**Docker使用手册**

目录

[Docker介绍 3](#_Toc135595412)

[Docker的由来 3](#_Toc135595413)

[Docker的应用场景 3](#_Toc135595414)

[Docker的优势 3](#_Toc135595415)

[Docker的架构 4](#_Toc135595416)

[Docker安装 4](#_Toc135595417)

[Docker操作 4](#_Toc135595418)

[Docker容器 4](#_Toc135595419)

[Docker镜像 6](#_Toc135595420)

[Docker仓库 7](#_Toc135595420)

[Dockerfile封装 6](#_Toc135595421)

[Dockerfile镜像使用 11](#_Toc135595430)

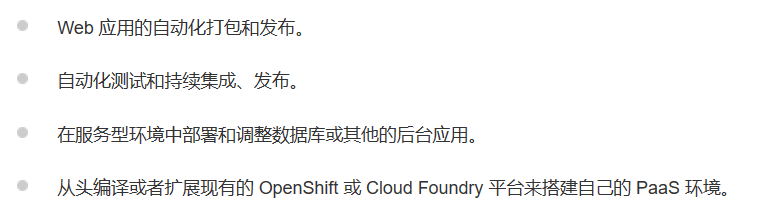
# Docker介绍

## Docker的由来

* + 1. Docker是一个开源的应用容器引擎，是基于Go语言+Apache2.0协议开源。
    2. Docker可以让开发者直接打包他们的应用和依赖包到一个容器之中。

1.1.3 Docker分为CE(社区版本)和EE(企业版本)两个版本。

## Docker的应用场景



## Docker的优势

* + 1. 能够快速、一致地交付应用程序

Docker能够让开发人员使用我提供的应用程序/本地的容器在标准化环境中工作，能够减少开发的时间。

* + 1. 能够响应式部署和扩展

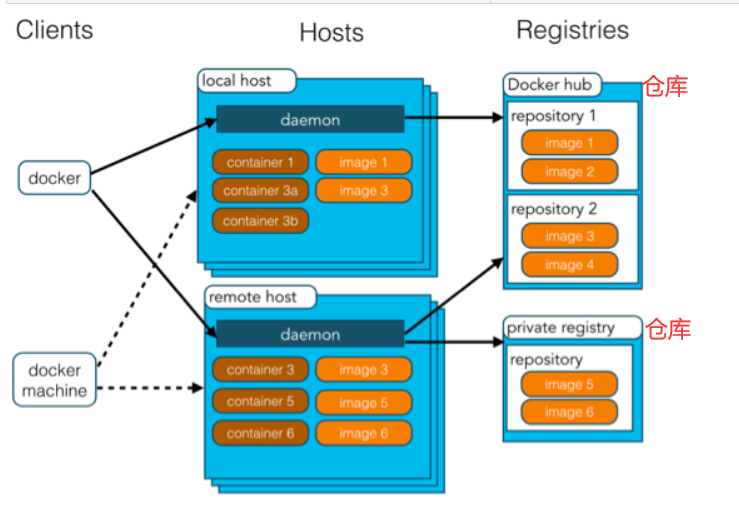
Docker能够在开发人员的本机上，数据中心的物理/虚拟机上/云服务上/混合环境中运行。

Docker还可以根据业务需求指示，动态拆除和扩展应用程序和服务。

* + 1. 在同一个硬件上运行更多工作负载

Docker能够减少资源做更多的事情。

## Docker的架构



* + 1. 镜像(image):相当于一个root文件夹，或者说是java语言中的类。（静态的定义）
    2. 容器(container):相当于一个文件，或者说是java语言中的实例。(镜像运行时的实体)。容器可以被创建、启动、停止、删除、暂停等。
    3. 仓库(repository):仓库可以看作是像github一样代码托管中心，可以用来保存镜像。

## Docker安装

Docker可以安装在Ubuntu/linux/windows/MacOS/CentOS等等上面。

可以参考本篇博客[🎉 Docker 简介和安装 - Docker 快速入门 - 易文档 (easydoc.net)](https://docker.easydoc.net/doc/81170005/cCewZWoN/lTKfePfP) 桌面版: <https://www.docker.com/products/docker-desktop>

# Docker操作

## Docker容器

* + 1. 启动容器 **docker run**

docker run -itd –name 容器名 镜像名

按照顺序加一个可有的参数

|  |  |  |  |
| --- | --- | --- | --- |
| -i | 交互式操作 |  |  |
| -t | 终端 |  |  |
| -d | 定义docker网络类型 | Bridge/overlay |  |
| -P/p | -P大写 | 随机映射 |  |
| -p小写 | 5000:5000 | Docker开放5000端口映射到主机端口5000上 |
| -p小写 | 127.0.0.1:5001:5000 | 绑定网络地址和端口 |
| -p小写 | 127.0.0.1:5001:5000/udp | 可以更改默认的tcp |
|  |  |  |
| --name | 给容器设定名称 |  |  |
| --network | 设定网络(可以用于容器连接) |  |  |

* + 1. 停止容器 **docker stop**

docker stop 容器名

* + 1. 启动已经停止运行的一个/多个容器 **docker start**

docker start 容器名

* + 1. 删除容器 **docker rm(容器必须是stop状态)**

docker rm -f 容器名

* + 1. 重启已经停止运行的一个容器 **docker restart**

docker restart 容器名

* + 1. 导出容器 **docker import**

docker import 容器名/http地址名

* + 1. 导入容器 **docker export**

docker export 容器名

* + 1. 进入容器
       1. **docker exec**

docker exec -it ID /bin/bash

ls(查看容器目录)

* + - 1. **docker cp**(只能将文件复制到容器对应位置，但是镜像不更新!)

docker cp 本地文件路径 容器名/ID：容器路径

* + - 1. **docker -v挂载**(可以将本地主机上的文件直接暴露给容器使用)

(将/aaa目录映射到主机的/a/b/c目录)

docker run -it -v /a/b/c:/aaa –name test image

* + 1. 容器常用指令
       1. 查看容器**docker ps**

docker ps -a(所有的容器)

docker ps (正在运行的容器)

docker ps -l (最后一次创建的容器)

docker port 容器名(查看容器的端口)

docker logs -f 容器名(查看容器的日志文件)

docker inspect 容器名(查看容器的底层信息json文件)

## Docker镜像

* + 1. 获取镜像 **docker pull**

docker pull 镜像名:tag(不写默认是latest标签)

* + 1. 查询镜像 **docker search**

docker search 镜像名

* + 1. 删除镜像 **docker rmi**

docker rmi 镜像名

* + 1. 创建镜像
       1. 从已有的容器中更新镜像，并且提交这个镜像 **docker commit**

先创建一个容器

docker run -it ubuntu:15.10 /bin/bash

docker commit -m=”提交的描述信息” -a=”指定镜像作者” 容器ID 目标镜像名

* + - 1. **docker build**

docker build -t 镜像名 . (一定注意最后有一个.表示这个dockerfile文件的绝对路径)

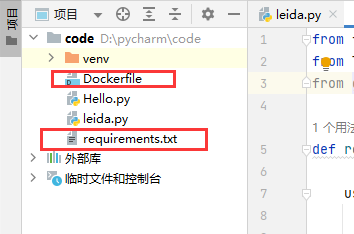
## Docker仓库

* + 1. 登录账号 docker login
    2. 注销账号 docker logout

# Dockerfile使用(封装tar文件)

参考本篇博客[简单Win10版Docker+Python封装\_docker封装python程序\_赫凯的博客-CSDN博客](https://blog.csdn.net/u010095372/article/details/125961361)

## 在执行代码路径下创建Dockerfile文件(一定要是文件而不是其他类型)和requirements文件



然后根据自己的需求编写dockerfile内容和代码所需的requirements内容

## 其中dockerfile内容的格式如下:

步骤可以参考本篇博客[将把python项目打包成Docker镜像（linux版）\_python打包docker镜像\_monkey\_susu的博客-CSDN博客](https://blog.csdn.net/sinat_37935727/article/details/125875287)

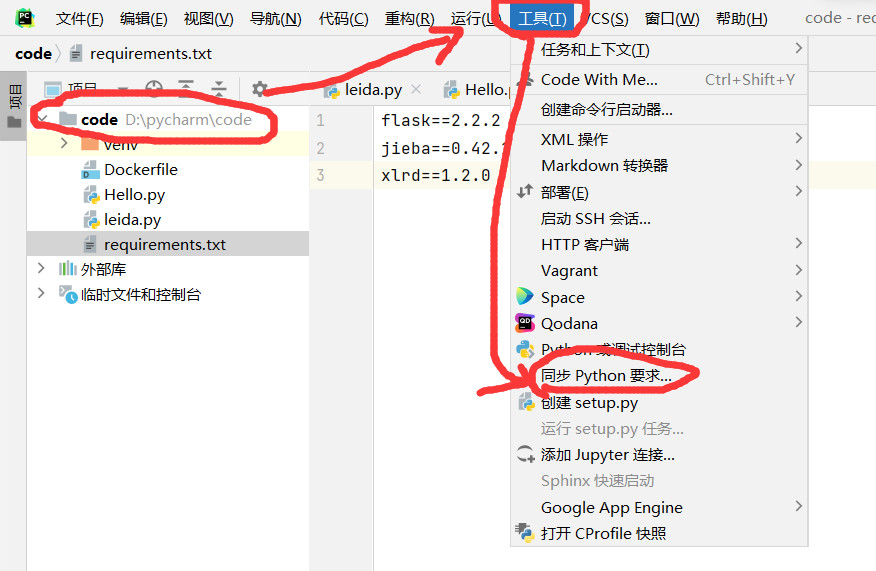


例如：

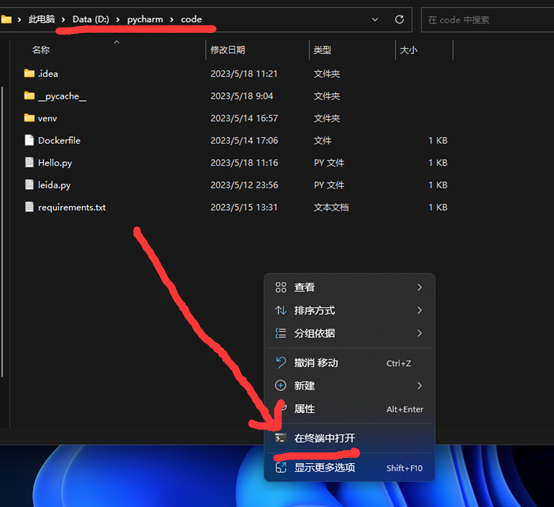


## Requirements内容可以根据如下步骤得出:

步骤可以参考本篇博客[如何使用PyCharm自动生成requirements.txt\_pycharm requirements\_COCO56（徐可可）的博客-CSDN博客](https://blog.csdn.net/COCO56/article/details/117452383)



## 在Dockerfile文件目录下右键选择在终端中打开：



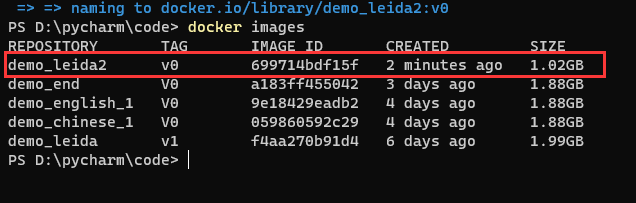
## 使用docker build语句生成docker镜像

docker build -t demo\_leida2:v0 .



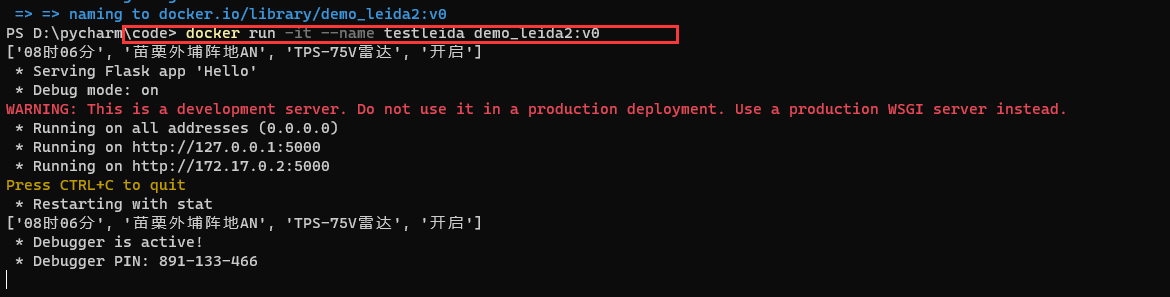
## docker images 查看是否生成成功

docker images



## docker run -it -–name 容器名 镜像名 启动一下

docker run -it --name testleida demo\_leida2:V0



要想退出就按Ctrl+v

## docker save -o 自定名.tar 镜像名

docker save -o leida.tar demo\_leida2:v0



