python flask??????????,??? docker??

??python web??????????docker?

> 参考 **：https://blog.51cto.com/fish/6023519**

(?)??????

创建目录

```
mkdir mytest
cd mytest

[root@python mytest]# pwd
/root/mytest
```

创建虚拟环境

```
[root@python mytest]# python3 -m venv myvenv
```

激活虚拟环境

```
[root@python mytest]# source myvenv/bin/activate
```

一旦激活虚拟环境后，将进入到该虚拟环境下的shell界面，如下：

```
(myvenv) [root@python mytest]#
(myvenv) [root@python mytest]# ll
total 16
-rw-r--r--. 1 root root 222 Mar 31 11:14 demo.py
-rw-r--r--. 1 root root 307 Mar 31 13:26 Dockerfile
-rw-r--r--. 1 root root 282 Mar 31 13:38 gunicorn.conf.py
drwxr-xr-x. 5 root root 100 Mar 31 11:45 myvenv
-rw-r--r--. 1 root root 261 Mar 31 13:03 requirements.txt
(myvenv) [root@python mytest]#
```

(?)??flask??

>

> 在文件夹 mytest下，创建一个 demo.py启动文件

```
from flask import Flask

app = Flask(__name__)

@app.route('/')
def hello():
    return 'hello docker & flask & 爱看书的小沐.'

if __name__ == '__main__':
    app.run(host="0.0.0.0", debug=True)
```

> 在该环境下执行 demo.py文件 ，由于没有 flask环境将报错

```
(myvenv) [root@python mytest]# python demo.py
...

安装flask
(myvenv) [root@python mytest]# python install flask
...

再次运行demo.py
(myvenv) [root@python mytest]# python demo.py
 * Serving Flask app 'demo' (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Running on all addresses.
   WARNING: This is a development server. Do not use it in a production deployment.
 * Running on http://10.0.0.101:5000/ (Press CTRL+C to quit)
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 128-311-800

监听5000端口,可以正常跑起来
```

(?)?? gunicorn gevent, ????

```
(myvenv) [root@python mytest]# pip install gunicorn gevent flask


在mytest目录下创建gunicorn.conf.py配置文件
(myvenv) [root@python mytest]# vim gunicorn.conf.py


文件内容如下:
workers = 5     # 定义同时开启的处理请求的进程数量，根据网站流量适当调整
worker_class = "gevent"    # 采用gevent库，支持异步处理请求，提高吞吐量
bind = "0.0.0.0:5000"    #端口随便写，但是注意是否已经被占用。netstap -lntp
```

> 启动 gunicorn,如下

```
(myvenv) [root@python mytest]# gunicorn demo:app -c gunicorn.conf.py
[2023-03-31 14:08:57 +0800] [58874] [INFO] Starting gunicorn 20.1.0
[2023-03-31 14:08:57 +0800] [58874] [INFO] Listening at: http://0.0.0.0:3000 (58874)
[2023-03-31 14:08:57 +0800] [58874] [INFO] Using worker: gevent
[2023-03-31 14:08:57 +0800] [58877] [INFO] Booting worker with pid: 58877
[2023-03-31 14:08:57 +0800] [58878] [INFO] Booting worker with pid: 58878
[2023-03-31 14:08:57 +0800] [58879] [INFO] Booting worker with pid: 58879
[2023-03-31 14:08:57 +0800] [58880] [INFO] Booting worker with pid: 58880
[2023-03-31 14:08:57 +0800] [58881] [INFO] Booting worker with pid: 58881
```

(?)??????docker?

> **(1)** 执行如下命令 ，生成 python项目所依赖的包文件 ，将其写入到 requirements.txt文件中

```
(myvenv) [root@python mytest]# pip freeze > requirements.txt
(myvenv) [root@python mytest]# ll
total 16
-rw-r--r--. 1 root root 222 Mar 31 11:14 demo.py
-rw-r--r--. 1 root root 282 Mar 31 13:38 gunicorn.conf.py
drwxr-xr-x. 5 root root 100 Mar 31 11:45 myvenv
drwxr-xr-x. 2 root root  69 Mar 31 14:08 __pycache__
-rw-r--r--. 1 root root 261 Mar 31 13:03 requirements.txt
(myvenv) [root@python mytest]# cat requirements.txt
```

```
click==8.0.4
dataclasses==0.8
Flask==2.0.3
gevent==22.10.2
greenlet==2.0.2
gunicorn==20.1.0
importlib-metadata==4.8.3
itsdangerous==2.0.1
Jinja2==3.0.3
MarkupSafe==2.0.1
typing_extensions==4.1.1
Werkzeug==2.0.3
zipp==3.6.0
zope.event==4.6
zope.interface==5.5.2
(myvenv) [root@python mytest]#
```

> **(2)** 创建 Dockerfile文件 ，用于构建自己的镜像

```
(myvenv) [root@python mytest]# vim Dockerfile
(myvenv) [root@python mytest]# cat Dockerfile
FROM python:3.8

WORKDIR /project/

COPY requirements.txt ./
RUN python -m pip install --upgrade pip
RUN pip install --ignore-requires-python dataclasses==0.8
RUN pip install -r requirements.txt -i https://pypi.tuna.tsinghua.edu.cn/simple

#将当前所有文件拷贝到要制作的docker镜像中
COPY . .

CMD ["gunicorn", "demo:app", "-c", "./gunicorn.conf.py"]
```

> **(3)** 配置 docker镜像加速地址

```
[root@python mytest]#  mkdir -p /etc/docker
[root@python mytest]#  tee /etc/docker/daemon.json <<-'EOF'
{
    "registry-mirrors": [
"https://o6ul5754.mirror.aliyuncs.com",
"https://ung2thfc.mirror.aliyuncs.com",
"https://registry.docker-cn.com",
"http://hub-mirror.c.163.com",
"https://docker.mirrors.ustc.edu.cn"
]
}
EOF
[root@python mytest]#  cat /etc/docker/daemon.json
{
    "registry-mirrors": [
"https://o6ul5754.mirror.aliyuncs.com",
"https://ung2thfc.mirror.aliyuncs.com",
"https://registry.docker-cn.com",
"http://hub-mirror.c.163.com",
"https://docker.mirrors.ustc.edu.cn"
]
}
[root@python mytest]#  systemctl daemon-reload
[root@python mytest]#  systemctl restart docker

□□□□□□□□
[root@python mytest]# docker info
Client:
 Context:    default
 Debug Mode: false
 Plugins:
  buildx: Docker Buildx (Docker Inc.)
    Version:  v0.10.4
    Path:     /usr/libexec/docker/cli-plugins/docker-buildx
  compose: Docker Compose (Docker Inc.)
    Version:  v2.17.2
    Path:     /usr/libexec/docker/cli-plugins/docker-compose
  scan: Docker Scan (Docker Inc.)
    Version:  v0.23.0
    Path:     /usr/libexec/docker/cli-plugins/docker-scan
```

```
Server:
 Containers: 5
  Running: 0
  Paused: 0
  Stopped: 5
 Images: 3
 Server Version: 23.0.2
 Storage Driver: overlay2
  Backing Filesystem: xfs
  Supports d_type: true
  Using metacopy: false
  Native Overlay Diff: true
  userxattr: false
 Logging Driver: json-file
 Cgroup Driver: cgroupfs
 Cgroup Version: 1
 Plugins:
  Volume: local
  Network: bridge host ipvlan macvlan null overlay
  Log: awslogs fluentd gcplogs gelf journald json-file local logentries splunk syslog
 Swarm: inactive
 Runtimes: io.containerd.runc.v2 runc
 Default Runtime: runc
 Init Binary: docker-init
 containerd version: 1e1ea6e986c6c86565bc33d52e34b81b3e2bc71f
 runc version: v1.1.4-0-g5fd4c4d
 init version: de40ad0
 Security Options:
  seccomp
   Profile: builtin
 Kernel Version: 3.10.0-957.el7.x86_64
 Operating System: CentOS Linux 7 (Core)
 OSType: linux
 Architecture: x86_64
 CPUs: 1
 Total Memory: 1.934GiB
 Name: python
 ID: IP2S:J2XT:35RB:KK7O:HFQ5:FMME:HNT2:LYFR:74GT:7CWE:GXGM:446B
 Docker Root Dir: /var/lib/docker
```

```
 Debug Mode: false
 Registry: https://index.docker.io/v1/
 Experimental: false
 Insecure Registries:
  127.0.0.0/8
 Registry Mirrors:
  https://o6ul5754.mirror.aliyuncs.com/
  https://ung2thfc.mirror.aliyuncs.com/
  https://registry.docker-cn.com/
  http://hub-mirror.c.163.com/
  https://docker.mirrors.ustc.edu.cn/
 Live Restore Enabled: false
```

> **(4)** □□ docker□□

```
[root@python mytest]# docker build -t myapp:1.0 .
[+] Building 18.5s (12/12)
FINISHED

 => [internal] load build definition from
Dockerfile
                                                    0.0s
 => => transferring dockerfile:
406B
                                                          0.0s
 => [internal] load
.dockerignore
                                                               0.0s
 => => transferring context:
2B
                                                             0.0s
 => [internal] load metadata for
docker.io/library/python:3.8
                                                           0.2s
 => [1/7] FROM
docker.io/library/python:3.8@sha256:4c4e6735f46e7727965d1523015874ab08f71377b3536b8789ee5742fc
737059                                                                    0.0s
 => [internal] load build
```

```
context
                                                                    0.2s
 => => transferring context:
370.95kB
                                                                0.2s
 => CACHED [2/7] WORKDIR
/project/
                                                                    0.0s
 => CACHED [3/7] COPY requirements.txt
./
                                                          0.0s
 => CACHED [4/7] RUN python -m pip install --upgrade
pip
                                          0.0s
 => [5/7] RUN pip install --ignore-requires-python
dataclasses==0.8
                                           2.3s
 => [6/7] RUN pip install -r requirements.txt -i
https://pypi.tuna.tsinghua.edu.cn/simple
                                            12.5s
 => [7/7] COPY .
.
                                                                          2.2s
 => exporting to
image
                                                                          1.2s
 => => exporting
layers
                                                                          1.1s
 => => writing image
sha256:4afe28f18625b8bf5cdcb9785d158e9917561d1d6de1367e71bf4fa3d40fc407
                                                                  0.0s
 => => naming to
docker.io/library/myapp:1.0
                                                                          0.0s
[root@python mytest]# docker images
REPOSITORY    TAG       IMAGE ID       CREATED          SIZE
myapp         1.0       4afe28f18625   59 minutes ago   1.01GB
hello-world   latest    feb5d9fea6a5   18 months ago    13.3kB
```

> **(5)** 运行该镜像成 docker容器

```
[root@python mytest]# docker run -d -p 5000:5000 myapp:1.0
a91f1d816659e5abcd74a34d4bfc2a049a60c98d0a797a49202a7cc83bbc0778
[root@python mytest]#
```

> 浏览器访问 http://10.0.0.101:5000 页面可以正常访问并获取到数据

> **(6)** 如果修改代码 ，需要重新制作该镜像文件 ，例如修改 demo.py文件如下

```
from flask import Flask

app = Flask(__name__)

@app.route('/')
def hello():
    return 'hello docker 666'

if __name__ == '__main__':
    app.run(host="0.0.0.0", debug=True, threaded=True)
```

> 同时修改端口 5000为 3000; 重新制作 docker镜像文件

```
[root@python mytest]# docker build -t myapp:1.1 .  # 这次打的标签tag版本为1.1
[root@python mytest]# docker build -t myapp:1.1 .
[+] Building 16.6s (12/12)
FINISHED

 => [internal] load build definition from
Dockerfile
                                                    0.0s
 => => transferring dockerfile:
406B
                                                          0.0s
```

```
 => [internal] load
.dockerignore
                                                                    0.0s
 => => transferring context:
2B
                                                                0.0s
 => [internal] load metadata for
docker.io/library/python:3.8
                                                            15.2s
 => [1/7] FROM
docker.io/library/python:3.8@sha256:4c4e6735f46e7727965d1523015874ab08f71377b3536b8789ee5742fc
737059                                                                  0.0s
 => [internal] load build
context
                                                               0.2s
 => => transferring context:
370.93kB
                                                            0.2s
 => CACHED [2/7] WORKDIR
/project/
                                                                0.0s
 => CACHED [3/7] COPY requirements.txt
./
                                                        0.0s
 => CACHED [4/7] RUN python -m pip install --upgrade
pip
                                          0.0s
 => CACHED [5/7] RUN pip install --ignore-requires-python
dataclasses==0.8
                                      0.0s
 => CACHED [6/7] RUN pip install -r requirements.txt -i
https://pypi.tuna.tsinghua.edu.cn/simple
                                        0.0s
 => [7/7] COPY .
.
                                                                    0.7s
 => exporting to
image
                                                                    0.5s
 => => exporting
```

```
layers
                                                                    0.5s
 => => writing image
sha256:c7df86df2dd926af25bf4194e18abbd5af873ef25ba0ed5c5cbf686d8f2df62d
                                                                   0.0s
 => => naming to
docker.io/library/myapp:1.1
                                                                    0.0s
[root@python mytest]# docker images
REPOSITORY    TAG       IMAGE ID       CREATED          SIZE
myapp         1.1       c7df86df2dd9   7 seconds ago    1.01GB
myapp         1.0       4afe28f18625   11 minutes ago   1.01GB
hello-world   latest    feb5d9fea6a5   18 months ago    13.3kB
[root@python mytest]# docker run -d -p 3000:3000 myapp:1.1
254176fcad70aebde1bb80b120b7b1f715ea1c67ce2a4d3e949b5be1bec13abc
[root@python mytest]#
```

> 再次访问 http://10.0.0.101:3000 已经正常访问

(?) ???????????

查看当前镜像

```
[root@python mytest]# docker images
REPOSITORY    TAG       IMAGE ID       CREATED         SIZE
check_tools   1.4       1175ff06f5c3   21 hours ago    1.06GB
check_tools   1.3       908c5ec78290   22 hours ago    1.04GB
check_tools   1.2       c584b78ae39d   22 hours ago    1.04GB
check_tools   1.1       770dd461a2cf   22 hours ago    1.04GB
check_tools   1.0       2692e79d33a2   6 days ago      1.01GB
myapp         1.1       c7df86df2dd9   7 days ago      1.01GB
myapp         1.0       4afe28f18625   7 days ago      1.01GB
hello-world   latest    feb5d9fea6a5   18 months ago   13.3kB
hello-world   latest    feb5d9fea6a5   18 months ago   13.3kB
```

保存镜像到指定目录下

```
[root@python mytest]# docker save 1175ff06f5c3>/root/check_tools.tar #check_tools.tar为打包的文件
```

查看保存的镜像文件 check_tools.tar

```
[root@python mytest]# ll /root/
```

```
total 1075124
-rw-------.  1 root root       1650 Oct 22  2020 anaconda-ks.cfg
-rw-r--r--.  1 root root 1083920896 Apr  6 16:10 check_tools.tar
drwxr-xr-x.  4 root root        140 Apr  7 13:16 mytest
drwxr-xr-x. 17  501  501       4096 Jan 12  2021 Python-3.6.4
-rw-r--r--.  1 root root   16992824 Jan 12  2021 Python-3.6.4.tar.xz
drwxr-xr-x.  2 root root         66 Jan 20  2021 test
drwxr-xr-x.  3 root root         21 Jan 20  2021 virtualenv_1
[root@python mytest]#
```

(?) ?????????????

> 将上述步骤生成的 `check_tools.tar` 文件拷贝到另一台机器上进行加载

```
[root@localhost ~]#  docker load < check_tools.tar        # check_tools.tar 为文件名称


查看镜像
[root@localhost ~]# docker images
REPOSITORY    TAG       IMAGE ID       CREATED         SIZE
check_tools   1.4       1175ff06f5c3   21 hours ago    1.06GB
[root@localhost ~]#


后台启动该镜像服务
[root@localhost ~]# docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
[root@localhost ~]# docker run -d -p 3000:3000 check_tools:1.4
22870b5ee2f07c2faf85b9f29bbec078355b9dd510154abb6a2f299a0078792d
[root@localhost ~]# docker ps
CONTAINER ID   IMAGE             COMMAND                  CREATED          STATUS
PORTS                                       NAMES
22870b5ee2f0   check_tools:1.4   "gunicorn check_tool…"   35 seconds ago   Up 34 seconds
0.0.0.0:3000->3000/tcp, :::3000->3000/tcp   flamboyant_keller
[root@localhost ~]#  netstat -tunlp
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program
name
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
7037/sshd
```

```
tcp        0      0 0.0.0.0:3000            0.0.0.0:*               LISTEN      14811/docker-
proxy
tcp        0      0 127.0.0.1:25            0.0.0.0:*               LISTEN
7125/master
tcp6       0      0 :::22                   :::*                    LISTEN
7037/sshd
tcp6       0      0 :::3000                 :::*                    LISTEN      14815/docker-
proxy
tcp6       0      0 ::1:25                  :::*                    LISTEN
7125/master
udp        0      0 127.0.0.1:323           0.0.0.0:*
6325/chronyd
udp6       0      0 ::1:323                 :::*
6325/chronyd
[root@localhost ~]#
```

(?)?????

> 为避免每次修改代码后 ，都要重新制作镜像文件 ，采用逻辑卷映射 ：
> 将宿主机的某个目录映射到镜像文件中的某个目录 ，以后只需要宿主机中的对应文件就可以

```
逻辑卷映射 check_tool_data
[root@python mytest]# docker run -p 3000:3000 -v check_tool_data:/project/ check_tools:1.7
[2023-04-07 07:14:49 +0000] [1] [INFO] Starting gunicorn 20.1.0
[2023-04-07 07:14:49 +0000] [1] [INFO] Listening at: http://0.0.0.0:3000 (1)
[2023-04-07 07:14:49 +0000] [1] [INFO] Using worker: gevent
[2023-04-07 07:14:49 +0000] [8] [INFO] Booting worker with pid: 8
[2023-04-07 07:14:49 +0000] [9] [INFO] Booting worker with pid: 9
[2023-04-07 07:14:49 +0000] [10] [INFO] Booting worker with pid: 10
[2023-04-07 07:14:49 +0000] [11] [INFO] Booting worker with pid: 11
[2023-04-07 07:14:50 +0000] [12] [INFO] Booting worker with pid: 12
^C[2023-04-07 07:17:02 +0000] [1] [INFO] Handling signal: int
[2023-04-07 07:17:02 +0000] [8] [INFO] Worker exiting (pid: 8)
[2023-04-07 07:17:02 +0000] [9] [INFO] Worker exiting (pid: 9)
[2023-04-07 07:17:02 +0000] [12] [INFO] Worker exiting (pid: 12)
[2023-04-07 07:17:02 +0000] [11] [INFO] Worker exiting (pid: 11)
[2023-04-07 07:17:02 +0000] [10] [INFO] Worker exiting (pid: 10)
[2023-04-07 07:17:03 +0000] [1] [INFO] Shutting down: Master
```

查找逻辑卷位置

```
[root@python mytest]# find / -name check_tool_data
/var/lib/docker/volumes/check_tool_data
[root@python mytest]# cd /var/lib/docker/volumes/check_tool_data
[root@python _data]# ll
total 24
-rw-r--r--. 1 root root 7112 Apr  6 16:03 check_tools.py
-rw-r--r--. 1 root root  222 Mar 31 11:14 demo.py
-rw-r--r--. 1 root root  389 Apr  6 16:04 Dockerfile
-rw-r--r--. 1 root root  281 Apr  6 15:27 gunicorn.conf.py
drwxr-xr-x. 2 root root   71 Apr  7 15:14 json_file
drwxr-xr-x. 5 root root  100 Apr  7 15:14 myvenv
drwxr-xr-x. 2 root root   76 Apr  7 15:14 __pycache__
-rw-r--r--. 1 root root  280 Apr  6 15:39 requirements.txt
[root@python _data]#
```

> 之后只需要修改 `/var/lib/docker/volumes/check_tool_data/check_tools.py` 文件就可以了 ，修改代码重新启动容器就可以加载到修改后的代码了
>
> ```
> [root@python mytest]# docker run -p 3000:3000 -v check_tool_data:/project/check_tools:1.7
> ```

您可

Revision #1
Created 2025-12-12 10:06:32 CST by eason
Updated 2025-12-12 10:07:49 CST by eason