1.本地运行neuroglancer

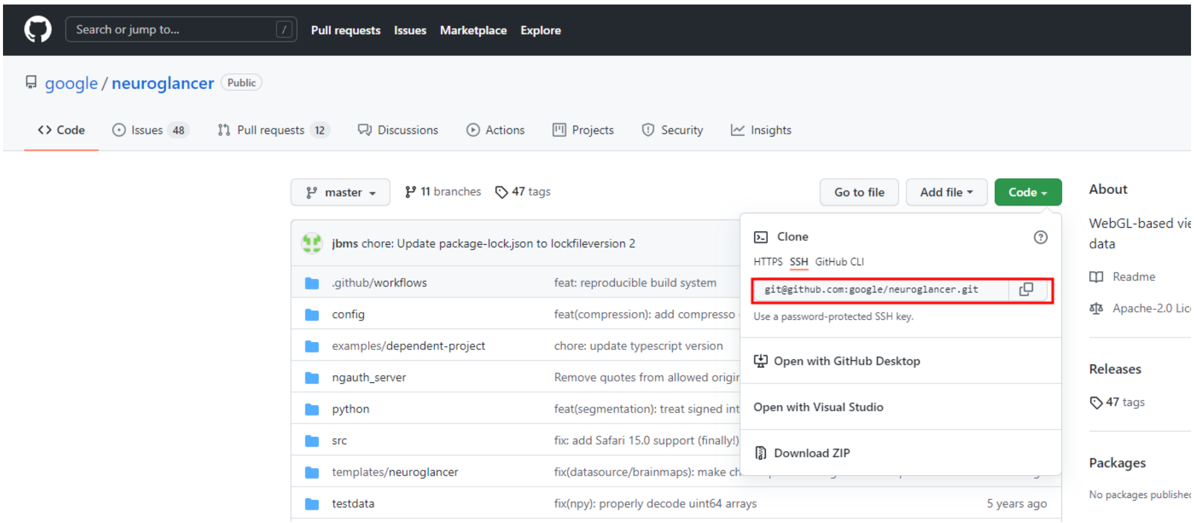
（1）从github上下载代码：

安装git（https://baijiahao.baidu.com/s?id=1601036689157983619&wfr=spider&for=pc），

创建GitHub账号，配置GitHub和本机电脑的联系（https://blog.csdn.net/weixin_43788566/article/details/84892542?ops_request_misc=%257B%2522request%255Fid%2522%253A%2522163394855316780265439001%2522%252C%2522scm%2522%253A%252220140713.130102334.pc%255Fall.%2522%257D&request_id=163394855316780265439001&biz_id=0&utm_medium=distribute.pc_search_result.none-task-blog-2~all~first_rank_ecpm_v1~rank_v31_ecpm-1-84892542.first_rank_v2_pc_rank_v29&utm_term=github%E7%9A%84%E6%9C%AC%E5%9C%B0%E9%85%8D%E7%BD%AE&spm=1018.2226.3001.4187）。

找到neuroglancer，　（https://github.com/google/neuroglancer）

找一文件夹，在命令行输入 **git clone git@github.com:google/neuroglancer.git**



git@github.com:google/neuroglancer.git

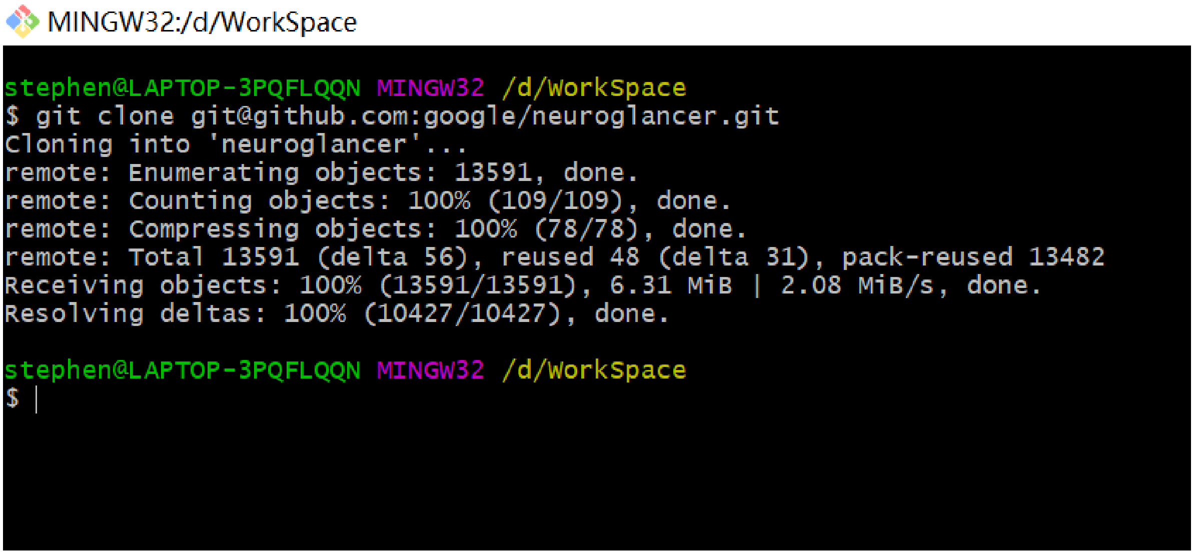
Use a password-protected SSH key.

jbms chore: Update package-lock.json to lockfileversion 2

.github/workflows	feat: reproducible build system	
config	feat(compression): add compresso	
examples/dependent-project	chore: update typescript version	
ngauth_server	Remove quotes from allowed origir	
python	feat(segmentation): treat signed int	
src	fix: add Safari 15.0 support (finally!)	
templates/neuroglancer	fix(datasource/brainmaps): make ch	
testdata	fix(npy): properly decode uint64 arrays	5 years ago

WebGL-based vie data

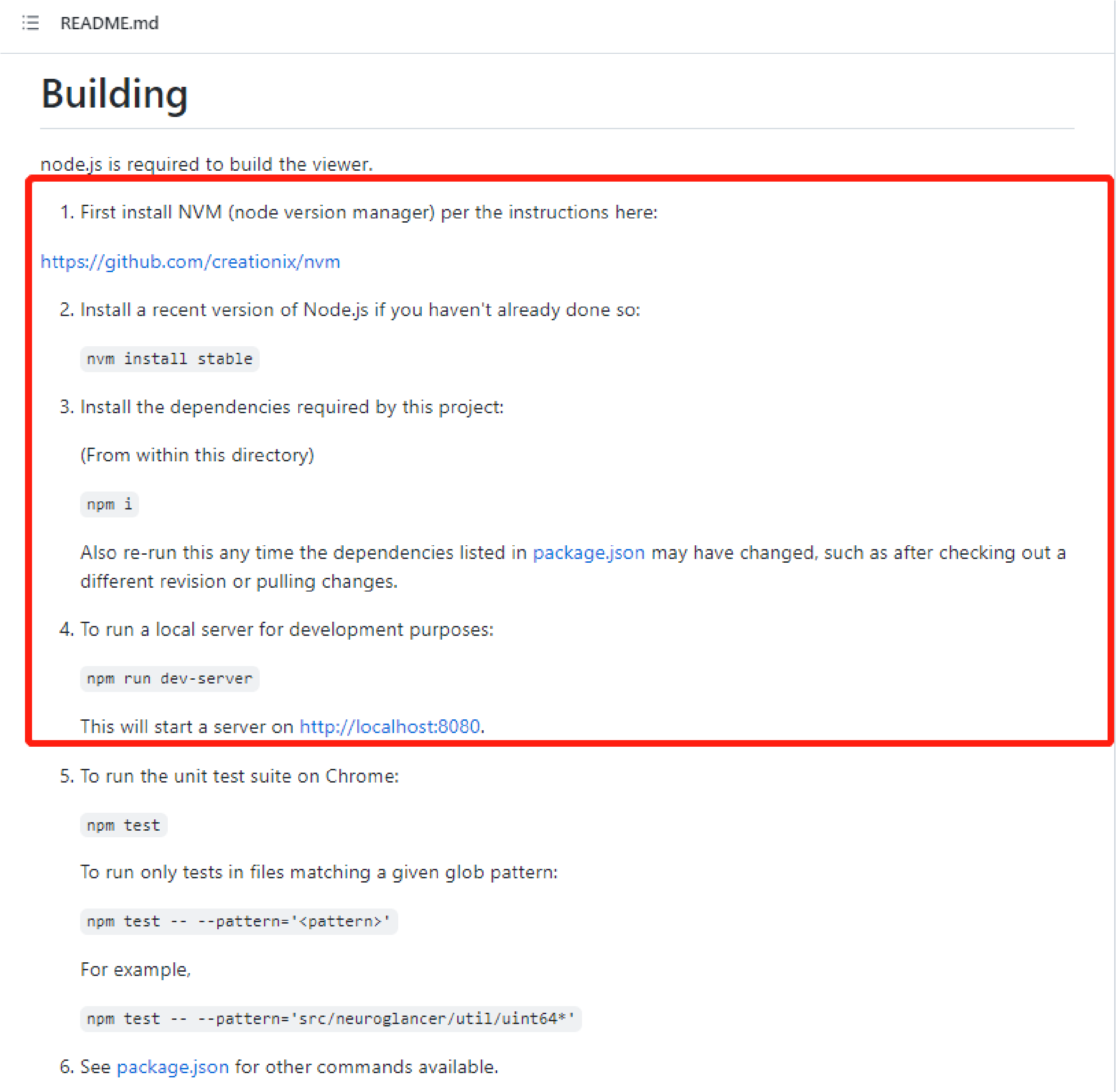
No packages publishe


MINGW32:/d/WorkSpace

```
stephen@LAPTOP-3PQFLQQN MINGW32 /d/WorkSpace
$ git clone git@github.com:google/neuroglancer.git
Cloning into 'neuroglancer'...
remote: Enumerating objects: 13591, done.
remote: Counting objects: 100% (109/109), done.
remote: Compressing objects: 100% (78/78), done.
remote: Total 13591 (delta 56), reused 48 (delta 31), pack-reused 13482
Receiving objects: 100% (13591/13591), 6.31 MiB | 2.08 MiB/s, done.
Resolving deltas: 100% (10427/10427), done.

stephen@LAPTOP-3PQFLQQN MINGW32 /d/WorkSpace
$ |
```

（2）本地运行neuroglancer

找到readme，按照步骤执行，执行完第四步，本地就可以运行起来了，建议下载个vscode，在其命令行上执行以下指令。



README.md

Building

node.js is required to build the viewer.

1. First install NVM (node version manager) per the instructions here:

https://github.com/creationix/nvm

2. Install a recent version of Node.js if you haven't already done so:

   `nvm install stable`

3. Install the dependencies required by this project:

   (From within this directory)

   `npm i`

   Also re-run this any time the dependencies listed in package.json may have changed, such as after checking out a different revision or pulling changes.

4. To run a local server for development purposes:

   `npm run dev-server`

   This will start a server on http://localhost:8080.

5. To run the unit test suite on Chrome:

   `npm test`

   To run only tests in files matching a given glob pattern:

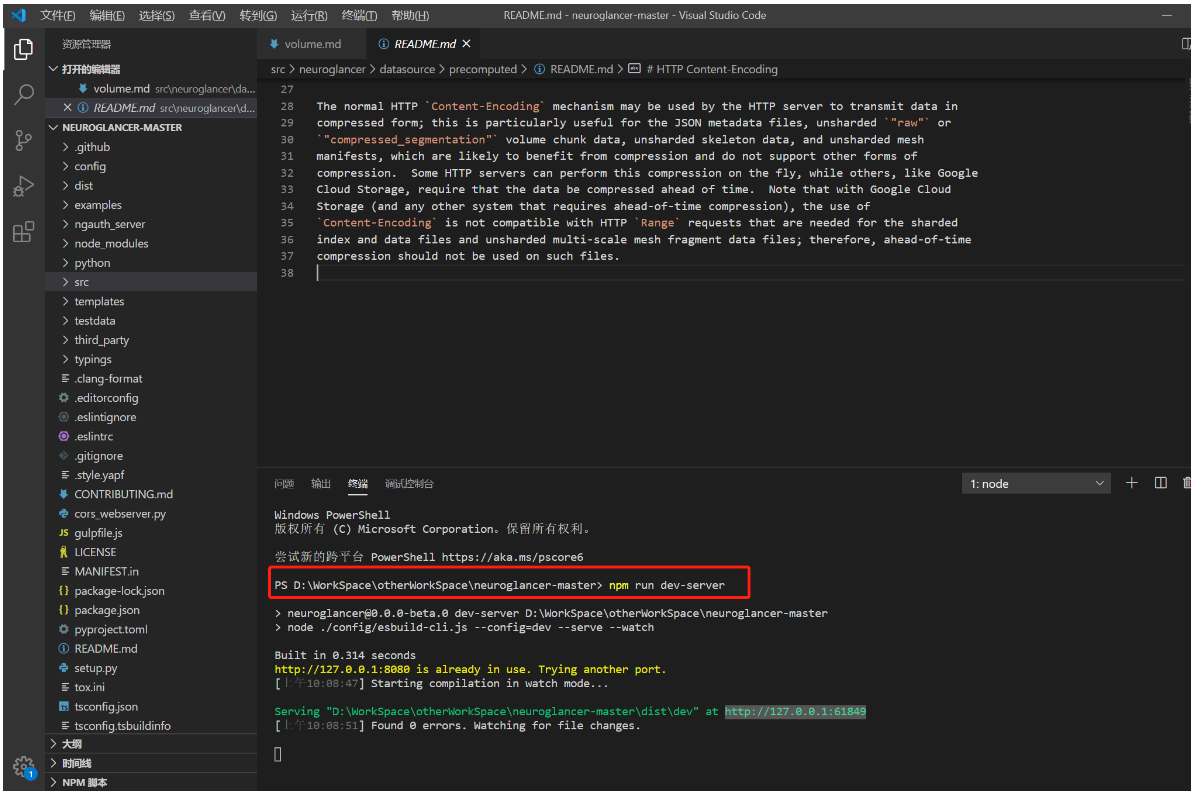
   `npm test -- --pattern='<pattern>'`

   For example,

   `npm test -- --pattern='src/neuroglancer/util/uint64*'`

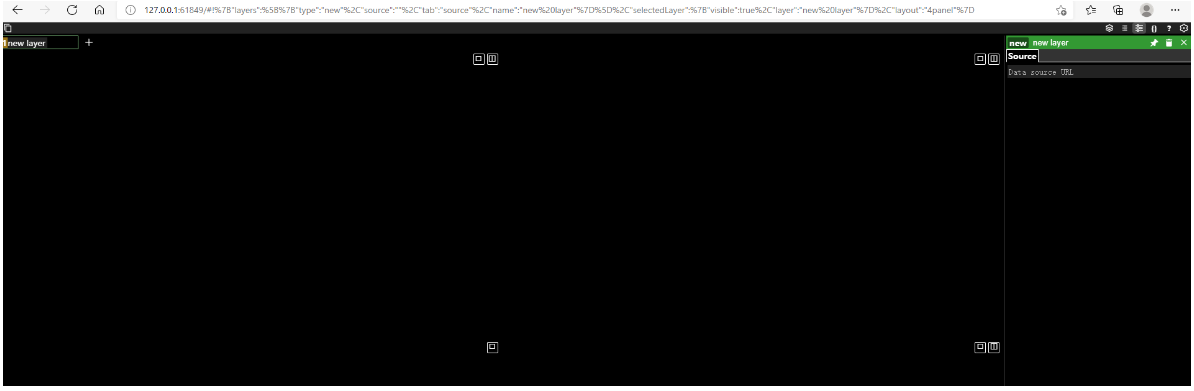
6. See package.json for other commands available.

最后一条运行指令执行完，在本地浏览器 输入 http://127.0.0.1:61849



```

The normal HTTP `Content-Encoding` mechanism may be used by the HTTP server to transmit data in
compressed form; this is particularly useful for the JSON metadata files, unsharded `"raw"` or
`"compressed_segmentation"` volume chunk data, unsharded skeleton data, and unsharded mesh
manifests, which are likely to benefit from compression and do not support other forms of
compression.  Some HTTP servers can perform this compression on the fly, while others, like Google
Cloud Storage, require that the data be compressed ahead of time.  Note that with Google Cloud
Storage (and any other system that requires ahead-of-time compression), the use of
`Content-Encoding` is not compatible with HTTP `Range` requests that are needed for the sharded
index and data files and unsharded multi-scale mesh fragment data files; therefore, ahead-of-time
compression should not be used on such files.

```

```
Windows PowerShell
版权所有 (C) Microsoft Corporation。保留所有权利。

尝试新的跨平台 PowerShell https://aka.ms/pscore6

PS D:\WorkSpace\otherWorkSpace\neuroglancer-master> npm run dev-server

> neuroglancer@0.0.0-beta.0 dev-server D:\WorkSpace\otherWorkSpace\neuroglancer-master
> node ./config/esbuild-cli.js --config=dev --serve --watch

Built in 0.314 seconds
http://127.0.0.1:8080 is already in use. Trying another port.
[上午10:08:47] Starting compilation in watch mode...

Serving "D:\WorkSpace\otherWorkSpace\neuroglancer-master\dist\dev" at http://127.0.0.1:61849
[上午10:08:51] Found 0 errors. Watching for file changes.

```



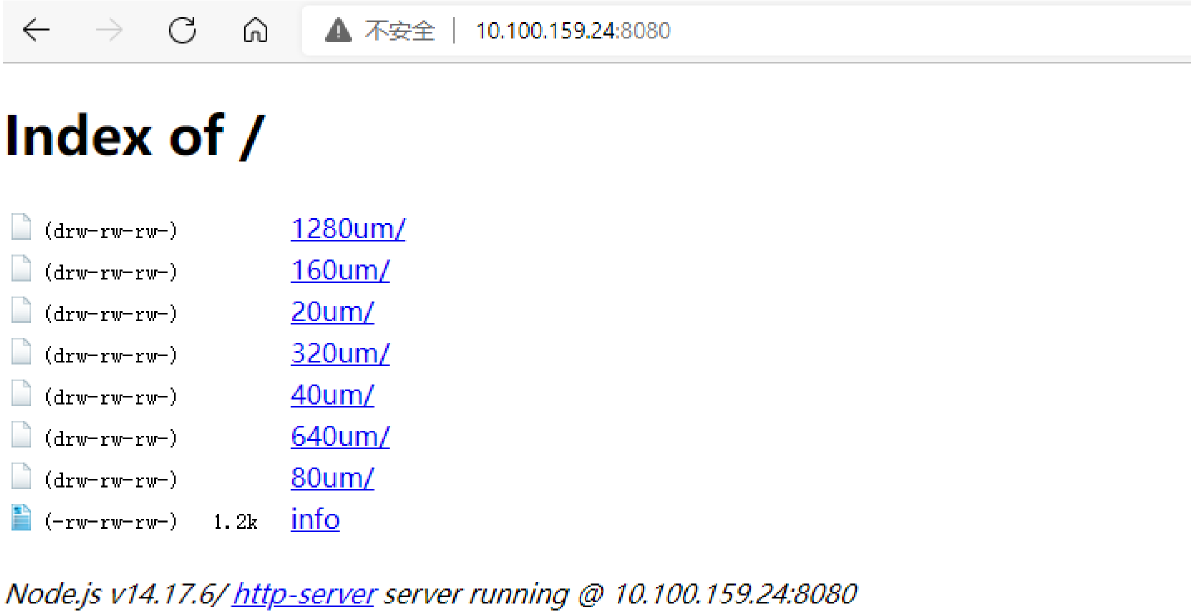
127.0.0.1:61849/#!%7B"layers":%5B%7B"type":"new"%2C"source":""%2C"tab":"source"%2C"name":"new%20layer"%7D%5D%2C"selectedLayer":%7B"visible":true%2C"layer":"new%20layer"%7D%2C"layout":"4panel"%7D

2.搭建本地文件服务器

经过多天调研，我选用的是http-server这款web服务器框架，

安装（https://blog.csdn.net/weixin_37856861/article/details/83274047?ops_request_misc=%257B%2522request%255Fid%2522%253A%2522163394966716780265411404%2522%252C%2522scm%2522%253A%252220140713.130102334.pc%255Fall.%2522%257D&request_id=163394966716780265411404&biz_id=0&utm_medium=distribute.pc_search_result.none-task-blog-2~all~first_rank_ecpm_v1~rank_v31_ecpm-6-83274047.first_rank_v2_pc_rank_v29&utm_term=windows+%E5%AE%89%E8%A3%85+http-server&spm=1018.2226.3001.4187）

安装成功后，文件服务器搭建成功，



不安全 | 10.100.159.24:8080

Index of /

```
(drw-rw-rw-)          1280um/
(drw-rw-rw-)          160um/
(drw-rw-rw-)          20um/
(drw-rw-rw-)          320um/
(drw-rw-rw-)          40um/
(drw-rw-rw-)          640um/
(drw-rw-rw-)          80um/
(-rw-rw-rw-)    1.2k  info
```

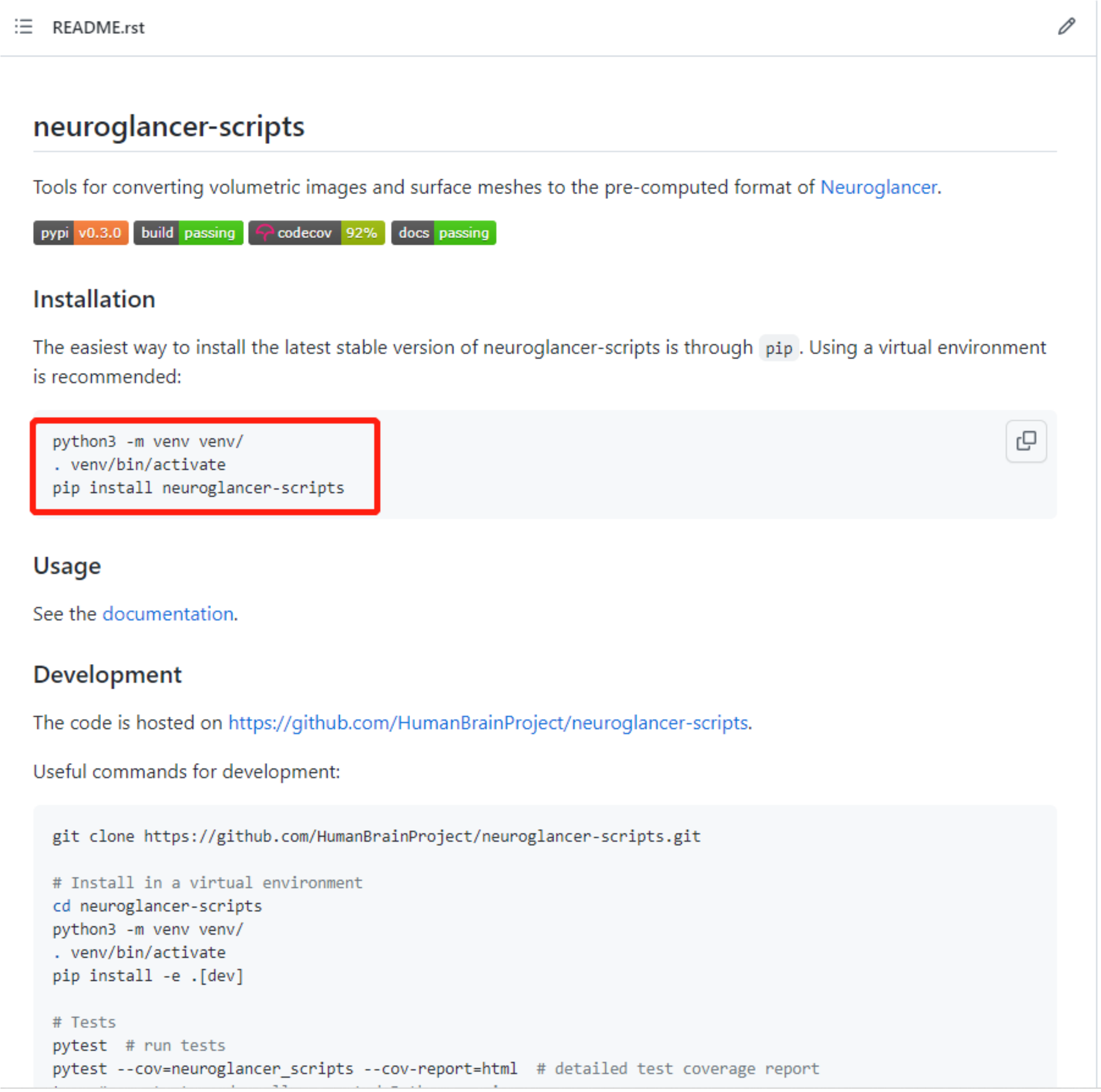
Node.js v14.17.6/ http-server server running @ 10.100.159.24:8080

3.转化数据为neuroglancer所需格式

（1）安装”转换脚本项目“的环境

我在GitHub上找到一转化格式的脚本（https://github.com/HumanBrainProject/neuroglancer-scripts），修改后既能生成neuroglancer所需数据源。

按照步骤安装此项目，运行此脚本前，需进入其自带的虚拟环境。note：**红框处命令必须执行**



neuroglancer-scripts

Tools for converting volumetric images and surface meshes to the pre-computed format of Neuroglancer.

Installation

The easiest way to install the latest stable version of neuroglancer-scripts is through `pip`. Using a virtual environment is recommended:

```
python3 -m venv venv/
. venv/bin/activate
pip install neuroglancer-scripts
```

Usage

See the documentation.

Development

The code is hosted on https://github.com/HumanBrainProject/neuroglancer-scripts.

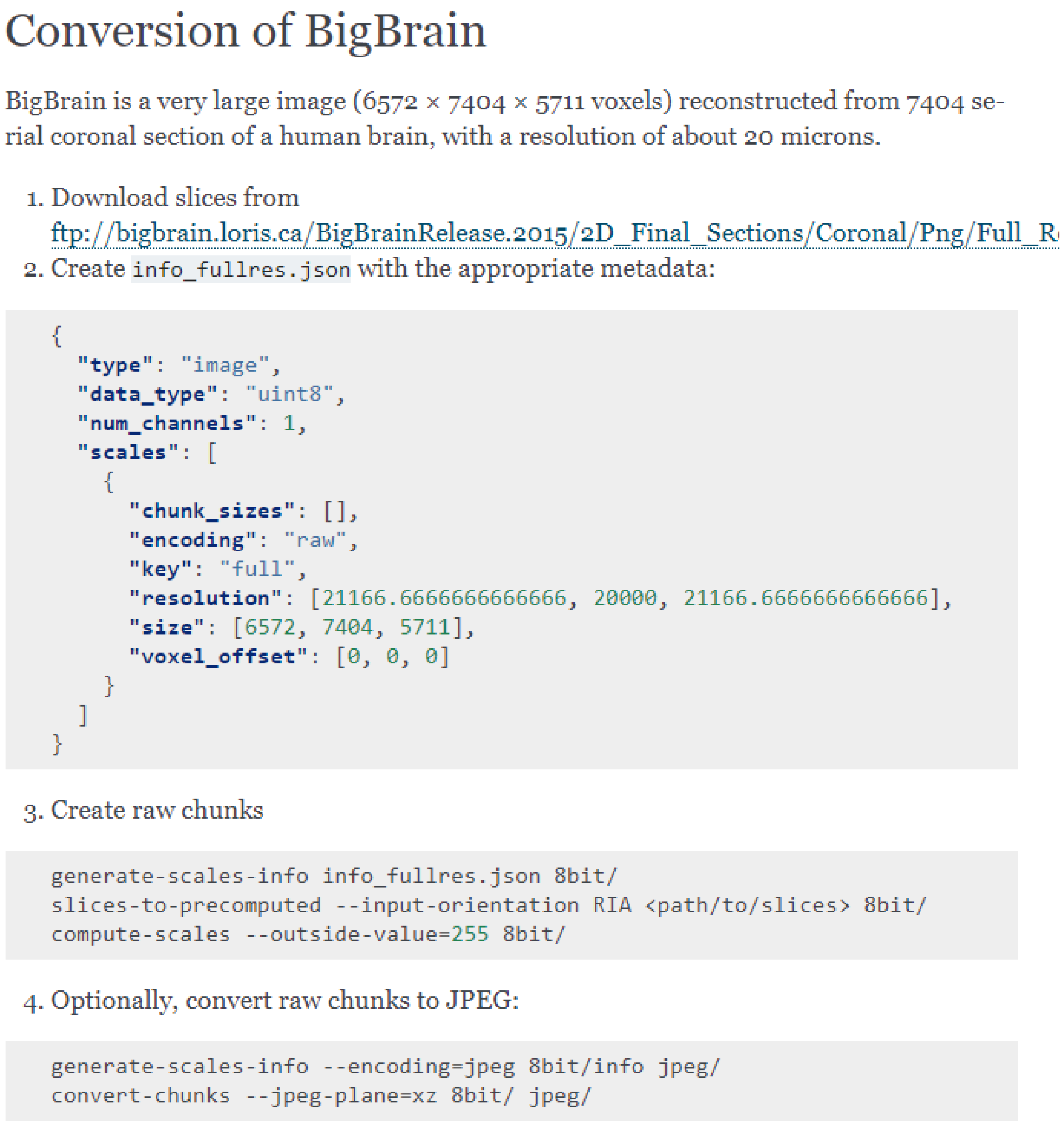
Useful commands for development:

```
git clone https://github.com/HumanBrainProject/neuroglancer-scripts.git

# Install in a virtual environment
cd neuroglancer-scripts
python3 -m venv venv/
. venv/bin/activate
pip install -e .[dev]

# Tests
pytest  # run tests
pytest --cov=neuroglancer_scripts --cov-report=html  # detailed test coverage report
```

（2）根据步骤转换格式

进入这个网站（https://neuroglancer-scripts.readthedocs.io/en/latest/examples.html）

执行以下4步（也可按照下面我详解的步骤操作）：



Conversion of BigBrain

BigBrain is a very large image (6572 × 7404 × 5711 voxels) reconstructed from 7404 serial coronal section of a human brain, with a resolution of about 20 microns.

1. Download slices from ftp://bigbrain.loris.ca/BigBrainRelease.2015/2D_Final_Sections/Coronal/Png/Full_R
2. Create `info_fullres.json` with the appropriate metadata:

```
{
  "type": "image",
  "data_type": "uint8",
  "num_channels": 1,
  "scales": [
    {
      "chunk_sizes": [],
      "encoding": "raw",
      "key": "full",
      "resolution": [21166.6666666666, 20000, 21166.6666666666],
      "size": [6572, 7404, 5711],
      "voxel_offset": [0, 0, 0]
    }
  ]
}
```

3. Create raw chunks

```
generate-scales-info info_fullres.json 8bit/
slices-to-precomputed --input-orientation RIA <path/to/slices> 8bit/
compute-scales --outside-value=255 8bit/
```

4. Optionally, convert raw chunks to JPEG:

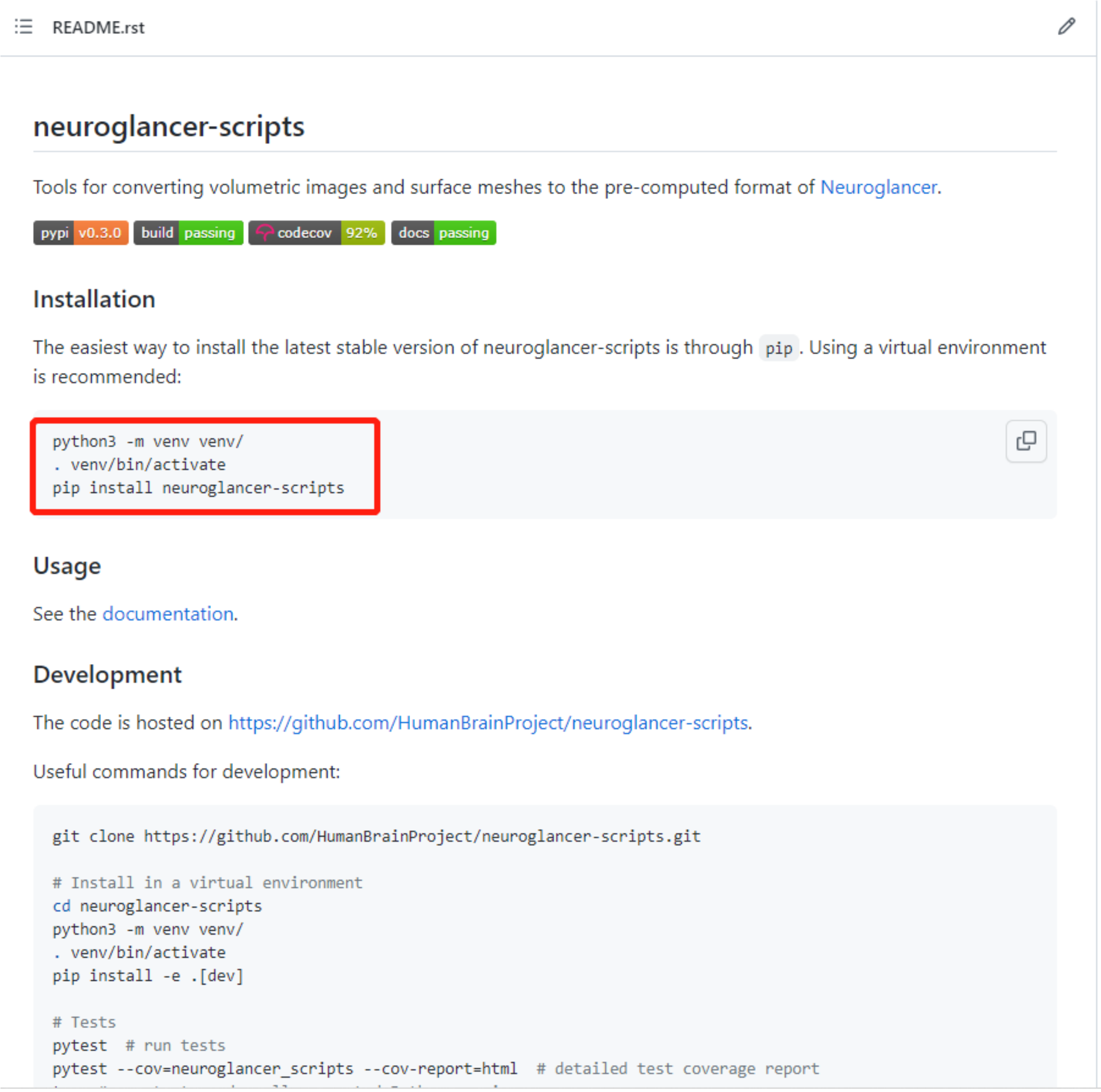
```
generate-scales-info --encoding=jpeg 8bit/info jpeg/
convert-chunks --jpeg-plane=xz 8bit/ jpeg/
```

详解步骤:

重新拉取我github上的neuroglancer-scripts项目（https://github.com/stephen-zhouyang/neuroglancer-scripts），我对其做了修改，建议用pycharm打开，：

①安装同上:

按照步骤安装此项目，运行此脚本前，需进入其自带的虚拟环境。note：**红框处命令必须执行**



neuroglancer-scripts

Tools for converting volumetric images and surface meshes to the pre-computed format of Neuroglancer.

Installation

The easiest way to install the latest stable version of neuroglancer-scripts is through `pip`. Using a virtual environment is recommended:

```
python3 -m venv venv/
. venv/bin/activate
pip install neuroglancer-scripts
```

Usage

See the documentation.

Development

The code is hosted on https://github.com/HumanBrainProject/neuroglancer-scripts.

Useful commands for development:

```
git clone https://github.com/HumanBrainProject/neuroglancer-scripts.git

# Install in a virtual environment
cd neuroglancer-scripts
python3 -m venv venv/
. venv/bin/activate
pip install -e .[dev]

# Tests
pytest  # run tests
pytest --cov=neuroglancer_scripts --cov-report=html  # detailed test coverage report
```

```
. venv/bin/activate
```

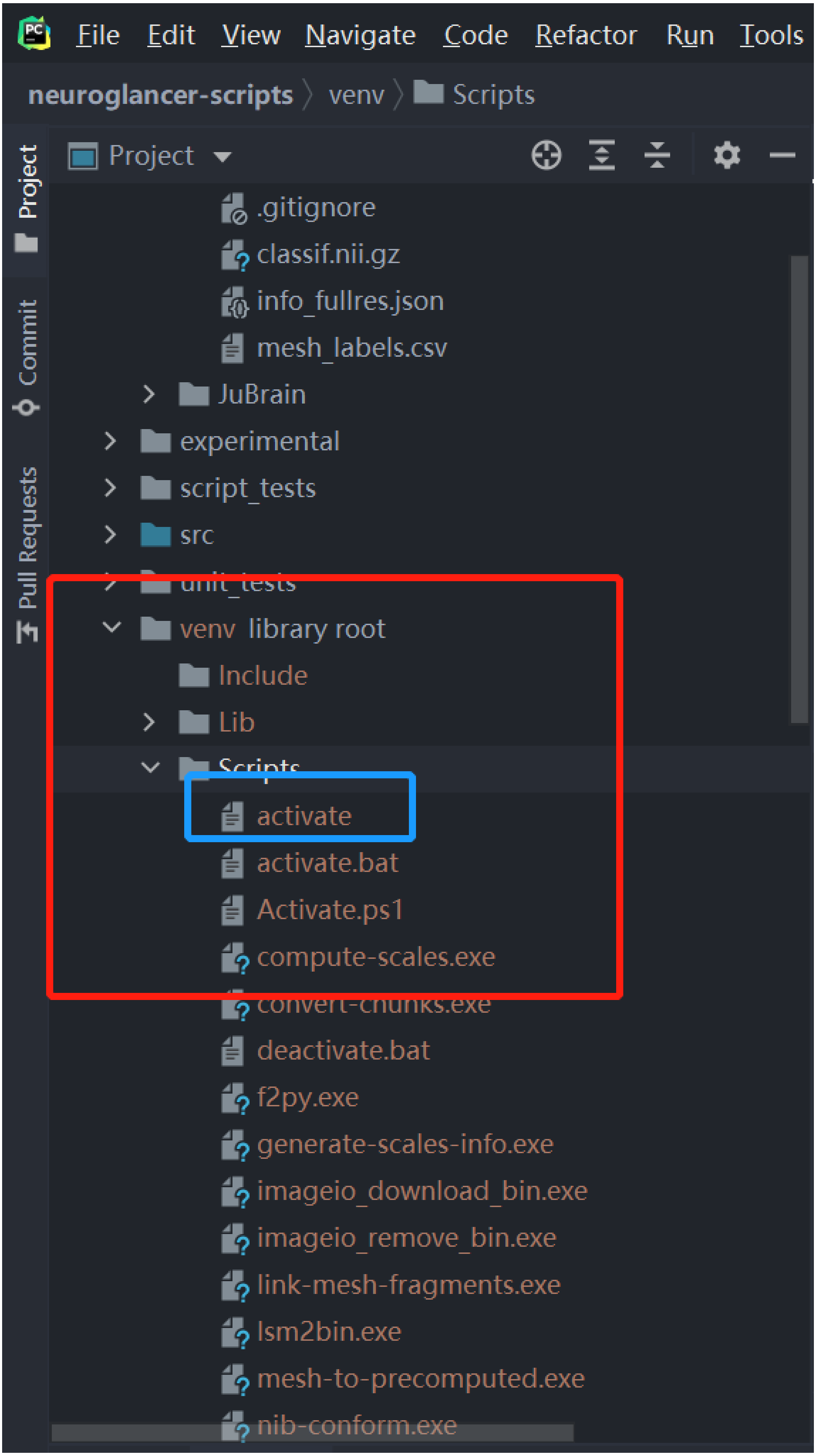
这条指令的含义是进入其自带虚拟环境，我们也可以这样做，当导入pycharm后，后自动生成venv library root，然后进入目录，执行

```
.\activate
```

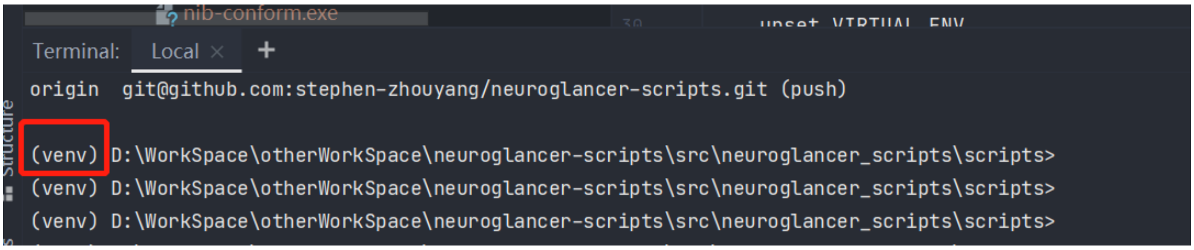
即进入其虚拟环境，再执行

```
pip install neuroglancer-scripts
```

虚拟环境如下：



File Edit View Navigate Code Refactor Run Tools
neuroglancer-scripts 〉 venv 〉 Scripts
Project
Commit
Pull Requests
Project
.gitignore
classif.nii.gz
info_fullres.json
mesh_labels.csv
JuBrain
experimental
script_tests
src
unit_tests
venv library root
Include
Lib
Scripts
activate
activate.bat
Activate.ps1
compute-scales.exe
convert-chunks.exe
deactivate.bat
f2py.exe
generate-scales-info.exe
imageio_download_bin.exe
imageio_remove_bin.exe
link-mesh-fragments.exe
lsm2bin.exe
mesh-to-precomputed.exe
nib-conform.exe

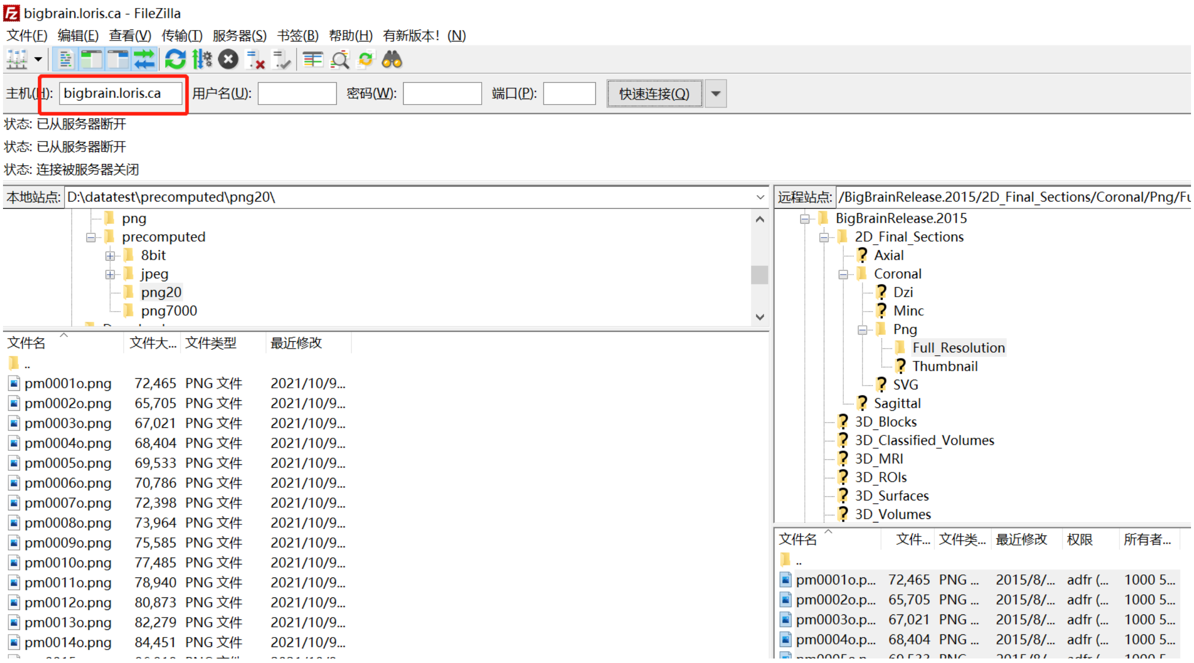


```
unset VIRTUAL_ENV
origin  git@github.com:stephen-zhouyang/neuroglancer-scripts.git (push)

(venv) D:\WorkSpace\otherWorkSpace\neuroglancer-scripts\src\neuroglancer_scripts\scripts>
(venv) D:\WorkSpace\otherWorkSpace\neuroglancer-scripts\src\neuroglancer_scripts\scripts>
(venv) D:\WorkSpace\otherWorkSpace\neuroglancer-scripts\src\neuroglancer_scripts\scripts>
```

②通过ftp工具filezilla（https://filezilla-project.org/download.php）下载源文件：

下载安装好后，红框输入主机名称，进入到以下目录：

```
ftp://bigbrain.loris.ca/BigBrainRelease.2015/2D_Final_Sections/Coronal/Png/Full_Resolution/
```



主机(H): bigbrain.loris.ca 用户名(U): 密码(W): 端口(P):

状态: 已从服务器断开
状态: 已从服务器断开
状态: 连接被服务器关闭

本地站点: D:\datatest\precomputed\png20\

远程站点: /BigBrainRelease.2015/2D_Final_Sections/Coronal/Png/Fu

文件名	文件大...	文件类型	最近修改
..			
pm0001o.png	72,465	PNG 文件	2021/10/9...
pm0002o.png	65,705	PNG 文件	2021/10/9...
pm0003o.png	67,021	PNG 文件	2021/10/9...
pm0004o.png	68,404	PNG 文件	2021/10/9...
pm0005o.png	69,533	PNG 文件	2021/10/9...
pm0006o.png	70,786	PNG 文件	2021/10/9...
pm0007o.png	72,398	PNG 文件	2021/10/9...
pm0008o.png	73,964	PNG 文件	2021/10/9...
pm0009o.png	75,585	PNG 文件	2021/10/9...
pm0010o.png	77,485	PNG 文件	2021/10/9...
pm0011o.png	78,940	PNG 文件	2021/10/9...
pm0012o.png	80,873	PNG 文件	2021/10/9...
pm0013o.png	82,279	PNG 文件	2021/10/9...
pm0014o.png	84,451	PNG 文件	2021/10/9...

文件名	文件...	文件类...	最近修改	权限	所有者...
..					
pm0001o.p...	72,465	PNG ...	2015/8/...	adfr (...	1000 5...
pm0002o.p...	65,705	PNG ...	2015/8/...	adfr (...	1000 5...
pm0003o.p...	67,021	PNG ...	2015/8/...	adfr (...	1000 5...
pm0004o.p...	68,404	PNG ...	2015/8/...	adfr (...	1000 5...

下载名称为pm0001-pm0020的20张图片d到本地。

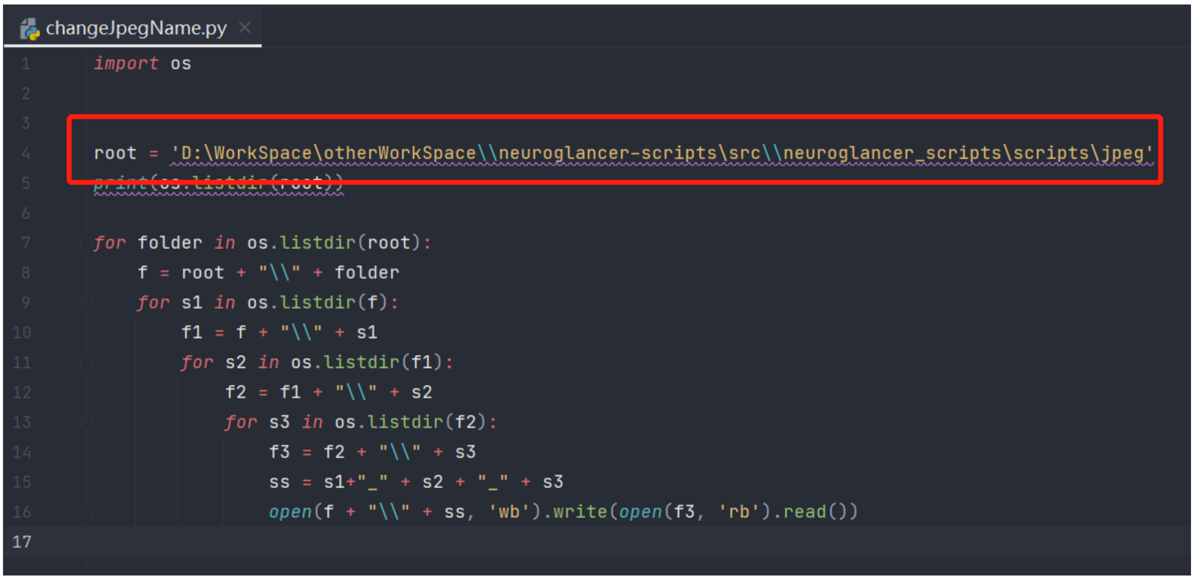
③然后依次执行以下脚本，记住更换20张图片所在目录，我的目录是 D:/datatest/precomputed/png20 8bit/，更换成你们自己的图片所在的目录。当执行python change8bitName.py和python changeJpegName.py时，也要更换其中的目录（见下图）。

生成raw型文件

```
generate-scales-info info_fullres.json 8bit/
slices-to-precomputed --input-orientation RIA D:/datatest/precomputed/png20
8bit/
compute-scales --outside-value=255 8bit/
python change8bitName.py
```

生成jpeg文件

```
generate-scales-info --encoding=jpeg 8bit/info jpeg/
convert-chunks --jpeg-plane=xz 8bit/ jpeg/
python changeJpegName.py
```



```python
import os


root = 'D:\WorkSpace\otherWorkSpace\\neuroglancer-scripts\src\\neuroglancer_scripts\scripts\jpeg'
print(os.listdir(root))

for folder in os.listdir(root):
    f = root + "\\" + folder
    for s1 in os.listdir(f):
        f1 = f + "\\" + s1
        for s2 in os.listdir(f1):
            f2 = f1 + "\\" + s2
            for s3 in os.listdir(f2):
                f3 = f2 + "\\" + s3
                ss = s1+"_" + s2 + "_" + s3
                open(f + "\\" + ss, 'wb').write(open(f3, 'rb').read())

```

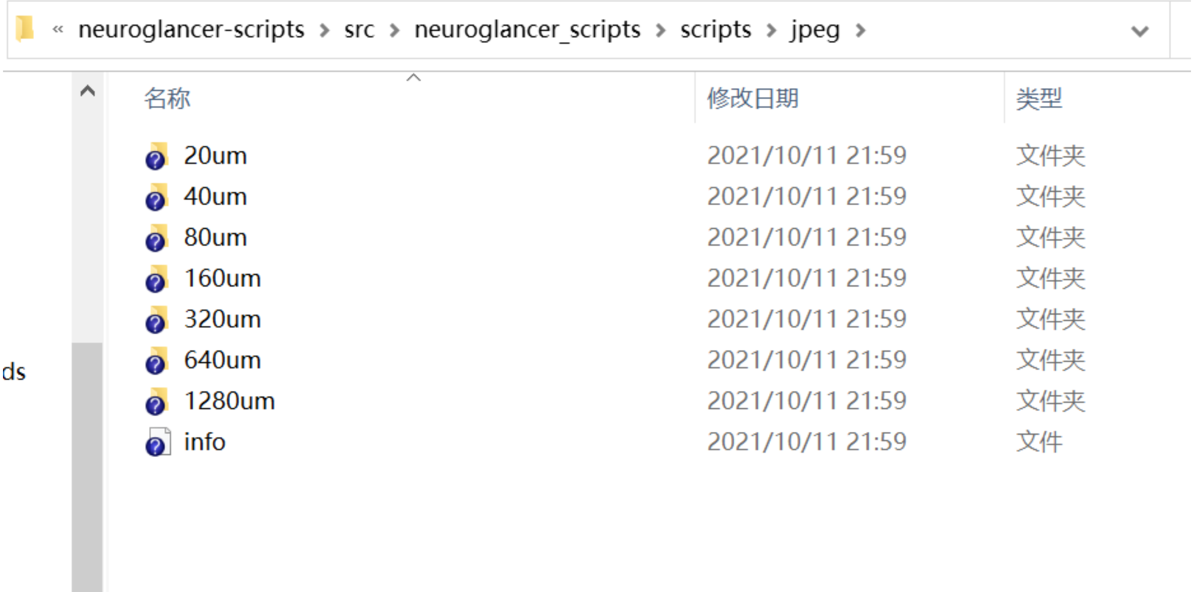
4.调试

1.启动http-server

在生成的jpeg目录下打开命令行，输入以下指令：

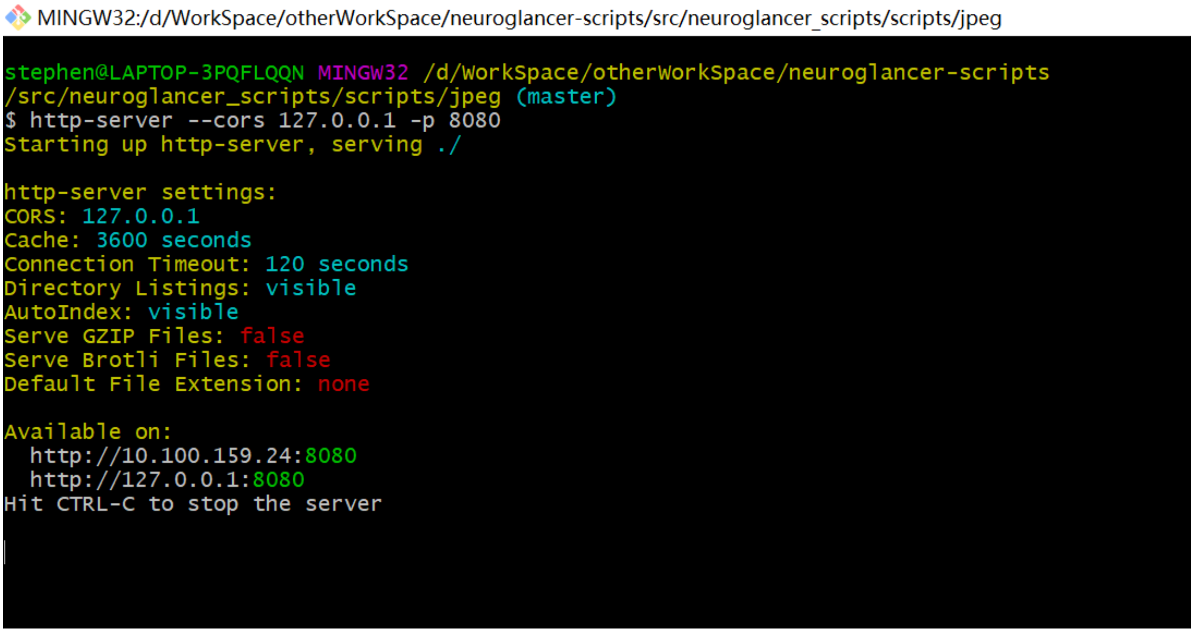
```
http-server --cors 127.0.0.1 -p 8080
```

note：--cors是解决跨域问题的。


« neuroglancer-scripts › src › neuroglancer_scripts › scripts › jpeg ›
ds

名称	修改日期	类型
20um	2021/10/11 21:59	文件夹
40um	2021/10/11 21:59	文件夹
80um	2021/10/11 21:59	文件夹
160um	2021/10/11 21:59	文件夹
320um	2021/10/11 21:59	文件夹
640um	2021/10/11 21:59	文件夹
1280um	2021/10/11 21:59	文件夹
info	2021/10/11 21:59	文件

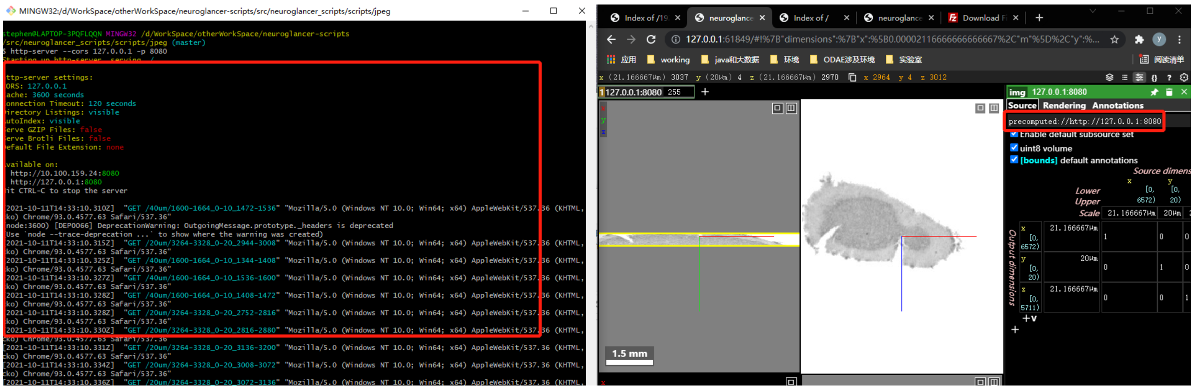
成功结果如下，



MINGW32:/d/WorkSpace/otherWorkSpace/neuroglancer-scripts/src/neuroglancer_scripts/scripts/jpeg

```
stephen@LAPTOP-3PQFLQQN MINGW32 /d/WorkSpace/otherWorkSpace/neuroglancer-scripts
/src/neuroglancer_scripts/scripts/jpeg (master)
$ http-server --cors 127.0.0.1 -p 8080
Starting up http-server, serving ./

http-server settings:
CORS: 127.0.0.1
Cache: 3600 seconds
Connection Timeout: 120 seconds
Directory Listings: visible
AutoIndex: visible
Serve GZIP Files: false
Serve Brotli Files: false
Default File Extension: none

Available on:
  http://10.100.159.24:8080
  http://127.0.0.1:8080
Hit CTRL-C to stop the server
```

在本地启动的neuroglance中输入 precomputed://http://127.0.0.1:8080 即完全成功



MINGW32:/d/WorkSpace/otherWorkSpace/neuroglancer-scripts/src/neuroglancer_scripts/scripts/jpeg

```
stephen@LAPTOP-3PQFLQQN MINGW32 /d/WorkSpace/otherWorkSpace/neuroglancer-scripts
/src/neuroglancer_scripts/scripts/jpeg (master)
$ http-server --cors 127.0.0.1 -p 8080
Starting up http-server, serving ./

ttp-server settings:
ORS: 127.0.0.1
ache: 3600 seconds
onnection Timeout: 120 seconds
irectory Listings: visible
utoIndex: visible
erve GZIP Files: false
erve Brotli Files: false
efault File Extension: none

vailable on:
 http://10.100.159.24:8080
 http://127.0.0.1:8080
it CTRL-C to stop the server

2021-10-11T14:33:10.310Z]  "GET /40um/1600-1664_0-10_1472-1536" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
ko) Chrome/93.0.4577.63 Safari/537.36"
(node:3600) [DEP0066] DeprecationWarning: OutgoingMessage.prototype._headers is deprecated
(Use `node --trace-deprecation ...` to show where the warning was created)
2021-10-11T14:33:10.315Z]  "GET /20um/3264-3328_0-20_2944-3008" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
ko) Chrome/93.0.4577.63 Safari/537.36"
2021-10-11T14:33:10.325Z]  "GET /40um/1600-1664_0-10_1344-1408" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
ko) Chrome/93.0.4577.63 Safari/537.36"
2021-10-11T14:33:10.327Z]  "GET /40um/1600-1664_0-10_1536-1600" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
ko) Chrome/93.0.4577.63 Safari/537.36"
2021-10-11T14:33:10.328Z]  "GET /40um/1600-1664_0-10_1408-1472" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
ko) Chrome/93.0.4577.63 Safari/537.36"
2021-10-11T14:33:10.328Z]  "GET /20um/3264-3328_0-20_2752-2816" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
ko) Chrome/93.0.4577.63 Safari/537.36"
2021-10-11T14:33:10.330Z]  "GET /20um/3264-3328_0-20_2816-2880" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
cko) Chrome/93.0.4577.63 Safari/537.36"
[2021-10-11T14:33:10.331Z]  "GET /20um/3264-3328_0-20_3136-3200" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
cko) Chrome/93.0.4577.63 Safari/537.36"
[2021-10-11T14:33:10.334Z]  "GET /20um/3264-3328_0-20_3008-3072" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
cko) Chrome/93.0.4577.63 Safari/537.36"
[2021-10-11T14:33:10.336Z]  "GET /20um/3264-3328_0-20_3072-3136" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
```

127.0.0.1:61849/#!%7B"dimensions":%7B"x":%5B0.000021166666666667%2C"m"%5D%2C"y":%...

x (21.166667㎛) 3037 y (20㎛) 4 z (21.166667㎛) 2970 x 2964 y 4 z 3012

img 127.0.0.1:8080

Source Rendering Annotations

precomputed://http://127.0.0.1:8080

☑ Enable default subsource set

☑ uint8 volume

☑ [bounds] default annotations