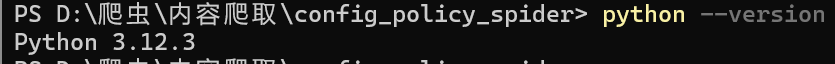
**介绍**

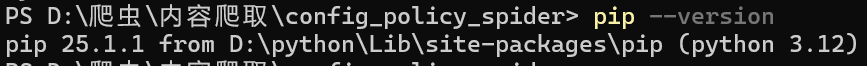
这个项目是一个基于Scrapy和FastAPI的政策爬虫管理系统，主要功能是通过Web前端配置爬虫参数，启动、暂停、恢复和终止爬虫实例，并实时查看爬虫输出。最后根据需要将爬取下来的csv文件存储到postgre数据库中。

**依赖安装**

需要python 3.7版本及以上（因为3.7以下版本不能保证python字典{}的顺序读取，可能会出bug）。可以在电脑PowerShell输入命令 python --version 来查看自己python版本。



Python一般自带pip。可以在PowerShell通过pip --version来查看自己有没有。后续的其他依赖都需要pip来安装。



使用pip安装以下依赖。在PowerShell输入以下命令：

pip install fastapi

pip install pydantic

pip install psutil

pip install uvicorn

pip install itemadapter

pip install scrapy

pip install scrapy-splash

pip install pandas

pip install psycopg2-binary

pip install psycopg2

如果安装失败，有可能是因为这些库是在国外，需要连上梯子，再次重新下载。

安装docker应用。

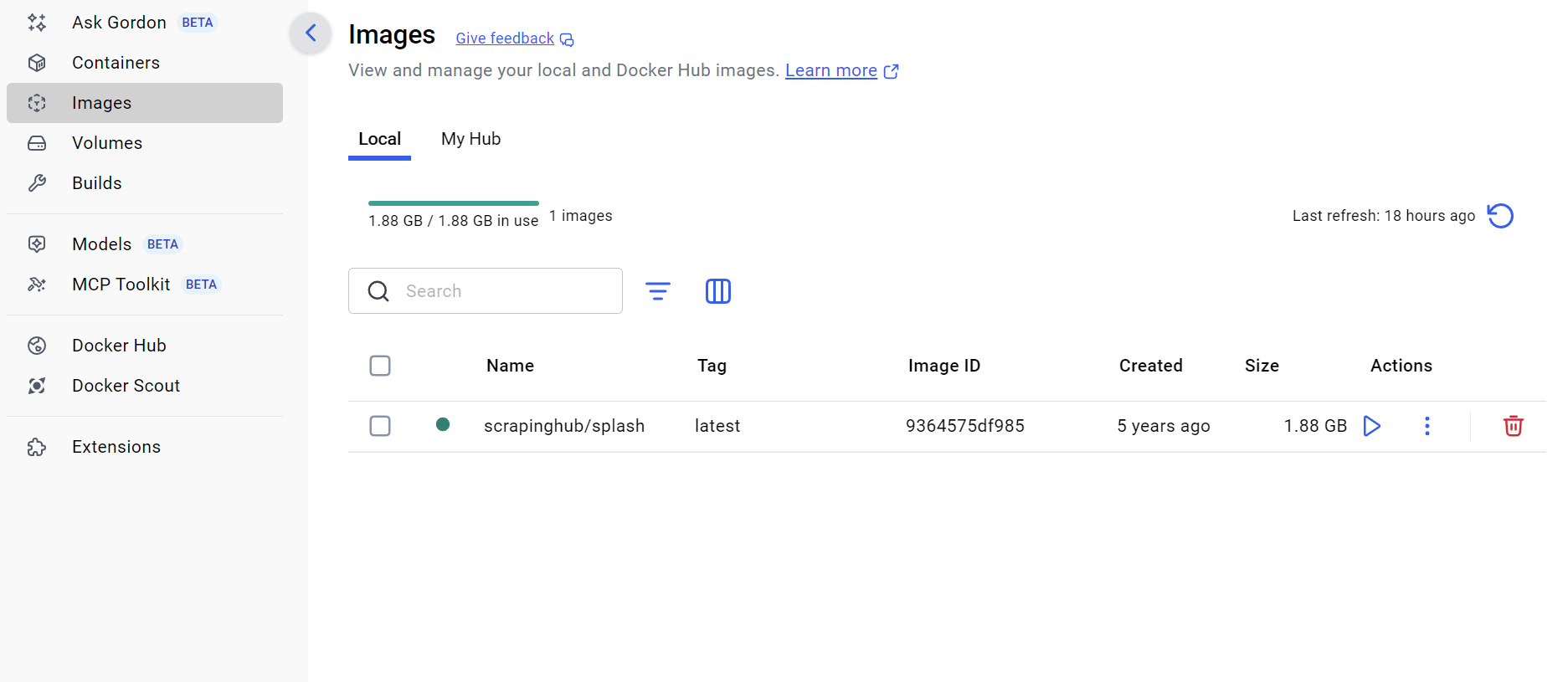
在打开docker应用的前提下，在控制台输入以下命令：

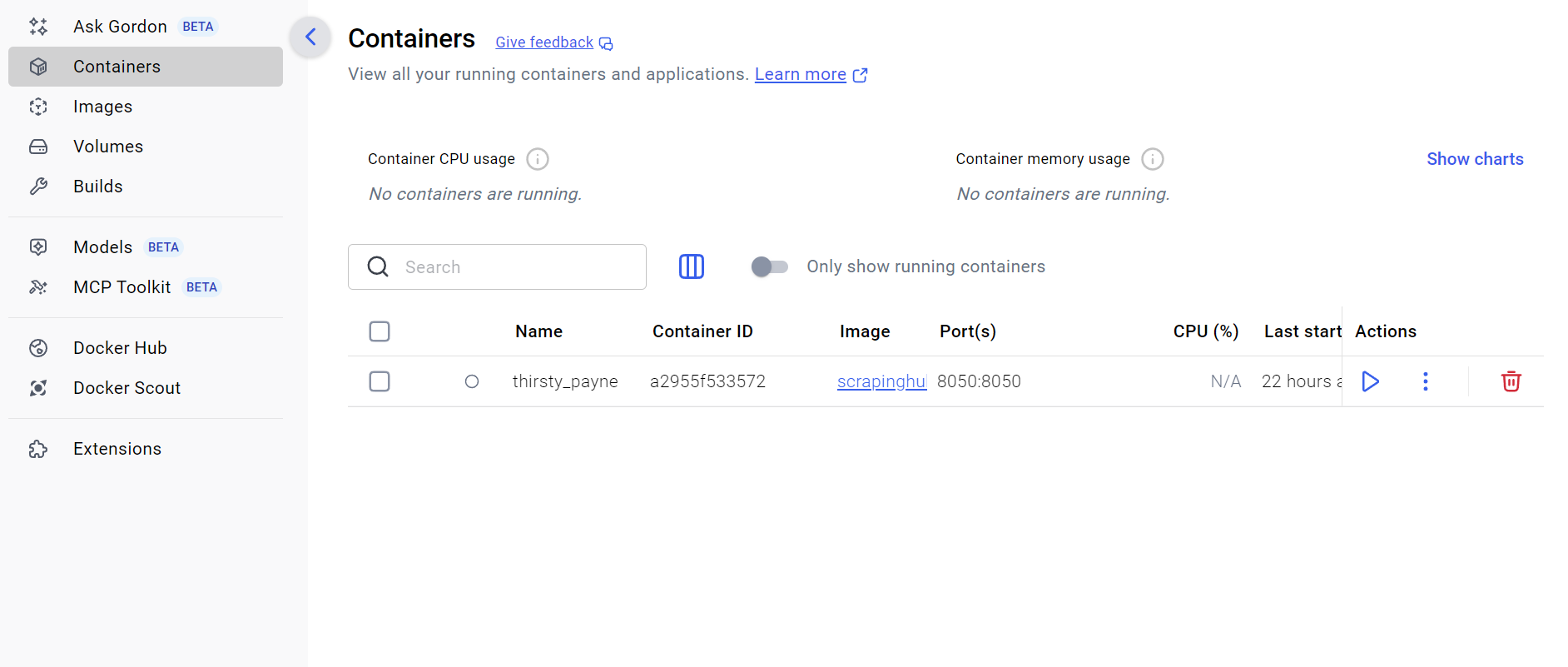
docker pull scrapinghub/splash

docker run -p 8050:8050 scrapinghub/splash

如果下载失败，有可能是没有连上梯子，连上后再试一遍。

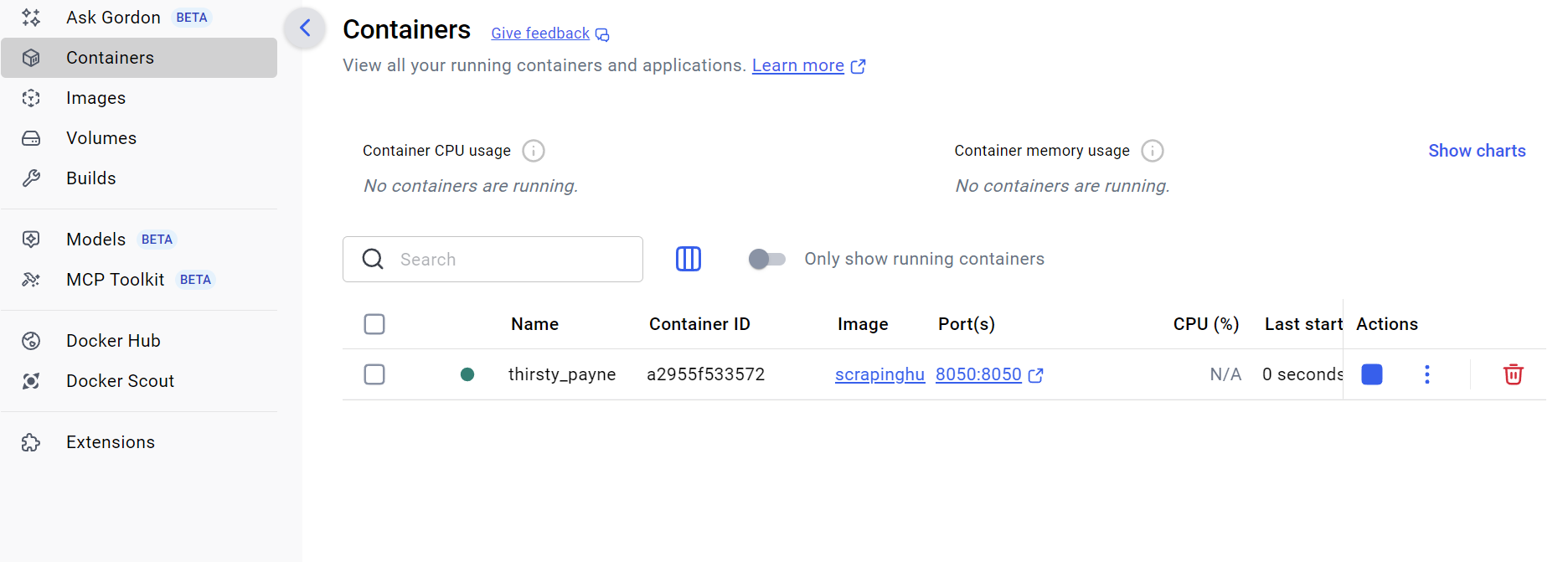
而后你会看到docker里有一个镜像（在images里）和一个容器（在containers里）。以后每次想要跑代码，要记得启动在containers的splash容器。



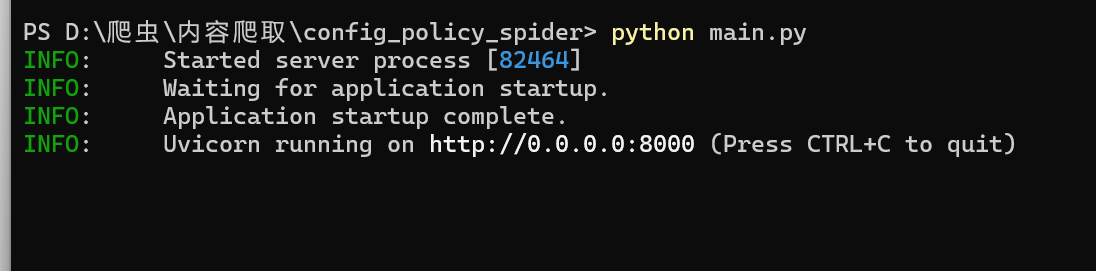


**使用教程**

先启动splash容器。（点击容器右边的三角形，启动后之后会变为方形，再次点击会结束容器变回三角形）



然后返回代码，在main.py的文件夹下打开PowerShell（目的是在正确的工作区下启动整个项目），使用命令python main.py启动后端服务。



双击打开main.html，应该来说是会打开一个网页的。这个网页即为前端。可以在可视化的界面来进行爬虫了。我会介绍各个模块的作用以及使用方法。

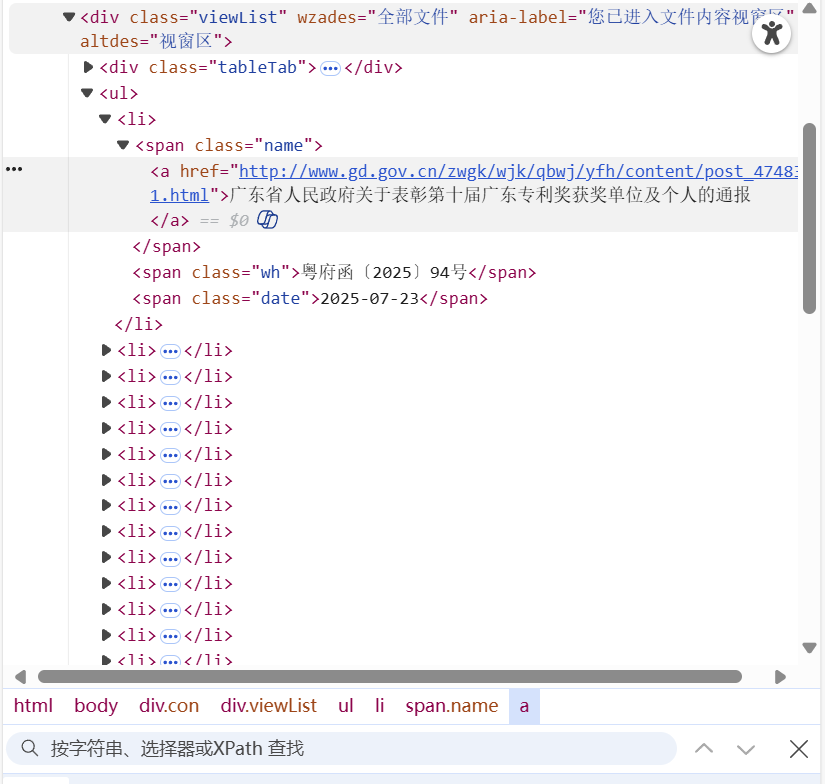
第一个部分是爬虫基本信息填写。首先，你可以给你想要爬虫的网页选择一个名称来作为标识。然后是网页的地址url。标题XPath对应的是政策列表的每个政策标题。链接XPath对应的是每个政策点击跳转的链接。正文XPath字段，是在进入链接后，对政策网页选择你想要爬取的部分（可以选择多个部分，右边绿色加号表示增加一个爬取字段）。然后因为政策列表不一定只有第一页，所以也需要下一页XPath来进行翻页。



关于XPath表达式常用语法，我填写在另一个文档中。（XPath 语法总结）。

当然，你也可以通过在浏览器对一个网页进行检查（鼠标右键出显示框，选择检查），然后在右侧键盘输入ctrl+f即可通过XPath表达式进行查找元素。从而以此练习。

我以广东省人民政府政策网页为例子。带你快速上手XPath表达式。



如你所见，我们想要的链接的路径为

div(class=”viewList”) ->ul->li->span(class=”name”)->href

因此，我们一层一层的查找。

首先，我们这个div(class=”viewList”)并不是在根标签。所以我们需要//来匹配所有div，但是这样范围太广了，所以我们加入限定查找，要求class一定等于“viewList”所以我们获得第一层//div[@class=”viewList”]

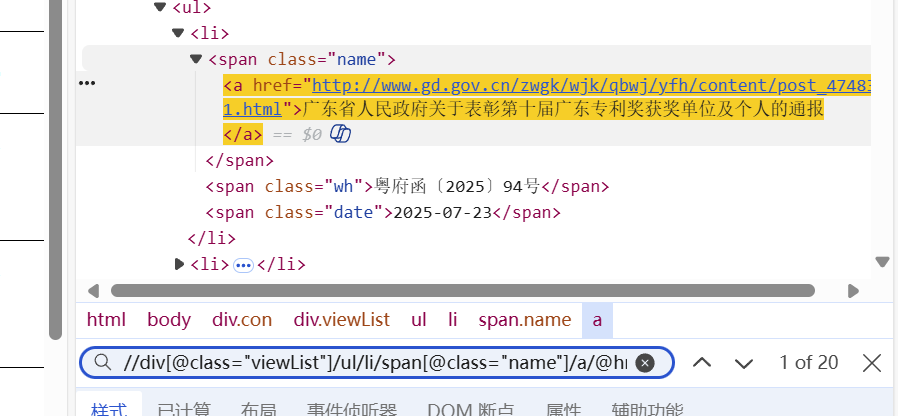
而后进入第二层，第二层有一个ul，因此可以不加限定。所以我们获得第二层//div[@class=”viewList”]/ul

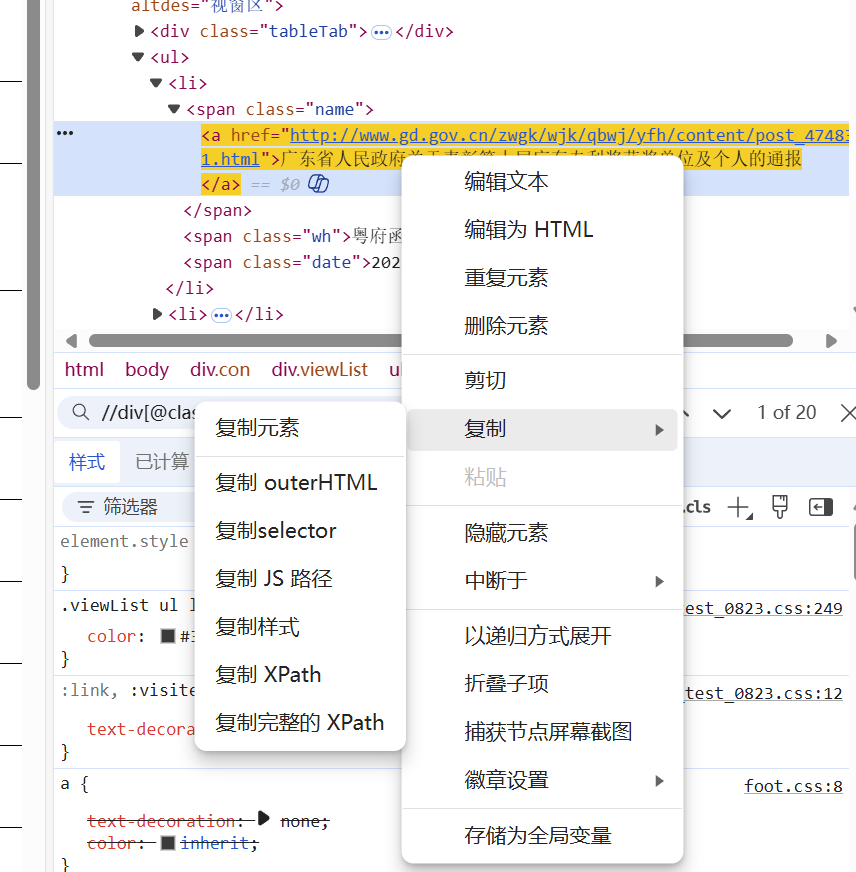
而后进入第三层，我们注意到li有很多个，其实是因为这是政策标题列表，每一个li对应一个政策，所以，我们在此处无形中表示了会选择列表中全部的政策来进行匹配并挖掘。所以第三层为//div[@class=”viewList”]/ul/li

第四层我们只需要class为“name”的span标签。所以第四层为//div[@class=”viewList”]/ul/li/span[@class=”name”]

第五层只有一个a标签，我们需要他的href属性。因此第五层为//div[@class=”viewList”]/ul/li/span[@class=”name”]/a/@href

最终实现查找到全部的20个政策链接

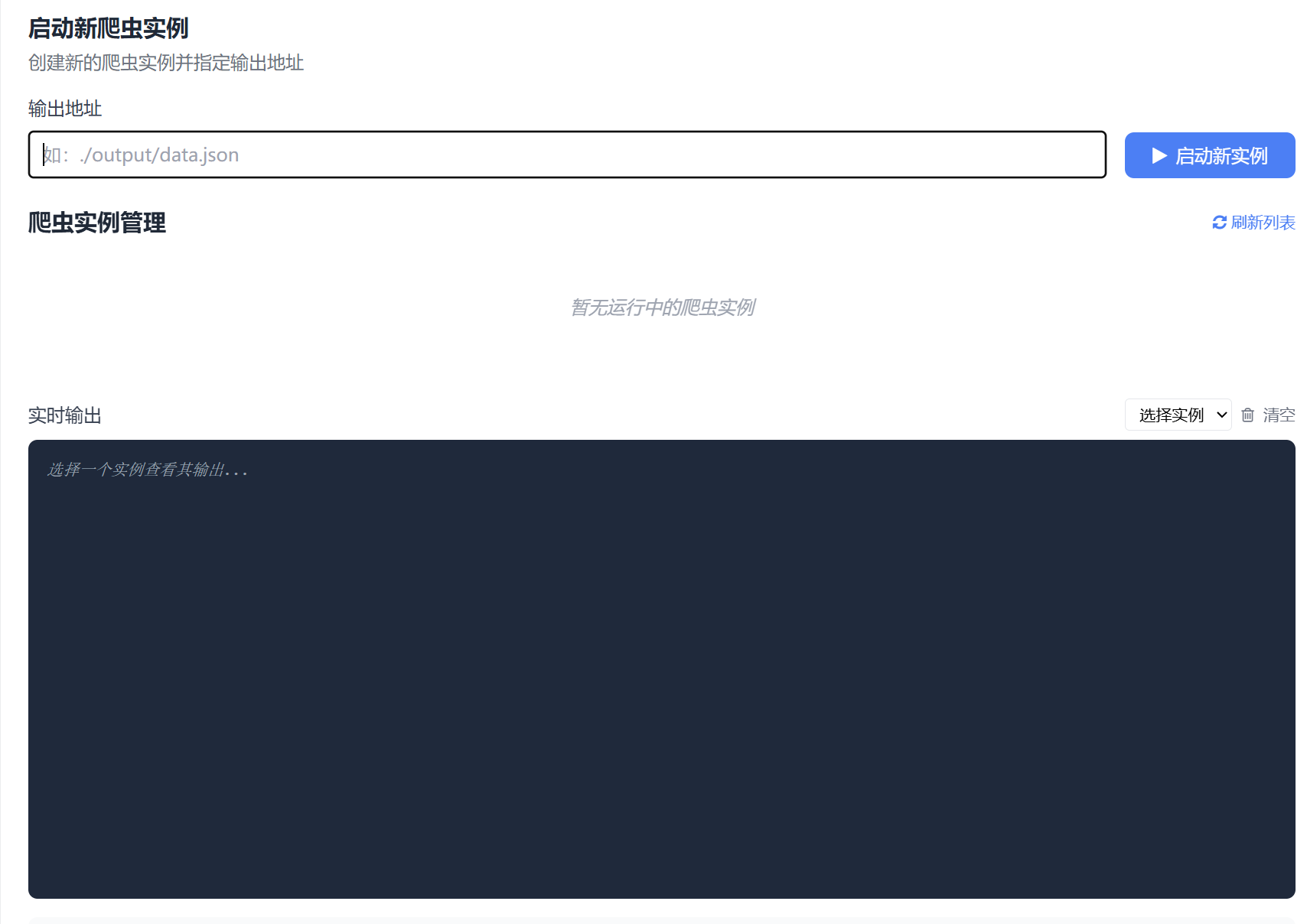


但是也可以不用这么麻烦，只需要鼠标右键点击你想要的元素，选择复制，然后选择复制完整的XPath即可获得。  


还可以对你爬取字段进行正则替换（点击蓝色方框【显示/隐藏正则替换规则】，即可出现正则替换界面），首先是标题进行替换，可以进行多个替换规则的配置（点击右侧绿色加号），他会将上一个替换后的文本交给第二个规则继续第二次替换。字段则根据上一步填写的正文文段设置相应的字段组，每一个字段组按顺序对正文字段进行替换。一个字段组内也可以设定多个替换规则，和标题替换相同，他会将第一个替换规则交给第二个替换规则继续替换。如果对文本不需要进行替换，则留空即可。



执行爬虫部分。选择一个输出地址，可以是相对地址，也可以是绝对地址。之后点击启动新实例，实例就会展示在爬虫实例管理列表中，你可以暂停，继续或者终止，也可以查看输出。你可以创建多个爬虫实例来爬取多个网站，但是每次爬一个新网站都要重新配置上面的规则部分，以及输出地址也要设置到新的一个文件地址。另提一嘴，终止一个爬虫实例要等一会才能最终终止，因为此时爬虫爬下来的信息还没全部存入到文件中，等到爬虫实例管理的对应实例消失，才算终止完毕。





**写入postgre数据库**

在上述步骤执行完成后，你应该获得一个csv文件，现在可以将你的csv文件导入到postgre数据库中了。填写下数据库名，用户名，密码，数据库地址以及端口，还有对应填写进入的表名，csv文件路径，即可自动导入。导入理论来说支持并发导入，但是效率依据表名而定，由于postgre数据库采用表级锁，因此当一个表正在被写入时，另一个进程导入同一个表会被阻塞。导入完成后，你可以利用一个软件叫pgAdmin4来查看表是否真的成功导入。写入逻辑是如果表名相同且列的数量正确，则能正确写入；如果表名相同，列数不同，则写入失败；如果表名不同，则创建一个对应表名的新表。

