

由于工作需要，这里我先创建一个vue的工程。

1.首先安装好gitlab相关插件：GitLab、GitLab Hook、NodeJS

插件安装参考：<https://www.cnblogs.com/jxd283465/p/11542680.html>

2.jenkins服务器安装git 和 nodejs

git安装：

- yum -y install git

nodejs安装：这里我安装的在home路径下，可自行更改。

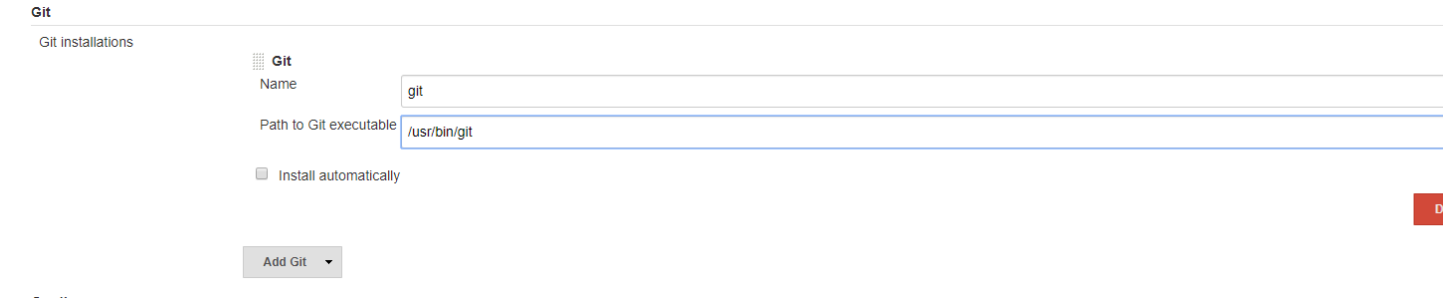
- wget https://nodejs.org/dist/v9.8.0/node-v9.8.0-linux-x64.tar.xz
- tar -xvf node-v9.8.0-linux-x64.tar.xz
- mv node-v9.8.0-linux-x64 node
- sudo ln -s /home/node/bin/node /usr/local/bin/
- sudo ln -s /home/node/bin/npm /usr/local/bin/
- sudo ln -s /home/node/bin/node /usr/bin

```
1 [root@localhost home]# node -v
2 v9.8.0
```

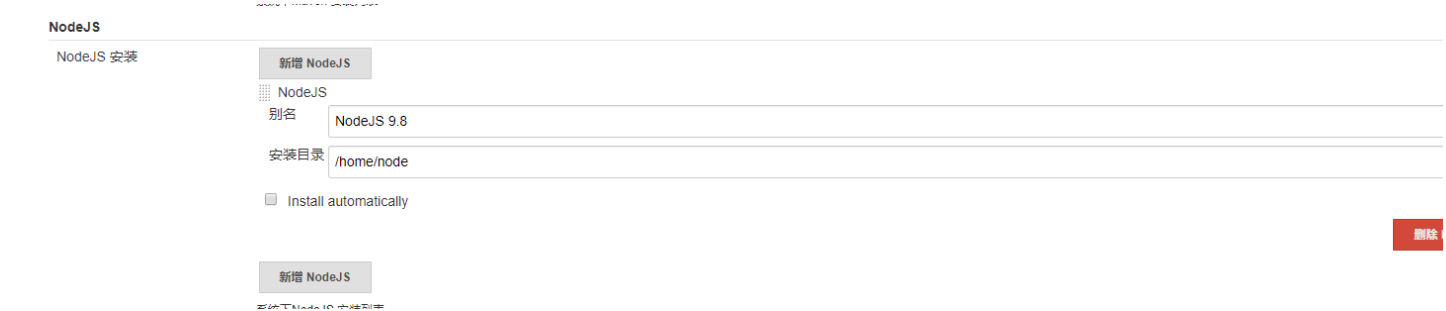
3.配置jenkins全局工具,这里暂时配置git和nodejs

点击 “系统管理” -----Global Tool Configuration

git配置：yum安装完成后git路径为/usr/bin/git



nodejs配置：这里选择安装路径为/home/node



配置git和nodejs环境结束后点击保存。


4.新建jenkins工程。

由于这里建的是vue的自动化部署项目，所以新建一个FreeStyle project.

输入一个任务名称


MingByteWeb

» 必填项




Freestyle project

This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for:




构建一个maven项目

构建一个maven项目.Jenkins利用你的POM文件,这样可以大大减轻构建配置.




流水线

精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作流），增加或者组织难以采用自由风格的任务




External Job

This type of job allows you to record the execution of a process run outside Jenkins, even on a remote machine. This is designed so that you c




构建一个多配置项目

适用于多配置项目,例如多环境测试,平台指定构建,等等.




MultiJob Project

MultiJob Project, suitable for running other jobs




Job Generator

A job which generates a new job when executed. It can be parameterized via the standard build parameters of Jenkins by adding Generator P




GitHub 组织

扫描一个 GitHub 组织（或者个人账户）的所有仓库来匹配已定义的标记。



GitLab Group

Scans a GitLab Group (or GitLab User) for all projects matching some defined markers.



多分支流水线

确定 后一个SCM仓库中检测到的分支创建一系列流水线。

General

源码管理

构建触发器

构建环境

构建

构建后操作

描述

工程描述

[Plain text] 预览

☒ Discard old builds

策略

Log Rotation

保持构建的天数

5

如果非空，构建记录将保存此天数

保持构建的最大个数

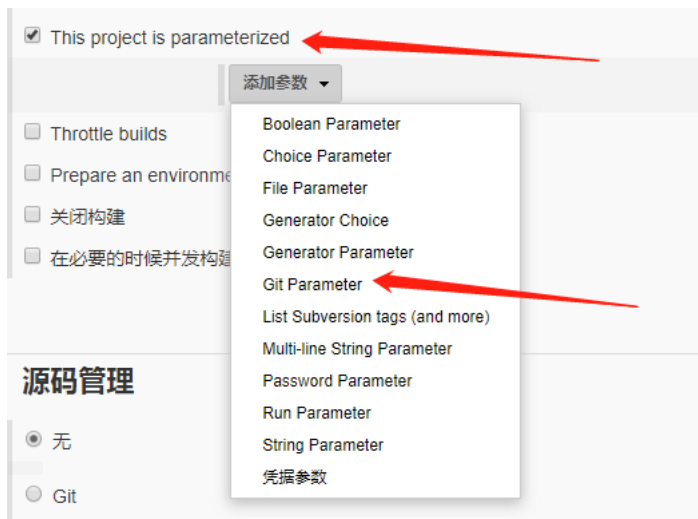
5

如果非空，最多此数目的构建记录将被保存

由于这里使用的是git，需要加一个参数化构建。

file:///C:/Users/jxd/Desktop/Jenkins文档/自动化运维三 Jenkins整合gitlab、docker发布vue项目.html

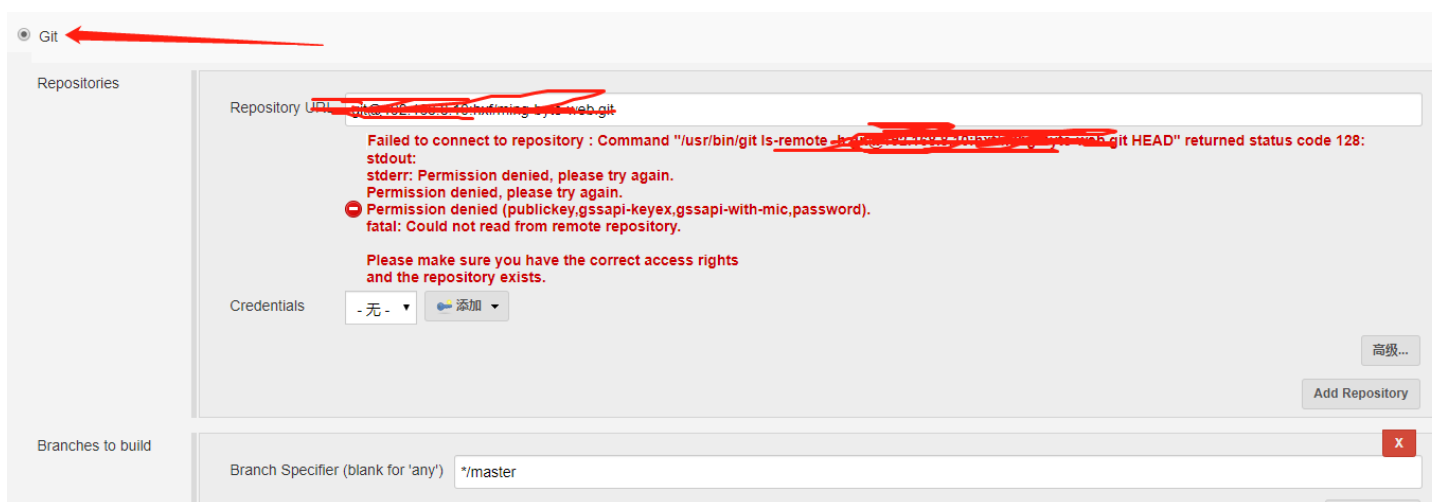
2/11



这里，我后续会配置，工程构建成功后会打一个git的tag，作为备份。以防构建失败后无法回滚。



代码管理工具选择---- "git"



Repository URL : 填写gitlab项目的git地址

此处报错是因为要添加Credentials凭证。

在jenkins的主机上执行

ssh-keygen -t rsa -C "<填写自己方便识别的注释>" -b 4096 没什么问题就执行三次空格。
三次问题是1.填入生成密钥对的路径名字。2 填入自定义passphrsa。3确认。

```
1 [root@localhost home]# ssh-keygen -t rsa -C "*****" -b 4096
2 Generating public/private rsa key pair.
3 Enter file in which to save the key (/root/.ssh/id_rsa):
```

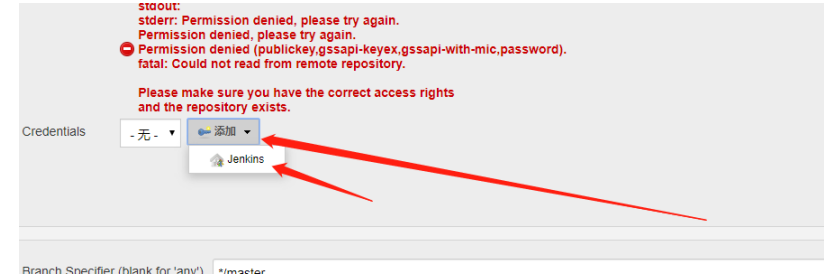
```
4 Enter passphrase (empty for no passphrase):
5 Enter same passphrase again:
6 Your identification has been saved in /root/.ssh/id_rsa.
7 Your public key has been saved in /root/.ssh/id_rsa.pub.
8 The key fingerprint is:
9 SHA256:oQYIN3KUKPmmCnOLrC25h5L0yVpiIvoHugawDuyCKuE *****
10 The key's randomart image is:
11 +---[RSA 4096]-----+
12 |*==|
13 ||=*,o|
14 |o+..|
15 |=+...|
16 |+B o o S|
17 |*o*..|
18 |BEo=..|
19 |# =+o|
20 |+*o.|
21 +-----[SHA256]-----+
```

生成的公钥和私钥在/root/.ssh目录下

```
1 [root@localhost .ssh]# pwd
2 /root/.ssh
3 [root@localhost .ssh]# ls -l
4 总用量 12
5 -rw-----. 1 root root 3247 9月 19 10:00 id_rsa          #私钥
6 -rw-r--r--. 1 root root 746 9月 19 10:00 id_rsa.pub      #公钥
7 -rw-r--r--. 1 root root 174 9月 18 16:58 known_hosts
```

```
1 # 查看私钥
2 [root@localhost .ssh]# cat id_rsa
3 -----BEGIN RSA PRIVATE KEY-----
4 MIIEJwIBAAKCAgEAtYMEUrqLRFVbBTXo89E8p+ci8N1ySimuuN1Shp0DrVNEkx89w
5 9gs1w6IU4xveb914L5jJ1TVeSGNq1HdPgTL1VDTgNWGQHxv94dT8XptPKcPC5NUN
6 I5/eJmvtpJRdVWPYuav1GAR9nDLZYUmTgwaI9DbXeTUuB0oyTzsR3KvcW0Uhyx2p
7 L5Rmdc4n0+K4MPigq51c5JCDvuJL6kjh3f1MI6VMSi5Jk+CrQWWYFq6PBcA6ZHv6
8 OQTR8rk/xKgeTNxtLZJxoI+OcHyH5vff9XffzKV1DbURu4a78rsWMRiXKSkvhZAV
9 VrdPVMT50pSrYrRha17sy4jh9ZNDQE69zNA9I3YjZ+9KFNKvjYXv3cqmq0/zojch8
10 dAs2xuC9hU+oZcBxv2GrbH2I6aYkMcBjHIuUCYxAN+9bS7ok3krYqPLKvN2BU71b
11 j1M15ynW12o9AcMocuzFyZs jJfhq/OgBQvtXhnWckmt3beS5H/MKZuzt/+5yfjrm
12 Drdw+38GnhvikKaMyhk7bbNvKJGLvXtXhN3JHDtRM1ruKQs7bWD064IQEm3c+1e
13 pvc7e1qgs2LSFNmxY6lLp1ygcqaJka8/NTDT4+QxNXH0d76qIT/bXd72ffxPQHAy
14 2pzKaJQjAo7T203EXJ4qSPcVxHOAACQ7B0fN8mqYWEVy8iEB8vkTz+j3w0UCAwEA
15 AQKCAgEAj1Uu.jMWSRBvDDJjNqnuIEm6LQPKdz0KrHoBnMq46RRyEK+S7pw+jHSFY
16 rR7MwDZSmgNk69aA5Vjce3rb5YJzZdN7Wav5SxennDW8NFkgYjKcLGeXQHh1zJfD
17 z6D5Mvtn3EGL7kcF8RvtEuNmOIINBE6hhRgjJEiut5EFKYCEpCd49NCzre5UT3C
18 ZaeB6G9Cz8gDetpDrvCOUGw3Dq1++ocFF26d25Cgq9W2C/M8LF3mrvn2mTnBn+ewf
19 VwQLKVG1xgkmgUv5iPpNty32RLIsmcXucSyTHCqBMGRcWdi5RVqGkEn/D8Mheicx
20 wAB2djNK2dyr003+wRXQ/W8wohOusoweJjywsOgGLuV5mIUz2gZ0eirGVoy/k2WH
21 ba2Cd5+TZGR4fj1Ds0ezS1P9++cW8C80qMnp/8gVMAzhnJ/EXVLTG+1+t14L1oaM
22 ZrZU1mqWErAq2Crh27v1VJ1tnNuRTf5/qjimiJkWr5qt4CzCAzmXUz1+Ldy0HK
23 nlkKfSKWdAfVFU+lgIEEuKMBVZ4xkzyxTFjgwn7Wg5I1aEklyeIf28hjcvUImyC
24 gpn1baYcziCFz04AJGUHohgBuRZ0kD1pnESK3U8djAhnX0hD0q1SzXkAzsBKc+bV
25 zFLARmnzA0tnyPV1m+Ej5eAJuPQ+AaBGd155XETeLb3DeEQqtuECggEBANzG0Z4K
26 VuG5n+0Hu0a5vQExEfi/suzFof5UC2okBgZq9a07Iran6Z7y/8j3XsAC4W9kMOf
27 9ssXgQgDaUvgYFEPB1NulmzYL0Iud3rRnxTzOORSNklwWzCAWFLPYy7YkU1VkQoh
28 MLUKU0A0VPPGGZCVfVBHPS81T1rz4fg14HcSBqaUKWT+PPOwRk+XxneDd/4Zv/JPH
29 7zrtgBvfGicGSWrrp2UPvu4bkXFXBrX9hNNdCkHR+uP43uCA0sHrC+W2deuPCnC
30 m40PvnfHk3EKJfJu+dTYSr64N1ZRg5sM+X9mLgDi6Rmx56g0/ipt0cKMPwPnogK
31 6rH6HqKMGH+TqwPMCggEBANJ5EdGFAdh4IUJwYqF1+1JM/UOOWRrZnfQqgr1bgJ01
32 +K11NdGQctGUpS3XOn1ho30y7uoa+0A6KSH6mTMedFYSEPFvIwiszmmEnS3nFpON
33 jRSHTRJyc+N2OI6Q7NpUxq19dwxmt6wInv4p5vK/7Pmh+gkN184HzmqofRRd6jYI
34 rAoFILIjY2whsnsLy/BqpC60/Odm6cmQtaQuvzJKyP6Vp9zsavXUD1idRfMEueaK
35 Nqa8K7K2MbXQ1LcQjM3BVxodpD9+MB55mo10oD2UnpEGaAXkx0eouv1o+UiVbz1i
36 6FB7PyMfm124S+2ZA9+Sna44NzPYhcLHMqFPm1JYLMcCggEBALYbwK0vZCk/v7ak
37 Xo3bSh9whZ2FKycJgq1SUDQWUPNDMWuiw7dpz0U8GrJzuBmVc8m263GiQWV1rZw
38 ceVhwks6t1IKNw74UR61K60KUMjtrCPLAHGm0GvrrKtqLCSZMwnyefg8FnxPbGb0
39 TP4BWBxHvQYP0zLci6uG4hIcjd1pBzF/Ds1R7vPFL/styYgSNtODDnosIpaed91t
40 skK2mdrZr1261ki8bskRY46891KN4GraNIQhpaJcRTEfvBVpNodW70rLq380Rq
41 H3jGpJvrKVaopW+Bj+9eL553/HtLKy0mp4pNyX18tjzYabjsnC1fD0q07pEhZIMx
42 gp9w3qcCgEBAKqV//Ad611Fg4dydS3uHw4WqjvbZ7KdDASSka1LUjpPxRvi3x
43 crCk1n/h6n3Iug6jHuvW50g8AefMZ3x9/5upBbaqhl10dLh7fbvTiGhWSojQz9J
44 Tk4FWCKi721aRS29M8zTxh8gEi1NR6smaf2NU4yvStWv1VfSONB/vs0aknF0t1a
45 ESSxzLda8y00kUc4cG3cpKw29eXH9XmMSPOLgBsCYD2/y2pQYsCxuu50ZTkcWBpn
46 +zDQc6fYHRcCtdLv5ViksOfyU+Mnu85Nb3tBgvwL4cxdoDVXdhVoyfUp/fqE+bZA
47 /s61+bw4AaCqF2zGGU/HFRq6ESjr2YqS9q8CggEBAKaf57DZtm4W7uBDekaBvg1J
48 ZBhF/h8UXNPsFePwFBM8u96taXN2noHrSuSuM59SSMfyQK/mpnaLXmyVPM3Scsw1
49 RT8MvtaPZAV08qTnc/AcUjFa2QrmTIU3TBOv8GNJCXnXbZTLewDwBW7xwcYApCSh
50 UL5b3B3RfByij3Vqhak6q2dFOE+6AhLVaHuuW6FKLb7NgioCVmzpIBJUSXfXiRH/S
51 h35/TgVtoY95+pIFS/xi+Y2hFEDZcdvKhlXxStYLANUS1dMdkcwnd+JiQ7RWhd
52 vJ2dqNaYK+Zqs1zXfqYLP1GFYvVZGAp67D6anoM7Rv7td+tHScSffjzMETuc1M=
53 -----END RSA PRIVATE KEY-----
54
55 # 查看公钥
56 [root@localhost .ssh]# cat id_rsa.pub
57 ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQCAQC1gwrSuotEVVsFNejz0Tyn5yLw3XJKKa643VKgk40tU0THz3D2CzXDohTjg695v2XgvmMmVNV5IY2rUd0+BMuVUN0A1YZAFG/3h1Pxem08pw8Lk1Q0jn94ma+2k1I
```

jenkins中配置私钥：



Jenkins 凭据提供者: Jenkins

添加凭据

Domain

全局凭据 (unrestricted)

类型

SSH Username with private key

范围

全局 (Jenkins, nodes, items, all child items, etc)

ID

描述

Username

jenkins

Private Key

Enter directly

Key

No Stored Value

Passphrase

添加

取消

此处输入上文
中生成的私钥

gitlab中配置公钥：

Search or jump to...

Administrator

@root

Set status

Profile

Settings

Sign out

User Settings

Profile

Account

Applications

Chat

Access Tokens

Emails

Password

Notifications

SSH Keys

GPG Keys

Preferences

Active Sessions

Authentication log

User Settings > SSH Keys

SSH Keys

SSH keys allow you to establish a secure connection between your computer and GitLab.

Add an SSH key

To add an SSH key you need to [generate one](#) or use an [existing key](#).

Key

Paste your public SSH key, which is usually contained in the file '~/.ssh/id_rsa.pub' and begins with 'ssh-ed25519' or 'ssh-rsa'. Don't use

Typically starts with "ssh-ed25519 ..." or "ssh-rsa ..."

Title

e.g. My MacBook key

Name your individual key via a title

Add key

Your SSH keys (2)

root@localhost.localdomain 50:0c:c2:0e:8d:6a:a0:51:fe:94:ca:b0:e5

Git

Repositories

Repository URL

https://root@localhost.localdomain:5000/

Credentials

jenkins

- 无 -

jenkins

Name

Refspec

这里选择新增加的凭证，已不报错

file:///C:/Users/jxd/Desktop/Jenkins文档/自动化运维三 Jenkins整合gitlab、docker发布vue项目.html

5/11

Branches to build

Branch Specifier (blank for 'any')

Add Branch

这里 “Branches to build” 处为上文中参数化构建的Name，此处为\$Tag, “\$” 表示调用变量。

至此，jenkins整合gitlab完成。

由于每次gitlab代码提交，都要发布jenkins很麻烦。所以这里使用gitlab hook自动触发，此处的webhook要复制。

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

☒ Build when a change is pushed to GitLab. GitLab webhook URL: <http://192.168.1.100:8080/job/vue-project/builds?token=1234567890>

Enabled GitLab triggers

Push Events☒

Opened Merge Request Events☒

Accepted Merge Request Events☒

Closed Merge Request Events☐

Rebuild open Merge Requests

Approved Merge Requests (EE-only)☒

Comments☒

Comment (regex) for triggering a build

Enable [ci-skip]☒

Ignore WIP Merge Requests☒

Set build description to build cause (eg. Merge request or Git Push)☒

Build on successful pipeline events☐

Pending build name for pipeline

Cancel pending merge request builds on update☐

Allowed branches

☒ Allow all branches to trigger this job

☐ Filter branches by name

☐ Filter branches by regex

☐ Filter merge request by label

保存

应用

这个URL要复制，待会要添加进gitlab中

配置gitlab相应项目的webhook

file:///C:/Users/jxd/Desktop/Jenkins文档/自动化运维三 Jenkins整合gitlab、docker发布vue项目.html

6/11

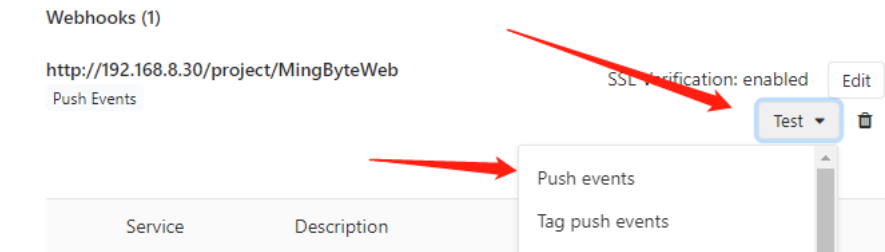
The screenshot shows the Jenkins 'Integrations' page. On the left sidebar, a red arrow points to 'Settings' and another points to 'Integrations'. The main content area shows the 'Integrations' section with a 'URL' field containing a redacted URL, a 'Secret Token' field with the text 'jenkins中的webhook', and a list of event triggers. The 'Push events' trigger is selected.

如果提交webhook报错 url is blocked request to the local network are not allowed.

gitlab 10.6 版本以后为了安全, 不允许向本地网络发送webhook请求, 如果想向本地网络发送webhook请求, 则需要使用管理员帐号登录, 默认管理员帐号是admin@example.com, 密码就是你gitlab搭建好之后第一次输入的密码, 登录之后进行如下配置即可。

The screenshot shows the GitLab Admin Area 'Network' settings page. Red arrows and numbered annotations indicate the steps to configure outbound requests:

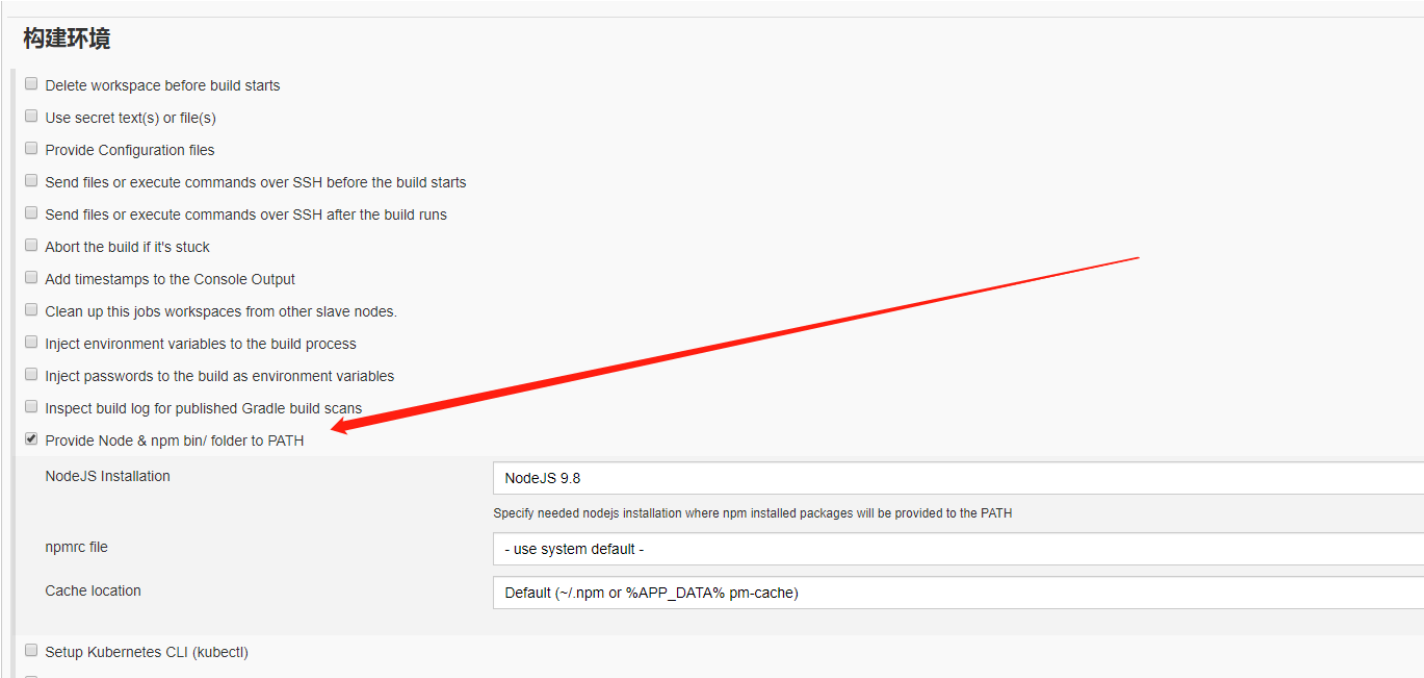
1. 点击Admin Area (Click Admin Area)
2. Settings (Click Settings in the left sidebar)
3. Network (Click Network in the left sidebar)
5. 勾选第一条 (Check the first checkbox: 'Allow requests to the local network from web hooks and services')
6. 保存 (Click 'Save changes')



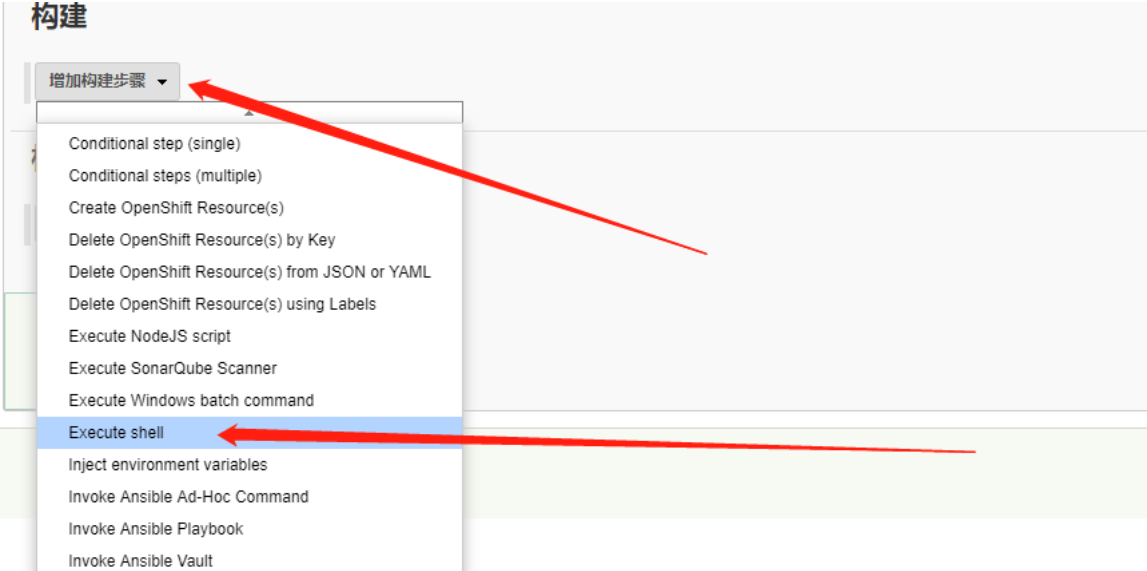
此处添加成功后，可以进行Test触发测试。这里我就不测试了。

继续我们的jenkins工程创建。

构建环境处由于这个工程是vue工程，这里选择node，此处的Node JS v9.8是上文中在全局工具中配置的。



接下来，添加构建步骤。由于我需要打包docker容器，这里选择的是执行shell脚本。



此处贴出shell脚本

```
# /home/jenkins/.jenkins/workspace为jenkins工作目录
# MingByteWeb为我jenkins创建的项目名称
cd /home/jenkins/.jenkins/workspace/MingByteWeb
# /home/jenkins/jenkins.sh这个脚本是我用于删除多余的tag和删除之前运行的容器
bash /home/jenkins/jenkins.sh 192.168.8.10:5000/mingbyteweb ming-byte-web
# 这里是vue打包
echo '=====vue打包====='
```



```
# docker推送镜像到docker私服
sudo docker push 192.168.8.10:5000/mingbyteweb:latest
echo '-----结束推送镜像-----'
# docker启动镜像
sudo docker run -d --name MingByteWeb -p:8081:8080 192.168.8.10:5000/mingbyteweb
echo "finished!"
# 构建成功!
```

jenkins.sh

```
t='sudo docker ps | grep $1|awk '{print $1}'|sed 's/%%/g'`;
b='';
if [ $t ];
then
sudo docker stop $t
echo "停止容器成功"
sudo docker rm $t
echo "删除容器成功"
else
echo "首次部署";
fi

tagnum=`git tag | wc -l`;
tag=(`git tag`)
a=0;
echo $tagnum
until [ $tagnum -lt 3 ]
do
echo $a 删除tag: ${tag[$a]}
git tag -d ${tag[$a]}
git push $2 :refs/tags/${tag[$a]}
a=`expr $a + 1`
tagnum=`expr $tagnum - 1`
done
```

构建项目:



返回面板



状态



修改记录



工作空间



Build with Parameters



删除 Project



配置



重命名

Project MingByteWeb

需要如下参数用于构建项目:

Tag

rc_59
rc_60
rc_61

如需发布新版本则点击“开始构建”。如需回退版本，请选择需要回退的Tag

开始构建

这里点击“开始构建”是直接拉取gitlab最新版的master分支代码。也可以选择tag构建。之后我会写构建生成tag的文档。

最后贴图构建成功的代码。

```
Started by user admin
Running as SYSTEM
[EnvInject] - Loading node environment variables.
Building in workspace /home/jenkins/.jenkins/workspace/MingByteWeb
using credential 26e43bbc-407f-42ff-8865-41496cf12fe9
> /usr/bin/git rev-parse --is-inside-work-tree # timeout=10
Fetching changes from the remote Git repository
> /usr/bin/git config remote.origin.url git@192.168.8.10:hxf/ming-byte-web.git # timeout=10
Fetching upstream changes from git@192.168.8.10:hxf/ming-byte-web.git
> /usr/bin/git --version # timeout=10
using GIT_SSH to set credentials
> /usr/bin/git fetch --tags --progress git@192.168.8.10:hxf/ming-byte-web.git +refs/heads/*:refs/remotes/origin/*
> /usr/bin/git rev-parse origin/master^{commit} # timeout=10
Checking out Revision f94b7302025142f80aa671d6d3dlffe0f5601807 (origin/master)
> /usr/bin/git config core.sparsecheckout # timeout=10
> /usr/bin/git checkout -f f94b7302025142f80aa671d6d3dlffe0f5601807
Commit message: "fix:修复接口baseUrl"
> /usr/bin/git rev-list --no-walk f94b7302025142f80aa671d6d3dlffe0f5601807 # timeout=10
[MingByteWeb] $ /bin/sh -xe /home/tomcat/temp/jenkins205814157553423786.sh
+ cd /home/jenkins/.jenkins/workspace/MingByteWeb
+ bash /home/jenkins/jenkins.sh 192.168.8.10:5000/mingbyteweb ming-byte-web
首次部署
3
0 删除tag: rc_59
已删除 tag 'rc_59' (曾为 40dc597)
fatal: 'ming-byte-web' does not appear to be a git repository
fatal: Could not read from remote repository.

Please make sure you have the correct access rights
and the repository exists.
+ echo =====vue打包=====
=====vue打包=====
+ sudo /usr/local/bin/npm install
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@1.2.9 (node_modules/fsevents):
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@1.2.9: wanted {"os":"darwin","arch":"any"} (current: {"os":"linux","arch":"x64"})

up to date in 8.204s
+ sudo /usr/local/bin/npm run build

> ming-byte@0.1.0 build /home/jenkins/.jenkins/workspace/MingByteWeb
> vue-cli-service build

WARN "baseUrl" option in vue.config.js is deprecated now, please use "publicPath" instead.
```

```
- Building for production...
WARNING Compiled with 2 warnings11:55:52

warning

asset size limit: The following asset(s) exceed the recommended size limit (244 KiB).
This can impact web performance.
Assets:
  assets/img/en-jiao.bc226a6e.png (736 KiB)
  assets/img/fzlc1.04b5a7d6.png (1.72 MiB)
  assets/img/bg.10e66f2d.png (269 KiB)
  assets/img/neng2.51572385.svg (708 KiB)
  assets/img/aboutus-bg.24b126e4.png (3.06 MiB)
  assets/img/homeImg.2d4c24fe.jpg (605 KiB)
  assets/img/banner.d0de1296.png (877 KiB)
  assets/img/en-hu.10ba4555.png (748 KiB)
  assets/img/shuzi.0c4edbf4.png (2.82 MiB)
  assets/img/en-jiao-bg.327842dc.png (352 KiB)
  assets/img/banner.dc7b97a9.png (535 KiB)
  assets/js/chunk-vendors.eabed4cd.js (826 KiB)

warning

entrypoint size limit: The following entrypoint(s) combined asset size exceeds the recommended limit (244 KiB). This can impact web performance.
Entrypoints:
  app (1.07 MiB)
    assets/css/chunk-vendors.2921d934.css
    assets/js/chunk-vendors.eabed4cd.js
    assets/css/app.8be713f0.css
    assets/js/app.a67be888.js

File                                         Size              Gzipped

dist/assets/js/chunk-vendors.eabed4cd.js  825.72 KiB        222.24 KiB
dist/assets/js/about.1e94fa7f.js         30.84 KiB         8.89 KiB
dist/assets/js/app.a67be888.js           26.48 KiB         9.19 KiB
dist/assets/js/chunk-e4921c92.8eb56c35.js 5.43 KiB          2.13 KiB
dist/assets/css/chunk-vendors.2921d934.css 228.07 KiB        34.52 KiB
dist/assets/css/about.f15fe60f.css        16.90 KiB         8.56 KiB
dist/assets/css/app.8be713f0.css          10.34 KiB         4.10 KiB
dist/assets/css/chunk-e4921c92.9451e9f4.css 2.12 KiB          0.83 KiB

Images and other types of assets omitted.

DONE Build complete. The dist directory is ready to be deployed.
INFO Check out deployment instructions at https://cli.vuejs.org/guide/deployment.html

+ echo =====打包docker=====
=====打包docker=====
+ docker build -t 192.168.8.10:5000/mingbyweb:latest .
Sending build context to Docker daemon 323.8MB

Step 1/6 : FROM nginx:1.15.5
--> dbfc48660aeb
Step 2/6 : RUN rm /etc/nginx/conf.d/default.conf
--> Using cache
--> 6f887258cdf3
Step 3/6 : ADD default.conf /etc/nginx/conf.d/
--> Using cache
--> b2292a96d1aa
Step 4/6 : COPY ./dist /usr/share/nginx/html/
--> Using cache
--> 717db4eaf72a
Step 5/6 : RUN chmod -R a+rx /usr/share/nginx/html/*
--> Using cache
--> fd948904b5d3
Step 6/6 : CMD nginx -g 'daemon off;'
--> Using cache
--> a58cefe00e53
Successfully built a58cefe00e53
Successfully tagged 192.168.8.10:5000/mingbyweb:latest
+ echo =====开始推送镜像=====
=====开始推送镜像=====
+ sudo docker push 192.168.8.10:5000/mingbyweb:latest
The push refers to repository [192.168.8.10:5000/mingbyweb]
8121fdecba88: Preparing
a44f0decdd50: Preparing
e3de46667e06: Preparing
38b7b8ba2bf9: Preparing
86df2alb653b: Preparing
bc5b41ec0cfa: Preparing
237472299760: Preparing
bc5b41ec0cfa: Waiting
237472299760: Waiting
e3de46667e06: Pushed
38b7b8ba2bf9: Pushed
86df2alb653b: Pushed
a44f0decdd50: Pushed
8121fdecba88: Pushed
bc5b41ec0cfa: Pushed
237472299760: Pushed
latest: digest: sha256:3808b7955c70fee07e92e75cf59c4b1585b20651a1094d7f6382e0441797d969 size: 1786
+ echo =====结束推送镜像=====
=====结束推送镜像=====
+ sudo docker run -d --name MingByteWeb -p:8081:8080 192.168.8.10:5000/mingbyweb
6d6160bcf0d7d638ae58368d1760d47e3d2ccd176aa282d31e859a26alf36db
+ echo 'finished!'
finished!
```

Notifying upstream projects of job completion
Finished: SUCCESS

最后注意gitlab这块有个问题。当提交代码到gitlab时，gitlab的webhook会触发jenkins自动构建。

但是可能会构建失败。

解决：jenkins的系统配置里面，将gitlab下的“Enable authentication for '/project' end-point”后面的√去掉

Gitlab

Enable authentication for '/project' end-point ☒

GitLab connections

Connection name

Gitlab connection name required.
A name for the connection

Gitlab host URL

Gitlab host URL required.
The complete URL to the Gitlab server (e.g. <http://gitlab.mydomain.com>)

Credentials

API Token for Gitlab access required
API Token for accessing Gitlab

vue项目需要用到的配置文件：

default.conf：

```
server {
    listen      7000;

    location / {
        root    /usr/share/nginx/html;
        index   index.html index.htm;
    }

    #error_page 404              /404.html;

    # redirect server error pages to the static page /50x.html
    #
    error_page   500 502 503 504  /50x.html;
    location = /50x.html {
        root    html;
    }

    # proxy the PHP scripts to Apache listening on 127.0.0.1:80
    #
    #location ~ /\.php$ {
    #    proxy_pass http://127.0.0.1;
    #}

    # pass the PHP scripts to FastCGI server listening on 127.0.0.1:9000
    #
    #location ~ /\.php$ {
    #    root            html;
    #    fastcgi_pass    127.0.0.1:9000;
    #    fastcgi_index   index.php;
    #    fastcgi_param   SCRIPT_FILENAME /scripts$fastcgi_script_name;
    #    include         fastcgi_params;
    #}

    # deny access to .htaccess files, if Apache's document root
    # concurs with nginx's one
    #
    #location ~ /\.ht {
    #    deny all;
    #}
}
```

Dockerfile：

```
FROM nginx:1.15.5
#dist到nginx部署
RUN rm /etc/nginx/conf.d/default.conf
#ADD
ADD default.conf /etc/nginx/conf.d/
COPY ./dist /usr/share/nginx/html/
RUN chmod -R a+rx /usr/share/nginx/html/*
CMD nginx -g 'daemon off;'
```