

[about](#)

[环境配置](#)

[安装docker](#)

[安装jenkins](#)

[docker简单使用](#)

[Jenkins插件安装及配置](#)

[安装allure插件](#)

[配置项目](#)

[构建执行](#)

[问题](#)

[如果遇到docker下载特别慢的情况。怎么办。](#)

[放开防火墙端口号](#)

[Failed to resolve host name updates.jenkins.io. Perhaps you need to configure HTTP proxy?](#)

about

这里展示了如何在centos上配置持续集成环境。

jenkins

- 开源
- 安装配置简单
- 跨平台、支持所有的平台
- web可视化管理界面
- 分布式构建
- 丰富的插件

docker

- 开源
- 快速搭建环境
- 自动化测试和持续集成

环境配置

首先安装docker，然后安装docker版本的Jenkins。

请以管理员权限进行下面的操作。

安装docker

可选的操作：查看内核版本

目前，CentOS 仅发行版本中的内核支持 Docker。

Docker 运行在 CentOS 7 上，要求系统为64位、系统内核版本为 3.10 以上。

Docker 运行在 CentOS-6.5 或更高的版本的 CentOS 上，要求系统为64位、系统内核版本为 2.6.32-431 或者更高版本。

```
uname -a
```

示例

```
[root@localhost ~]# uname -a
Linux localhost 3.10.0-514.el7.x86_64 #1 SMP Tue Nov 22 16:42:41 UTC 2016 x86_64
x86_64 x86_64 GNU/Linux
```

更新yum源

```
[root@localhost ~]# yum update -y
```

可选的操作：卸载旧版的docker

```
yum remove docker docker-common docker-selinux docker-engine
```

安装依赖包

`yum-util` 提供 `yum-config-manager` 功能，另外两个是 `devicemapper` 驱动依赖的。

```
yum install -y yum-utils device-mapper-persistent-data lvm2
```

示例

```
[root@localhost ~]# yum install -y yum-utils device-mapper-persistent-data lvm2
```

设置yum源

```
yum-config-manager --add-repo https://download.docker.com/linux/centos/docker-
ce.repo
```

示例

```
[root@bogon ~]# yum-config-manager --add-repo
https://download.docker.com/linux/centos/docker-ce.repo
已加载插件: fastestmirror
adding repo from: https://download.docker.com/linux/centos/docker-ce.repo
grabbing file https://download.docker.com/linux/centos/docker-ce.repo to
/etc/yum.repos.d/docker-ce.repo
repo saved to /etc/yum.repos.d/docker-ce.repo
```

查询及安装docker

```
yum list docker-ce --showduplicates | sort -r
```

示例

```
[root@bogon ~]# yum list docker-ce --showduplicates | sort -r
Loading mirror speeds from cached hostfile
Loaded plugins: fastestmirror
docker-ce.x86_64          3:19.03.5-3.el7          docker-ce-stable
docker-ce.x86_64          3:19.03.4-3.el7          docker-ce-stable
docker-ce.x86_64          3:19.03.3-3.el7          docker-ce-stable
docker-ce.x86_64          3:19.03.2-3.el7          docker-ce-stable
docker-ce.x86_64          3:19.03.1-3.el7          docker-ce-stable
```

在版本列表中，选择合适的版本下载即可。

```
yum install docker-ce-17.12.1.ce -y
```

示例

```
[root@bogon ~]# yum install docker-ce-17.12.1.ce -y
```

检查是否安装成功

```
docker version
```

示例

```
[root@bogon ~]# docker version
```

Client:

Version: 17.12.1-ce

API version: 1.35

Go version: go1.9.4

Git commit: 7390fc6

Built: Tue Feb 27 22:15:20 2018

OS/Arch: linux/amd64

Cannot connect to the Docker daemon at unix:///var/run/docker.sock. Is the docker daemon running?

启动docker并加入开机启动

```
systemctl start docker # 启动
```

```
systemctl enable docker # 加入开机启动
```

```
ps -ef | grep docker
```

安装Jenkins

上面的docker环境配置好之后，我们再来配置docker版本的Jenkins。

选择docker版本的Jenkins

选择长期支持版本。

在新的页面中，选择LTS版本的安装命令。

→ × 🏠 ⓘ hub.docker.com/r/jenkins/jenkins



By [jenkins](#) • Updated 44 minutes ago

The leading open source automation server

Container

Overview Tags

Jenkins Continuous Integration and Delivery server.

This is a fully functional Jenkins server, based on the weekly and LTS releases .



Jenkins

- To use the latest LTS: `docker pull jenkins/jenkins:lts`
- To use the latest weekly: `docker pull jenkins/jenkins`
- Lighter alpine based image also available

也就是复制 `docker pull jenkins/jenkins:lts` 这个命令并且centos中去执行该命令。

```
docker pull jenkins/jenkins:lts
```

示例

```
[root@bogon ~]# docker pull jenkins/jenkins:lts
lts: Pulling from jenkins/jenkins
9a0b0ce99936: Pull complete
db3b6004c61a: Pull complete
f8f075920295: Pull complete
6ef14aff1139: Pull complete
962785d3b7f9: Pull complete
631589572f9b: Pull complete
c55a0c6f4c7b: Pull complete
4e96cf3bdc20: Pull complete
e0b44ce6ec69: Pull complete
d961082c76f4: Pull complete
5a229d171c71: Pull complete
64514e4513d4: Pull complete
6797bb506402: Pull complete
b8d0a307156c: Pull complete
b17b306b4a0a: Pull complete
e47bd954be8f: Pull complete
b2d9d6b1cd91: Pull complete
fa537a81cda1: Pull complete
Digest: sha256:64576b8bd0a7f5c8ca275f4926224c29e7aa3f3167923644ec1243cd23d611f3
Status: Downloaded newer image for jenkins/jenkins:lts
```

PS:下载速度稍慢.....

下载完毕后，我们来使用下面命令看一下是否下载成功。

```
[root@bogon ~]# docker images
REPOSITORY          TAG                 IMAGE ID            CREATED
SIZE
jenkins/jenkins     lts                22b8b9a84dbe       6 days ago
568MB
[root@bogon ~]#
```

启动Jenkins

```
docker run -d -p 80:8080 -p 50000:50000 -v jenkins:/var/jenkins_home -v
/etc/localtime:/etc/localtime --name jenkins docker.io/jenkins/jenkins:lts
```

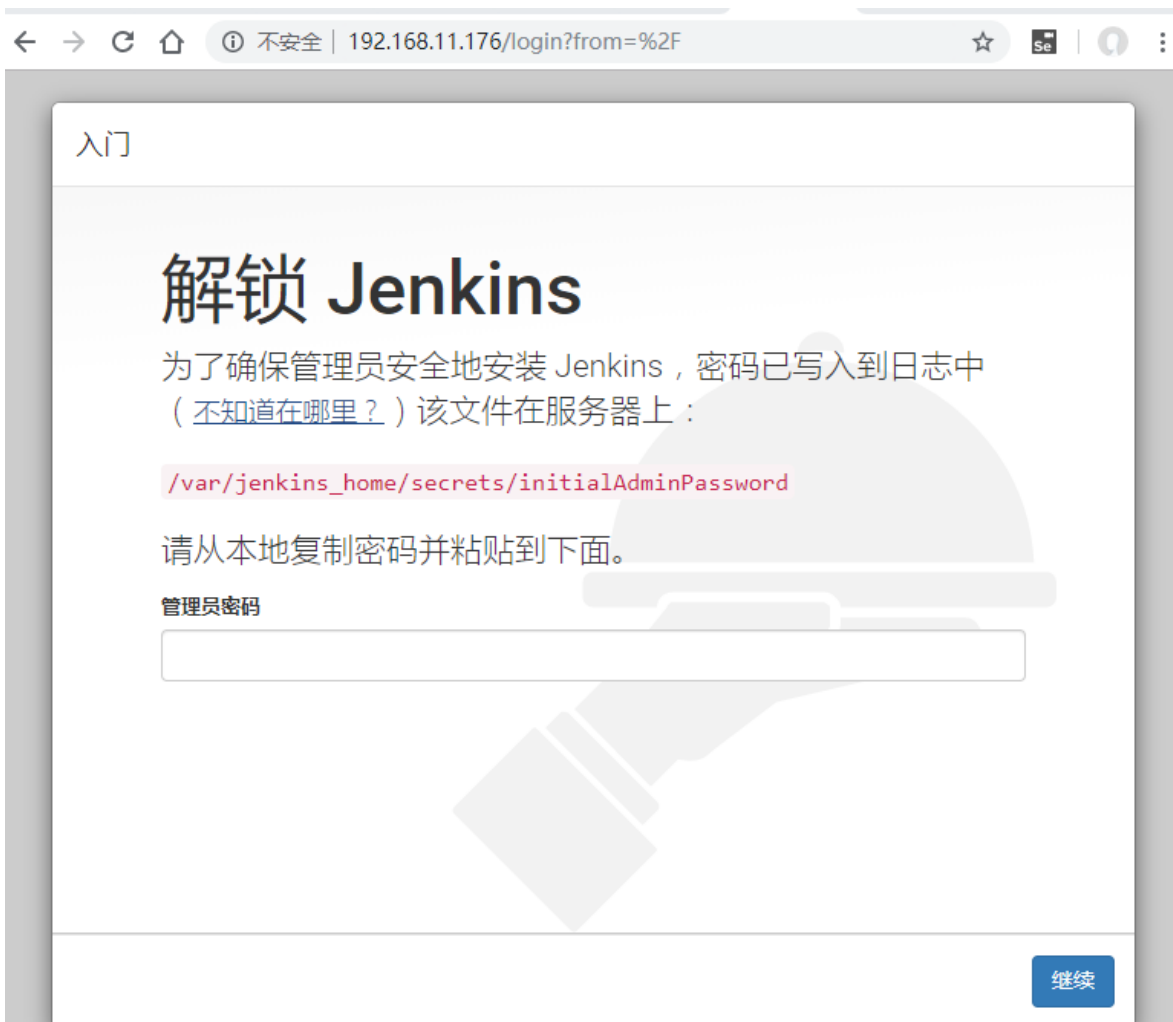
示例

```
[root@bogon ~]# docker run -d -p 80:8080 -p 50000:50000 -v
jenkins:/var/jenkins_home -v /etc/localtime:/etc/localtime --name jenkins
docker.io/jenkins/jenkins:lts
9f1d7ae974968f7465f98a7fbc25fc8f59cdebae561db33db7916fc68ee0f408
[root@bogon ~]#
```

各参数的意义：

- `-d` 后台运行镜像。
- `-p 80:8080` 将镜像的 8080 端口映射到服务器的 80 端口。
- `-p 50000:50000` 将镜像的 50000 端口映射到服务器的 50000 端口。
- `-v jenkins:/var/jenkins_home`，其 `/var/jenkins_home` 为 Jenkins 的工作目录，我们将硬盘上的一个目录挂在到这个位置，方便后续更新镜像后继续使用原来的工作目录。
- `-v /etc/localtime:/etc/localtime` 让容器使用和服务器同样的时间设置。
- `--name jenkins` 给容器起一个别名。
- `docker.io/jenkins/jenkins:lts` 最后说的是 docker 拉去的是 Jenkins 的 lts 版本。

现在，让我们使用本机的浏览器去访问虚拟机的 centos 的 ip。



由于咱们安装的是docker版的Jenkins，所以不能直接去这个位置去找密码（Windows版可以根据提示去上面那个位置去找密码）。那个路径不是服务器的路径，而是docker容器的路径，所以我们还需要使用docker命令去获取密码，让我们拷贝上面的那个路径。

```
docker exec jenkins tail /var/jenkins_home/secrets/initialAdminPassword
```

示例

```
[root@bogon ~]# docker exec jenkins tail  
/var/jenkins_home/secrets/initialAdminPassword  
683a4afce37d44a780c287a131e9388b
```

上面那个 683a4afce37d44a780c287a131e9388b 字符串就密码，填到之前页面的输入框中。

注意，在输入密码后会进入新手操作阶段，下插件啥的，如果这个过程遇到失败等问题，会需要你重新输入这个密码，所以请记住它

经过上面的种种操作，我们终于进入到了新手入门阶段，首先映入眼帘的是**安装Jenkins插件**，如无特殊需求，我们选择**安装推荐的插件**即可。

新手入门

自定义Jenkins

插件通过附加特性来扩展Jenkins以满足不同的需求。

安装推荐的插件

安装Jenkins社区推荐的插件。

选择插件来安装

选择并安装最适合的插件。

Jenkins 2.190.3

漫长的等待.....

① 不安全 | 192.168.11.176 ☆

新手入门

新手入门

🔗 Folders	🔗 OWASP Markup Formatter	🔗 Build Timeout	🔗 Credentials Binding	** Trilead API Folders
🔗 Timestampers	🔗 Workspace Cleanup	🔗 Ant	🔗 Gradle	
🔗 Pipeline	🔗 GitHub Branch Source	🔗 Pipeline: GitHub Groovy Libraries	🔗 Pipeline: Stage View	
🔗 Git	🔗 Subversion	🔗 SSH Slaves	🔗 Matrix Authorization Strategy	
🔗 PAM Authentication	🔗 LDAP	🔗 Email Extension	🔗 Mailer	
🔗 Localization: Chinese (Simplified)				

** - 需要依赖

Jenkins 2.190.3

在漫长的等待过程中，可能会发生各种意外，但请记住一点：**如果发生意外，请间歇性的刷新浏览器，很可能就.....好了.....好不了，喝杯咖啡考虑一下人生回来....继续刷新.....**，我反正两次插件安装失败，我都是刷新解决的.....

如果遇到有的插件安装失败，就点击重试....

安装失败

部分插件安装失败，请重试或继续

✓ Folders Plugin	✓ OWASP Markup Formatter	✗ Build Timeout	✗ Credentials Binding
✗ Timestampers	✗ Workspace Cleanup	✗ Ant	✗ Gradle
✗ Pipeline	✗ GitHub Branch Source	✗ Pipeline: GitHub Groovy Libraries	✗ Pipeline: Stage View
✗ Git	✗ Subversion	✗ SSH Slaves	✗ Matrix Authorization Strategy
✗ PAM Authentication	✗ LDAP	✗ Email Extension	✗ Mailer
✗ Localization: Chinese (Simplified)			

Jenkins 2.190.3

继续重试

创建管理员用户

当上面的插件安装完毕后，会进入到创建管理员用户界面。

创建第一个管理员用户

用户名:

root

密码:

.....

确认密码:

.....

全名:

root

电子邮件地址:

1206180815@qq.com

Jenkins 2.190.3

使用admin账户继续保存并完成

各项填写完毕，选择 保存并完成。

配置实例

根据需要配置ip地址，如果你的远程服务器的，就配个公网的ip或者绑定域名也行；本地的话，就直接 localhost 吧。

实例配置

Jenkins URL:

http://localhost/

Jenkins URL 用于给各种Jenkins资源提供绝对路径链接的根地址。 这意味着对于很多Jenkins特色是需要正确设置的，例如：邮件通知、PR状态更新以及提供给构建步骤的BUILD_URL环境变量。

推荐的默认值显示在尚未保存，如果可能的话这是根据当前请求生成的。 最佳实践是要设置这个值，用户可能会需要用到。这将会避免在分享或者查看链接时的困惑。

Jenkins 2.190.3

现在不要

保存并完成

安装完成

新手入门

Jenkins已就绪！

Jenkins安装已完成。

开始使用Jenkins

Jenkins 2.190.3

进入你的工作台。



docker简单使用

重启docker服务

```
systemctl restart docker.service
```

列出本地主机上的所有镜像

```
docker images
```

示例

```
[root@bogon ~]# docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED
jenkins/jenkins	lts	22b8b9a84dbe	6 days ago
568MB			

查看容器状态

```
docker ps -a
```

示例

```
[root@bogon ~]# docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED
STATUS	PORTS		NAMES
9f1d7ae97496	jenkins/jenkins:lts	"/sbin/tini -- /usr/..."	26 minutes ago
Up 26 minutes	0.0.0.0:50000->50000/tcp, 0.0.0.0:80->8080/tcp		jenkins

容器启动命令

```
docker start container-name
docker stop container-name
docker restart container-name
```

示例

```
[root@bogon ~]# docker stop jenkins
jenkins
[root@bogon ~]# docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED
9f1d7ae97496	jenkins/jenkins:lts	"/sbin/tini -- /usr/..."	28 minutes ago

```

Exit (143) 22 seconds ago
jenkins
[root@bogon ~]# docker start jenkins
jenkins
[root@bogon ~]# docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED
9f1d7ae97496	jenkins/jenkins:lts	"/sbin/tini -- /usr/..."	28 minutes ago

```

Up 3 seconds
0.0.0.0:50000->50000/tcp, 0.0.0.0:80->8080/tcp
jenkins
[root@bogon ~]# docker restart jenkins
jenkins
```

进入、退出正在运行的镜像

```
docker exec
```

相关参数：

- `-d` 分离模式，在后台运行。
- `-i` 及时没有附加2也保持STDIN打开。
- `-t` 分配一个伪终端。

示例，演示一下检查在docker中的Jenkins是否能ping通网络。

```
docker exec -it jenkins bash

# 示例
[root@bogon ~]# docker exec -it jenkins bash
jenkins@9f1d7ae97496:/$ ping www.baidu.com
PING www.a.shifen.com (182.61.200.7) 56(84) bytes of data.
64 bytes from 182.61.200.7 (182.61.200.7): icmp_seq=1 ttl=54 time=8.34 ms
64 bytes from 182.61.200.7 (182.61.200.7): icmp_seq=2 ttl=54 time=7.84 ms
64 bytes from 182.61.200.7 (182.61.200.7): icmp_seq=3 ttl=54 time=6.51 ms
```

如果ping不通需要在docker中执行。

```
nmcli connection modify docker0 connection.zone trusted # 添加信任
systemctl restart docker.service # 重启docker服务
docker ps -a # 查看Jenkins是否启动
docker start jenkins # 没有启动则重启
```

完事之后再重新进行ping百度的过程。

退出就是 `CTRL + D` 或者输入 `exit` 退出。

```
jenkins@9f1d7ae97496:/$ exit
```

Jenkins插件安装及配置

安装allure插件

安装allure插件

1. 进入Jenkins ▶ 系统管理 ▶ 管理插件 ▶ 可选插件
2. 搜索框中输入 allure ▶ 勾选allure插件 点击 直接安装 ▶ 安装完成 ▶ 重启Jenkins



正在下载。



安装/更新 插件中

准备

- Checking internet connectivity
- Checking update center connectivity
- Success

Loading plugin extensions

- Success
- Success
- Allure 安装中
- Loading plugin extensions Pending

返回首页
(返回首页使用已经安装好的插件)

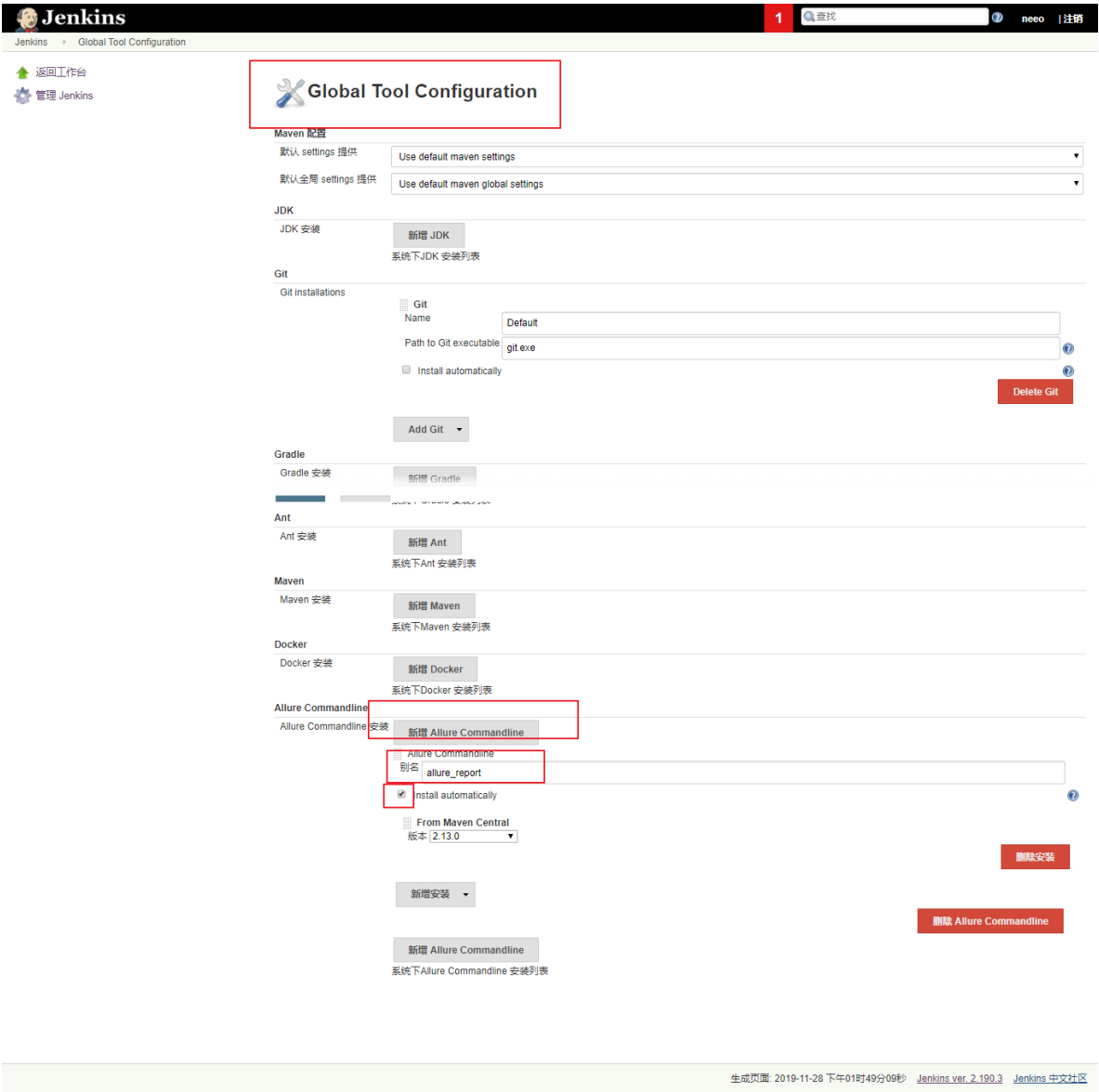
安装完成后重启Jenkins(空闲时)

你也可以选择 安装完成后重启Jenkins。

安装allure commandline

安装 allure commandline 的目的是为了生成HTML报告。

安装方法是 系统管理 ► 全局工具配置 ► allure commandline

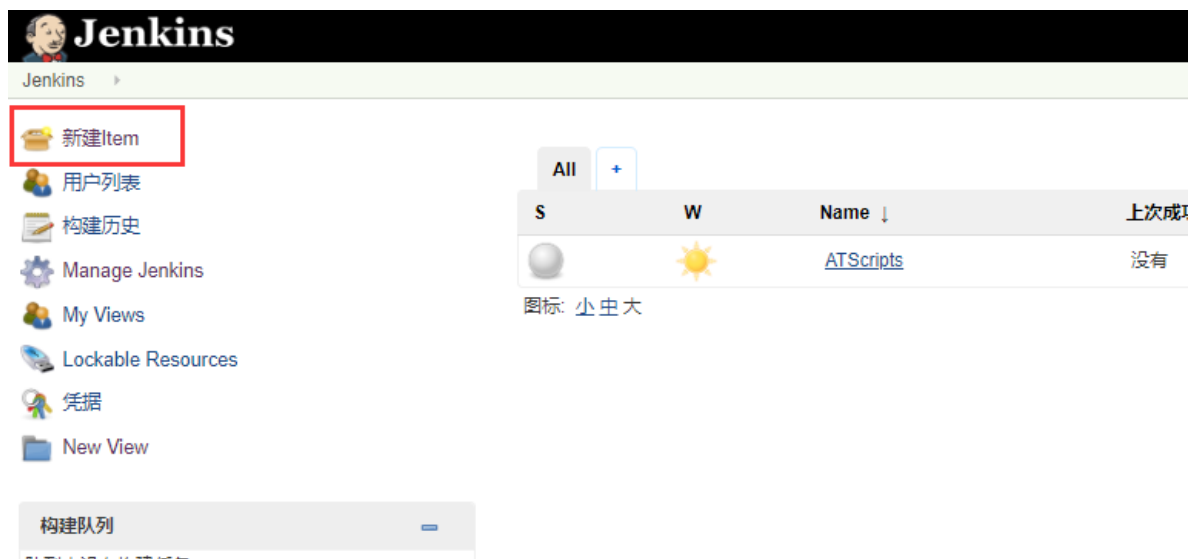


最后点击保存按钮。

配置项目

创建一个自由风格的项目

点击 新建Item，创建项目。



创建一个自由风格的项目。



当我们在Jenkins中构建一个新的项目之后，可以有以下操作：

- general，描述信息，还报告其他的节点信息。
- 源码管理，如何管理代码。
- 构建触发器，也就是定时任务。
- 构建配置，你应该如何去执行你的代码。
- 报告。
- 邮件。

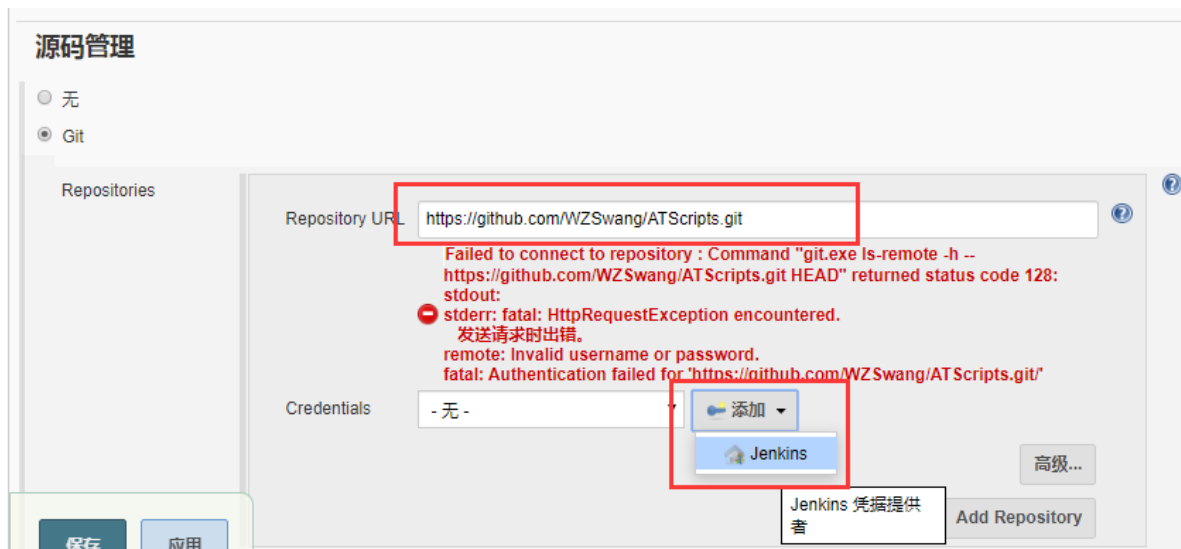
general配置

由于我们本次只是单节点运行，所以暂不执行分布式操作，比如添加节点等。

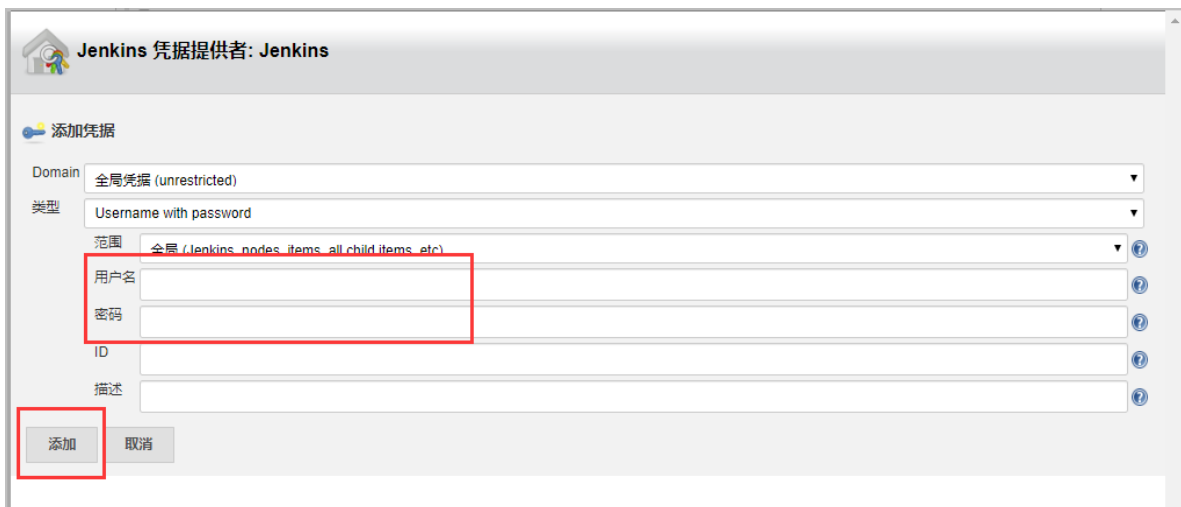
源码管理

- 选择Git
- 输入github上的项目连接。
- 添加Jenkins凭证。

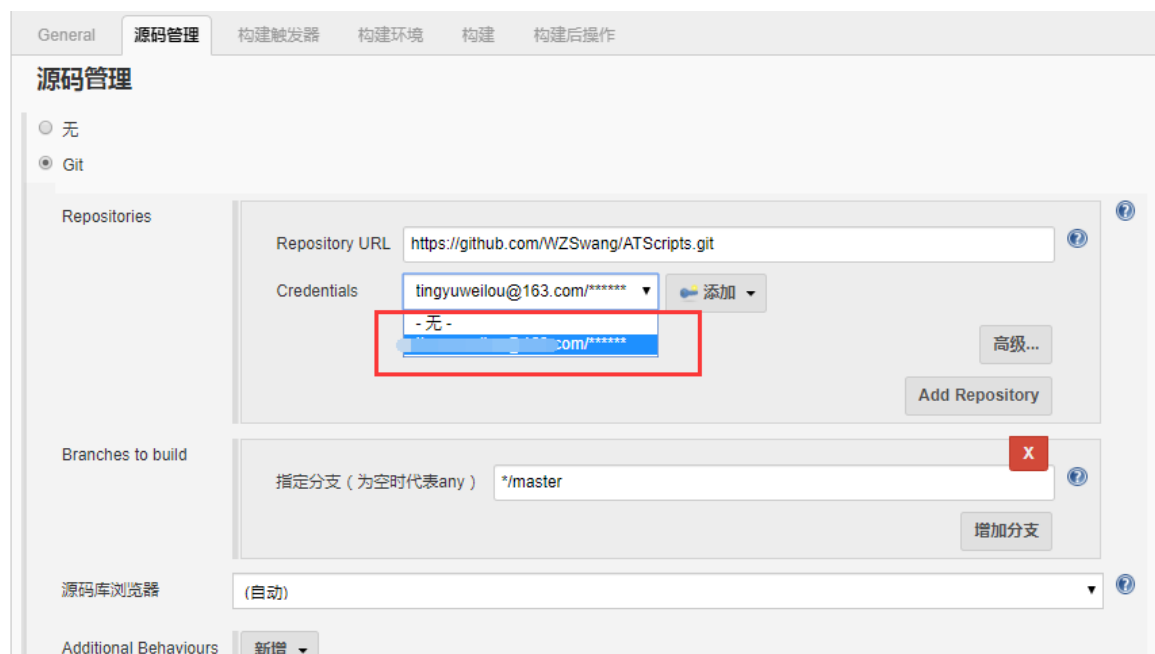
在输入框输入github上的项目连接。点击添加Jenkins认证。



输入你的github账号和密码，点击添加。



然后下拉选择生成的凭证。



构建触发器

日程表的几颗星表示:

- 第一颗星表示分钟，取值 0~59。
- 第二颗星表示小时，取值 0~23。
- 第三颗星表示一个月的第几天，取值 1~31。
- 第四颗星表示一年中的第几个月，取值 1~12。
- 第五颗星表示一周中的第几天，取值 0~7，其中 0 和 7 都表示周日。

构建触发器

☐ 触发远程构建 (例如,使用脚本)

☐ Build after other projects are built

☐ Build periodically

☒ Poll SCM

日程表: `H H/3 ***`

Would last have run at 2019年11月28日 星期四 上午11时32分46秒 CST; would next run at 2019年11月28日 星期四 下午 02时32分46秒 CST.

☐ 忽略钩子 post-commit

示例中表示每隔3小时执行一次。

构建环境

此环节没有操作，略过。

构建配置

构建这里根据平台的不同使用不同的构建命令：

- Windows平台，执行的是 `execute windows batch command`。
- Mac平台执行的是 `execute shell command`。

我这里是Windows平台为示例。

构建

增加构建步骤

- Execute Windows batch command
- Execute shell
- Invoke Ant
- Invoke Gradle script
- Invoke top-level Maven targets
- Run with timeout

进入 `execute windows batch command`，写入你要执行的命令。

构建

Execute Windows batch command

命令: `python run.py`

参阅 [可用环境变量列表](#)

高级...

当然，万一找不到Python环境，你也可以写成绝对路径去执行：

```
C:\python36\python run.py
```

构建后的操作

构建后操作

Allure Report

Disabled ☐

Results:

Path

新增

Paths to Allure results directories relative from workspace.
E.g. target/allure-results.

Properties 新增

高级...

增加构建后操作步骤

保存 应用

上面的path是我们项目的 `report` 目录下的 `result`。

在新增一个发邮件功能。

高级...

E-mail Notification

Recipients

Whitespace-separated list of recipient addresses. May reference build parameters like \$PARAM. E-mail will be sent when a build fails, becomes unstable or returns to stable.

☒ 每次不稳定的构建都发送邮件通知

☐ 单独发送邮件给对构建造成不良影响的责任人

增加构建后操作步骤

保存 应用

构建执行

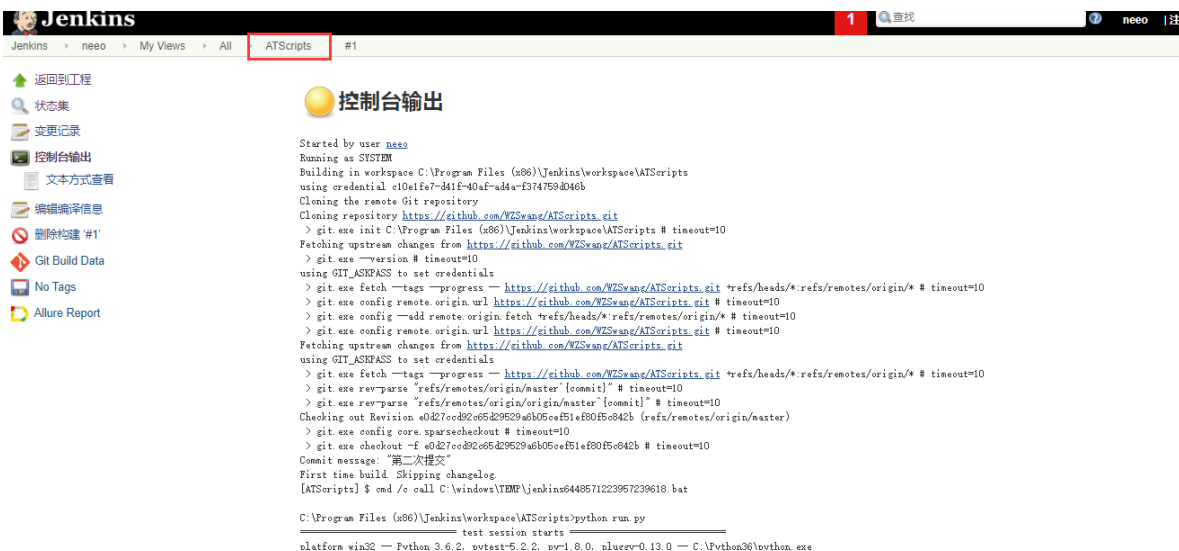
当上面的构建环节执行完毕，可以点击 [立即构建](#)，下面会展示进度，点击它会进入新的页面。



选择 控制台输出，我们就能看到相关的执行信息。



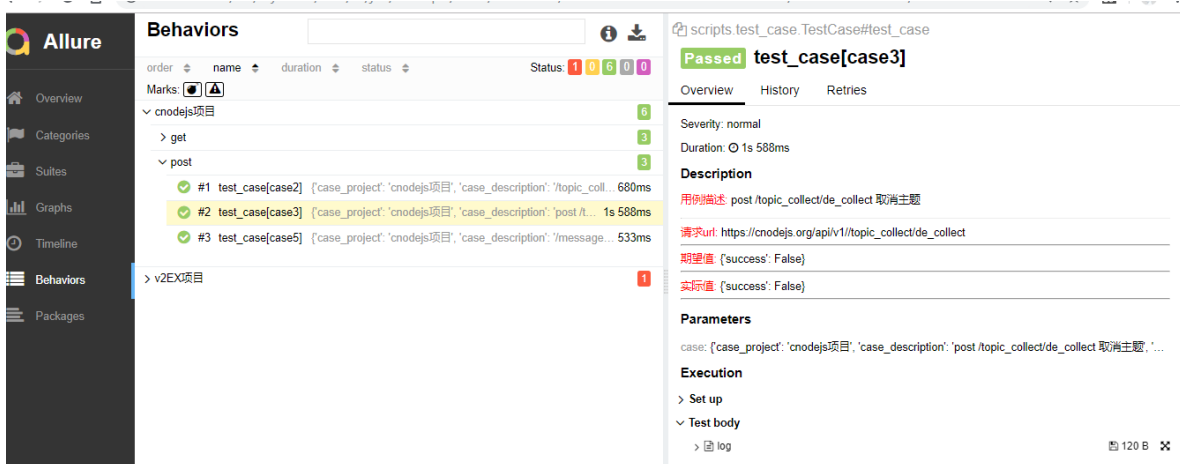
当执行完毕，没啥问题后，我们点击项目名回到项目的工作区。



此时点击 allure report 就有了执行结果。



allure执行报告。



问题

如果遇到docker下载特别慢的情况。怎么办。

- 编辑docker配置文件

```
[root@bogon ~]# vi /lib/systemd/system/docker.service
```

- 将ExecStart 值更改为阿里云。

```
# ExecStart=/usr/bin/dockerd
ExecStart=/usr/bin/dockerd --registry-mirror=https://u1qbyfsc.mirror.aliyuncs.com
```

这样的话，docker再去拉去镜像就会快很多。

放开防火墙端口号

如果是虚拟机或者远程服务器，有可能造成防火墙没有放开端口号，导致我们远程连接jenkins失败，来看解决办法。

```
firewall-cmd --list-ports    # 查看一下当前开放的端口

# 如果该命令提示
[root@bogon ~]# firewall-cmd --list-ports
Firewalld is not running

# 那么启动它
[root@bogon ~]# systemctl start firewalld    # 启动
[root@bogon ~]# firewall-cmd --list-ports    # 然后重新执行
```

完事之后，再执行：

```
firewall-cmd --zone=public --add-port=80/tcp --permanent

# 示例
[root@r ~]# firewall-cmd --zone=public --add-port=80/tcp --permanent
success
```

接下来再重新加载防火墙设置后再去查看是否开启成功：

```
firewall-cmd --reload    # 重新加载
firewall-cmd --list-ports    # 查看

# 示例
[root@r ~]# firewall-cmd --reload
success
[root@r ~]# firewall-cmd --list-ports
80/tcp
```

Failed to resolve host name updates.jenkins.io. Perhaps you need to configure HTTP proxy?

环境是阿里云centos7

安装 allure 插件报错，提示配置 `configure HTTP proxy`。那就按照提示去配置。

在 `系统管理` 下的 `插件管理` 选择 `高级` 选项。下来找到 `升级站点`，修改其中的 `URL`，将 `https` 改为 `http`，其余不动，最后点击提交。

Jenkins

插件管理

返回工作台

管理 Jenkins

更新中心

可更新

可选插件

已安装

高级

代理设置

服务器

端口

用户名

密码

不通过代理的主机

提交

高级...

上传插件

您可以通过上传一个.hpl文件来安装插件。

文件:

选择文件

 未选择任何文件

上传

升级站点

URL:

https://updates.jenkins.io/update-center.json

 将https改为http，其余不动

提交

21 分 之前获取了更新信息

立即获取

生成时间: 2019-11-28 上午01时15分20秒 [REST API](#) [Jenkins ver. 2.190.3](#) [Jenkins 中文社区](#)

完事之后，在浏览器访问 `http://localhost:8080/restart` 路由，提示你是否重启Jenkins，点击是重启即可。

要是还不行，就换个国内的镜像吧

将
`http://updates.jenkins.io/update-center.json`
替换为
`http://mirror.xmission.com/jenkins/updates/current/update-center.json`

提交后重启Jenkins，

see also: [docker: CentOS安装 docker和默认安装目录](#) | [如何进入、退出docker的container](#) | [Centos7上安装docker](#) | [Jenkins技巧: 如何启动、停止、重启、重载Jenkins](#) | [jenkins插件下载地址](#)