Jenkins

linux安装

先安装java

```
#先安装java jdk
#下载
 wget --no-check-certificate --no-cookies --header
"Cookie: oraclelicense=accept-securebackup-cookie"
http://download.oracle.com/otn-pub/java/jdk/8u131-
b11/d54c1d3a095b4ff2b6607d096fa80163/jdk-8u131-
linux-x64.rpm
#授权限
 chmod +x jdk-8u131-linux-x64.rpm
 #安装
rpm -ivh jdk-8u131-linux-x64.rpm
```

```
# Jenkins 系统管理-脚本命令行,执行指令,修改时间

System.setProperty('org.apache.commons.jelly.tags.fm
t.timeZone', 'Asia/Shanghai')

# 下载jenkins.war ,切换到当前目录 执行
#修改配置文件之前,不要后台启动 关闭窗体就停止
java -jar jenkins.war --httpPort=8081
#后台运行 --后台启动,窗体关闭了,也在后台启动
nohup java -jar jenkins.war --httpPort=9999 &

输入jps 查看当前jenkins启动的进程号
ps ef|gr jenkins 查询当前启动的jenkins的进程号
```

```
如果要关闭,kill -9 端口号
```

启动完成之后打开地址 http://ip:8081

然后根据提示找到密码输入就OK

修改下面操作,更新使用, 如果更新不成功,则多更新几次

修改下列文件之后,记得要给两个文件权限

1、修改根目录文件/home/jenkins/hudson.model.UpdateCenter.xml

```
<sites>
  <site>
    <id>default</id>

 <url>http://mirror.xmission.com/jenkins/updates/update-center.json</url>
  </site>
</sites>
```

2、等jenkins初始化完成后找到跟目录下面updates文件中`default.json`文件做如下修改

cd /home/jenkins/updates/

设置 default.json 权限 安装插件什么的时候,不需要google.com,改成百度

切换当刚开始进去的时候,需要密码的那个路径

cd /root/.jenkins/updates/

然后执行下面的

```
sed -i 's/http:\/\/updates.jenkins-ci.org\/download/https:\/\/mirrors.tuna.tsinghua.edu.cn\/jenkins/g' default.json && sed -i 's/http:\/\/www.google.com/https:\/\/www.baidu.com/g' default.json
```

修改完之后,需要重新启动

新增工作流

docker安装

```
docker run -d --name jenkins -p 9090:8080 -v /home/jenkins:/var/jenkins_home -u 0 jenkins

#查看第一次登录的密码
/home/jenkins/secrets/initialAdminPassword

#升级
wget http://updates.jenkins-ci.org/download/war/2.239/jenkins.war
```

1、修改根目录文件/home/jenkins/hudson.model.UpdateCenter.xml

```
<sites>
  <site>
    <id>default</id>

 <url>http://mirror.xmission.com/jenkins/updates/update-center.json</url>
  </site>
</sites>
```

2、等jenkins初始化完成后找到跟目录下面updates文件中`default.json`文件做如下修改

cd /home/jenkins/updates/

设置 default.json 权限

```
sed -i 's/http:\/\/updates.jenkins-ci.org\/download/https:\/\/mirrors.tuna.tsinghua.edu.cn\/jenkins/g' default.json && sed -i 's/http:\/\/www.google.com/https:\/\/www.baidu.com/g' default.json
```

3、修改完成后必须重启，还有如果文件没有修改权限添加写的权限

手动升级

可以去Jenkins的官网下载好最新jar包上传到服务器，也可以使用`wget`命令。

```
wget http://jenkins新版本的下载地址
#目前最新2.239
wget http://updates.jenkins-ci.org/download/war/2.239/jenkins.war
```

Jenkins的更新主要是替换jenkins镜像里面的war包，我们可以把下载好的war包使用`docker cp`直接进行复制命令如下：

```
docker cp jenkins.war jenkins:/usr/share/jenkins
```

重新启动Jenkins即可完成升级。

```
docker restart jenkins
```

安装svn

```
docker run --privileged=true --restart always --name
svn-server -d -v /home/svn:/var/opt/svn -p 3690:3690
garethflowers/svn-server

#进入运行的svn容器，做一些配置工作
docker exec -it svn-server /bin/sh

创建仓库
svnadmin create repo
在进入的容器中 使用此命令
创建成功后生成repo目录，进入该目录下的conf配置文件夹。有以下
几个文件authz, passwd, svnserve.conf
authz 是权限控制，可以设置哪些用户可以访问哪些目录,
```

使用.netcore 实现服务持续集成和发布

https://github.com/ 注册一个用户名和密码

```
在安装jenkins的服务执行这句指令,安装git服务
yum install -y git  **/bin  **/obj
```

持续集成和发布的指令

```
#这些指令都是docker

#!/bin/bash
# 获取短版本号
GITHASH=`git rev-parse --short HEAD`
echo ---------------开始编译程序...------------------
echo ---------------Building Docker Image...--------
----------
docker build -t demo:$GITHASH .
docker tag demo:$GITHASH demo:latest
echo ---------------Launching Container...----------
--------
docker run -d -p 5001:80 --name demo1  demo:latest
```

```
# 第二次
#!/bin/bash
# 获取短版本号
GITHASH=`git rev-parse --short HEAD`
echo ---------------开始编译程序...------------------
echo ---------------Building Docker Image...--------
----------
docker build -t demo:$GITHASH .
docker tag demo:$GITHASH demo:latest
echo ---------------Launching Container...----------
--------
docker rm -f demo1
docker run -d -p 5001:80 --name demo1  demo:latest
```

首先我们需要一个镜像仓库--

```
# 第二次
#!/bin/bash
# 获取短版本号
GITHASH=`git rev-parse --short HEAD`
echo ---------------开始编译程序...------------------
echo ---------------Building Docker Image...--------
----------
docker build -t demo:$GITHASH .
docker tag demo:$GITHASH demo:latest
echo ---------------Launching Container...----------
--------

 # 需要把当前的镜像推送到我们的仓库里面
 # 吧当前的镜像名称+版本储存在数据库
```

我们界面化的管理我们所有的linux服务

当前最后一个镜像名称+版本号 右边选择我们自己服务--10000

docker pull image:tag

docker run ----

回滚---

一级二级三级--

代码单元测试

docker

msbulid --编译.net 代码

2.提供的接口--- 你需要把这些接口名称参数通过界面化配置--限流..

3.当前这个模块打包的时候,需要哪些配置选择,打包的时候,需要哪些dll

image 提交上去

开发,运行在开发模板机上面,自己测试 ---申请

测试人员

镜像发布到测试环境--申请

开发--运行在我们灰度环境--申请

每一次发布--除了开发人员做的之外,其他时间程序员---

LINUX 服务监控工具

```
docker run -d --name=netdata -p 19999:19999 -v
netdatalib:/var/lib/netdata -v
netdatacache:/var/cache/netdata  -v
/etc/passwd:/host/etc/passwd:ro -v
/etc/group:/host/etc/group:ro  -v
/proc:/host/proc:ro -v /sys:/host/sys:ro -v /etc/os-
release:/host/etc/os-release:ro  --restart unless-
stopped   --cap-add SYS_PTRACE   --security-opt
apparmor=unconfined   netdata/netdata
```

f