链滴

Docker 的使用

作者：howie404

原文链接：https://ld246.com/article/1561778938923

来源网站：链滴

<img src="https://ld246.com/images/img-loading.svg" alt="" data-src="https://b3logfile
com/bing/20190407.jpg?imageView2/1/w/960/h/540/interlace/1/q/100">
<h2 id="Docker容器技术">Docker 容器技术</h2>
<h3 id="Docker安装与启动">Docker 安装与启动</h3>
注意：因为 docker 需要联网操作，所以需要将虚拟机配置成 NAT 网络模式。Docker 必须安装
centos7.0 以上

挂载资料中提供好的 docker-test 镜像　可以通过 ip addr 命令查看 Linux 系统 ip 地址
安装 docker　yum install docker　通过 docker -v 查看安装的 docker 版本
docker 启动与停止

<pre><code class="highlight-chroma"><span class="highlight
cl">启动docker：systemctl start docker
停止docker：syst
mctl stop docker
重启docker：syst
mctl restart docker
查看docker状态：s
stemctl status docker
开机启动：systemc
l enable docker
</code></pre>
<h3 id="Docker镜像操作">Docker 镜像操作</h3>

列出镜像　docker images

配置使用镜像加速器提交拉取速度。修改/daemon.json 文件，vi /etc/docker/daemon.json</
>
<pre><code class="highlight-chroma"><span class="highlight
cl">{
"registry-mirrors":
"https://docker.mirrors.ustc.edu.cn"]
}

</code></pre>

重启 docker　 systemctl restart docker

拉取 centos 镜像
<pre><code class="highlight-chroma"><span class="highlight
cl">docker pull centos:7
</code></pre>

搜索镜像　搜索网络上镜像
<pre><code class="highlight-chroma"><span class="highlight
cl">//例如搜索tomcat
docker search to
cat
</code></pre>

删除镜像 **注意：**包起 docker images -q 命令的符合是~,是键盘上数字 1 左侧的符合。</
>
<pre><code class="highlight-chroma"><span class="highlight
cl">1、 docker rmi IMAGE_ID：删除指定镜像
2、 docker rmi `
ocker images -q`：删除所有镜像
</code></pre>

<h3 id="Docker容器操作">Docker 容器操作</h3>
<pre><code class="highlight-chroma"><span class="highlight
cl">-i:运行容器
-t:启动后进入命令

--name=为创建的
器命名
-d:后台启动
-v:目录映射关系
-p:端口映射
</code></pre>

创建交互式容器启动
<pre><code class="highlight-chroma"><span class="highlight
cl">// 当前版本 进入后需要执行的解释器
docker run -it --n
me=mycentos centos:7 /bin/bash
执行完该命令后，
接进入容器内部，退出通过exit，容器不能重名，一旦退出，容器也处于停止状态
</code></pre>

创建守护式容器启动 命令：。 执行完该命令后，。。。
<pre><code class="highlight-chroma"><span class="highlight
cl">//不进入容器内部
docker run -di --n
me=mycentos2 centos:7
//进入容器的命令
（exit退出时，容器不会停止）
docker exec -it co
tainer_name (或者 container_id) /bin/bash
</code></pre>

查看容器
<pre><code class="highlight-chroma"><span class="highlight
cl"> //查看正在运行的容器：docker ps
 //查看所有的容器
docker ps -a
</code></pre>

停止与启动容器
<pre><code class="highlight-chroma"><span class="highlight
cl">//停止容器：docker stop container_name/ID
//启动容器：docke
 start container_name/id
</code></pre>

删除容器　注意：只能删除停止的容器，正在运行的容器不能被删除
<pre><code class="highlight-chroma"><span class="highlight
cl">删除指定的容器：docker rm $CONTAINER_ID/NAME
删除所有容器：do
ker rm `docker ps -a -q`
</code></pre>

<h3 id="部署应用">部署应用</h3>
<h4 id="mysql部署">mysql 部署</h4>

拉取 mysql 镜像
<pre><code class="highlight-chroma"><span class="highlight
cl"> docker pull mysql
</code></pre>

创建 mysql 容器　（守护式）

<pre><code class="highlight-chroma"><span class="highlight
cl">//   宿主机端口:容器端口   容器mysql密码 镜像名
docker run -di --n
me=pinyougou_mysql -p 33306:3306 -e MYSQL_ROOT_PASSWORD=123456 mysql
</code></pre>
<ol start="3">
进入 mysql 容器

<pre><code class="highlight-chroma"><span class="highlight
cl">docker exec -it pinyougou_mysql /bin/bash
</code></pre>
<ol start="4">

在容器中登陆 mysql
<pre><code class="highlight-chroma"><span class="highlight
cl"> 进入命令：mysql -u root -p
 退出mysql 命令：
uit
</code></pre>

远程登录 mysql **注意：**在 navicat 链接 mysql8 以后的版本时，会出现 2059 的错误，这错误出现的原因是在 mysql8 之前的版本中加密规则为 mysql_native_password，而在 mysql8 以的加密规则为 caching_sha2_password。解决此问题有两种方法，一种是更新 navicat 驱动来解决问题，一种是将 mysql 用户登录的加密规则修改为 mysql_native_password。本文采用第二种方式

<pre><code class="highlight-chroma"><span class="highlight
cl">ALTER USER 'root'@'%' IDENTIFIED WITH mysql_native_password BY '123456';#更新一下
户的密码
FLUSH PRIVILEGES
#刷新权限
</code></pre>
<ol start="6">
在 Navicat 中运行品优购建表语句的 sql 脚本文件，生成品优购数据库、表以及插入测试数据</l
>
查看容器 ip **注意：**在宿主机中查看，因为后续需要容器连接容器数据库

<pre><code class="highlight-chroma"><span class="highlight
cl">我们可以通过以下命令查看容器运行的各种数据
docker inspect pi
yougou_mysql
也可以直接执行下
的命令直接输出IP地址
docker inspect --f
rmat='{{.NetworkSettings.IPAddress}}' pinyougou_mysql
</code></pre>
<h4 id="tomcat部署">tomcat 部署</h4>

拉取 tomcat 镜像
<pre><code class="highlight-chroma"><span class="highlight
cl">   //版本和tomcat中的环境
docker pull tomcat
7-jre7
</code></pre>

运行 tomcat 容器
<pre><code class="highlight-chroma"><span class="highlight
cl">docker run -di --name pinyougou_tomcat -p 9000:8080 tomcat:7-jre7

//也可以用如下方
进行创建 这样将宿主机的/usr/local/myhtml文件夹映射到容器内的/usr/local/tomcat/webapps目
中，privilege是为了修改一些如/etc/sys里面的文件的时候才需要
docker run -di --n
me=pinyougou_tomcat -v /usr/local/myhtml:/usr/local/tomcat/webapps --privileged=true -p
9000:8080 tomcat:7-jre7
</code></pre>

部署 web 应用 CAS 单点登录部署
1.修改 cas 系统的配置文件，修改数据库连接的 url
<pre><code class="highlight-chroma"><span class="highlight
cl"><bean id="dataSource" class="com.mchange.v2.c3p0.ComboPooledDataSource"
        p:driverClass
"com.mysql.jdbc.Driver"
        p:jdbcUrl="j

```
bc:mysql://172.17.0.3:3306/pinyougoudb?characterEncoding=utf8"
</span></span><span class="highlight-line"><span class="highlight-cl">        p:user="root"
</span></span><span class="highlight-line"><span class="highlight-cl">        p:password=
123456" /&gt;
</span></span></code></pre>
<p>2.将 cas 文件夹上传至宿主机  通过 alt+p 进入上传窗口，上传 cas 文件夹</p>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight
cl">// -r 递归上传
</span></span><span class="highlight-line"><span class="highlight-cl">put -r E:\item\pin
ougou306\apache-tomcat-cas\webapps\cas
</span></span></code></pre>
<p>3.从宿主机中，将 cas 文件夹拷贝到到容器/usr/local/tomcat/webapps 目录</p>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight
cl">docker cp cas pinyougou_tomcat:/usr/local/tomcat/webapps
</span></span></code></pre>
<p>4.测试：地址栏输入：  <a href="https://ld246.com/forward?goto=http%3A%2F%2F192.168
25.143%3A9000" target="_blank" rel="nofollow ugc">http://192.168.25.143</a>/cas/login
主机的 ip 地址</p>
</li>
</ol>
<h4 id="Nginx部署">Nginx 部署</h4>
<ol>
<li>
<p>拉取 Nginx 镜像</p>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight
cl">docker pull nginx
</span></span></code></pre>
</li>
<li>
<p>创建 Nginx 容器</p>
</li>
</ol>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight
cl">docker run -di --name=pinyougou_nginx -p 80:80  nginx
</span></span></code></pre>
<ol start="3">
<li>
<p>测试 Nginx  访问：  <a href="https://ld246.com/forward?goto=http%3A%2F%2F192.168.25
143" target="_blank" rel="nofollow ugc">http://192.168.25.143</a></p>
</li>
<li>
<p>配置反向代理 从容器拷贝配置文件到宿主机</p>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight
cl">//从容器拷贝反向代理文件到本地，修改后再拷贝回容器
</span></span><span class="highlight-line"><span class="highlight-cl">默认挂载html目录:
usr/share/nginx/html
</span></span><span class="highlight-line"><span class="highlight-cl">docker cp pinyou
ou_nginx:/etc/nginx/nginx.conf nginx.conf
</span></span></code></pre>
<p>配置反向代理</p>
</li>
</ol>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight
cl">upstream tomcat-cas {
```

    server 172.17.0.
:8080;
}
server {
    listen 80;
    server_name pa
sport.pinyougou.com;
    location / {
        proxy_pass ht
p://tomcat-cas;
        index index.h
ml index.htm;
    }
}
</code></pre>
将修改后的配置文件拷贝到容器
<pre><code class="highlight-chroma"><span class="highlight
cl">docker cp nginx.conf pinyougou_nginx:/etc/nginx/nginx.conf
</code></pre>
重启容器
<pre><code class="highlight-chroma"><span class="highlight
cl">docker restart pinyougou_nginx
</code></pre>
通过 SwitchHosts 新增 ip 地址和域名绑定
<pre><code class="highlight-chroma"><span class="highlight
cl">192.168.184.88 passport.pinyougou.com
</code></pre>
重新启动容器
<pre><code class="highlight-chroma"><span class="highlight
cl">docker restart pinyougou_nginx
</code></pre>
<h4 id="Redis部署">Redis 部署</h4>

拉取 Redis 镜像
<pre><code class="highlight-chroma"><span class="highlight
cl">//拉取镜像
docker pull redis
</code></pre>

创建 Redis 容器

<pre><code class="highlight-chroma"><span class="highlight
cl"> docker run -di --name=pinyougou_redis -p 6379:6379 redis
</code></pre>
<ol start="3">
在你的本地电脑命令提示符下，用 window 版本 redis 测试

<pre><code class="highlight-chroma"><span class="highlight
cl">redis-cli -h 192.168.247.135
</code></pre>
<h3 id="备份与迁移">备份与迁移</h3>

容器保存为镜像
<pre><code class="highlight-chroma"><span class="highlight
cl">   容器名   保存的镜像名
docker commit pi
yougou_nginx mynginx
</code></pre>
镜像备份
<pre><code class="highlight-chroma"><span class="highlight
cl">   文件名   容器名
docker save -o m
nginx.tar mynginx
</code></pre>
镜像恢复与迁移
<pre><code class="highlight-chroma"><span class="highlight
cl">测试删掉 docker rmi myniginx
//恢复加载镜像
docker load -i my
ginx.tar
</code></pre>
修改 ip 地址
vim /etc/sysconfig/network-scripts/ifcfg-ens33
查看版本
cat /etc/os-release
rpm -qa|grep docker -查看 docker
yum -y remove docker --删除 docker
rm -rf /var/lib/docker