

使用 docker 部署大型 scrapy 爬虫

Docter

问题：使用了 Scrapy-Client 成功将 Scrapy 项目部署到 Scrapyd 运行，前提是需要提前在服务器上安装好 Scrapyd 并运行 Scrapyd 服务，而这个过程比较麻烦。如果同时将一个 Scrapy 项目部署到 100 台服务器上，我们需要手动配置每台服务器的 Python 环境，更改 Scrapyd 配置吗？

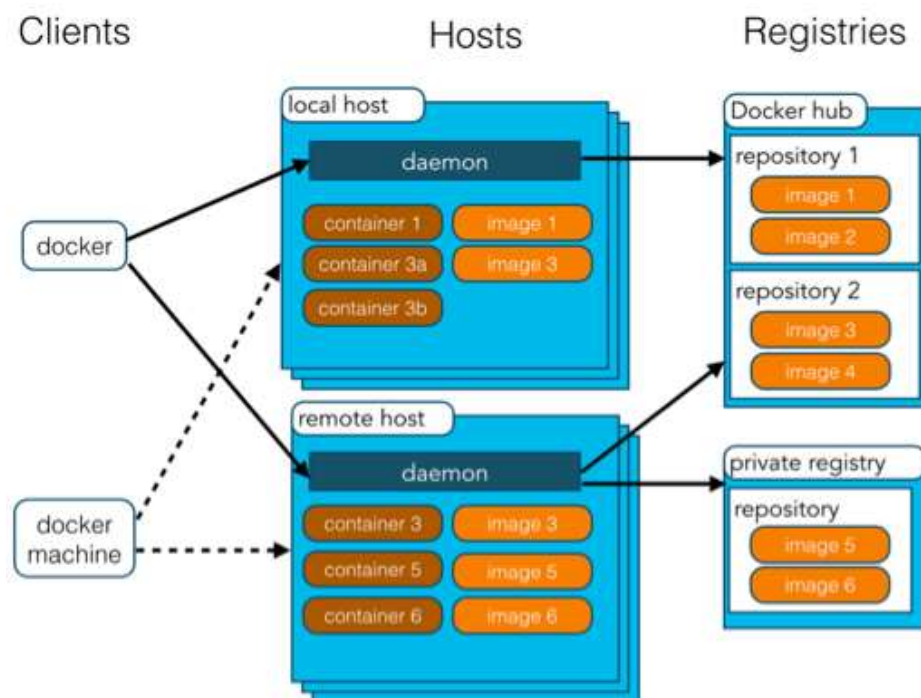
Docker 架构

Docker 是一个开源的应用容器引擎，让开发者可以打包他们的应用以及依赖包到一个可移植的容器中，然后发布到任何服务器上。容器使用沙箱机制，互相隔离，优势在于让各个部署在容器的里的应用互不影响，独立运行，提供更高的安全性。

Docker 使用客户端-服务器 (C/S) 架构模式，使用远程 API 来管理和创建 Docker 容器。

Docker 容器通过 Docker 镜像来创建。

容器与镜像的关系类似于面向对象编程中的对象与类。



Docker 镜像(Images): 用于创建 Docker 容器的模板。

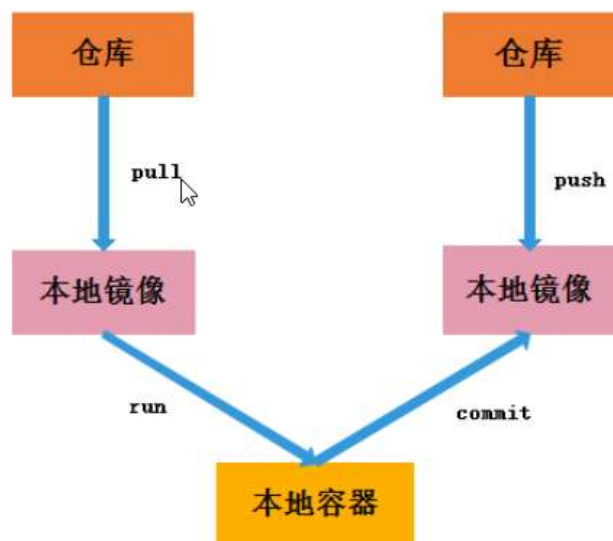
Docker 容器(Container): 独立运行的一个或一组应用。

Docker 客户端 (Client): 通过命令行或者其他工具使用 Docker API (https://docs.docker.com/reference/api/docker_remote_api) 与 Docker 的守护进程通信。

Docker 主机(Host): 一个物理或者虚拟的机器用于执行 Docker 守护进程和容器。

Docker 仓库(Registry): 用来保存镜像，可以理解为代码控制中的代码仓库。

Docker Hub(<https://hub.docker.com>): 提供了庞大的镜像集合供使用。



从仓库下载的镜像，变成容器，并在容器中制作项目，再将容器变成镜像，然后将镜像推送到仓库

Docker 的安装

Docker 是一种容器技术，可以将应用和环境等进行打包，应用可以被分发到任意一个支持 Docker 的环境中，极大方便了应用服务的部署。容器是完全使用沙箱机制，每个应用彼此互相隔离。

windows 下安装

从 Docker 官网下载 Docker for window，直接安装：

<https://docs.docker.com/docker-for-windows/install/>

Ubuntu Docker 安装

```
wget -qO- https://get.docker.com/ | sh
```

安装完成后有个提示：

```
+ sudo -E sh -c docker version
Client:
Version:      1.11.0
API version:  1.23
Go version:   go1.5.4
Git commit:   4dc5990
Built:        Wed Apr 13 18:38:59 2016
OS/Arch:      linux/amd64

Server:
Version:      1.11.0
API version:  1.23
Go version:   go1.5.4
Git commit:   4dc5990
Built:        Wed Apr 13 18:38:59 2016
OS/Arch:      linux/amd64

If you would like to use Docker as a non-root user, you should now consider
adding your user to the "docker" group with something like:

    sudo usermod -aG docker runoob

Remember that you will have to log out and back in for this to take effect!
```

启动 docker 后台服务

```
sudo service docker start
```

测试运行 hello-world

```
C:\Users\admin>docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
d1725b59e92d: Pull complete
Digest: sha256:0add3ace90ecb4adbf7777e9aacf18357296e799f81cab9fde470971e499788
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.
```

第一次使用该命令时，Docker 会从云端将 image（Docker 镜像）检出到本地。

该命令完成了一个简单的功能，但是 Docker 内核做的工作包括依赖镜像的检出，容器的创建和销毁等都已经有了。

scrapyd 对接对接 Docter

使用了 Scrapyd-Client 成功将 Scrapy 项目部署到 Scrapyd 运行, 前提是需要提前在服务器上安装好 Scrapyd 并运行 Scrapyd 服务, 而这个过程比较麻烦。如果同时将一个 Scrapy 项目部署到 100 台服务器上, 我们需要手动配置每台服务器的 Python 环境, 更改 Scrapyd 配置吗?

接下来, 我们就将 Scrapyd 打包制作成一个 Docker 镜像。

创建配置文件

新建一个 scrapyd.conf, 即 Scrapyd 的配置文件, 内容如下:

```
[scrapyd]
eggs_dir      = eggs
logs_dir      = logs
items_dir     =
jobs_to_keep  = 5
dbs_dir       = dbs
max_proc      = 0
max_proc_per_cpu = 10
finished_to_keep = 100
poll_interval = 5.0
bind_address  = 0.0.0.0
http_port     = 6800
debug         = off
runner        = scrapyd.runner
application   = scrapyd.app.application
launcher      = scrapyd.launcher.Launcher
webroot       = scrapyd.website.Root

[services]
schedule.json      = scrapyd.webservice.Schedule
cancel.json        = scrapyd.webservice.Cancel
addversion.json    = scrapyd.webservice.AddVersion
listprojects.json  = scrapyd.webservice.ListProjects
listversions.json  = scrapyd.webservice.ListVersions
listspiders.json   = scrapyd.webservice.ListSpiders
```

```
delproject.json    = scrapyd.webservice.DeleteProject
delversion.json    = scrapyd.webservice.DeleteVersion
listjobs.json      = scrapyd.webservice.ListJobs
daemonstatus.json = scrapyd.webservice.DaemonStatus
```

实际上是修改自官方文档的配置文件:

<https://scrapyd.readthedocs.io/en/stable/config.html#example-configuration-file> ,

其中修改的地方有两个。

- `max_proc_per_cpu=10` , 原本是 4, 即 CPU 单核最多运行 4 个 Scrapy 任务, 也就是说 1 核的主机最多同时只能运行 4 个 Scrapy 任务, 这里设置上限为 10, 也可以自行设置。
- `bind_address = 0.0.0.0` , 原本是 127.0.0.1, 不能公开访问, 这里修改为 0.0.0.0 即可解除此限制。

创建 requirements.txt

pip freeze >requirements.txt

```
(env2) E:\Python\代码\scrapy\scrapyd_test>pip freeze >requirements.txt
```

主要的内容:

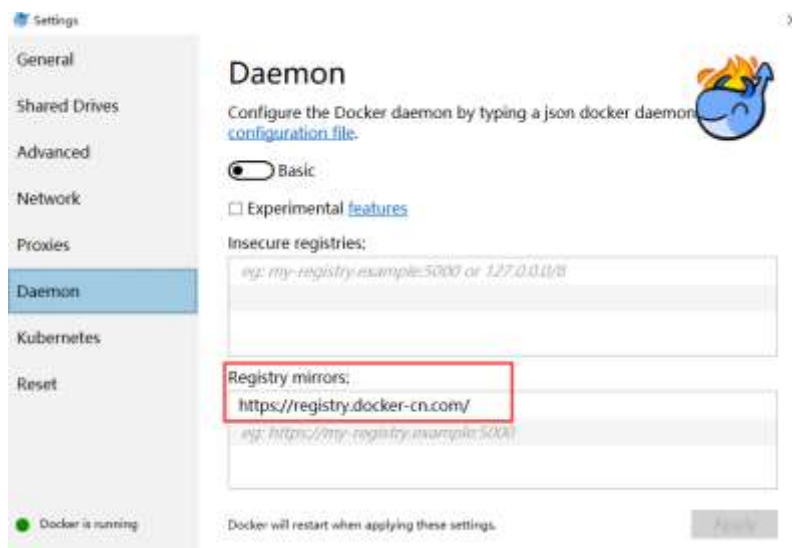
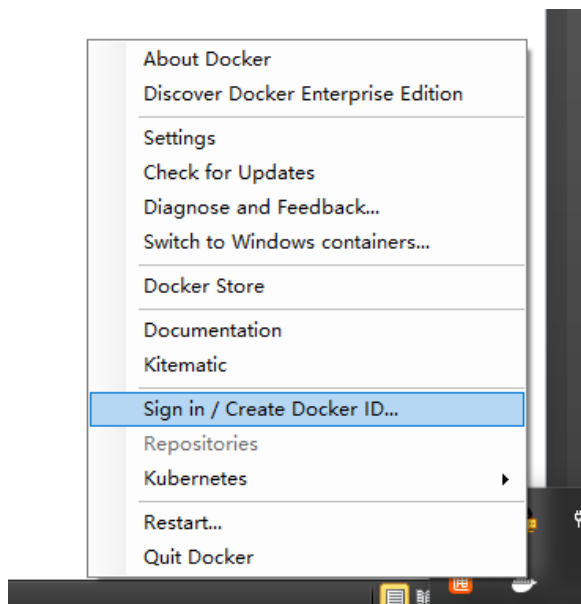
```
requests
selenium
aiohttp
beautifulsoup4
pyquery
pymysql
redis
pymongo
django
scrapy
scrapyd
scrapyd-client
scrapy-redis
```

使用 docker 创建 python 容器

Docker 国内官方镜像地址设置

<https://registry.docker-cn.com>

右键菜单，选择 Settings



安装 python 镜像

搜索 python 镜像

docker search python3.5

安装并运行镜像

docker pull python:3.5

搜索 python 镜像

docker search python3.6

安装并运行镜像

docker pull python:3.6

查看当前的镜像

docker images

```
(env2) E:\Python\代码\scrapy\scrapyd_test>docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
python	3.5	1419e5e87f7c	7 hours ago	917MB
hello-world	latest	4ab4c602aa5e	5 weeks ago	1.84kB
ubuntu	latest	cd6d8154f1e1	5 weeks ago	84.1MB

进入 ubuntu 容器交互系统:

docker run -i -t python:3.5

-t 是启动终端

-i 是允许进行交互

直接进入 python 交互环境

```
(env2) E:\Python\代码\scrapy\scrapyd_test>docker run -i -t python:3.5
Python 3.5.6 (default, Oct 16 2018, 07:29:38)
[GCC 6.3.0 20170516] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> print('hello, world')
hello, world
>>>
```

新建一个 Dockerfile

FROM python:3.5

ADD . /code

WORKDIR /code

COPY ./scrapyd.conf /etc/scrapyd/

EXPOSE 6800

RUN pip3 install -r requirements.txt

CMD scrapyd

第一行的 FROM 是指在 python:3.5 这个镜像上构建，也就是说在构建时就已经有了 Python 3.5 的环境。

第二行的 ADD 是将本地的代码放置到虚拟容器中。它有两个参数：第一个参数是 .，即代表本地当前路径；第二个参数 /code 代表虚拟容器中的路径，也就是将本地项目所有内容放置到虚拟容器的/code 目录下。

第三行的 WORKDIR 是指定工作目录，这里将刚才添加的代码路径设成工作路径，这个路径下的目录结构和当前本地目录结构是相同的，所以在这个目录下可以直接执行库安装命令。

第四行的 COPY 是将当前目录下的 scrapyd.conf 文件复制到虚拟容器的/etc/scrapyd/目录下，Scrapyd 在运行的时候会默认读取这个配置。

第五行的 EXPOSE 是声明运行时容器提供服务端口，注意这里只是一个声明，运行时不一定会在此端口开启服务。这个声明的作用，一是告诉使用者这个镜像服务的运行端口，以方便配置映射，二是在运行使用随机端口映射时，容器会自动随机映射 EXPOSE 的端口。

第六行的 RUN 是执行某些命令，一般做一些环境准备工作。由于 Docker 虚拟容器内只有 Python 3 环境，而没有 Python 库，所以我们运行此命令来在虚拟容器中安装相应的 Python 库，这样项目部署到 Scrapyd 中便可以正常运行。

第七行的 CMD 是容器启动命令，容器运行时，此命令会被执行。这里我们直接用 scrapyd 来启动 Scrapyd 服务。

镜像构建

docker build -t scrapyd:latest .

```
(env2) E:\Python\代码\scrapy\scrapyd_test>docker build -t scrapyd:latest .
Sending build context to Docker daemon 33.28kB
Step 1/7 : FROM python:3.5
--> 1419e5e87f7c
Step 2/7 : ADD . /code
--> 5291d2077aa5
Step 3/7 : WORKDIR /code
--> Running in b880f43d2195
Removing intermediate container b880f43d2195
--> 6610adb7b43e
Step 4/7 : COPY ./scrapyd.conf /etc/scrapyd/
--> 388807a57293
Step 5/7 : EXPOSE 6800
--> Running in f5ec910399b4
Removing intermediate container f5ec910399b4
--> 251ead3e7640
Step 6/7 : RUN pip3 install -r requirements.txt
--> Running in 21e5ccdb8147
Collecting Appium-Python-Client==0.28 (from -r requirements.txt (line 1))
  Downloading https://files.pythonhosted.org/packages/65/f4/b31229c97ecd03015f2e2abd79ee998f0b91977062d559270abda9f1f3fe/Appium-Python-Client-0.28.tar.gz
Collecting asn1crypto==0.24.0 (from -r requirements.txt (line 2))
  Downloading https://files.pythonhosted.org/packages/ea/cd/35485615f45f30a510576f1a56d1e0a7ad7bd8ab5ed7cdc600ef7cd06222/asn1crypto-0.24.0-py2.py3-none-any.whl (101kB)
Collecting atomicwrites==1.2.1 (from -r requirements.txt (line 3))
  Downloading https://files.pythonhosted.org/packages/3a/9a/9d878f8d885706e2530402de6417141129a943802c084238914fa6798d97/atomicwrites-1.2.1.tar.gz
Removing intermediate container 7bd8fce711e0
--> e902e60ae2f4
Step 7/7 : CMD scrapyd
--> Running in 006cf3a6a397
Removing intermediate container 006cf3a6a397
--> f181837ea10e
Successfully built f181837ea10e
Successfully tagged scrapyd:latest
SECURITY WARNING: You are building a Docker image from Windows against a non-Windows Docker host. All files and directories added to build context will have '-rwxr-xr-x' permissions. It is recommended to double check and reset permissions for sensitive files and directories.
```

docker tag

标记本地镜像，将其归入某一仓库。

docker images

```
(env2) E:\Python\代码\scrapy\scrapy_test>docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
scrapy	latest	f181837ea10e	About an hour ago	1.05GB
<none>	<none>	35309826ba85	About an hour ago	917MB
<none>	<none>	c67732c60a84	20 hours ago	917MB
<none>	<none>	1f7368525d09	20 hours ago	917MB
<none>	<none>	332b14ae5dfd	21 hours ago	917MB
<none>	<none>	251ead3e7640	21 hours ago	917MB
python	3.5	1419e5e87f7c	2 days ago	917MB
hello-world	latest	4ab4c602aa5e	5 weeks ago	1.84kB
ubuntu	latest	cd6d8154f1e1	6 weeks ago	84.1MB

运行测试

`docker run -d -p 6800:6800 scrapyd`

```
(env2) E:\Python\代码\scrapy\scrapy_test>docker run -d -p 6800:6800
scrapyd
c3593668505c0b13bce1a15153ba369d7450f3ba55efc48932fcc2ecf72350fb
```

```
(env2) E:\Python\代码\scrapy\scrapy_test>
```

-p: 端口映射, 格式为: 主机(宿主)端口:容器端口, 将容器的 6800 端口映射到主机的 6800 端口

观察 Scrapyd 服务

打开: <http://localhost:6800>, 即可观察到 Scrapyd 服务



Jobs

[Go back](#)

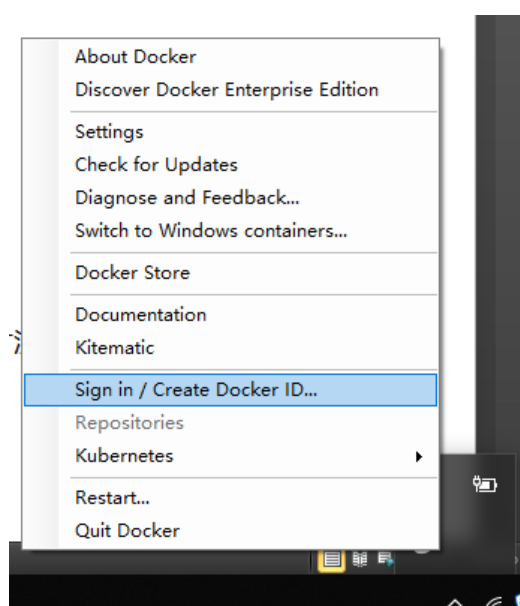
Project	Spider	Job	PID	Start	Runtime	Finish	Log
Pending							
Running							
Finished							

Scrapyd Docker 镜像构建完成并成功运行。

Docker Cloud

Docker 提供了在云端注册存储和获取 Docker 映像的功能。 可以使用私人或公开地存储 Docker 映像。它是一个完整的 GUI 界面，允许我们管理构建，映像，群集，节点和应用程序。


Docker Cloud 帐号的创建



输入 ID, email, 密码, 进行注册

New to Docker?

Create your free Docker ID to get started.

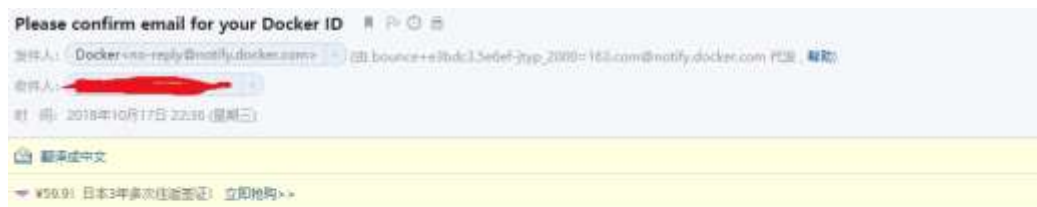
☒ 进行人机身份验证  reCAPTCHA
隐私权 · 使用条款

☒ * I agree to Docker's [Terms of Service](#).

☒ * I agree to Docker's [Privacy Policy](#) and [Data Processing Terms](#).

☐ I would like to receive email updates from Docker, including its various services and products.

邮箱激活



Please confirm your email address

You have created a Docker ID with the username: kwok2018

[Confirm Your Email](#)

(This link will expire in 2 days.)

创建 Repository



Create Repository

kwok2018 / tom

Tom's repository

Visibility

Using 0 of 1 private repositories. [Get more](#)

<input checked="" type="radio"/> Public	<input type="radio"/> Private
Public repositories appear in Docker Hub search results	Only you can see private repositories

Build Settings (optional)

Autobuild triggers a new build with every git push to your source code repository [Learn more](#)



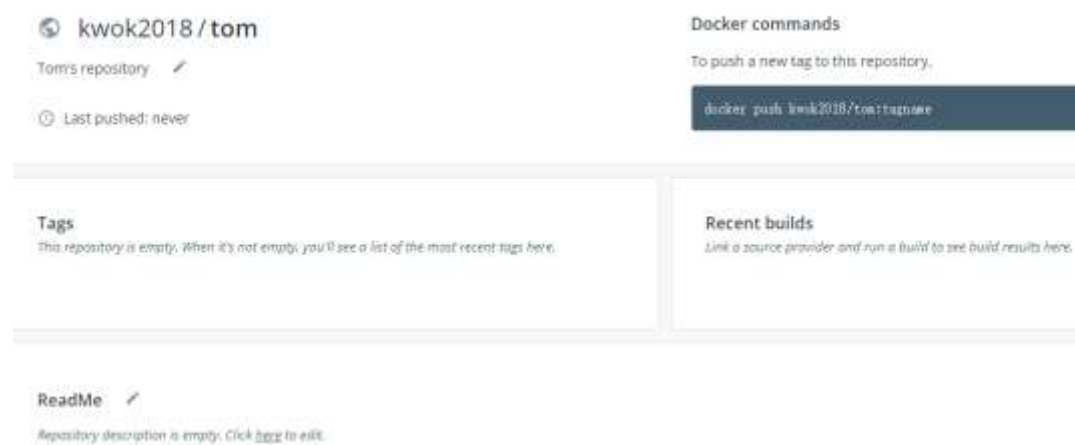
Cancel

Create

Create & Build

0 OF 1 FREE PRIVATE REPOSITORY IN USE
Purchase plans starting at \$7/ month for 5 private repositories

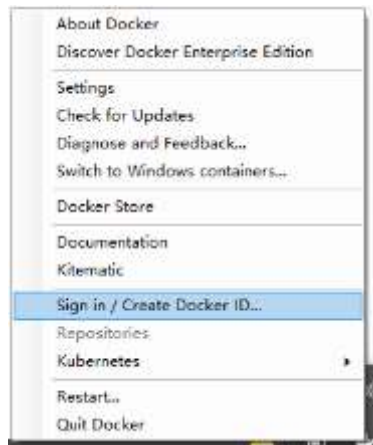
可以免费创建一个私有的 repositories，每月 7 美元购买 5 个私有的 repositories



登录 Docker Cloud

注意：一定要登录，否则上传镜像是会报错：

denied: requested access to the resource is denied





上传到 Docker Cloud

语法:

```
docker tag local-image:tagname reponame:tagname
```

```
docker push reponame:tagname
```

我的 Docker Hub 用户名为 **kwok2018**, repository 为 **tom**, 操作:

```
docker tag scrapyd:latest kwok2018/tom:zhaopin
```

```
(env2) E:\Python\代码\scrapy\scrapy_test>docker tag scrapyd:latest kwok2018/tom:zhaopin
```

docker images


```
(env2) E:\Python\代码\scrapy\scrapy_test>docker images
REPOSITORY          TAG                 IMAGE ID           CREATED            SIZE
kwok2018/scrapyd     latest             f181837ea10e      2 hours ago       1.05GB
kwok2018/tom/scrapyd latest             f181837ea10e      2 hours ago       1.05GB
kwok2018/tom         zhaopin            f181837ea10e      2 hours ago       1.05GB
scrapyd              latest             f181837ea10e      2 hours ago       1.05GB
<none>               <none>            35309826ba85      2 hours ago       917MB
<none>               <none>            c67732c60a84      21 hours ago      917MB
<none>               <none>            1f7368525d09      21 hours ago      917MB
<none>               <none>            332b14ae5dfd      22 hours ago      917MB
<none>               <none>            251ead3e7640      22 hours ago      917MB
python               3.5               1419e5e87f7c      2 days ago        917MB
hello-world          latest            4ab4c602aa5e      5 weeks ago       1.84kB
ubuntu               latest            cd6d8154f1e1      6 weeks ago       84.1MB
```

push,将此镜像上传到 Docker Hub

语法:

docker push reponame:tagname

操作:

docker push kwok2018/tom:zhaopin

```
(env2) E:\Python\代码\scrapy\scrapy_test>docker push kwok2018/tom:zhaopin
The push refers to repository [docker.io/kwok2018/tom]
fb25514c881d: Pushing 4.421MB/131.6MB
c1442e70ae4a: Pushed
d43a37baac4b: Pushing 3.872MB/64.12MB
2f4f74d3821e: Pushing 5.319MB/16.85MB
aaa5e8fca5a4: Pushed
d43a37baac4b: Pushing 3.315MB/64.12MB
2f4f74d3821e: Pushing 4.532MB/16.85MB
9978d084fd77: Waiting
1191b3f5862a: Waiting
08a01612ffca: Waiting
8bb25f9cdc41: Waiting
f715ed19c28b: Waiting
```

其他主机运行

其他主机运行此命令即可启动 Scrapyd 服务:

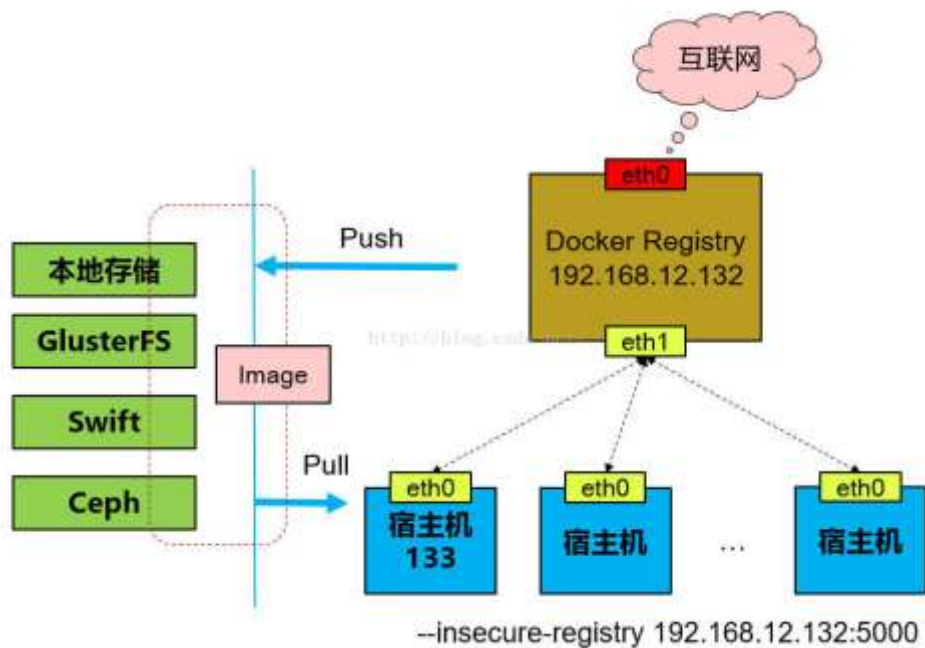
docker run -d -p 6800:6800 kwok2018/tom:zhaopin

-d: 后台运行容器, 并返回容器 ID;

Docker 私有仓库 Registry

有时候我们的服务器无法访问互联网, 或者不希望将自己的镜像放到公网当中, 那么你就需

要 Docker Registry, 它可以用来存储和管理自己的镜像。

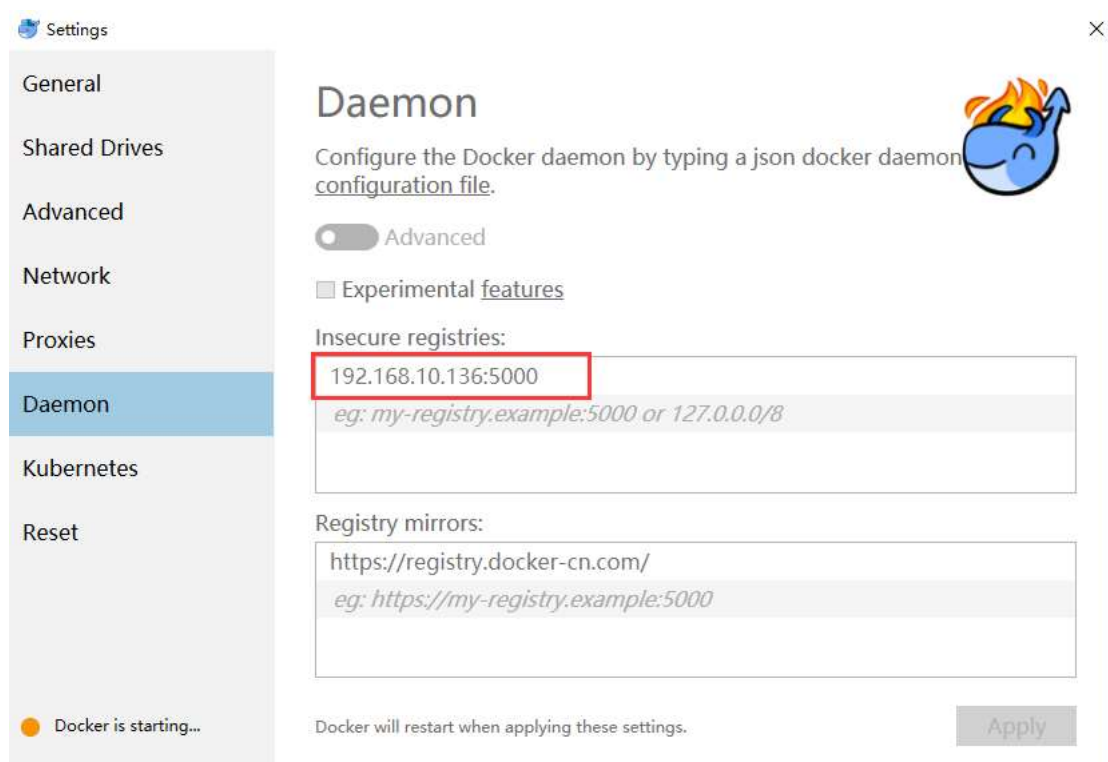


安装 Registry

设置 insecure 参数

由于默认是需要 https 证书支持的, 我们启用的 registry 服务不是安全可信赖的, 所以我们需要在所有参与的物理服务器添加可信参数

ip: Registry 的机器 ip 地址, 在安装 registry 的节点和客户端需要访问私有 Registry 的节点都需要执行此步操作。



点击应用后会自动重启 docker 服务

安装 Registry

从 docker cloud 下载 docker registry 镜像，然后启动，执行如下命令：

```
docker run -d -p 5000:5000 -v /data/registry:/var/lib/registry --name registry --restart=always registry
```

-p: 端口映射，格式为：主机(宿主)端口:容器端口，将容器的 5000 端口映射到主机的 5000 端口

-d: 后台运行容器，并返回容器 ID

--name registry: 为容器指定一个名称

```
(env2) C:\Users\admin>docker run -d -p 5000:5000 -v /data/registry:/var/lib/registry --name registry --restart=always registry
Unable to find image 'registry:latest' locally
latest: Pulling from library/registry
36a5679aa3cf: Pull complete
ad0eac849f8f: Pull complete
2261be058a15: Pull complete
f296fda86f10: Pull complete
bcd4a541795b: Pull complete
Digest: sha256:5a156ff125e5a12ac7fdec2b90b7e2ae5120fa249cf62248337b6d04abc574c8
Status: Downloaded newer image for registry:latest
793ebe3fdd33564d847b8d3929439f1c0a4da341f7911718c3adfbb4be50388
```

重命名镜像

通过 docker tag 重命名镜像，使之与 registry 匹配

docker tag scrapyd:latest 192.168.10.136:5000/scrapyd:zhaopin

```
(env2) C:\Users\admin>docker tag scrapyd:latest 192.168.10.136:5000/scrapyd:zhaopin
```

```
(env2) C:\Users\admin>docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
scrapyd	latest	5f42d5e2f7d5	3 days ago	1.05GB
192.168.10.136:5000/scrapyd	zhaopin	5f42d5e2f7d5	3 days ago	1.05GB
kwok2018/tom	zhaopin	5f42d5e2f7d5	3 days ago	1.05GB
python	3.5	1419e5e87f7c	6 days ago	917MB
registry	latest	2e2f252f3c88	5 weeks ago	33.3MB

上传镜像到 Registry

docker push 192.168.10.136:5000/scrapyd:zhaopin

```
(env2) C:\Users\admin>docker push 192.168.10.136:5000/scrapyd:zhaopin
33de0cd7ca33: Pushing [=====>] 70.57MB/131.6MB
33de0cd7ca33: Pushing [=====>] 70.04MB/131.6MB
117ebf1a69bf: Pushed
3978d084fd77: Pushed
1191b3f5862a: Pushed
1191b3f5862a: Pushing [=====>] 106.1MB/141.8MB
143a37baac4b: Pushed
f715ed19c28b: Pushed
```

查看 Registry 中所有镜像信息

curl http://192.168.10.136:5000/v2/_catalog

```
(env2) C:\Users\admin>curl http://192.168.10.136:5000/v2/_catalog
{"repositories":["scrapyd"]}
```

其他 Docker 服务器下载镜像

docker pull 192.168.10.136:5000/scrapyd:zhaopin

启动镜像

```
docker run -it 192.168.10.136:5000/scrapyd:zhaopin
```

说明：latest 标签

latest 在使用中不是最新的意思，而是默认值(defalut)的意思。

如果在 tag 为可选的命令中，我们没有写上 tag

如：

```
docker pull entel_zmc_images:zmc_base
```

```
docker pull entel_zmc_images
```

第一个语句有确定的 tag，而后者没有，这时系统会自动添加一个:latest 标签，然后去匹配。

本地镜像的管理

保存镜像

我们的镜像做好之后，我们要保存起来，以供备份使用

使用 docker save 命令，保存镜像到本地。

```
docker save -o rocketmq.tar rocketmq
```

参数说明：

-o：指定保存的镜像的名字

rocketmq.tar：保存到本地的镜像名称；

rocketmq：镜像名字，通过"docker images"查看

载入镜像

在需要使用的时候可以使用 docker load 将本地保存的镜像再次导入 docker 中

docker load --input rocketmq.tar

或

docker load < rocketmq.tar

删除镜像

docker rmi -f image_id

参数说明:

-f: 表示强制删除镜像

image_id: 镜像 id

docker images

```
(env2) C:\Users\admin>docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
kwok2018/scrapyd	latest	f181837ea10e	3 hours ago	1.05GB
kwok2018/tom/scrapyd	latest	f181837ea10e	3 hours ago	1.05GB
kwok2018/tom	zhaopin	f181837ea10e	3 hours ago	1.05GB
scrapyd	latest	f181837ea10e	3 hours ago	1.05GB
<none>	<none>	332b14ae5dfd	23 hours ago	917MB
<none>	<none>	251ead3e7640	24 hours ago	917MB
python	3.5	1419e5e87f7c	2 days ago	917MB
hello-world	latest	4ab4c602aa5e	5 weeks ago	1.84kB
ubuntu	latest	cd6d8154f1e1	6 weeks ago	84.1MB

docker rmi -f 332b14ae5dfd

```
(env2) C:\Users\admin>docker rmi -f 332b14ae5dfd
Deleted: sha256:332b14ae5dfda5ba51552db8347c27fd621893858f2e26eb530aaa61359717bb
Deleted: sha256:7f9723cdfe0d79a9c5fceedc4233e96b6c6a951640e9d8e9a0859ac9b1d9afe4
Deleted: sha256:d46cca3b214e3d4b042898a630226f4b36eb938eacab7b0c5dacb0117413a772
Deleted: sha256:f8400f961b1e17f83fa0718b693bea4e69c817663660a800251fd91cf91ece28
Deleted: sha256:6bacd69975e98a45d4a2d1904e7b613b654994d914c5395e123a56180fbbd956
Deleted: sha256:ea6154e79b2b8882fe2a3cdb09b85d15f8d539ddc5e6b0ea139c30d2403c8120
```

docker images

```
(env2) C:\Users\admin>docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
kwok2018/scrapyd	latest	f181837ea10e	3 hours ago	1.05GB
kwok2018/tom/scrapyd	latest	f181837ea10e	3 hours ago	1.05GB
kwok2018/tom	zhaopin	f181837ea10e	3 hours ago	1.05GB
scrapyd	latest	f181837ea10e	3 hours ago	1.05GB
<none>	<none>	251ead3e7640	24 hours ago	917MB
python	3.5	1419e5e87f7c	2 days ago	917MB
hello-world	latest	4ab4c602aa5e	5 weeks ago	1.84kB
ubuntu	latest	cd6d8154f1e1	6 weeks ago	84.1MB

Docker 容器操作

查看容器的信息

`docker ps -n 5`

```
(env2) C:\Users\admin>docker ps -n 5
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS
NAME					
:3593668505c	scrapyd	"/bin/sh -c scrapyd"	4 hours ago	Up 4 hours	0.0.0.0:6800->6800/tcp
lucid_sammet	35309826ba85	"/bin/sh -c 'pip3 in...'"	4 hours ago	Exited (1) 4 hours ago	
889d87041d79	35309826ba85	"/bin/sh -c 'pip3 in...'"	4 hours ago	Exited (1) 4 hours ago	
suspicious_elion	35309826ba85	"/bin/sh -c 'pip3 in...'"	4 hours ago	Exited (1) 4 hours ago	
71b958d8805b	35309826ba85	"/bin/sh -c 'pip3 in...'"	4 hours ago	Exited (1) 4 hours ago	
zen_lumiere	c67732c60a84	"/bin/sh -c 'pip3 in...'"	4 hours ago	Exited (2) 4 hours ago	
:50c39028607	c67732c60a84	"/bin/sh -c 'pip3 in...'"	4 hours ago	Exited (2) 4 hours ago	
quizzical_engelbart	c67732c60a84	"/bin/sh -c 'pip3 in...'"	4 hours ago	Exited (2) 4 hours ago	
3007d0b7ed5	c67732c60a84	"/bin/sh -c 'pip3 in...'"	4 hours ago	Exited (2) 4 hours ago	
friedly_meninsky					

启动容器

`docker run my/python:v1 cal`

参数说明:

my/python:v1 为镜像名和标签

```
[root@rocketmq-nameserver4 ~]# docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
my/python	v1	36b6e288656c	2 days ago	281MB
my/centos_width_python	v1.0.1	36b6e288656c	2 days ago	281MB
my/sinatra	v2	8bald6a3ce4e	3 days ago	453MB
hello-world	latest	725dcfab7d63	4 months ago	1.84kB
centos	latest	d123f4e55e12	4 months ago	197MB
daocloud.io/library/centos	latest	196e0ce0c9fb	6 months ago	197MB
training/sinatra	latest	49d952a36c58	3 years ago	447MB

```
[root@rocketmq-nameserver4 ~]# docker run my/python:v1 cal
```

```

      March 2018
Su Mo Tu We Th Fr Sa
                1  2  3
 4  5  6  7  8  9 10
11 12 13 14 15 16 17
18 19 20 21 22 23 24
25 26 27 28 29 30 31

[root@rocketmq-nameserver4 ~]#
```

终止容器

`docker stop $CONTAINER_ID`

重启容器

终止状态的容器，可以使用 `docker start` 来重新启动。


```
[root@rocketmq-nameserver4 ~]# docker ps -n 5
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS
bad9a582e592       centos             "/bin/sh -c 'while..." 10 minutes ago     Up 32 seconds
b9985ff33181       my/python:v1      "/bin/echo hello test" 17 minutes ago     Exited (0) 16 minutes ago
21db53466358       my/python:v1      "/bin/echo hello w..." 33 minutes ago     Exited (0) 32 minutes ago
ae0eaf5cb897       my/python:v1      "/bin/echo hello w..." 2 hours ago        Exited (0) 2 hours ago
167965f9d732       my/python:v1      "/bin/echo 'hello..." 2 hours ago        Exited (0) 2 hours ago
[root@rocketmq-nameserver4 ~]# docker stop bad9a582e592
[root@rocketmq-nameserver4 ~]# docker start bad9a582e592
[root@rocketmq-nameserver4 ~]# docker ps -n 5
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS
bad9a582e592       centos             "/bin/sh -c 'while..." 10 minutes ago     Up 2 seconds
b9985ff33181       my/python:v1      "/bin/echo hello test" 18 minutes ago     Exited (0) 17 minutes ago
21db53466358       my/python:v1      "/bin/echo hello w..." 33 minutes ago     Exited (0) 33 minutes ago
ae0eaf5cb897       my/python:v1      "/bin/echo hello w..." 2 hours ago        Exited (0) 2 hours ago
167965f9d732       my/python:v1      "/bin/echo 'hello..." 2 hours ago        Exited (0) 2 hours ago
```

附录：curl 的安装

<https://www.cnblogs.com/zhuzhenwei918/p/6781314.html>

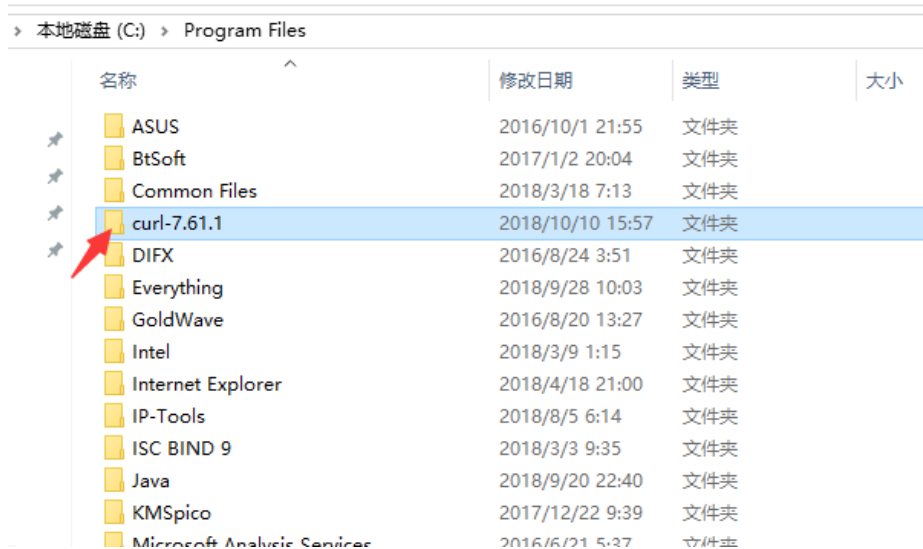
curl 是利用 URL 语法在命令行方式下工作的开源文件传输工具。它被广泛应用在 Unix、多种 Linux 发行版中，并且有 DOS 和 Win32、Win64 下的移植版本。

第一步：进入 curl 下载官网，下载合适的版本。

<https://curl.haxx.se/download.html>

Windows 32 bit			
Windows 32 bit	7.61.1	binary	the curl project
Windows 32 bit	7.61.1	binary	Stefan Kanthak
Windows 32 bit	7.61.1	binary	Chocolatey
Windows 32 bit	7.61.1	binary	Viktor Szakats
Windows 32 bit	7.61.0	binary	Marc Hörsken
Windows 32 bit	7.61.0	binary	Dirk Paehl
Windows 32 bit - cygwin			
Windows 32 bit cygwin	7.59.0	binary	Cygwin
Windows 32 bit cygwin	7.59.0	libcurl	Cygwin
Windows 64 bit			
Windows 64 bit	7.61.1	binary	the curl project
Windows 64 bit	7.61.1	binary	Chocolatey
Windows 64 bit	7.61.1	binary	Stefan Kanthak
Windows 64 bit	7.61.1	binary	Viktor Szakats
Windows 64 bit	7.59.0	binary	Marc Hörsken
Windows 64 bit	7.53.1	binary	Darren Owen

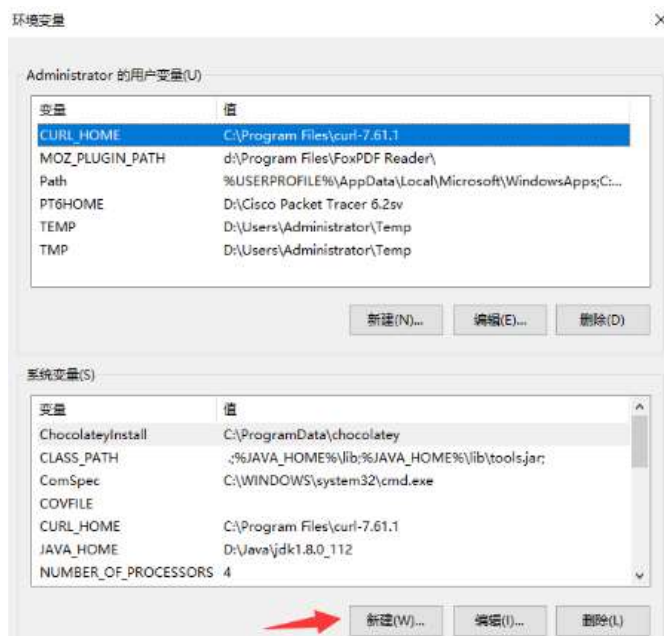
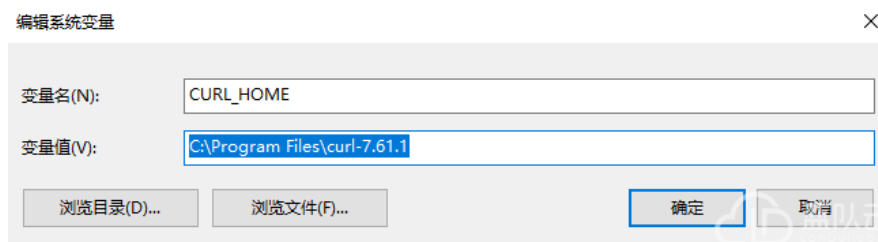
下载后解压出来存放我丢在了 C 盘 Program Files



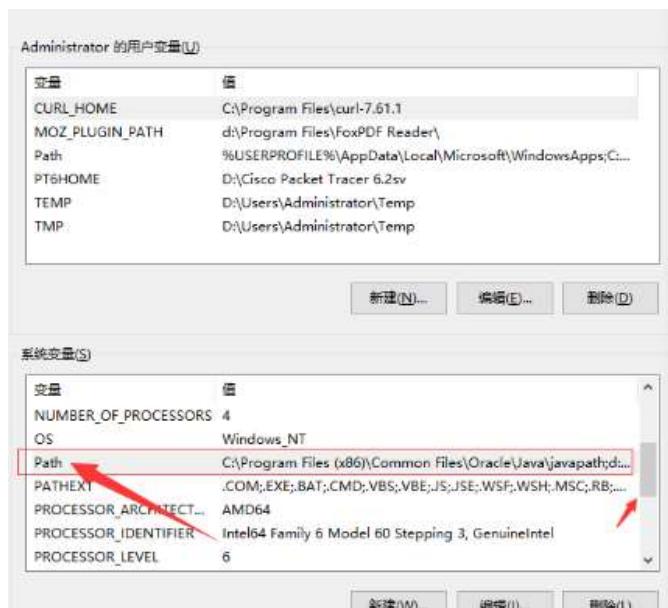
设置变量环境

新建一个环境变量，名字随意取，这里：CURL_HOME

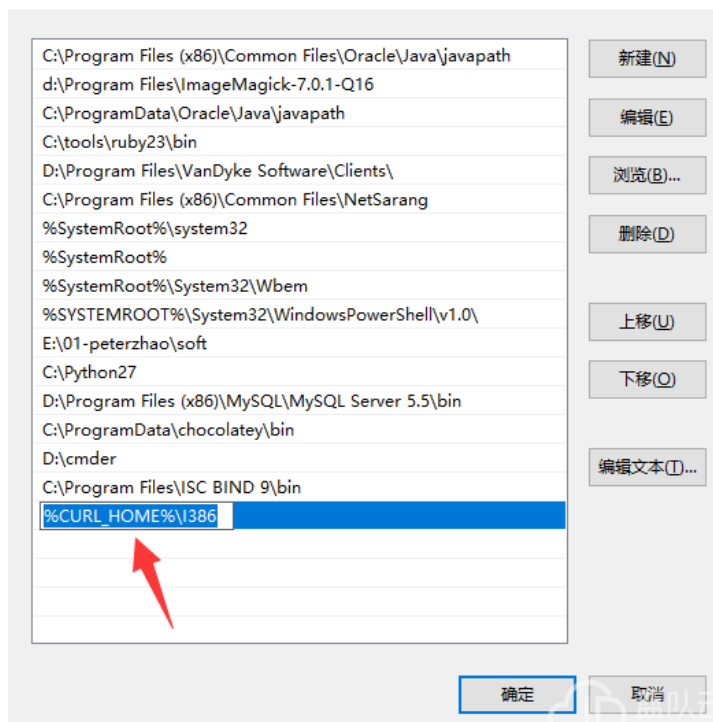
变量值为文件路径:C:\Program Files\curl-7.61.1



编辑 Path 变量



编辑环境变量



cmd 下测试

curl http://www.landui.com -v

```
C:\Users\Administrator>curl http://www.landui.com -v
* Rebuilt URL to: http://www.landui.com/
* Trying 182.247.238.19...
* TCP_NODELAY set
* Connected to www.landui.com (182.247.238.19) port 80 (#0)
> GET / HTTP/1.1
> Host: www.landui.com
> User-Agent: curl/7.61.1
> Accept: */*
>
< HTTP/1.1 301 Moved Permanently
< Server: kangle/3.5.10.1
< Date: Wed, 10 Oct 2018 09:40:59 GMT
< Location: https://www.landui.com/
< Content-Length: 0
< Connection: close
<
* Closing connection 0
```

C:\Users\admin\Dockerfile: