

- 作者: MingGH
- 原文链接: https://ld246.com/article/1672835504167
- 来源网站:链滴
- 许可协议: 署名-相同方式共享 4.0 国际 (CC BY-SA 4.0)

原文发布于: Java开发者必看! 手把手带你搞定Jenkins+Maven仓库+Docker, 欢迎使用 RSS 订阅 取最新更新。

1. 写在开头

为了调通这长长的一串花费了整整4天时间。。。写完后将之前写的API都采取了这样的方式进行部署极大程度的压缩了需要我部署的时间,还是非常值得的。

比起上次写的 使用Jenkins对springboot项目进行docker镜像一键部署,jenkins + docker + sprin boot 集成了更多的内容,也更加动态化。

因为流程很长,如果你在看这篇博客的时候有任何问题可以通过博客主页的邮箱联系我,我会提供我帮助到的。

1.1 实现的功能

● 一次构建, 可以完成如下所有操作

checkout code \rightarrow build jar package \rightarrow push package to nexus \rightarrow build docker image \rightarrow pu h image to repositories \rightarrow deploy \rightarrow update pom.xml version

Average stage times: (Average full run time: -3min 27s)	Declarative: Checkout SCM 4s	Build 4s	Unit Test 3s	Push Nexus	Package Image 28s	Push Image	Deploy 1s	Update version 9s
(#125) Jan 03 18 08:04 commits	4s	4s	35	1min 18s	28s	1min 9s	1s	95

当中没有接入Sonarqube或者墨菲安全之类的stage是因为我把这个部份做到了code一提交时触发的 itHub Actions中。

废话不多说,马上开始教程。

2. 搭建过程

2.1 准备内容

搭建环境:

● 一个简单的Spring Boot项目,使用Java17,我已经准备好了这个,建议先clone到本地: https:/github.com/MingGH/demo-springboot-simple

● 阿里云账号 (推送nexus和docker image)

● 一台已经安装了docker的电脑或者服务器,如果服务器安装docker有难度,可以参考这篇博客: 用官方安装脚本自动安装

2.2 通过Docker安装Jenkins

创建一个目录,用来存在Jenkins的数据

mkdir -p /dockerData/jenkins/jenkins-data

进入到 /dockerData/jenkins 目录,我们在这创建 Dockerfile

cd /dockerData/jenkins vim Dockerfile

复制以下内容到文件 Dockerfile 中

FROM jenkins/jenkins:2.375.1 USER root RUN apt-get update && apt-get install -y lsb-release RUN curl -fsSLo /usr/share/keyrings/docker-archive-keyring.asc \ https://download.docker.com/linux/debian/gpg RUN echo "deb [arch=\$(dpkg --print-architecture) \ signed-by=/usr/share/keyrings/docker-archive-keyring.asc] \ https://download.docker.com/linux/debian \ \$(lsb_release -cs) stable" > /etc/apt/sources.list.d/docker.list RUN apt-get update && apt-get install -y docker-ce-cli USER jenkins RUN jenkins-plugin-cli --plugins "blueocean:1.26.0 docker-workflow:563.vd5d2e5c4007f"

从Dockerfile中构建镜像

docker build -t myjenkins-blueocean:2.375.1-1.

构建完成之后,使用docker images可以看到刚刚构建的镜像。

运行一个Jenkins容器

docker run \ -u root \ --name jenkins \ --restart=on-failure \ --detach \ --publish 8080:8080 \ --publish 50000:50000 \ --volume /etc/localtime:/etc/localtime \ --volume /var/run/docker.sock:/var/run/docker.sock -v /usr/bin/docker:/usr/bin/docker \ --volume /dockerData/jenkins/jenkins-data:/var/jenkins_home \ myjenkins-blueocean:2.375.1-1

对当中的一些参数进行解释:

- -u root 容器中的进程以root用户权限运行
- --restart=on-failure如果容器由于错误而退出,则将其重新启动
- --detach 保持容器在后台持续运行
- --publish 8080:8080 映射宿主机8080端口给容器8080端口
- --publish 50000:50000映射宿主机50000端口给容器50000端口
- --volume /etc/localtime:/etc/localtime 容器时间如何与宿主机同步

• --volume /var/run/docker.sock:/var/run/docker.sock -v /usr/bin/docker:/usr/bin/docker 挂 宿主机的 /var/run/docker.sock 给容器,这样Jenkins容器就可以调用宿主机的docker,创建其他的

器服务于CICD

• --volume /dockerData/jenkins/jenkins-data:/var/jenkins_home 挂载Jenkins容器的数据到宿 机目录下

解决ECDSA host key is known for github.com and you have requested strict checking问题

运行成功之后,别急,还有一步需要操作,进入到Jenkins容器配置ssh,如果不进行配置当拉取代码 时候会抛出异常ECDSA host key is known for github.com and you have requested strict checki g,你也可以在这篇文章找到更加详细的说明:Jenkins执行pipeline抛出异常No ECDSA host key is nown for github.com and you****

进入Jenkins容器并执行以下操作

docker exec -it -u root jenkins /bin/bash
mkdir -p /root/.ssh
cd ~/.ssh/
touch known_hostsknown_hosts
ssh-keyscan github.com >> ~/.ssh/known hosts

你会看到以下输出,就表示ok了

```
root@240a8671ca6c:~/.ssh# ssh-keyscan github.com >> ~/.ssh/known_hosts
# github.com:22 SSH-2.0-babeld-408889af
```

然后输出 exit可以直接推出容器

如果你是某某云的服务器,需要到某某云的控制台开放安全组端口才能进行访问

现在我们就可以以 ip+端口的形式进行访问到Jenkins, 打开浏览器 http://服务器公网ip:8080 提示 们需要输入密码。

入门

解锁 Jenkins

为了确保管理员安全地安装 Jenkins, 密码已写入到日志中(**不知道在哪里?**)该文件在服务器上:

/var/jenkins_home/secrets/initialAdminPassword

请从本地复制密码并粘贴到下面。

管理员密码

我们可以直接使用 docker logs jenkins 拿到密码

2023-01-03 15:00:58.233+0800	[id=1]	INFO	o.e.j.s.s.DefaultSessionIdManager#doStart: Session workerName=node0
2023-01-03 15:00:58.705+0000			hudson.WebAppMain#contextInitialized: Jenkins home directory: /var/jenkins_home found at: EnvVars.masterEnv
2023-01-03 15:00:58.850+0000 ins_home/war}			o.e.j.s.handler.ContextHandler#doStart: Started w.@6b8280e6{Jenkins v2.375.1,/,file:///var/jenkins_home/war
2023-01-03 15:00:58.865+0000	[id=1]	INFO	o.e.j.server.AbstractConnector#doStart: Started ServerConnector@64a8c844{HTTP/1.1, (http/1.1)}{0.0.0:0888
2023-01-03 15:00:58.878+0800	[id=1]	INFO	org.eclipse.jetty.server.Server#doStart: Started Server@278bb07e{STARTING}[10.0.12,sto=0] @2786ms
			winstone.Logger#logInternal: Winstone Servlet Engine running: controlPort=disabled
2023-01-03 15:00:59.121+0000	[id=32]	INFO	jenkins.InitReactorRunner\$1#onAttained: Started initialization
2023-01-03 15:00:59.259+0000			hudson.PluginManager#considerDetachedPlugin: Loading a detached plugin as a dependency: /var/jenkins.home/p
2023-01-03 15:00:59.314+0000 jpi			hudson.PluginManager#considerDetachedPlugin: Loading a detached plugin as a dependency: /var/jenkins_home/p
2023-01-03 15:01:00.186+0000 r.jpi			hudson.PluginManager#considerDetachedPlugin: Loading a detached plugin as a dependency: /var/jenkins_home/p
2023-01-03 15:01:00.194+0000	[id=45]	INFO	hudson.PluginManager#considerDetachedPlugin: Loading a detached plugin as a dependency: /var/ienkins.home/p
2023-01-03 15:01:01.671+0000	[1d=33]	INFO	jenkins.InitReactorRunnerS1#onAttained: Listed all plugins
2023-01-03 15:01:05.482+0000	[id=32]	INFO	jenkins.InitReactorRunner\$1#onAttained: Prepared all plugins
2023-01-03 15:01:05.495+0800	[id=44]	INFO	jenkins.InitReactorRunner\$1#onAttained: Started all plugins
2023-01-03 15:01:05.512+0000	[1d=38]	INFO	jenkins.InitReactorRunner\$1#onAttained: Augmented all extensions
WARNING: An illegal reflectiv	ve access	operatio	n has occurred
WARNING: Illegal reflective a	access by	org.code	haus.groovy.vmplugin.v7.lava7\$1 (file:/var/ienkins.home/war/WEB-INF/lib/groovy-all-2.4.21.jar) to constructor
odHandles\$Lookup(java.lang.C)	lass.int)		
WARNING: Please consider repo	orting thi		maintainers of org.codehaus.groovy.vmplugin.v7.Java751
WARNING: Use	s=warn to	enable w	arnings of further illegal reflective access operations
WARNING: All illegal access of	operations	will be	denied in a future release
2023-01-03 15:01:06.529+0800	[id=39]	INFO	jenkins.InitReactorRunner\$1#onAttained: System config loaded
2023-01-03 15:01:06.529+0000	[fid=39]	INFO	jenkins.InitReactorRunner\$1#onAttained: System config adapted
2023-01-03 15:01:06.534+0800			jenkins.InitReactorRunner\$1#onAttained: Loaded all jobs
2023-01-03 15:01:06.545+0000	[id=45]		jenkins.InitReactorRunner\$1#onAttained: Configuration for all jobs updated
2023-01-03 15:01:06.561+0800	[id=59]		hudson.util.Retrier#start: Attempt #1 to do the action check updates server
			er has been created and a password generated.
			o installation:
0129c6de6904416fb7d8b6e0047da			
	var/jenkin		ecrets/initialAdminPassword

			h.m.DownloadService\$Downloadable#load: Obtained the updated data file for hudson.tasks.Maven.MavenInstaller
			h.m.DownloadService\$Downloadable#load: Obtained the updated data file for hudson.tools.JDKInstaller
2023-01-03 15:03:05.540+0000			hudson.util.Retrier#start: Performed the action check updates server successfully at the attempt #1

2.3 Jenkins配置:时区

选择安装完推荐的插件之后, Jenkins会进行重启, 等待一段时间, 然后刷新登录



时区配置

进入到 系统管理→脚本命令行

Dashboard 》 系统管理	which has an even one permities process and so could bypass security restrictions. 1 scripts pending approval.
	状态信息
	系统信息 系统日志 系统日志 系统日志 负载统计 金示系统环境信息以帮助解决问题。 系统日志从java.util.logging擁获Jenkins相 关的日志信息。 本 負載於计 检查您的資源利用情况,看看是否需要更多的计算机未帮助您构建。
	A 关于Jenkins 童者版本以及证书信息。
	问题排查
	管理旧数應 从旧的、早期版本的插件中请理配置文件。
	工具和动作
	建取设置 放弃当前内存中所有的设置信息并从配置文件中 重新设取 仅用于当您手动排改配置文件时重新 读取设置。 Jenkins 命令行確却本访问或管理您的Jenkins。 载家命令行。 城市令行或期本访问或管理您的Jenkins。 如本命令行 执行用于管理或放弃提到或诊断的任意期本命 令。
	也 准备关机 停止执行新的构造任务以安全关闭计算机。

执行如下命令,设置时区为上海

System.setProperty('org.apache.commons.jelly.tags.fmt.timeZone', 'Asia/Shanghai')

十 新建任务		脚本命令行
22、用户列表		Type in an arbitrary Groovy script and execute it on the server. Useful for trouble-shooting and diagnostics. Use the 'printin' command to see the output (if you use system.out, it will go to the
圆 构建历史		server's stdout, which is harder to see.) Example:
③ 项目关系		println(Jenkins.instance.pluginManager.plugins)
参 检查文件指纹		All the classes from all the plugins are visible. jenkins.*, jenkins.model.*, hudson.*, and hudson.model.* are pre-imported.
③ 系統管理		1 System.setProperty('org.apache.commons.jelly.tags.fmt.timeTone', 'Asia/Shanghai')
□ 我的视图		
● 打开 Blue Ocean		
勾織队列	v	
列中没有构建任务		
如建执行状态	~	
空闲		
214		

2.4 Jenkins配置: GitHub凭据

回到系统管理,点击 Manage Credentials

Dashboard) 系统管理										
十 新建任务		Man	age Jenkins							
&空。用户列表										
四 构建历史		版向作	(建设置有误				更多信息 不再显示			
③ 项目关系		新版本的 Jenkins (2.384) 可以下置 (空間记录)。								
检查文件指纹										
③ 系統管理		系统配								
□ 我的视图										
🥥 打开 Blue Ocean		0	系統配置 配置全局设置和路径	P	全局工具配置 工具配置,包括它们的位置和自动安装器	2	備件管理 添加。删除、禁用或应用Jenkins功能扩展语 件。			
构建队列	~		11 A MID							
队列中设有构建任务		\bigcirc	》放業項 添加、删除、控制和监视系统运行任务的节点。							
档建执行状态	*									
1 空闲		安全								
2 空州		ð	全局安全配置 Jenkins 安全,定义谁可以访问或使用系统。	Co	Manage Credentials Configure credentials	P	決擴配置 配置凭線的提供者和类型			
		22	管理用户 创建,删除或终改 Jenkins 用户	11	In-process Script Approval Allows a Jenkins administrator to review proposed scripts (written e.g. in Groovy) which run inside the Jenkins process and so could bypass security restrictions. 1 scripts pending approval.					

创建新的凭据

SSH Username with private key 选择SSH Username with key	
128 2	
金局 (Jenkins, nodes, items, all child items, etc)	
ID II	
git-with-privatekey id58.8	
描述 2	
git-with-privatekey	
Usemane	
不用填写	
Treat username as secret	
Private Key	
Enter directly	
Key	
	Enter New Secret Below
编入完整private key	

private-key的来源:在部署Jenkins的服务器上生成对应的公钥

ssh-keygen -t rsa -b 4096 -C "your_email@example.com"

会有三个文件生成

[root@ecs-205431 .ssh]# ls
authorized_keys config config.save id_rsa id_rsa.pub known_hosts
[root@ecs-205431 .ssh]#

然后公钥文件 id_rsa.pub中的内容复制到GitHub SSH配置中

Search or jump to Pull requests	issues Codespaces Marketplace Explore	¢ +• 🕵
Earch or jump to Pull request 1	SSH Keys / Add new Title Key type Authentication Key 9 Begins with "sah-rsa", 'ecdsa-sha2-nistp256", 'ecdsa-sha2-nistp521', 'sah-ed25519, 'sak- ecdsa-sha2-nistp256@openssh.com' or 'sk-sah-ed25519@openssh.com'	Ge to your personal profile
왕 Copilot 한 Pages 수 Saved replies Security		

然后把私钥文件 id_rsa 中的内容配置到刚刚Jenkins的那个地方

Priv	ate
0	Enter directly Key
	Enter New Secret Below

id_rsa的文件内容类似:

-----BEGIN OPENSSH PRIVATE KEY-----

xxxxxxxxxxxxxxxxx == -----END OPENSSH PRIVATE KEY-----

生成公钥的问题可以参考这篇博客: Git SSH公钥配置

2.5 准备阿里云效的Maven仓库

地址: https://packages.aliyun.com/maven

注册并登录进去之后,我们可以看到有两个对应的仓库,**生产库-release**和**非生产库-snapshot**,别用来存放我们不同环境的jar包,如果你的 pom.xml中的version带 -**snapshot**,那么推送的就是 **生产库,**如果不带 -**snapshot**,推送的就是生产库。这里需要注意的是,生产库的jar包是不允许重 推送的。

… P 制品仓库 Packa	iges	自2022.12.12起,受 Maven 中央仓库网络限制,阿里云云戏 Maven 中央代理仓库可能会出现部分新增依赖查找不到的情况,但不影响已有依赖,请知悉		٥	0	0 ×
M Maven 登库	0	Maven 仓库 生产库-reviease Maren Belease番目下存成功成計干级定。1988年19月1日、1928年19月1日日本平 2023-01-03 13 15:38			221	(8)
○ 通用税品仓库 (Nor)		非			201	rit.
			ŧ	1	£	

点进**非生产库-snapshot**,在仓库指南中,我们下载对应的settings.xml文件,这个文件当Jenkins需 推送Jar到Maven仓库的时候需要用到

	自2022.12.12起,受 Maven 中央仓库网络限制,阿里云云戏 Maven 中央代理仓库可能会出现部分新增依赖壹找不到的情况,但不影响已有依赖,请知悉		
合库指南 白文件列表		,凡 成员 - @	仓库证
M Maven方式	推进(修改方式, 推送(覆盖)式) 拉取(修改方式) 拉取(覆盖方式)		
ale Gradie方式	通过本指用快速设置仓库凭证,并完成制品文件的推送。		
。 企库权限管理	本方法适用于本地没有配置过 settlings.xmi 可以通过我们提供给您的示例 settlings.xml 文件完成设置。若忍在本地进行过 Maven 配置,请参考 推送(修改方式)。		
□3. 本地仓库迁移	步覆一:设置仓库凭证		
〇 文件上传	下载示明配置文件 settings.xml 并将其存放员 \$HOME/m2/ 文件目录下。 示例文件 settings.xml 中約 username 和 password 为忽约个人凭证,个人对仓库制品文件的操作权跟根据仓库用户角色设置,重要说明		
	示例 settings.xml 中已为你自动填写 username 和 password ,可以直接使用、无语手动修改		
	配置文件 settings.xml	b	
	步骤二:推送		
	进行以下命令完成和品推进。		
	推送命令	ð	
	mvn clean install org.apache.maven.plugins:maven-deploy-plugin/2.8:deploy -DskipTests		
	1250		
	云波默认为你避惧了两个仓库:		
	Maven Release 库用于存储功能趋于稳定、当前更新停止,可以用于发行的成本。 Maven Snapshot 库用于存储不稳定,尚处于升发中的版本,即快照版本。		
	您的新品文件具体推送到哪个库、根据您项目目录的 pom.xml 文件中 <version></version> 字现中最否配置了 -SNAPSHOT 。		
	推送至 Release 挥 pom.uml 示明:		

在旁边的包文件列表可以看到之后Jenkins推送上去的包

仓库	指南 包文件列表					A、成员 ③ 仓库设置
生产(军-snapshot 非主作用 企业约可见					Q
	包名	最新版本	最近更新时间	版本数	操作	
	run.runnable:demo-springboot-simple	0.0.3-SNAPSHOT	2023-01-03 23:12:36	3	ŵ	
	run.runnable:Service-Registry	0.0.11-SNAPSHOT	2023-01-03 22:20:57	7	ŵ	

将这个文件上传到服务器 /root/.m2/目录下

[root@ecs-205431 .m2]# ll total 24 -rw-r--r-- 1 root root 5979 Jan 3 23:11 copy_reference_file.log drwxr-xr-x 34 root root 4096 Jan 3 00:08 repository -rw-r--r-- 1 root root 327 Jan 2 14:18 settings-docker.xml -rw-r--r-- 1 root root 5337 Jan 1 16:51 settings.xml root@ecs-205431 .m2]# <mark>原文链接</mark>:Java 开发者必看!手把手带你搞定 Jenkins+Maven 仓库 +Docker 仓库 + 部署 + 自动更新 pom 版本 一条龙部署服务

2.6 准备Docker^{容器镜像仓库}

阿里云容器镜像服务地址: https://cr.console.aliyun.com/

登录之后选择个人版

三(一)阿里云	✿ 工作台 Ⅲ 图号全部资源 ~ 受 华东1 (杭州) ~		Q 188	费用	ICP 香蕉	22	支持	工单	0	0	Ħ	9	1	前体
容器镜像服务	容器镇缘服务 / 实例列表													
实例列表	实例列表													
制品中心	10日を2015年1月 全部展示 ~													
被像工具	◇ 个人实例	个人题	企业版											
	个人容器镜像托管服务, 即發使用		① 企业级安全托管和高效分发服务, 限时代惠	购买										
	正行状态 运行中 🗸		+											
	实例提销 个人版 操作 特入企业账	8	什么是企业版实例? 如何无缝从自建迁移企业版	实例?	8	33								

创建命名空间

☰ (-) 阿里	5 A	Inth @ 21			Q 87.	费用	ICP ≣ II	皇皇	支持	ΞØ	₿	0	Ħ	9	⑦ 第件
8888888 / ¥5 ← 个人3	^{893表/◎3} 实例	名空间													
概算 合库管理 現像合库	^	台城4名空间 命名空间	权限	命名空间状态	niddill da O		1	R V G W	#1 0					1817	
命名空间 代码源 访问凭证		runnable-run-public	1913 1913	 ✓ 正常 	 на та 			 公开 公开 	 8 	有				授权	8133 8174
2															

创建API对应的镜像仓库

		Q 514	08 22 XH	I# 🕀 🖾 .	0° H 0 10 10 10 10 10 10 10 10 10 10 10 10 1
\$888\$\$888 / \$989\$ / \$1890\$ ← 个人实例 ± #51 (8.40) ~	创建镜像仓库		×		ma
42 0004095 02595 02550	全部地名空闲 仓库信息		2 代码源		sin.
14997 9529 1193	地域 • 命名空间	华乐1(REHR) runnable-run-public ~ 选择例例创建的命名空间		30 19:47:30	** **
访问凭证	* 仓库名称	demo-springboot-simple 22/64 可以使用API名字 抗進为2-64个学符、可使用小耳英文字母、数字、可使用分類時*_*、**、** (分編符不1 来位)	能在普位城	63 17:49:02	64 85
	位用类型	○ 公开 ● 私有			
		oemo-springooot-simple 22/100			
	描述法意	长度量长100个字符			
		更将Markdown相式			
			F− \$ ≌≋		

原文链接: Java 开发者必看! 手把手带你搞定 Jenkins+Maven 仓库 + Docker 仓库 + 部署 + 自动更新 pom 版本 一条龙部署服务

创建完成点击进去 基本信息 一栏有公网地址,这就是Docker Image需要推送的地址,我们可以先复出来。

古田狭康風券 / 実例列表 / 領像仓库 / 基本信息	4:000 ✓ E3 ▶ N
第二 (成用) 101 なよ信息 基本信息 101 な年信息 基本信息 101 合作現収 企作現収 企作用以 公用地 (M) Colspan="2">Colspan="2">Colspan="2">Colspan="2">Colspan="2">Colspan="2">Colspan="2">Colspan="2">Colspan="2">Colspan="2">Colspan="2">Colspan="2">Colspan="2">Colspan="2">Colspan="2"Colspan="2">Colspan="2"Colspan	\$1805 ✔ IN }
基本信息 基本信息 Mill 仓库很收 仓库名标 demo-springboot-simple 公用地車 cgistry.cn-hangzhou.aliyuncs.com/tunnable-ru/d 触发着 仓车地車 学东1 (欧州) 专有用店 @ registry-vpc.cn-hangzhou.aliyuncs.com/tunnable-ru/d 健康成本 仓车 私有 包用店 @ registry-internal.cn-hangzhou.aliyuncs.com/tunnable-ru/d 機像均步 代用店 季 无 服養 ● Short description is empty for this repo.	A
台岸模板 台座布特 demo-springboot-simple 公用地しの egistry-on-hangzhou aliyuncs.com/unnable-ruv(d_1 触发着 台岸地域 毕気1 (抗州) 学有用信の registry-spc.on-hangzhou aliyuncs.com/unnable-ruv(d_1 健康原本 台岸東型 私有 经用目 学有用信の registry-spc.on-hangzhou aliyuncs.com/unnable-ruv(d_1 健康同歩 代目白库 天 損費の Short description is empty for this repo.	
触发器 台環地域 年気1 (統州) 专有同信 @ registry-spc.cn-hangzhou.aliyuncs.com/nunnable-f5 機像版本 台環奥型 私有 经典网语 @ registry-internal.cn-hangzhou.aliyuncs.com/nunnable_f5 機像同步 代目白库 无 損費 @ Short description is empty for this repo.	(11)
健康版本 台岸类型 私有 经用用语 Field 经用用语 Field registry-internal.cn-hangshou.allyuncs.com/runnab 健康同步 代目位率 无 損要 Field 指要 Field Short description is empty for this repo.	
機像同步 代目位率 无 損要 Short description is empty for this repo.	(86
操作指案 制品描述	
1. 登录阿里云Docker Registry	
\$ docker loginusername= registry.cn-hangzhou.aliyumcs.com	
用于整象的用户名为阿累五张号金名,银码为开通服务时设置的密码。	
您可以在访问先进页重伸起洗证把锅。	
2. 从Registry中拉取镜像	
\$ docker pull registry.cn-hangzhou.aliyuncs.com/runnable-run/demo-springboot-simple:[暖像版本号]	
3. 将镜像推送到Registry	
\$ docker login —-username= <mark></mark> registry.cn-hangzhou.aliyuncs.com \$ docker tag [ImageId] registry.cn-hangzhou.aliyuncs.com/runnable=run/demo-springboot-simple:[镜像版本号] \$ docker push registry.cn-hangzhou.aliyuncs.com/runnable=run/demo-springboot-simple:[镀像版本号]	
请积据实际领量信息替换示例中的[Imageld]和[镜像版本号]参数。	
4. 选择合适的镜像仓库地址	
从ECS推送领播时,可以选择使用植像仓库内网地址。履送遗推将每到很升升且将不会指耗宽的公网凤凰。	
如果您使用的机器位于VPC网络、通使用 registry-vpc.cn-hangzhou.allyuncs.com 作为Registry的域名登录。	

设置密码

在个人版中,访问凭证中,设置固定密码,这个密码之后会在项目demo-springboot-simple 中的 eploy_docker.sh 用到。

容器镜像服务 / 实例列表 /	和外证	
← 个人实例		所在地域 全国
概见 仓库管理 ^ 機像仓库 命名空间 代码源	我取外证 在拉取私有環像或者上時環像病,需要docker login输入型的洗证强急。您可以选择通过 编射撤码 或者 關定證码 作为访问洗证。 國定密码 国定密码 国定密码 国定密码 国定部码 国定部码 通过本时获增新,请注意妥善保健。如果通失,可以通过两次设置重要固定密码。 临时密码 通过A时获取福时提出访问镜像服务实例,使考 和协交档。	设置固定时行
访问凭证	 2 登录实例 	(8)
		83

之后Jenkins推送镜像上来,可以在这里镜像版本找到对应的镜像

≡ €Э開里云	â 178			Q 188	费用 iCP	备果 企业 支持 工車 {	9 🖂 🗘 🖉	7 Q 004
#書稿集展秀 / 文明列表 ← demo-sp #±400	/ meax / meas= pringboot-sim						华东1(杭州)	私有 本地会際 🗸
企库模权 1000 1000 1000 1000 1000 1000 1000 100	☐ 版#	tanio 😡	itő	Digest 🕢	植像大小 🖗	最近推送时间	操作	
11200 11205本 技術同步	0.0.3-SNAPSHOT	53222d9c7	✓ 正常	4a4c79f08588bd816e423f 236156fa8cc3d5b399a58a cdb954d38687c40a0c28	224.094 MB	2023-01-03 23:13:17	安全扫描	(瓜仏泉)同步(副雄
	0.0.2-SNAPSHOT	a77ef181d1	√ E8	ac626/4411/5a0d69/d0/1cb e2ebb127d694dffte4a47b9 52b05ac4a1ad9dc6d	224.094 MB	2023-01-03 22:14:42	安全扫描	双位是 同步 数 码
	0.0.1-SNAPSHOT	4bb662aeb	✓ ER	66f31af172093e8113cd4fcc 8fcc1625f33799f9578f050 c65beec38926e24e9	224.094 MB	2023-01-03 22:12:24	安全日曜	- RAB - RO - BO

3. 通过pipeline把上面内容都串起来

有了上面的准备的内容,那我们就可以通过Jenkins pipeline把这些内容都串在一起了。

3.1 Jenkins新建pipeline

再次登录Jenkins,这次我们新建一条pipeline,pipeline的名字就叫API的名字,这里大家照着我的配一样的填写就行,需要注意的地方我会说出来。

🧌 Jenkins		Q 查找(X+K)	0	φ.	(1) Asher.tang ~	₿注销
Dashboard () 所有()						
	输入一个任务名称					
	demo-springboat-simple = di/0/07					
	内建一个自由风格的软件项目 这是 Jenkins的主要功能 Jenkins将会结合任何SCM和任何构建系统束构建你的项目, 甚至可以构)建软件认外的系统.				
	滚水线 幕心地组织一个可以长期运行在多个节点上的任务,适用于构建流水线(更加正式地应当称为工 任务类型。	(作流) , 増加成者组织淹以采用自由风格的				
	約讓一个多配置项目 透用于多配置项目,例如多环境期试,平台指定构建,等等.					
	(0) 文件夹 Creates a set of multibranch project subfolders by scanning for repositories.					
	多分支流水线 每度一个SCM仓库中检测到的分支创建一系列流水线。					
	文件夹 创建一个可以兼靠存储的容器。利用它可以进行分组。 模倣仅仅是一个过滤器。而文件夹则是一同名称的的内容。只要它们在不同的文件 夹里即可。	一个独立的命名空间, 因此你可以有多个相				
	····································					
点击确定后配置一些	参数					

🦣 Jenkins	Q. 重控(30+K)	⑦
Dashboard) demo-springboot-simp	at > Configuration	
Configure	General	Enabled
General	權送	
▶ 高级项目选项 ↓ 流水线	demo-springboot-simple 描述,随便填填	
	 GitHub 项目 現日 URL ? git@gitHub.com:MingGH(demo-springboot-simple.git/ 需要集成的API GitHubb地址,这里我用的是示例的这一个, 你可以克羅到你的CitHub仓库填着你的仓库地址 アreserve stashes from completed builds ? Preserve stashes from completed builds ? Throttle builds ? 不允许并发构键 委兵旧的构建 ? 有时候我们不需要记录这么多次的构建过程,所以选择开启丢弃旧的构建 策略 	
	Log Rotation 保持构建的天数	~
	947 EM	

添加参数化构建过程,总共会添加三个参数ENV_INFO,GIT_Branch,Deploy_Port

ENV_INFO

Configure	22++13#37998 2 策略	
ුරු General	Log Rotation	
♪ 高級項目选項 屵 流水线	保持构建的天数 如果非空、构建记录将保存此天数 8	
	保持构建的最大个数 如果非空,最多此数目的构建记录将被保存 10	
	▲ 書題:	
	≕ 选项参数 ? 选择选项参数 名称 ?	×
	ENV_INFO 配置环境参数,用来决定部署在dev还是prod	
	透项 ? dev 每一行代表一个选项 prod	

GIT_Branch

GIT_Branch	这个参数用于决定检出代码时需要用哪个分支	
大认值 ?		
develop	因为我们多半部署的都是dev环境,所以填写测试环境用的分支即可	

Deploy_Port

≡ 字符参数 ?		
3称 ?		
Deploy_Port	Docker部署时,容器通过宿主机暴露出去的端口	
状认值 ?		
default	这里配置default是和Jenkinsfile中结合起来的,如果是default就会使用项目文件deploy_docker.sh中写好的端口, 如果在构建时使用新的端口那么则会覆盖deploy_docker.sh中的配置,	
default 韵述 ?	这里配置default是和Jenkinsfile中结合起来的,如果是default就会使用项目文件deploy_docker.sh中写好的端口, 如果在构建时使用新的端口那么则会覆盖deploy_docker.sh中的配置, 这样做的原因是考虑到很多个人开发者一台机既是测试又是prod,可能会导致端口冲突	
default 護述 ? Ports that will be exp	这里配置default是和Jenkinsfile中结合起来的,如果是default就会使用项目文件deploy_docker.sh中写好的端口, 如果在构建时使用新的端口部么则会覆盖deploy_docker.sh中的配置, 这样做的原因是考虑到很多个人开发者一台机既是测试又是prod,可能会导致端口冲突	
default 誕生(? Ports that will be exp	这里配置default是和Jenkinsfile中结合起来的,如果是default就会使用项目文件deploy_docker.sh中写好的端口, 如果在构建时使用新的端口部么则会覆盖deploy_docker.sh中的配置, 这样做的原因是考虑到很多个人开发者一台机既是测试又是prod,可能会导致端口冲突	
default 謎述 ? Ports that will be exp 电文本] 预览	这里配置default是和Jenkinsfile中结合起来的,如果是default就会使用项目文件deploy_docker.sh中写好的端口, 如果在构建时使用新的端口部么则会覆盖deploy_docker.sh中的配置, 这样做的原因是考虑到很多个人开发者一台机既是测试又是prod,可能会导致端口冲突 posed when deployed	

Dashboard > demo-springboot-simpl >	Configuration	
Configure	□ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □	
♪ 商級項目造項 一 流水銭	其他工程构建后触发 ? 定时构建 ? GitHub hook trigger for GITScm polling ? 轮询 SCM ? 静默期 ? 触发远程构建 (例如,使用圆本) ?	这些保留为空即可,如果有特殊需求可以进行勾选
	高级项目选项 ^{显示的名称} ?	

Pipeline script from SCM

Configure	流水线	
ලි General	定义	
₯ 高級項目透頂	Pipeline script from SCM 选择Pipeline script from SCM	
で 流水线	SCM ?	
	Git 选择Git	33 33
	Repository URL ? git@github.com:MingGH/demo-springboot-simple.git Credentials ?	0
	Credentials ? root (github-with-privatekey) 选择在文章前半部份已经配置的github凭据 + 混版 Add Repository	
	Branches to build 🕐	
	指定分支(为空时代表any) ?	

Dashboard > demo-springboot-simpl	Configuration	
Configure	Add Repository	
ැටි General	Branches to build (?)	
》高级项目选项	指定分支(为空时代表any) ?	×
ℓ [」] 流水线	*/\$(GIT_Branch) \$(GIT_Branch)的参数来源是在上面配置的字符参数设置的 Add Branch	
	源码库浏览器 ? (自动) 保持默认即可	~
	Additional Behaviours	
	脚本路径 ?	
	Jenkinsfile 保持Jenkinsfile即可,对应的是项目文件中的Jenkinsfile, Jenkins会去项目中找同名文件 發量級检出 ② 取消勾选	
	流水线语法	
	94.49 (12.70)	

点击新建,此时一条pipeline就创建成功,是不是已经迫不及待的想点击build试试?等等,先听我把 面这一部份讲完,当出现问题时才知道去哪找解决方案

3.2 项目demo-springboot-simple 讲解

回到最开始需要让你下载那个项目: demo-springboot-simple

application-dev.yaml 和 application-prod.yaml

项目是一个很简单的Spring Boot项目, 默认端口是 5002。定义了两个不同的配置文件, 分别是 appli ation-dev.yaml 和 application-prod.yaml 用来模拟不同环境时切换不同的参数。

HelloController

创建了一个 HelloController用来演示最简单的GET请求, 会根据不同的部署环境, 返回不同的内容。

HelloController的内容如下:

import org.springframework.beans.factory.annotation.Autowired; import org.springframework.stereotype.Controller; import org.springframework.web.bind.annotation.GetMapping; import org.springframework.web.bind.annotation.ResponseBody; import run.runnable.demospringbootsimple.config.AppConfig;

```
/**

* @author Asher

* on 2023/1/3

*/

@Controller

public class HelloController {
```

```
@GetMapping("/hello")
@ResponseBody
public String hello(){
    return "hello " + appConfig.getEnv();
}
private AppConfig appConfig;
@Autowired
public void setAppConfig(AppConfig appConfig) {
    this.appConfig = appConfig;
}
}
```

3.3 项目配置: Jenkinsfile (有需要修改内容)

当Jenkins在面板上点击构建时,会找到对应目录下的Jenkinsfile,所以第一个我们需要说明的就是Jenk nsfile。

在Jenkinsfile中我定义了7个stage,分别是Build, Unit Test, Push Nexus, Package Image, Push I age, Deploy, Update version

当Jenkins进行build的时候就会出现对应的这几个阶段。

	Declarative: Checkout Build SCM		Unit Test	Push Nexus	Package Image	Push Image	Deploy	Update version
Average stage times: (Average <u>full</u> run time: ~2min 5s)	4s	3s	35	37s	11s	38s	15	95
Jan 03 1 15:11 commit	4s	Зs	35	31s	14s	28s	1s	11s

这里因为篇幅过长,使用了新的一篇文章来进行讲解:demo-springboot-simple中Jenkinsfile详解

需要修改的内容为:

在这个 Update version Stage中需要修改你的git邮箱地址和username,因为在步骤2.4中已经**配置** 公钥在GitHub上,且在Jenkins agent 中配置了 /root/.ssh的目录映射,所以这里只需要配置用户名邮箱就行。



3.5 项目配置: deploy_docker.sh

说完了Jenkinsfile,希望你已经明白Jenkins执行时的主要的流程,那么在 deploy_docker.sh中的内是docker部署到本地和docker推送到**Docker容器镜像仓库时的一些命令。**

你需要配置的内容如图:

```
#!/bin/bash
DEFAULT_PORT=5002
APP=demo-springboot-simple
PROG_NAME=$0
ACTION=$1
TAGNAME=$2
PROFILE=$3
PASS_PORT=$4
APP_PORT=${PASS_PORT:-$DEFAULT_PORT}
APP_NAME=${APP}-${PROFILE}
APP_HOME=/dockerData/runnable-run/${APP_NAME}
APP_OUT=${APP_HOME}/logs
# 阿里云仓库命名空间
APP_NAMESPACE=
APP_REGISURL=registry.cn-hangzhou.aliyuncs.com
APP_RESP=${APP_REGISURL}/${APP_NAMESPACE}/${APP}
# 阿里云账号
APP_USERNAME=
# 阿里云配置个人容器镜像服务时,设置的密码
APP_PASSWORD=
mkdir -p ${APP_HOME}
mkdir -p ${APP_HOME}/logs
usage() {
   echo "Usage: $PROG_NAME {start|stop|restart|push}"
start_application() {
    docker run \
    -e "SPRING_PROFILES_ACTIVE=${PROFILE}" \
    --log-opt max-size=1024m \
```

3.4 项目配置: pom.xml (有需要修改内容)

需要修改的配置

在 pom.xml 文件中你需要修改properties中的内容,

• docker.namespace是在阿里云的Docker容器镜像仓库时创建的。

• docker.registry.address 则是镜像仓库基本信息中的公网地址,例如: registry.cn-hangzhou.aliy ncs.com

在plugin中我增加了dockerfile-maven-plugin,通过这个plugin可以使用maven构建docker imag,但是需要项目中存在Dockerfile,这个会马上说到。

```
<plugin>
<groupId>com.spotify</groupId>
<artifactId>dockerfile-maven-plugin</artifactId>
<version>1.4.13</version>
<configuration>
<repository>${docker.registry.address}/${docker.namespace}/${project.artifactId}</repos
tory>
<tag>${project.version}</tag>
<buildArgs>
<JAR_NAME>${project.artifactId}</JAR_NAME>
<JAR_FILE>target/${project.build.finalName}.jar</JAR_FILE>
</plugin>
```

这里需要注意的是标签 < buildArgs>下定义的 < JAR_NAME> 和 < JAR_FILE> 将会传递到Dockerfil 中用于进行构建镜像

3.4 项目配置: Dockerfile (无修改)

Dockerfile中配置了构建image时的一些参数,如下:

FROM openjdk:17-jdk-slim-buster

ARG JAR_NAME ENV PROJECT_NAME \${JAR_NAME} ENV PROJECT_HOME /usr/local/\${PROJECT_NAME}

RUN In -sf /usr/share/zoneinfo/Asia/Shanghai /etc/localtime RUN echo 'Asia/Shanghai' >/etc/timezone RUN mkdir \$PROJECT HOME && mkdir \$PROJECT HOME/logs

ARG JAR_FILE COPY \${JAR_FILE} \$PROJECT_HOME/\${JAR_NAME}.jar

ENTRYPOINT java -jar -Xmn128m -Xms256m -Xmx256m \$PROJECT_HOME/\$PROJECT_NAME. ar

当中一些参数的解释,如果需要更加详细的说明,可以参考这个: 什么是 Dockerfile?

• FROM openjdk:17-jdk-slim-buster image基于openjdk:17-jdk-slim-buster

• ENV PROJECT_NAME \${JAR_NAME} 设置环境变量,定义了环境变量,那么在后续的指令中,就以使用这个环境变量。这个参数中的 \${JAR_NAME}的来源是pom.xml中的 <JAR_NAME>\${project.rtifactId}</JAR NAME>

- RUN In -sf /usr/share/zoneinfo/Asia/Shanghai /etc/localtime 指定容器运行时区
- RUN mkdir \$PROJECT_HOME && mkdir \$PROJECT_HOME/logs 创建项目路径
- COPY \${JAR_FILE} \$PROJECT_HOME/\${JAR_NAME}.jar 将maven构建出来的Jar文件复制到容器 具体位置

• ENTRYPOINT java -jar -Xmn128m -Xms256m -Xmx256m \$PROJECT_HOME/\$PROJECT_NA E.jar 容器的入口,通过java -jar进行启动,并指定运行时的一些参数

完成以上配置之后, 提交你的代码到具体的分支。

4 Build with Parameters

终于到了最后一步,回到Jenkins的面板,点击Build with Parameters,填入你的参数,进行构建

🦣 Jenkins		Q 查找 (31+K)	③
Dashboard > demo-springboot-simpl >			
₱ 状态	Pipeline demo-springboot-simpl		
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>	需要如下参数用于构建项目:		
D Build with Parameters	ENV_INFO		
() REM	Which environment do you want to deploy to?		
前 删除 Pipeline			
Q,完整阶段视图	GIT_Branch		
O GitHub	develop V		
分 打开 Blue Ocean	Deploy_Port		
ℓ 重命名	Porta cial nel de exposed verier depoyed		
⑦ 瀧水機谱法			
△ Build History 档望历史 ∨	开始问题		
Q Filter builds /			
② 49 2023年1月3日 王午11:11			
※#8 2023年1月3日下午11.09			
② 記 2023年1月3日 下午10.13			

此时你就可以看到pipeline已经在跑了

🧌 Jenkins					Q. 查找 (₩+K)	0	្ជិ 🚺 😩 Ashe	er.tang ~ [→ 注销
Dashboard > demo-springboot-simpl >									
e ta	Pipeline demo-sprin	gboot-sim	pl						
◇ 変更历史	demo-springboot-simple								
Build with Parameters									
I RE							361-744	10 th H.	MARINE
· 劃除 Pipeline						1	Passed - 1	Skipped Falled	0
Q、完整阶段视图									
O GitHub									
🥥 打开 Blue Ocean									
0 11#88						an #2	#3 #4	#5 #6	a7 a9
② 流水线谱法	阶段视图	1							
△ Build History 档建历史 ~	(Declarative: Checkout	Build	Unit Test	Push Nexus	Package	Push Image	Deploy	Update
Q. Filter builds		SCM	T			Image			version
✓ #10 2023年1月4日 7年7.55	Average stage times (Average <u>full</u> run time: ~2min 5s)	45	- 35	35	37s	115	385	ts	95
② 約 1 2023年1月3日 下午11.03	Jan 04 Hill TLSS Changes	2.							
(2) #3 2023年1月3日 下午11:09		-			-	-			
● #2 1 2023年1月3日下午10.13	Jan 03 1 15:11 Commit	45	21	:34	31s	145	28s	: ts	118

执行完成的状态应该是所有的stage都是绿色,且在你服务器上能看到有一个新的容器产生

	Declarative: Checkout SCM	Build	Unit Test	Push Nexus	Package Image	Push Image	Deploy	Update version
Average stage times: (Average <u>full</u> run time: ~2min 1s)	45	35	3s	37s	11s	37s	1s	95
11:55 commit	3s	35	Зs	31s	12s	28s	1s	115
(#3) Jan 03 1 15:11 commit	4s	35	35	31s	14s	28s	1s	11s

[root@ecs-205431 ~]# docker ps -s CONTAINER ID IMAGE COMMAND REATED STATUS PORTS NAMES SIZE 9c0ee6e2726f registry.cn-hangzhou.aliyuncs.com/xxxx/demo-springboot-simple:0.0.4-SNAP HOT "/bin/sh -c 'java -j..." 3 minutes ago Up 3 minutes 0.0.0.0:5002->5002/tcp, :::5002-> 002/tcp demo-springboot-simple-dev 32.8kB (virtual 419MB)

5. 问题排查

因为流程很长,出现问题不要慌。直接点击那一个构建过程,进去看看日志。



6. 参考内容

The Problem With 'src refspec does not match any'

Error: src refspec master does not match any – How to Fix in Git

git 获取当前分支名

git: rename local branch failed

Change System Time Zone Jenkins分布式构建与并行构建 build-a-java-app-with-maven Installing Jenkins cut or awk command to print first field of first row How to automatically increment pom version with maven, for example 1.2.0 to 1.3.0 Starting Spring Boot Application in Docker With Profile 使用Jenkins对springboot项目进行docker镜像一键部署,jenkins + docker + springboot