Docker安装开发环境

Docker安装Nginx

Nginx 是一个高性能的 HTTP 和反向代理 web 服务器，同时也提供了 IMAP/POP3/SMTP 服务 。

1、查看可用的 Nginx 版本

访问 Nginx 镜像库地址： https://hub.docker.com/_/nginx?tab=tags

可以通过 Sort by 查看其他版本的 Nginx，默认是最新版本 **nginx:latest**。

2、取最新版的 Nginx 镜像

```
docker pull nginx:latest
```

3、查看本地镜像

使用以下命令来查看是否已安装了 nginx：

```
docker images
```

4、运行容器

安装完成后，我们可以使用以下命令来运行 nginx 容器

```
docker run -it --name nginx-80 --rm -d -p 80:80 nginx
```

> 参数说明：
>
> - **--name nginx-test**：容器名称。
> - **-p 80:80**： 端口进行映射，将本地 8080 端口映射到容器内部的 80 端口。
> - **-d nginx**： 设置容器在在后台一直运行。

5、安装成功

最后我们可以通过浏览器可以直接访问 80 端口的 nginx 服务

6、自定义配置

> mkdir -p /usr/local/docker/nginx //创建指定目录配置

示例1 静态资源配置：

静态资源直接访问。直接将静态资源文件复制到nginx html下即可。

```
# docker cp  宿主机：本地目录   将需要进行静态处理的文件复制到 /usr/local/docker/nginx目录下
docker cp f4:/usr/share/nginx/html /usr/local/docker/nginx

#启动Nginx 静态资料配置
docker run -it --name nginx-80 --rm -d -p 80:80 -v
/usr/local/docker/nginx/html:/usr/share/nginx/html nginx
```

浏览器直接访问： http://服务器地址:80/qfnj/index.html

示例2 反向代理：

为方便，可将 nginx.conf 和 conf.d/default.conf 两个文件合并成一个文件再重新配置访问

```
 server_name  exam_qf;
 location / {
        proxy_pass http://127.0.0.1:8080;
  }
```

```
docker run -it --name nginx-80 --rm -d -p 80:80  -v
/usr/local/docker/nginx/nginx.conf:/etc/nginx/nginx.conf  nginx
```

示例3 负载均衡

nginx.conf

```
 upstream nginxCluster{
        server 192.168.20.136:8080;
        server 192.168.20.136:8081;
   }

    server {
    listen       80;
    server_name  localhost;

    #charset koi8-r;
    #access_log  /var/log/nginx/host.access.log  main;

     location /{
          proxy_pass http://nginxCluster;
     }
    }
```

```
#复制 docker容器内nginx.conf 到本地指定目录下
docker run -it --name nginx-80 --rm -d -p 80:80  -v
/usr/local/docker/nginx/nginx.conf:/etc/nginx/nginx.conf  nginx
```

Docker安装MySQL

安装MySQL 5.* 版本

1、搜索镜像

```
docker search mysql
```

2、下载镜像

```
docker pull mysql:5.6
```

3、创建并启动MySQL容器

```
docker run -d  --name mysql5.6-3306  -p 3306:3306   -e MYSQL_ROOT_PASSWORD='guoweixin'  mysql:5.6
```

docker run -d --name mysql5.6-3306 -p 3306:3306 -e MYSQL_ROOT_PASSWORD='Guoweixin927!' mysql:5.6

4、访问测试 进入到容器内部 Guoweixin927!

```
docker exec -it mysql5.6-3306 bash
```

连接mysql数据库：

```
mysql -u root -p
```

输入数据库密码 即可完成。

5、授权其他机器登陆 1、授权主机访问：

```
MySQL>GRANT ALL PRIVILEGES ON *.* TO 'root'@'%' IDENTIFIED BY 'guoweixin' WITH GRANT OPTION;
```

2、刷新权限：

```
MySQL>FLUSH PRIVILEGES;
```

3、退出：

```
MySQL>EXIT;
```

> 注意：（如果阿里云服务器部署和访问，防火墙默认是关闭的，需要手动开启） https://www.cnblogs.com/kcc dzz/p/8110143.html 阿里云控制组中加入该权限： https://blog.csdn.net/yenange/article/details/89499840

安装MySQL 8.* 版本

```
docker pull mysql
#启动
docker run -d --name mysql8 -p 3306:3306 -e MYSQL_ROOT_PASSWORD=guoweixin  mysql

#进入容器
docker exec -it mysql8 bash

#登录mysql
mysql -u root -p
ALTER USER 'root'@'localhost' IDENTIFIED BY 'guoweixin';
```

> 参数说明：
>
> - **-p 3306:3306** ： 映射容器服务的 3306 端口到宿主机的 3306 端口，外部主机可以直接通过 **宿主机ip:3306** 访问到 MySQL 的服务。
> - **MYSQL_ROOT_PASSWORD=guoweixin** ： 设置 MySQL 服务 root 用户的密码。

docker run -d --name mysql8 -p 3306:3306 -e MYSQL_ROOT_PASSWORD=Guoweixin927! mysql

ALTER USER 'root'@'localhost' IDENTIFIED BY 'Guoweixin927!';

Docker 安装Redis

安装单机版Redis

1、搜索redis

```
docker search redis
```

2、下载镜像

```
docker pull redis:4.0.1
```

3、创建并运行容器

```
docker run --rm -d --name redis6379 -p 6379:6379 redis:4.0.1 --requirepass "guoweixin"
```

4、测试Redis

```
docker exec -it redis6379 bash   //进入redis命令
redis-cli    //开启客户端功能
```

低内存空间开启swap(阿里云)

https://www.techieliang.com/2018/11/1283/

内存过小docker可能会出现exited(137)错误，随机关闭某个容器，我这mysql被关了好几次。这实际上是os关的，并非docker。可以建立swap交换空间。

查看当前已有swap大小

```
free -m
total used free shared buff/cache available
Mem: 992 436 75 34 480 364
Swap: 0 0 0
```

可以看到 Swap 只有0，下面我们来扩大到2G，为什么2G，因为我这个物理内存是1G，一般大小建议如下：

物理内存	建议的交换空间大小	如果开启休眠功能建议的交换空间大小
? 2GB	2倍	3倍
> 2GB – 8GB	相等	2倍
> 8GB – 64GB	>4GB	1.5倍
> 64GB	>4GB	不推荐休眠

创建一个 Swap 文件

首先cd到一个想要创建文件的地方，要注意这个目录所在硬盘分区要有大于所要创建大小的空间

```
mkdir /swap
cd /swap
sudo dd if=/dev/zero of=swapfile bs=1024 count=2000000
```

然后会提示创建成功，创建的大小、用时、拷贝速度

将普通文件转换成 Swap 文件

```
sudo mkswap -f swapfile
```

给出如下提示：

```
Setting up swapspace version 1, size = 1999996 KiB
no label, UUID=XXXXXXXXXXX
```

激活 Swap 文件

```
sudo swapon swapfile
```

再次查看 free -m 的结果

卸载Swap文件及自动挂载

如果需要卸载这个 swap 文件，可以进入建立的 swap 文件目录。执行下列命令。

```
sudo swapoff swapfile
```

如果需要一直保持这个 swap开机自动挂载，可以把它写入 /etc/fstab 文件。

```
/XXXX/swapfile /XXXX swap defaults 0 0
```

补充dd指令含义

dd：用指定大小的块拷贝一个文件，并在拷贝的同时进行指定的转换

注意：指定数字的地方若以下列字符结尾，则乘以相应的数字：b=512；c=1；k=1024；w=2

```
1. if=文件名：输入文件名，缺省为标准输入。即指定源文件。< if=input file >
2. of=文件名：输出文件名，缺省为标准输出。即指定目的文件。< of=output file >
3. ibs=bytes：一次读入bytes个字节，即指定一个块大小为bytes个字节。
  obs=bytes：一次输出bytes个字节，即指定一个块大小为bytes个字节。
  bs=bytes：同时设置读入/输出的块大小为bytes个字节。
4. cbs=bytes：一次转换bytes个字节，即指定转换缓冲区大小。
5. skip=blocks：从输入文件开头跳过blocks个块后再开始复制。
6. seek=blocks：从输出文件开头跳过blocks个块后再开始复制。
注意：通常只用当输出文件是磁盘或磁带时才有效，即备份到磁盘或磁带时才有效。
7. count=blocks：仅拷贝blocks个块，块大小等于ibs指定的字节数。
8. conv=conversion：用指定的参数转换文件。
  ascii：转换ebcdic为ascii
  ebcdic：转换ascii为ebcdic
  ibm：转换ascii为alternate ebcdic
  block：把每一行转换为长度为cbs，不足部分用空格填充
  unblock：使每一行的长度都为cbs，不足部分用空格填充
  lcase：把大写字符转换为小写字符
  ucase：把小写字符转换为大写字符
  swab：交换输入的每对字节
  noerror：出错时不停止
  notrunc：不截短输出文件
  sync：将每个输入块填充到ibs个字节，不足部分用空（NUL）字符补齐。
```

上述建立过程就是bs=1024 count=2000000，以1024位块大小，建立2000000个块，总共2048000000字节，所以显示的不是2GB大小，因为不是2097152字节。