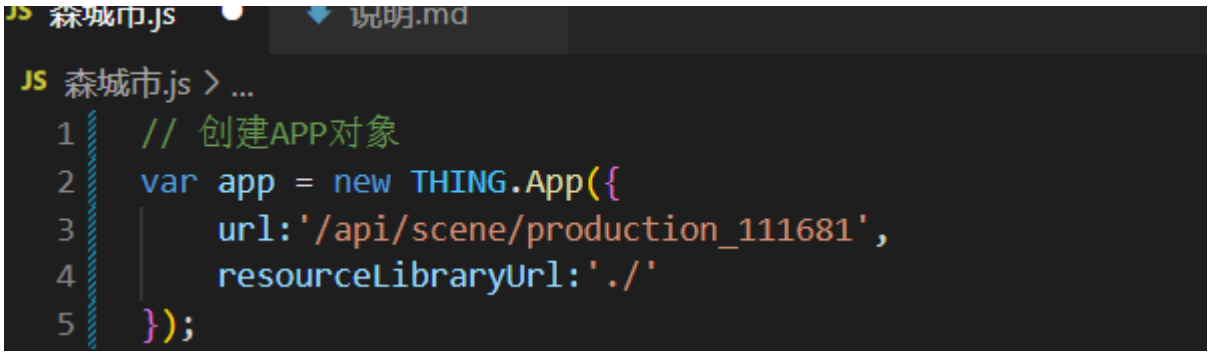


上传成功后提示框弹出“XXX.tjs 包上传成功！”，如下图所示：

上传成功后，界面切换到显示“园区”的列表下，该资源可“应用”“预览”。

②如您想要使用该资源：在资源列表中找到目标资源，点击右侧“应用”，此时代码块已复制到您的剪切板中，打开或创建项目后，在您 vscode 中“ctrl+v”即可完成复制。

注：项目中离线官方场景单资源加载需要配置“resourceLibraryUrl:'./'”字段。如下图所示：



③如您需要更新本资源：点击“导入”按钮，上传需要更新的资源文件，并在是否覆盖文件处，选择“是”，点击上传即可。

更新资源后，在项目中引用的该资源也会同步更新。

8.4 地图

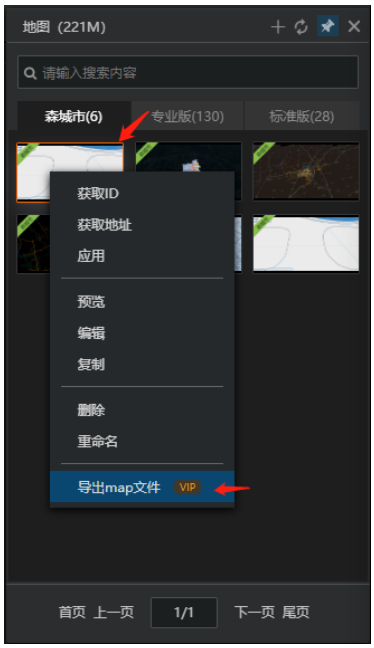
本节依次介绍地图资源的获取渠道、使用方式和更新方法。

官方支持获取地图资源的途径有以下三种，分别为森城市、专业版、标准版。

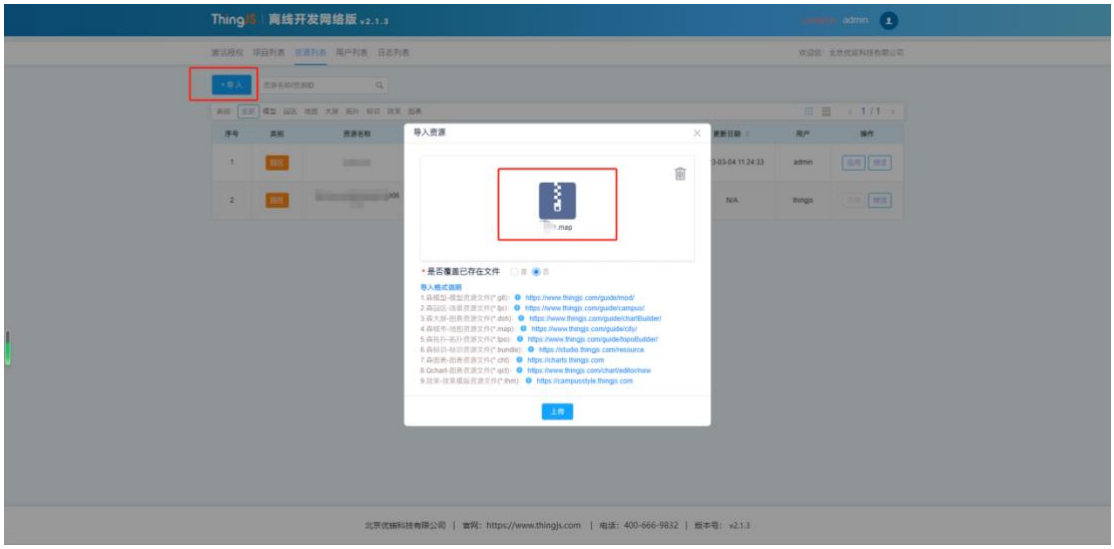
如果您想将上述地图资源引用到离线开发的项目中，或是将项目中已有的地图资源进行更新，您都可以参照本节。本章以森城市资源为例。

①若您想将在线环境中的地图资源在离线环境中使用，具体操作步骤如下：

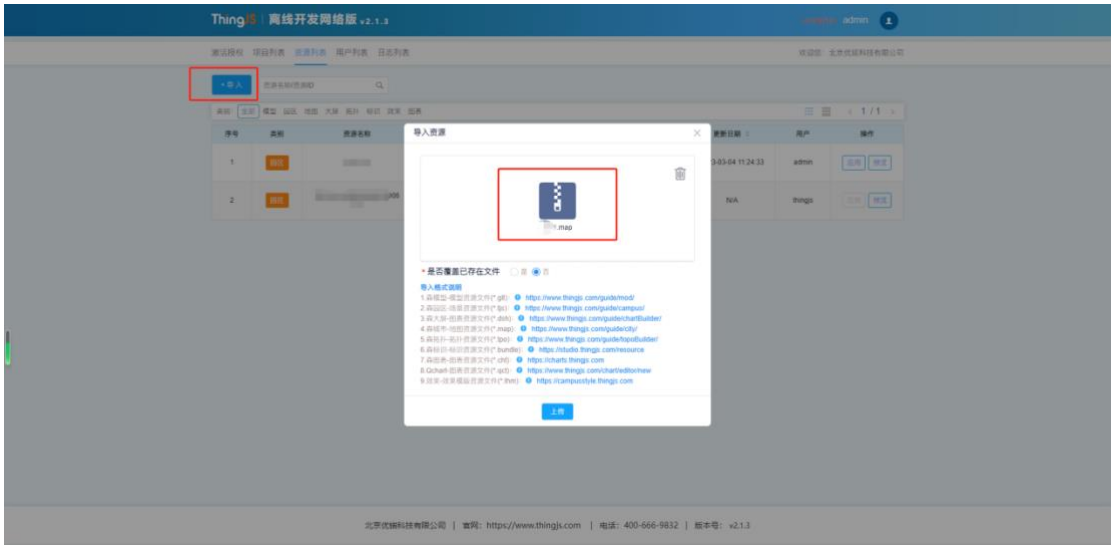
首先，在“在线开发”找到所需要的地图资源，右键点击导出.map 文件。如下图所示：



然后，将导出的 map 包上传到离线开发网络版资源列表中，点击导入按钮上传对应 map 文件。如下图所示：



上传成功后提示框弹出“XXX.map 包上传成功！”，如下图所示：



上传成功后，界面切换到显示“地图”的列表下，该资源可“应用”“预览”。

注：系统管理员可以“预览”应用管理员资源，不可“应用”其资源。

②如您想要使用该资源：在资源列表中找到目标资源，点击右侧“应用”，此时代码块已复制到您的剪切板中，打开或创建项目后，在您 vscode 中“ctrl+v”即可完成复制。

③如您需要更新本资源：点击“导入”按钮，上传需要更新的资源文件，并在是否覆盖文件处，选择“是”，点击上传即可。

更新资源后，在项目中引用的该资源也会同步更新。

8.5 图表

本节依次介绍图表资源的获取渠道、使用方式和更新方法。

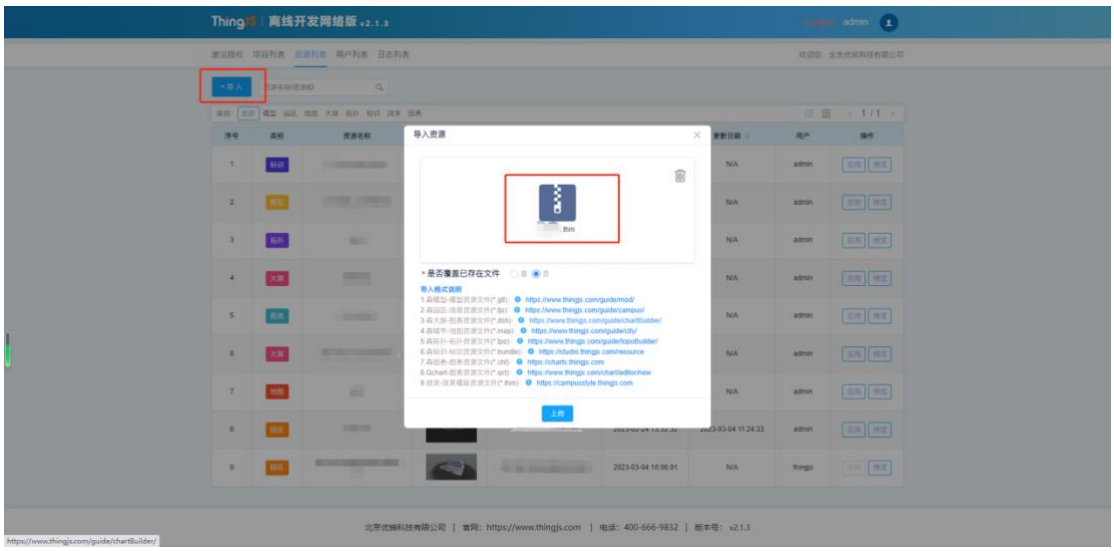
官方支持获取图表资源的途径有以下两种，分别为 QuickChart、森大屏。

如果您想将上述图表资源引用到离线开发的项目中，或是将项目中已有的图表资源进行更新，您都可以参照本节。本章以 QuickChart 资源为例。

①若您想将在线环境中的 QuickChart 资源在离线环境中使用，具体操作步骤如下：

首先，在“在线开发”找到所需要的 QuickChart 资源，右键点击导出.qct 文件。如下图所示：

然后，将导出的 qct 包上传到离线开发网络版资源列表中，点击导入按钮上传对应 qct 文件。如下图所示：



上传成功后提示框弹出“XXX.qct 包上传成功！”，如下图所示：

上传成功后，界面切换到显示“图表”的列表下，该资源可“应用”“预览”。

②如您想要使用该资源：在资源列表中找到目标资源，点击右侧“应用”，此时代码块已复制到您的剪切板中，打开或创建项目后，在您 vscode 中“ctrl+v”即可完成复制。
③如您需要更新本资源：点击“导入”按钮，上传需要更新的资源文件，并在是否覆盖文件处，选择“是”，点击上传即可。
更新资源后，在项目中引用的该资源也会同步更新。

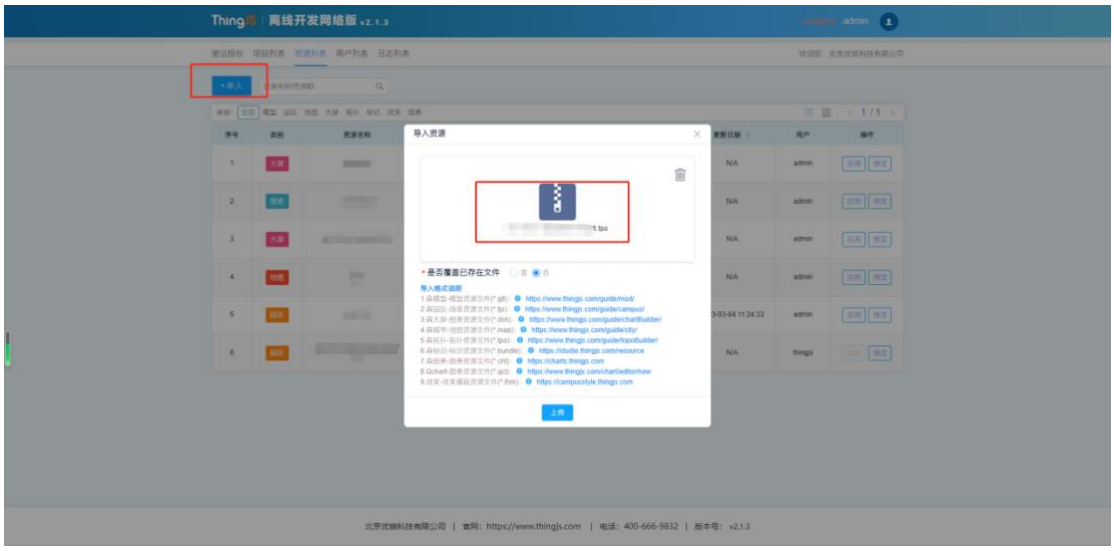
8.6 拓扑

本节依次介绍拓扑资源的获取渠道、使用方式和更新方法。

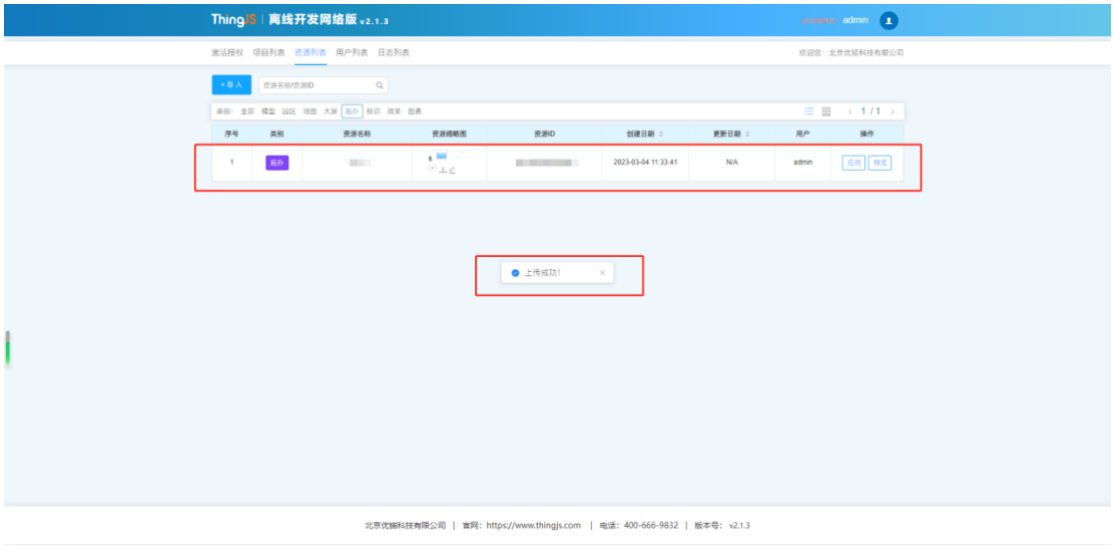
如果您想将拓扑资源引用到离线开发的项目中，或是将项目中已有的图表资源进行更新，您都可以参照本节。

①若想将在线环境中的拓扑资源在离线环境中使用。首先，在“在线开发”找到所需要的拓扑，右键点击导出 tpo 文件。如下图所示：

然后，将导出的 tpo 包上传到离线开发网络版资源列表中，点击导入按钮上传对应 tpo 文件。如下图所示：



上传成功后提示框弹出“XXX.tpo 包上传成功！”，如下图所示：



上传成功后，界面切换到显示“拓扑”的列表下，该资源可“应用”“预览”。

注：系统管理员可以“预览”应用管理员资源，不可“应用”其资源。

②如您想要使用该资源：在资源列表中找到目标资源，点击右侧“应用”，此时代码块已复制到您的剪切板中，打开或创建项目后，在您 vscode 中“ctrl+v”即可完成复制。

③如您需要更新本资源：点击“导入”按钮，上传需要更新的资源文件，并在是否覆盖文件处，选择“是”，点击上传即可。

更新资源后，在项目中引用的该资源也会同步更新。

8.7 效果

本节依次介绍效果模板资源的获取渠道、使用方式和更新方法。

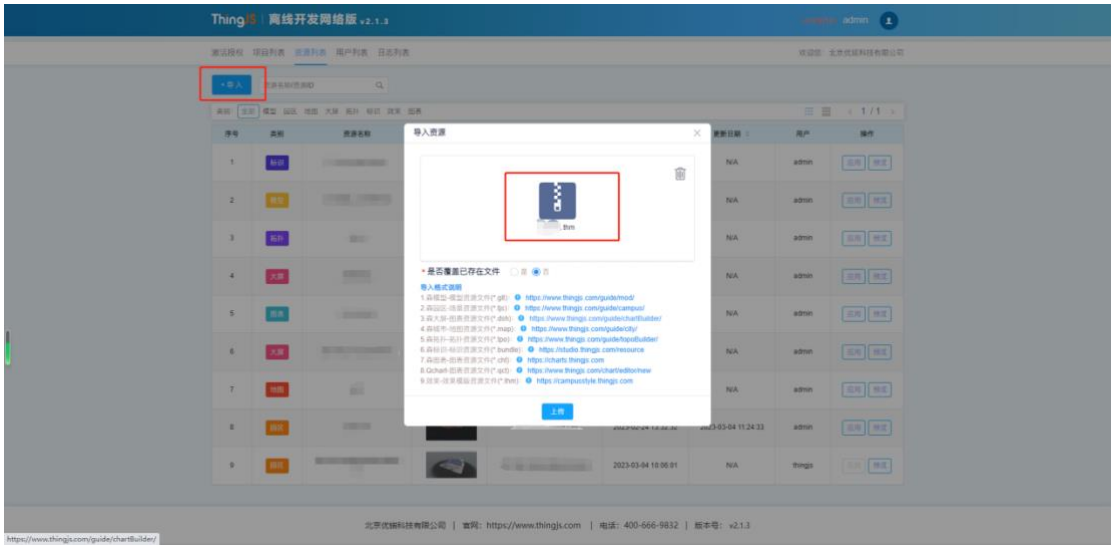
如果您想将园区和地图效果模板资源引用到离线开发的项目中，或是将项目中已有的效果模板资源进行更新，您都可以参照本节。

注：目前仅支持官方效果模板复制后的资源进行下载，其他方式产生的个人模板后续将逐步进行支持。

①若想将在线环境中的效果模板资源在离线环境中使用。首先，在“在线开发”效果模板列表的园区/地图页签下，选择所需要的效果模板资源，并右键点击“复制”按钮，该资源将复制到个人页签下，如下图所示：

在个人页签下，选择该效果模板，右键点击导出 thm 文件。如下图所示：

然后，将导出的 thm 包上传到离线开发网络版资源列表中，点击导入按钮上传对应 thm 文件。如下图所示：



上传成功后提示框弹出“XXX.thm 包上传成功！”，如下图所示：

上传成功后，界面切换到显示“效果”的列表下，该资源可“应用”“预览”。

注：系统管理员可以“预览”应用管理员资源，不可“应用”其资源。

②如您想要使用该资源：在资源列表中找到目标资源，点击右侧“应用”，此时代码块已复制到您的剪切板中，打开或创建项目后，在您 vscode 中“ctrl+v”即可完成复制。

③如您需要更新本资源：点击“导入”按钮，上传需要更新的资源文件，并在是否覆盖文件处，选择“是”，点击上传即可。

更新资源后，在项目中引用的该资源也会同步更新。

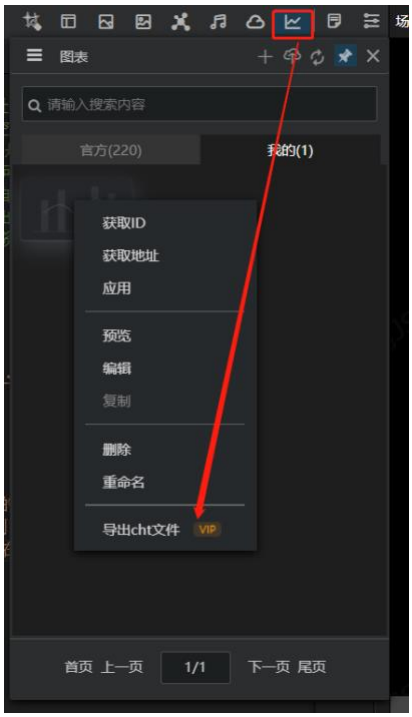
8.8 图表组件

本节依次介绍图表组件资源的获取渠道、使用方式和更新方法。

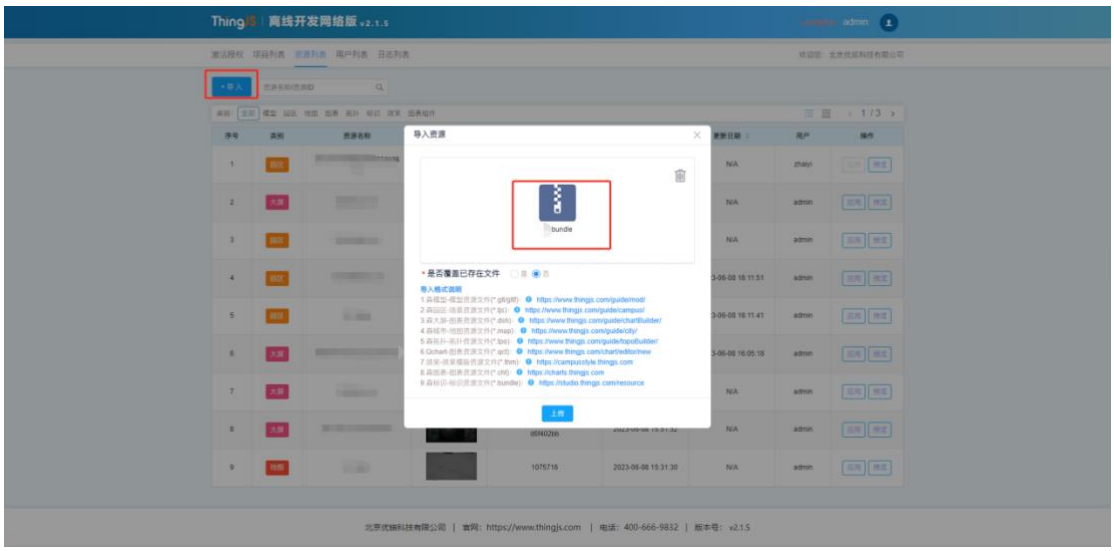
如果您想将上述图表组件资源引用到离线开发的项目中，或是将项目中已有的图表组件资源进行更新，您都可以参照本节。

①若想将在线环境中的图表资源在离线环境中使用，具体操作步骤如下：

首先，在“在线开发”找到所需要的图表场景，右键点击导出 cht 文件。如下图所示：



然后，将导出的 cht 包上传到离线开发网页版资源列表中，点击导入按钮上传对应 cht 文件。如下图所示：



上传成功后提示框弹出“XXX.cht 包上传成功！”，如下图所示：

②如您想要使用该资源：在资源列表中找到目标资源，点击右侧“获取代码块”，此时代码块已复制到您的剪切板中，打开或创建项目后，在您 vscode 中“ctrl+v”即可完成复制。

③如您需要更新本资源：点击“导入”按钮，上传需要更新的资源文件，并在是否覆盖文件处，选择“是”，点击上传即可。

更新资源后，在项目中引用的该资源也会同步更新。

8.9 标记

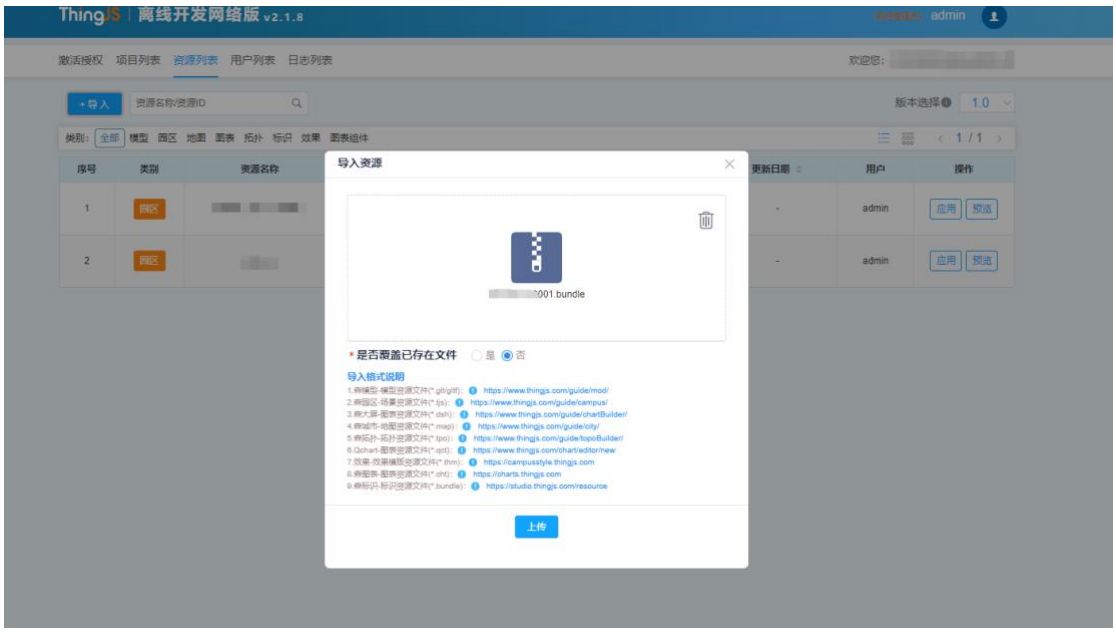
本节依次介绍标记资源的获取渠道、使用方式和更新方法。

如果您想将上述标记资源引用到离线开发的项目中，或是将项目中已有的标记资源进行更新，您都可以参照本节。

①若想将在线环境中的图表资源在离线环境中使用，具体操作步骤如下：

首先，在“在线开发”找到所需要的图表场景，右键点击导出 bundle 文件。如下图所示：

然后，将导出的 bundle 包上传到离线开发网页版资源列表中，点击导入按钮上传对应 bundle 文件。如下图所示：



上传成功后提示框弹出“XXX.bundle 包上传成功！”，如下图所示：

②如您想要使用该资源：在资源列表中找到目标资源，点击右侧“获取代码块”，此时代码块已复制到您的剪切板中，打开或创建项目后，在您 vscode 中“ctrl+v”即可完成复制。

③如您需要更新本资源：点击“导入”按钮，上传需要更新的资源文件，并在是否覆盖文件处，选择“是”，点击上传即可。

更新资源后，在项目中引用的该资源也会同步更新。

9 部署

9.1 配置 CPS 文件

项目开发完成后，可对项目进行离线部署。

若项目中引用了第 8 章中资源列表的单资源，需手动配置与主文件同名的 cps 文件，配置如下图所示：

```
{
    // 当前项目使用的ThingJS包（thing.min.js）版本号
    "thingjs_version" : "1.4.2",
    // 举例：项目中引用的第一个场景URL为 "/api/scene/d370cad09e74f42d932b083d",
    //      第二个场景URL为 "/api/scene/b422fd26d4c7874df3992068",
    //      为能正确打包上述两个场景或单资源园区，需配置如下：
    //"scenes": [
    //    "/api/scene/d370cad09e74f42d932b083d",
    //    "/api/scene/b422fd26d4c7874df3992068"
    // ],
    // 举例：项目中动态引用了模型 "/api/models/8CF6171F7EE046968B16E10181E8D941/0/gltf/"，为能正确打包该引用模型，需配置如下：
    // "models": [
    //     "/api/models/8CF6171F7EE046968B16E10181E8D941/0/gltf/"
    //  ],
    // 举例：项目中引用了单资源模型 "/Resources/Model/admin/447aebc08c5e4bc1a2f0d84a0747b763"，为能正确打包该引用模型，需配置如下：
    //"glts": [
    //  "/Resources/Model/admin/447aebc08c5e4bc1a2f0d84a0747b763"
    //],
    // 举例：项目中引用了单资源QChart "/api/chart/638ede8aee63ce091cc770ca"，为能正确打包该引用模型，需配置如下：
    // "qcts": [
    //     "/api/chart/638ede8aee63ce091cc770ca"
    //  ],
    // 举例：项目中引用了单资源森大屏 "/Resources/BigScreen/admin/178eb10990544ac6abe00388209c143e"，为能正确打包该引用模型，需配置如下：
    //"bigScreens": [
    //    "/Resources/BigScreen/admin/9a32fc758a44499d94a955680b30e369"
    // ],
    // 举例：项目中引用了单资源森图表 "/Resources/senChart/admin/f5e187be77074a22833becd4e3492d8a"，为能正确打包该引用模型，需配置如下：
    // "chts": [
    //     "/Resources/senChart/admin/f5e187be77074a22833becd4e3492d8a"
    // ],
    // 举例：项目中引用了单资源标识 "/Resources/bundle/admin/RMK100000000187590"，为能正确打包该引用模型，需配置如下：
    // "bundles": [
    //     "/Resources/bundle/admin/RMK100000000187590"
    // ],
    // 举例：项目中引用了单资源效果 "/Resources/EffectTemplate/admin/9869"，为能正确打包该引用模型，需配置如下：
    // "thms": [
    //     "/Resources/EffectTemplate/admin/9869"
    // ],
    // 举例：项目中引用了单资源拓扑 "/Resources/topo/admin/9d124b930cefdf59"，为能正确打包该引用模型，需配置如下：
    // "tpos": [
    //     "/Resources/topo/admin/9d124b930cefdf59"
    // ],
    // 举例：项目中引用了单资源地图 "/Resources/CityMap/admin/43933"，为能正确打包该引用模型，需配置如下：
    // "maps": [
    //     "/Resources/CityMap/admin/1059084"
    // ]
}
```

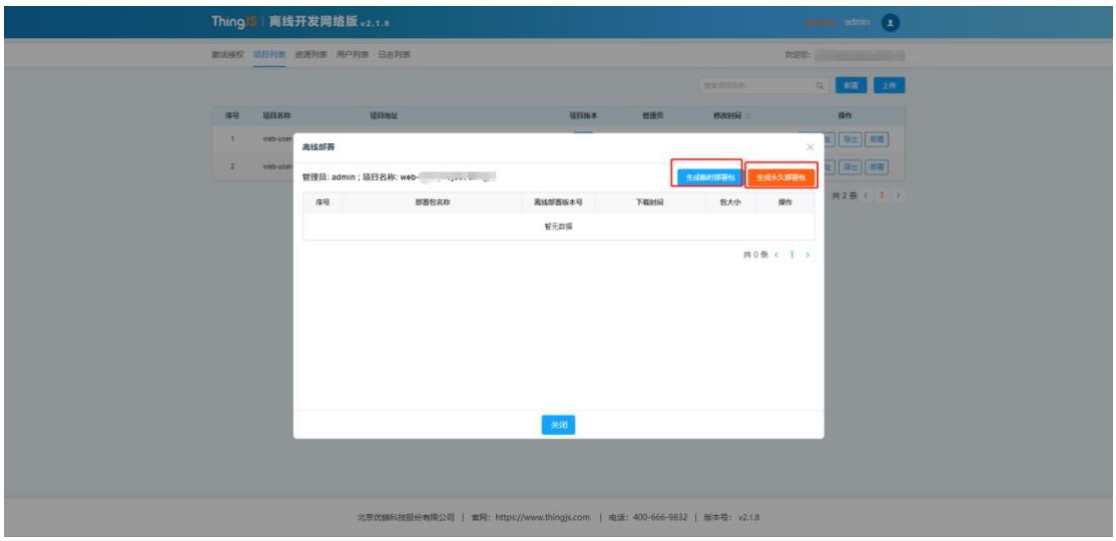
举例：如项目中使用园区和模型资源，配置内容如下：

```
{
    // 当前项目使用的ThingJS包（thing.min.js）版本号
    "thingjs_version" : "1.2.7.16",
    // 举例：项目中引用的第一个场景URL为 "/api/scene/d370cad09e74f42d932b083d",
    //       第二个场景URL为 "/api/scene/b422fd26d4c7874df3992068",
    //       为能正确打包上述两个场景，需配置如下：
    "scenes": [
        "/api/scene/15fc31d71fba01bc0a093
    ],
    // 举例：项目中动态引用了模型 "/api/models/8CF6171F7EE046968B16E10181E8D941/0/gltf/",
    //       为能正确打包该引用模型，需配置如下：
    "models": [
        "/api/models/ABCF7A3F0BDA48B7B7A8C9C94824359D/0/gltf/"
    ]
}
```

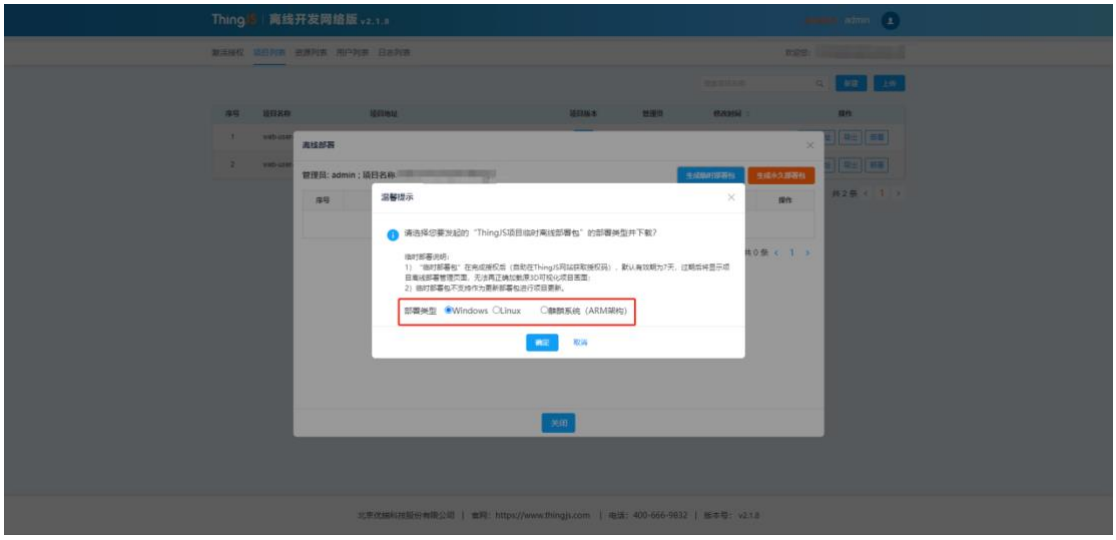
注：资源路径即为在代码中创建时使用的路径地址

9.2 下载离线部署包

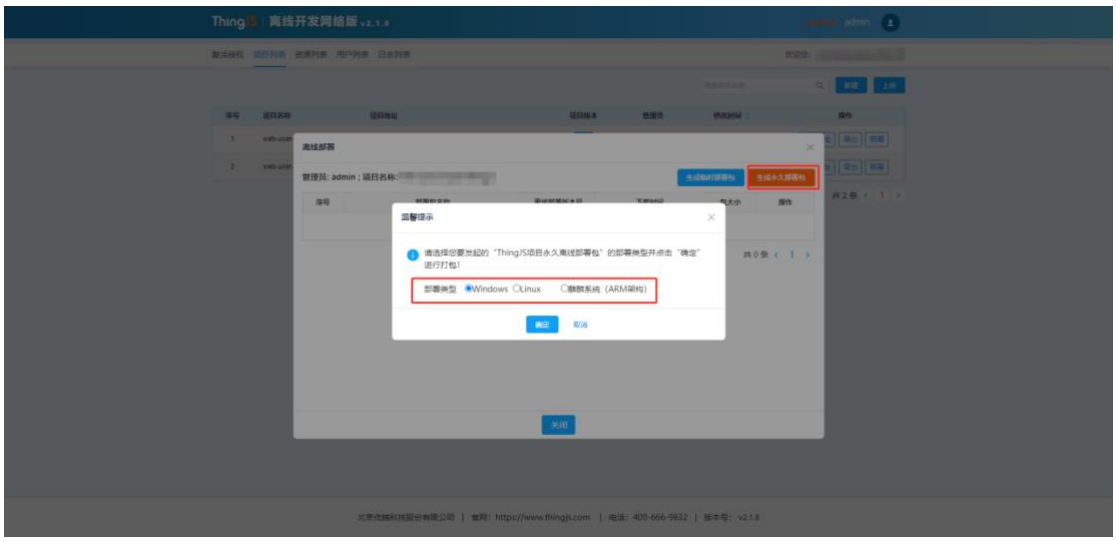
访问“ThingJS 离线开发网络版”管理界面，切换到“项目列表”页签，对要发布的项目点击“离线部署”按钮，可支持生成临时部署包和永久部署包，如下图所示：



点击“生成临时部署包”按钮，并选择“部署类型”，等待部署包生成即可。



点击“生成永久部署包”按钮，并选择“部署类型”，等待部署包生成即可。



关于离线部署包的使用，请到 https://store.thingjs.com/tools 查看“项目离线部署包用户手册”进行后续操作。

10 迁移

变更部署服务器，或改变部署服务器的硬件信息（如硬盘、CPU、网卡等）需重新授权，如确需对“ThingJS 离线开发网络版”进行环境变更，请按本章节内容迁移。

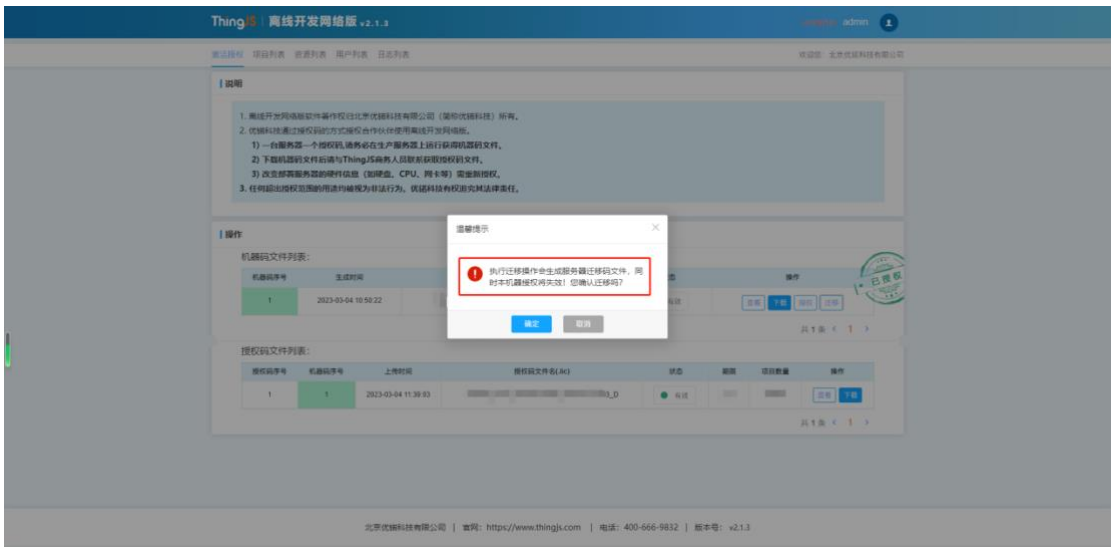
10.1 获取新机器码文件

将“ThingJS 离线开发网络版”部署包正确部署到新的服务器环境中，启动后访问部署在新服务器环境中的“ThingJS 离线开发网络版”管理界面，下载当前有效的新的“机器码文件”至本地。

10.2 获取迁移码文件

返回到原已部署的“ThingJS 离线开发网络版”管理界面，点击“迁移”按钮，获取“迁移码文件”。

（注：执行迁移操作后，当前机器授权将失效，且无法再次授权激活。）



将生成的“迁移码文件”下载至本地。

10.3 获取新授权码文件

将 9.1 节获取的“新机器码文件”和“迁移码文件”发送给 ThingJS 商务人员，获取“新授权码文件”。

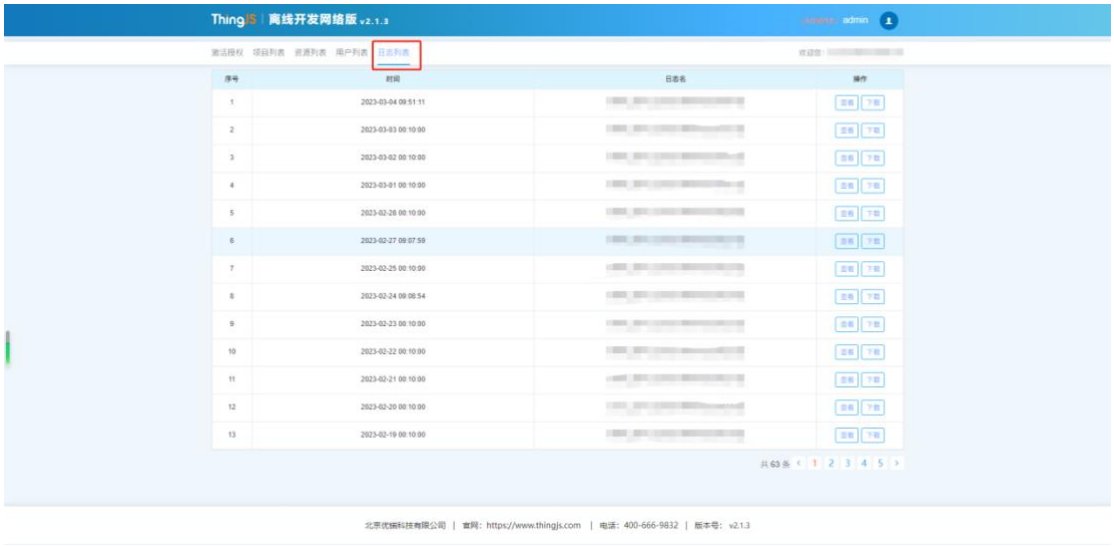
10.4 完成新服务器的授权

访问部署在新服务器环境中的“ThingJS 离线开发网络版”管理界面，在当前有效的机器码文件一栏中选择“授权”按钮，上传 9.3 节中获取的新的对应的授权码文件，即可完成新服务器的授权。

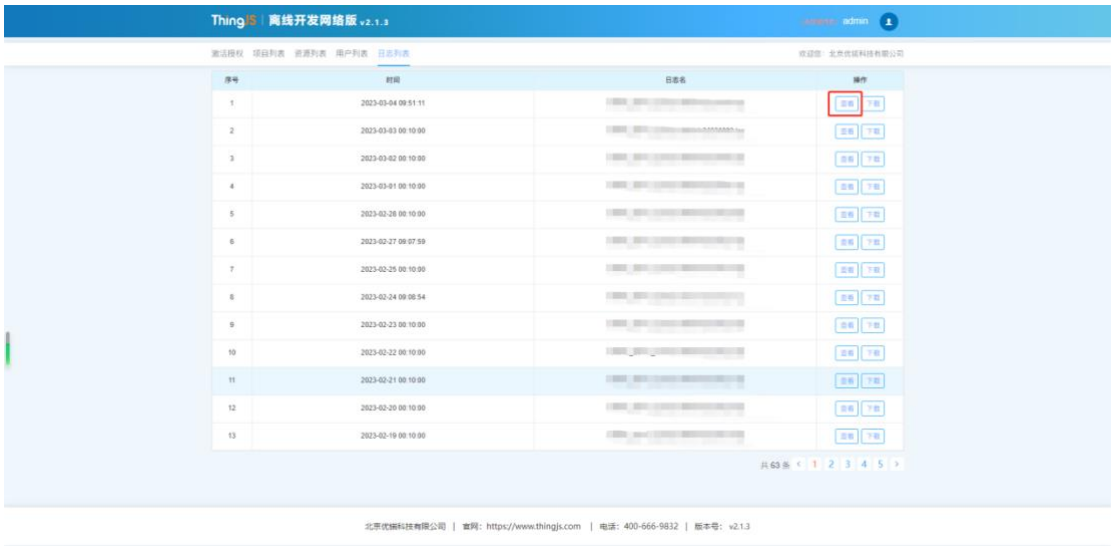
11 日志

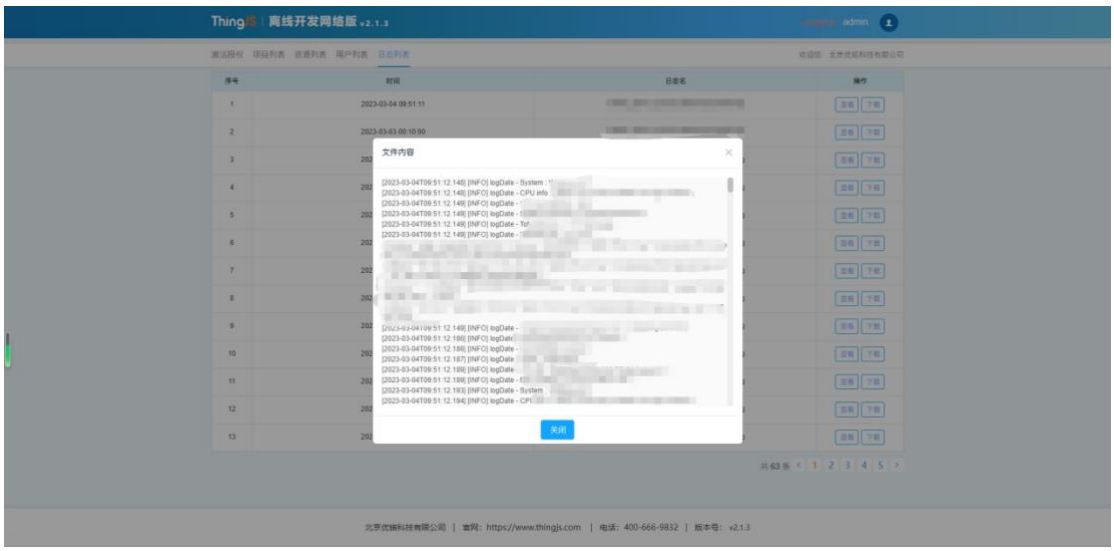
11.1 查看日志

访问离线开发网络版管理页面，点击日志列表页签，如下图所示：



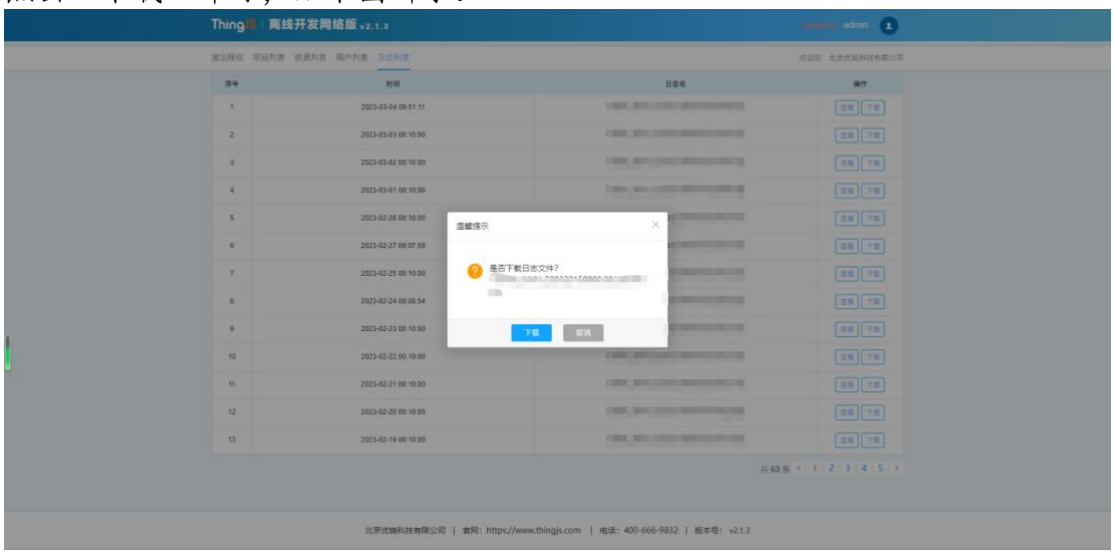
选择所需日期的日志文件，点击“查看”即可查看详细日志内容，如下图所示：



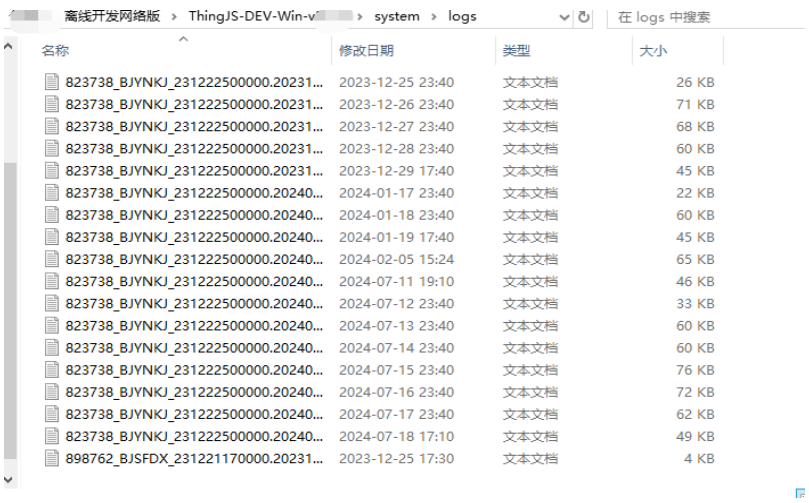


11.2 下载日志

访问离线开发网络版管理页面，点击日志列表页签，选择所需日期日志文件，点击“下载”即可，如下图所示：



日志文件也可在部署目录/system/logs 文件夹中获取，如下图所示：



12 常见问题（FAQ）

12.1 访问 3D 场景展示出错

答：用于部署的网络服务器一般不强调显示，故显卡较差，甚至无独立显卡。而 3D 场景显示对显卡要求较高，直接在服务器上 127.0.0.1 访问，服务器显卡不足以支持 3D 场景渲染展示，往往场景会显示出错。故可用有独立显卡的客户机浏览器访问该服务器场景，来验证 3D 场景显示是否正常。

12.2 Windows 环境下，start.exe 启动失败

答：常见的 start.exe 启动失败、发生闪退的情况是，当前 Windows 环境中部署服务的端口号已被其他服务占用，请检查端口占用情况，或为服务配置其他可使用的端口号。

12.3 Windows 环境下，start.exe 卡住，服务无反应

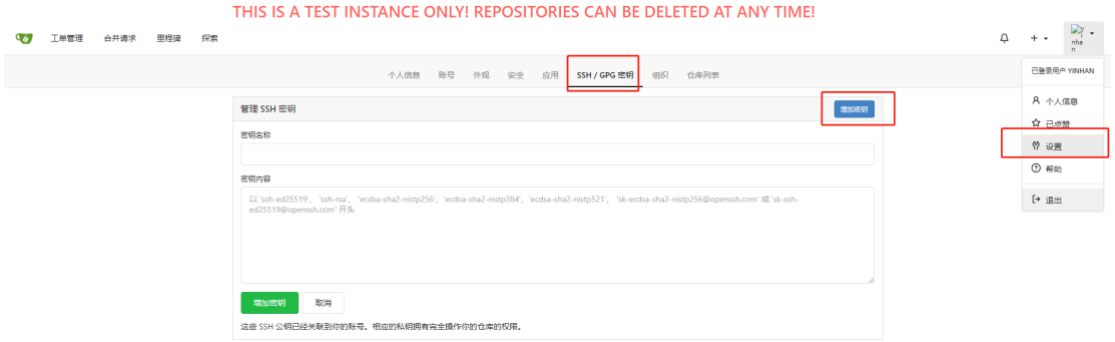
答：这种情况一般是鼠标不小心点击了服务程序窗口，导致服务程序暂停，需手动于服务程序窗口中敲击回车来解决。

12.4 上传包含 3DMax 模型的场景 tjs 包，预览场景，无法加载模型

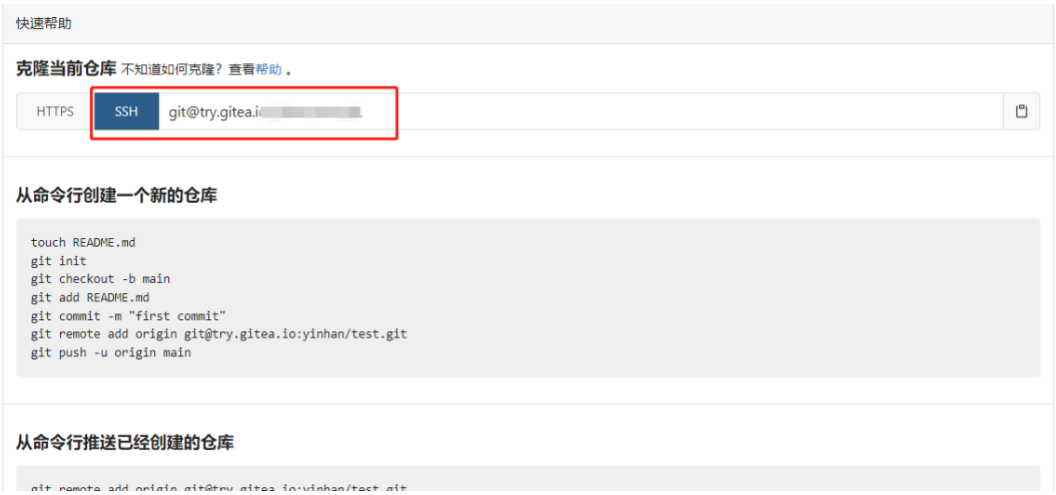
答：将包含 3DMax 模型的场景添加到在线开发的项目中，通过下载离线开发包的方式使用该场景，详见 7.2.5.2 章节内容。

12.5 Linux 环境下，添加项目 git 地址时报错

答：这种情况一般是创建的 git 仓库为私有仓库，有两种解决方式：
（1）通过在 git 地址中添加用户名和密码，如 http://yourname:password@git...
（2）通过配置 SSH 来实现正常访问。首先在 Linux 环境中配置公私钥，Linux 环境中可通过以下命令来进行配置：
生成密钥：ssh-keygen –t rsa –C “一般这里写邮箱”
切换到密钥文件夹：cd ~/.ssh
在本机配置私钥：ssh-add id_rsa
获取公钥： cat id_rsa.pub
在获取到公钥后，需要在 Git 服务器中配置公钥，这里以 Gitea 为例，配置如下图所示：



配置完成后，在 ThingJS 离线开发网络版中添加项目时，git 地址使用 SSH 地址即可，如下图所示：

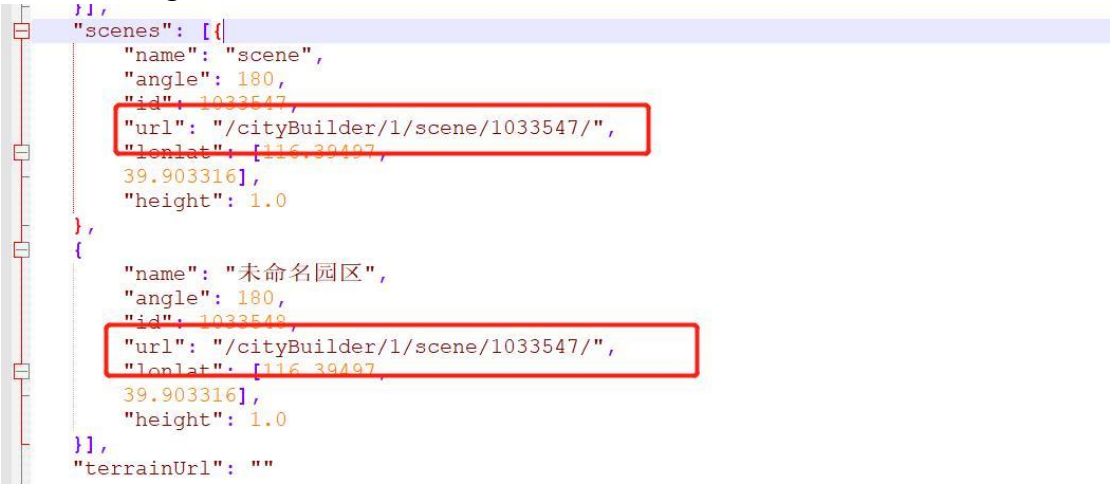


12.6 森城市携带园区上传离线开发包后预览场景加载不到如何配置

答：打开项目文件目录/cityBuilder 中查看对应地图场景目录下是否有“scene”目录，若没有“scene”目录，需到在线开发重新进行离线开发打包，如下图所示：

打开/cityBuilder 中对应地图场景目录下的“source.cfg”文件，找到配置场景的地方：

将“source.cfg”文件中的场景路径改为相对路径，如图所示：

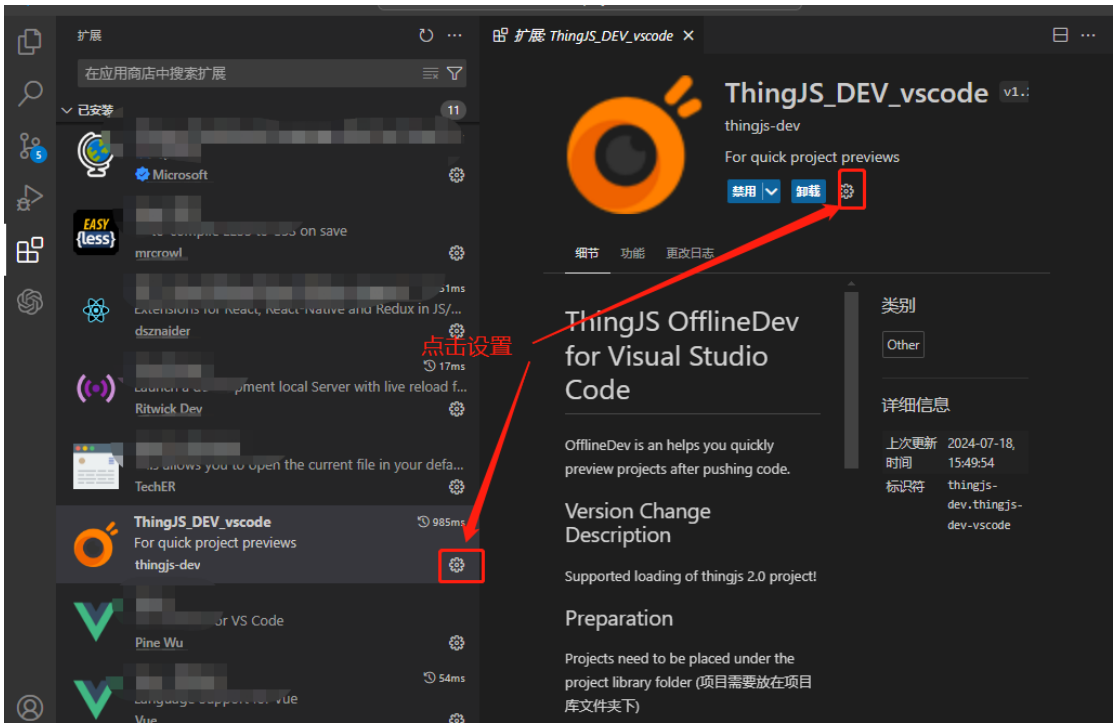


打开加载地图代码块的文件，添加如下配置即可：

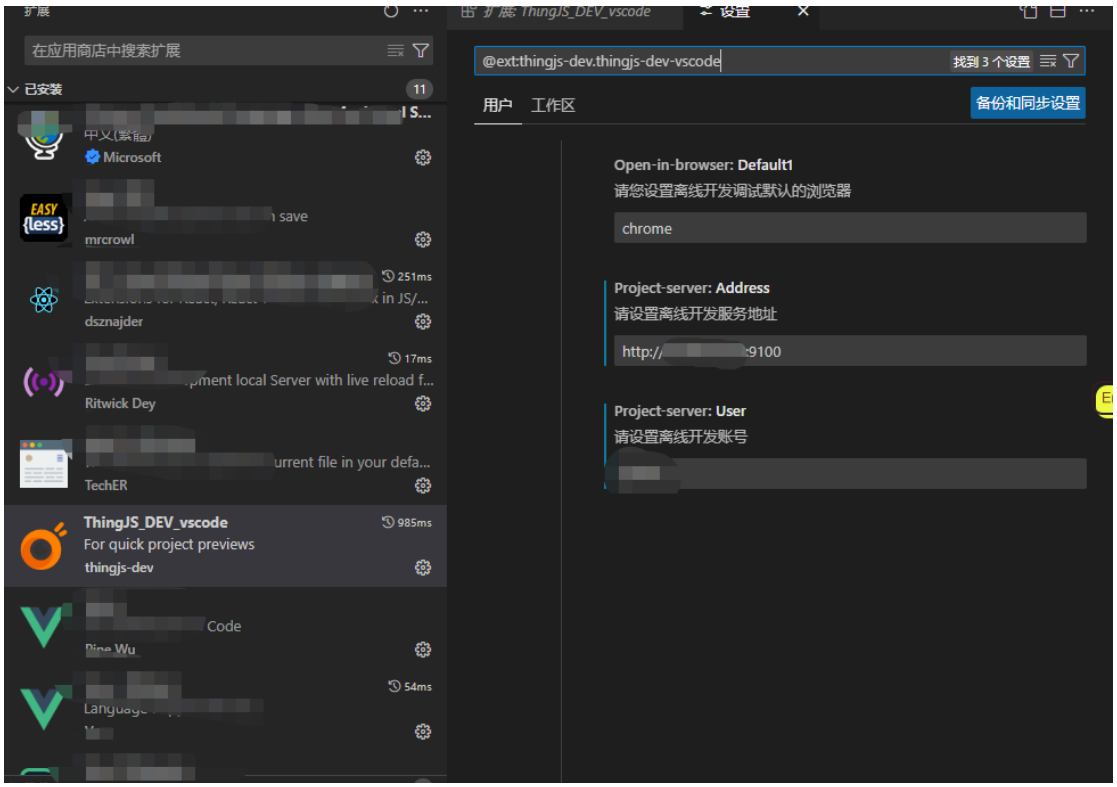
```
var app = new THING.App();
app.background = [0, 0, 0];
THING.Utils.dynamicLoad(['./source/uearth.min.js'], function () {
    app.create({
        type: 'Map',
        url: "./cityBuilder/1/map.bundle.json",
        resourceConfig: {
            resourcePrefix: './',
            loadDataFirst:true,
            isProxima:true,
            changeCampusLevel:false,
            externalConfigUrl:'./cityBuilder/1/source.cfg'
        },
        complete: function (event) {
            console.log(event.object.userLayers.length);
        }
    });
});
```

12.7 vscode 本地预览插件修改配置项

答：打开扩展插件，选择“ThingJS OfflineDev”，点击“设置”，如下图所示：



点击菜单中“扩展设置”，即可修改对应配置项，如下图所示：



12.8 森大屏资源离线加载时，数据对接错误

答：离线部署包中默认不具备森大屏接口服务请求功能，离线环境下若要使用数据对接服务，需要按以下步骤进行操作：

①获取 dataSource-server 数据对接服务（软件依赖 JRE V1.8~V1.10）：https://charts.thingjs.com/s-static/external/thing.charts.min/demo/dataSource-server.zip

②将 dataSource-server 数据对接服务 zip 包解压至部署环境中，启动服务：windows 环境中启动 dataSource-server/bin/start.bat，linux 环境中启动 dataSource-server/bin/start.sh。

③数据对接服务默认端口为 7000，可在 dataSource-server/conf/application.yml 文件中进行配置。

④修改离线部署项目中引用森大屏的配置，将数据源更改为离线数据对接服务地址（ws://ip:port/spray/websocket），如下图所示：

```
// 说明：在离线环境中使用森大屏，需使用以下脚本。
// 离线开发引入森大屏图表脚本
THING.Utils.dynamicLoad([
    '/static/release/thing.charts.min.js',
    '/static/plugins/vue.min.js'
],
async function () {
    const scenebundle = THING.CHARTS.Utils.loadBundle(
        './mock/设备站点可视化', // 前缀
        {
            container: document.getElementById('div2d') || '#div2d', // 挂载节点
            // 可选参数
            hide3D: true, // 隐藏场景内的3D
            hideCanvasBackground: true, // 画布背景色是否设置为透明的，默认false
            dataSource: {
                disable: true, // 是否禁用RestAPI数据，默认false
                http: {
                server: 'ws://ip:port/spray/websocket', // 数据源配置
                }
            }
        }
    )
    const instance = await scenebundle.waitForComplete() // 等待场景加载完成
    instance // ui场景实例
})
```

12.9 上传或新建项目时后台提示需要输入用户名和密码

答：解决在上传或新建项目的时后台提示需要输入用户密码（确保服务器和本地环境都对 git 进行了相关配置）有以下两种方式可以尝试：

（1） 使用 SSH 密钥：设置 SSH 密钥进行身份验证。首先生成一个 SSH 密钥对（公钥和私钥），可以使用命令“ssh-keygen –t rsa –b 4096 –C “your_email@example.com””，按照提示完成生成密钥过程，将生成的公钥文件（通常是‘~/.ssh/id_rsa.pub’）中的内容添加到您私有仓库的设置中。

（2） 使用 Git 凭据存储：可以使用命令“git config –global credential.helper store”以及设置用户名和邮箱 git config --global user.name 开启了权限的账号”，“git config --global user.email 开启了权限的账号对应的邮箱”

13 修订历史

版本	日期	更新内容描述
Rev. 1.1.1	2021/11/09	ThingJS 离线开发网络版功能说明。
Rev. 1.2.1	2021/11/26	更新场景、模型引用路径。
Rev. 1.2.2	2022/01/08	优化离线开发网络版服务接口。
Rev. 1.2.3	2022/02/22	优化服务接口。
Rev. 1.2.4	2022/03/17	修复 Git 服务相关命令。
Rev. 1.2.5	2022/05/07	更新部署包选择及常见问题说明。
Rev. 1.2.5	2022/06/23	新增日志列表说明模板。
Rev. 1.2.6	2022/07/01	新增支持森城市资源开发。
Rev. 2.0.1	2022/07/22	新增多用户管理，与之前版本不兼容。
Rev. 2.1.0	2022/08/22	新增本地开发项目预览功能。
Rev. 2.1.1	2022/12/23	优化服务接口。
Rev. 2.1.2	2023/02/06	支持森大屏、拓扑、地图的上传加载部署。
Rev. 2.1.2.1	2023/03/01	支持森图表、QChart、标识、效果、模型上传加载部署。
Rev. 2.1.3	2023/03/08	新增单资源预览、优化前端列表。
Rev. 2.1.4	2023/05/15	优化园区、城市、模型、图表等单资源上传。
Rev. 2.1.5	2023/07/10	优化图表组件等单资源上传； 支持导出项目离线开发包。
Rev. 2.1.6	2023/11/23	支持输出 ARM 架构离线部署包； 新增上传项目、优化新建项目。
Rev. 2.1.7	2023/12/21	解决森城市的上传解析问题； 更新默认携带的 ThingJS、uearth 版本。
Rev. 2.1.8	2024/07/18	支持 ThingJS 2.0 项目上传与开发； 支持 ThingJS 2.0 资源上传与使用； 升级离线部署包 v3.3.4； 升级插件预览 v1.2.7； 优化项目列表； 优化手册“配置 CPS 文件”章节； 修复部分已知问题。
Rev. 2.1.9	2025/02/20	升级安全检测； 修改部分漏洞。