九、Jenkins构建java项目（分布式）

前言：

先把构建思路说一下，不然闷头看步骤有些晦涩。

1】微服务+前后端分离项目，要启动多个“**服务**”，包括**jar包服务**和一个**前端服务**。

每个服务都要打包一个镜像，然后生成容器去运行。

2】打包镜像，就需要提前准备打包镜像的依赖镜像。

3】jenkins创建的item/任务，只纳管一个服务，所有需要创建多个item/任务。

1.准备java依赖镜像

1）查看可用的 jdk/jre 版本

访问 docker 镜像库地址： https://hub.docker.com/，搜索jdk或者jre，挑选自己需要的版本

2）拉取镜像

```
docker pull kdvolder/jdk8:latest #下载最新版本
```

3）查看本地镜像

```
docker images
```

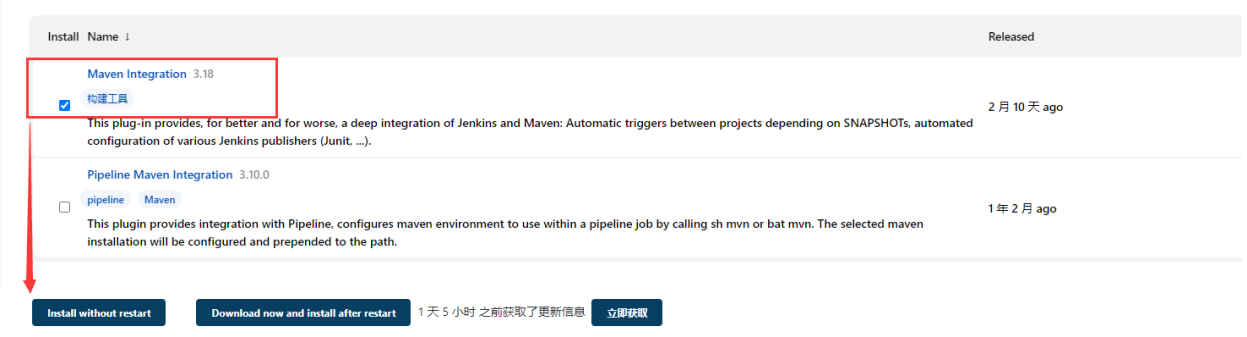
2.新建item/新建任务

3.构建一个maven项目

构建微服务的gateway网关服务为例，起名manstro-gateway

如果没有“**构建一个maven项目**”选项，是因为缺少插件：Maven Intergration

【系统管理】==>【管理插件】==>搜索**Maven Integration**插件==>点击 install without restart

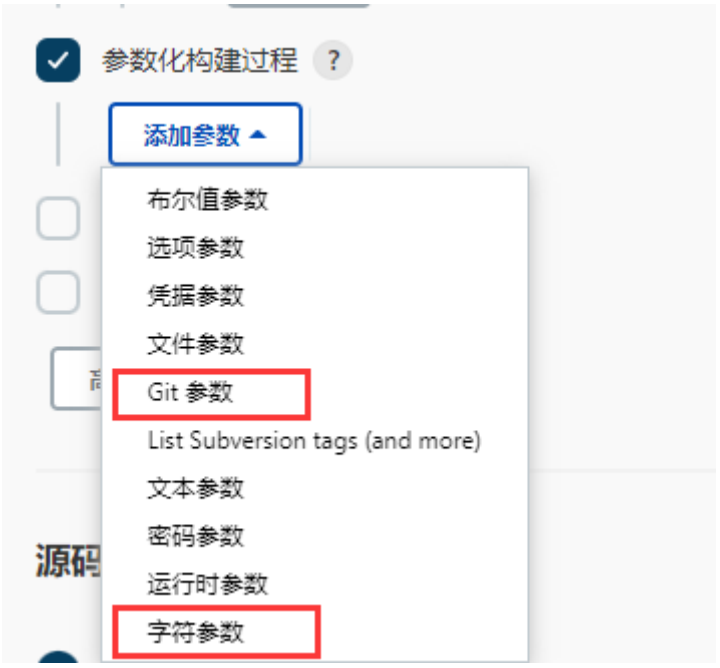


4.配置item/任务

1】Discard old builds/丢弃旧的构建

提供了“Discard old builds”的选项可以配置保留几天/最多几个的build 和 artifacts，缺省这个选项不会勾上。不勾选时，默认值为365，显然这个值过大了。

2】参数化构建过程



如果没有“**Git 参数**”，需要安装Git Parameter插件

【系统管理】==>【管理插件】==>搜索**Git Parameter**插件==>点击 install without restart

1）添加Git参数：branch（必配）
名称：branch
描述：git分支
参数类型：分支或标签
默认值：*/master

2）添加字符参数：host_port（必配）
名称：host_port
默认值：8080
描述：服务端口（宿主机暴露的端口，可以和服务配置的端口保持一致）

3）添加字符参数：docker_ip（必配）
名称：docker_ip
默认值：192.168.16.5
描述：服务IP（宿主机IP）

4）添加选项参数：profile_flag（必配）
名称：profile_flag
选项：
dev
prod
test
描述：运行环境

5）添加字符参数：images_name
名称：images_name
默认值：manstro-gateway
描述：镜像名称

6）添加字符参数：container_name
名称：container_name
默认值：manstro-gateway

描述：容器名称

7）添加字符参数：jar_name

名称：jar_name

默认值：manstro-gateway

描述：jar包名称

注意：这里的jar包名不是自定义的，打包出来是啥jar包可以先试试。

例如system打包出来就叫manstro-modules-system.jar。

8）添加字符参数：ssh_jar_file

名称：ssh_jar_file

默认值：/docker/project/manstro/gateway

描述：jar包文件路径

9）添加字符参数：ssh_logs_file

名称：ssh_logs_file

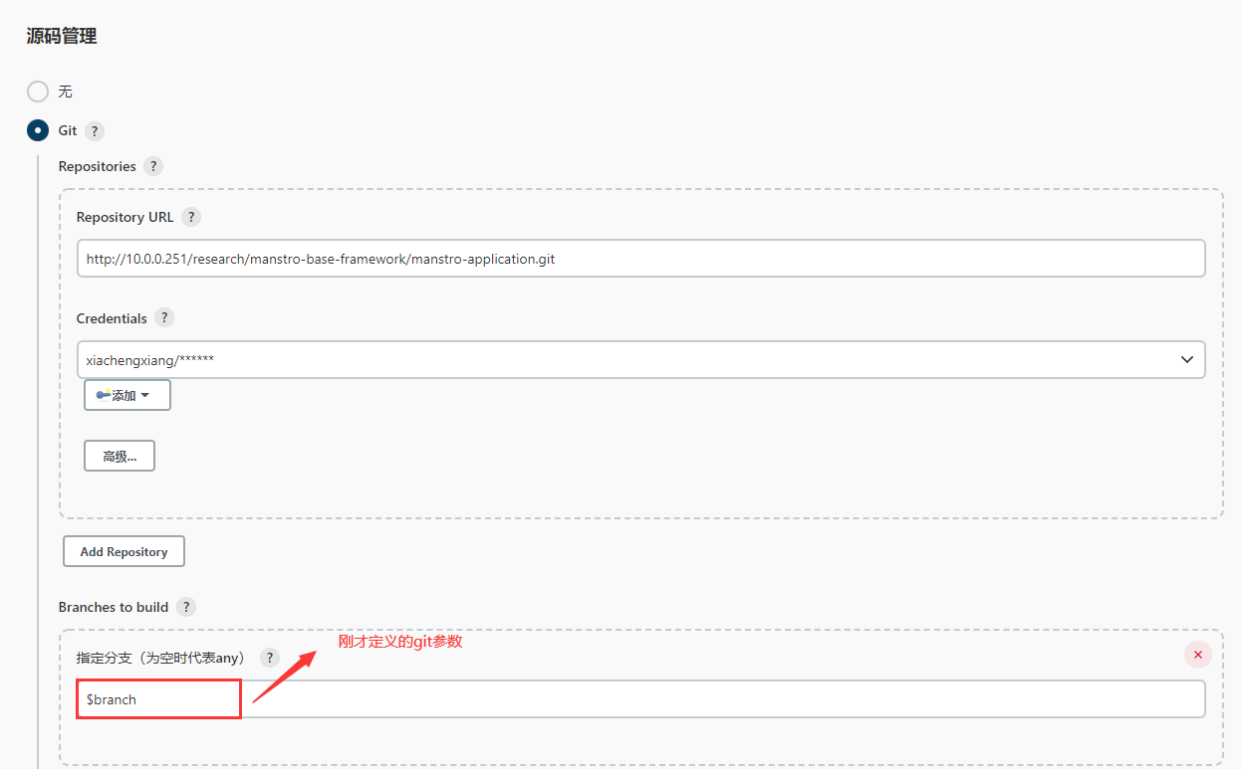
默认值：/docker/project/manstro/logs

描述：服务日志文件路径

3】源码管理

http://10.0.0.251/research/manstro-base-framework/manstro-application.git

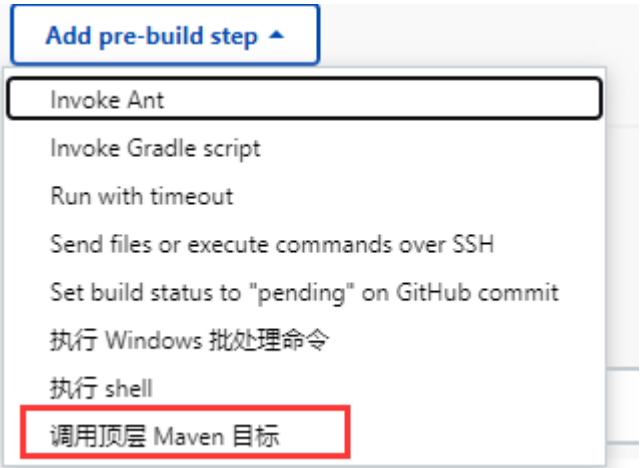
$branch



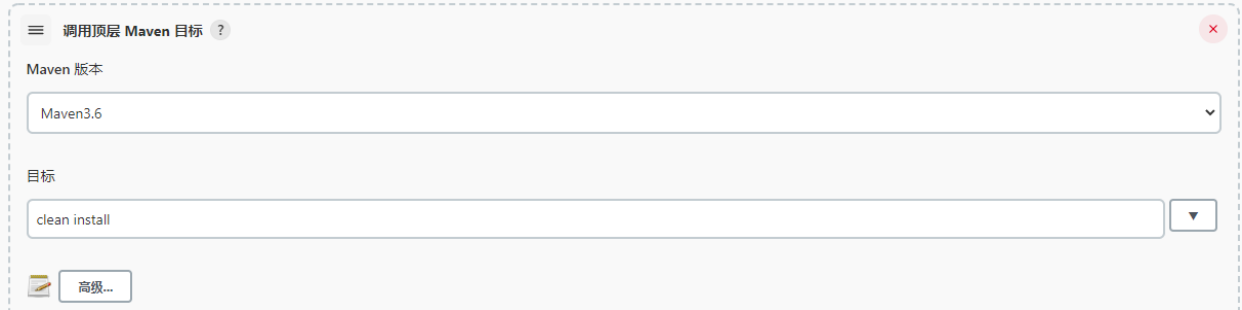
4】Pre Steps

拉取代码后优先执行此配置

1）选择 Invoke top-level Maven targets（调用顶层Maven目标）



2）选择maven版本，填写指令：clean install



5】Build

打包命令执行完成后根据build配置所在目录的pom文件进行docker镜像构建

Root POM：manstro-gateway/pom.xml

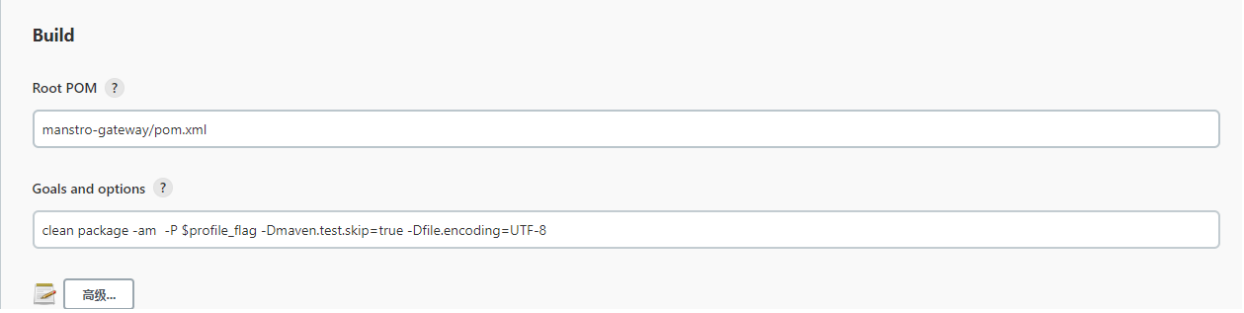
Goals and options：clean package -am -P $profile_flag -Dmaven.test.skip=true -Dfile.encoding=UTF-8

备注：

-am：构建指定模块，同时构建指定模块依赖的其他模块

-P：激活指定的profile文件列表(用逗号[,]隔开)

-D:定义系统变量



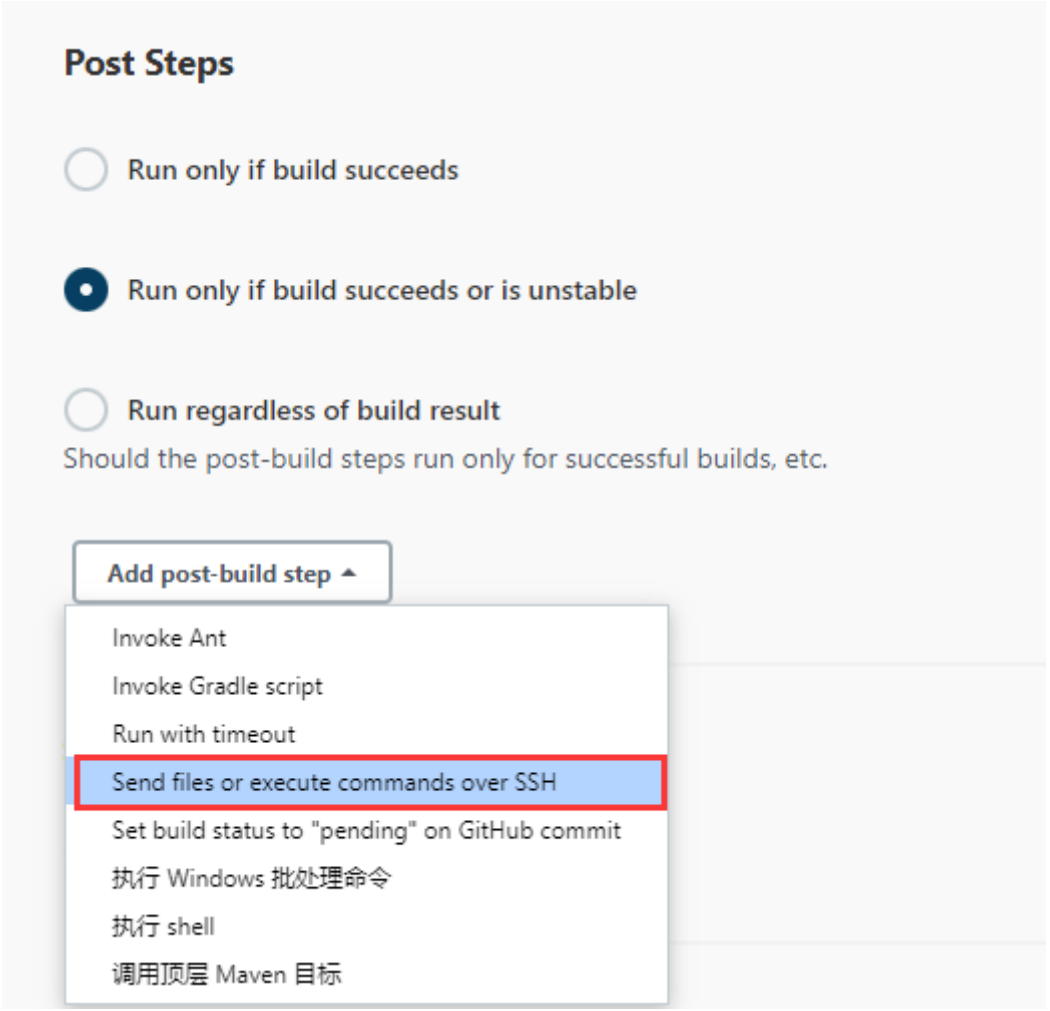
6】Post Steps

选择：Run only if build succeeds or is unstable

1）Send files or execute commands over SSH（上传文件到远程服务器）

点击：Add post-build step（增加构建步骤）按钮

选择：Send files or execute commands over SSH



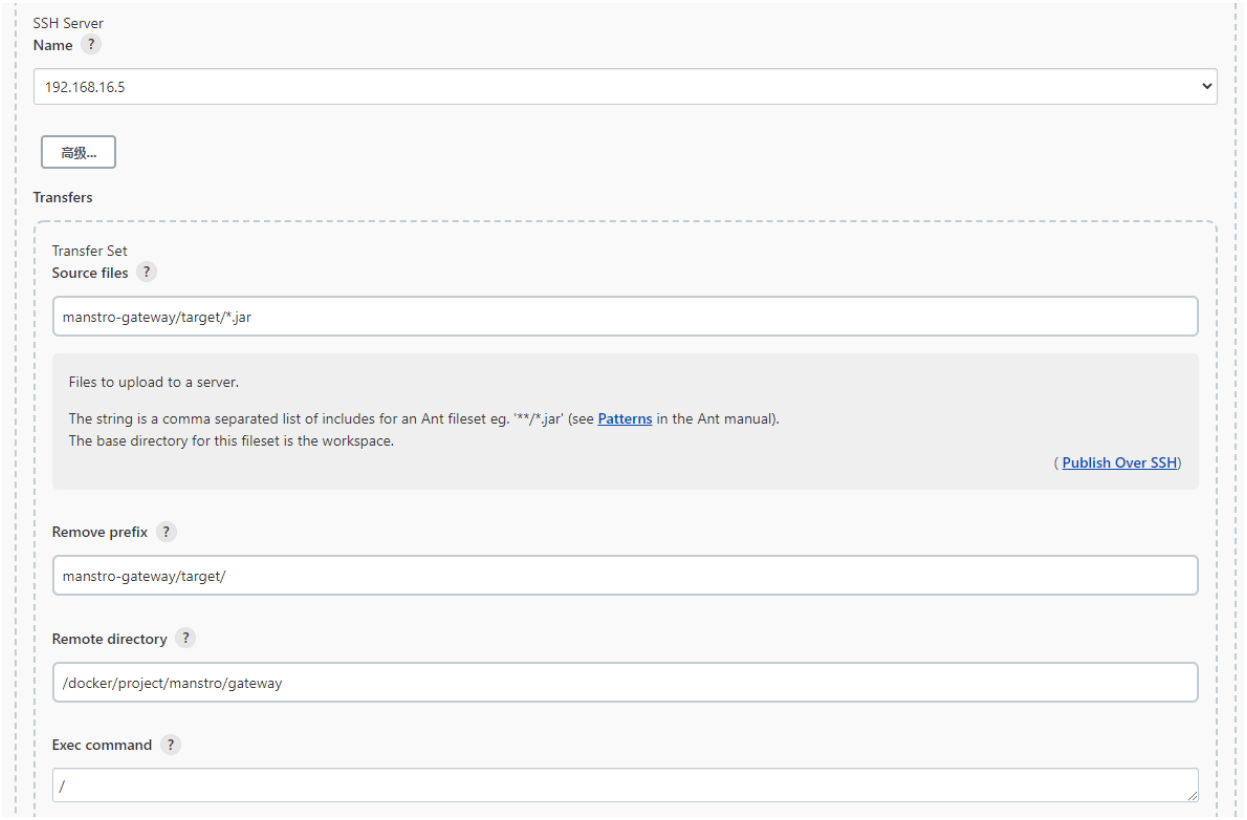
SSH Server：选择之前配置ssh地址

Source files（准备上传的源文件，=本地打包后的jar包路径）：manstro-gateway/target/*.jar

Remove prefix（删除前缀）：manstro-gateway/target/

Remote directory（远程目录，和上面配置的**ssh_jar_file**保持一致）：/docker/project/manstro/gateway

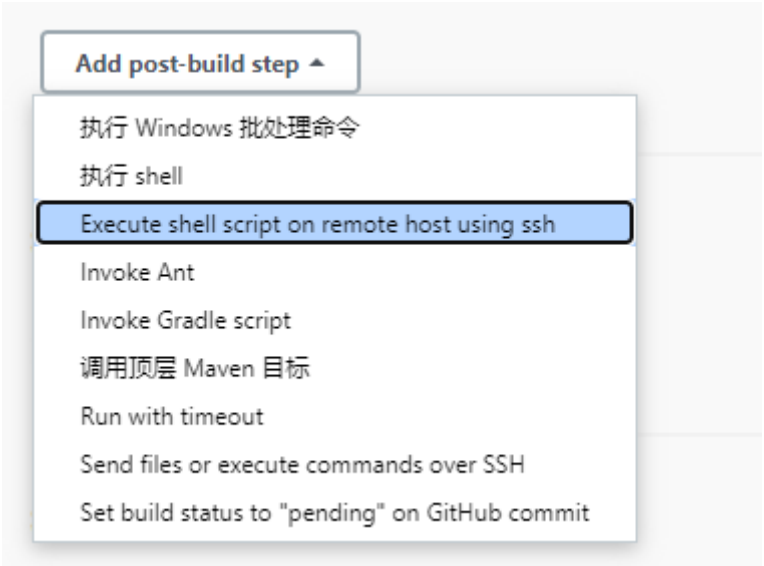
Exec command（Exec命令）：/



2）Execute shell script on remote host using ssh（在远程服务器上执行脚本）

点击：Add post-build step（增加构建步骤）按钮

选择：Execute shell script on remote host using ssh



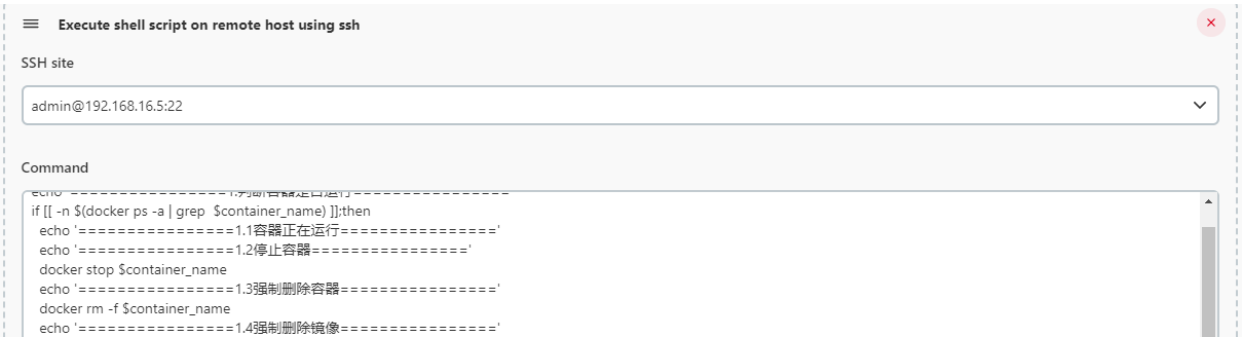
如果没有这个选项，请参考第三部分jenkins配置，下载ssh插件并在系统配置中配置相关内容。

Commad输入：

```
echo '================1.判断容器是否运行================'
if [[ -n $(docker ps -a | grep $container_name) ]];then
```

```
 echo '================1.1容器正在运行================'
 echo '================1.2停止容器================'
 docker stop $container_name
 echo '================1.3强制删除容器================'
 docker rm -f $container_name
 echo '================1.4强制删除镜像================'
 docker rmi -f $images_name
elseE
 echo '================2.1容器未运行================'
fi

echo '================2.进入jar包文件夹================'
if [ ! -d "$ssh_jar_file" ]; then
 mkdir -p $ssh_jar_file
fi
cd $ssh_jar_file
echo '================3.重新写入配置================'

echo "FROM kdvolder/jdk8" > Dockerfile
echo '# 指定字符集'
echo -e "ENV LC_ALL \"zh_CN.UTF-8\"" >> Dockerfile
echo '# 设置时区'
echo -e "ENV TZ \"Asia/Shanghai\"" >> Dockerfile
echo -e "RUN ln -snf /usr/share/zoneinfo/Asia/Shanghai /etc/localtime && echo 'Asia/Shanghai' >/etc/timezone" >> Dockerfile
echo '# 将本地文件挂载到当前容器'
echo "VOLUME /tmp" >> Dockerfile
echo '# 复制文件到容器'
echo "ADD ./$jar_name.jar /usr/local/$jar_name.jar" >> Dockerfile
echo '# 声明暴露的端口号'
echo "EXPOSE $host_port">> Dockerfile
echo '# 配置容器启动后执行的命令'
echo -e "ENTRYPOINT [\"java\",\"-jar\",\"-Xmx512m\",\"-Dfile.encoding=utf-8\",\"/usr/local/$jar_name.jar\"]" >> Dockerfile

echo '================4.打包镜像================'
docker build -t $images_name .
echo '================5.启动容器================'
if [ ! -d "$ssh_logs_file" ]; then
 mkdir -p $ssh_logs_file
fi
```

```
42  docker run --restart=always -p $host_port:$host_port -e docker_ip="$dock
er_ip" -e host_port="$host_port" --name $container_name -v
$ssh_logs_file:/logs -d $images_name:latest
```



5.确认配置文件是否正确

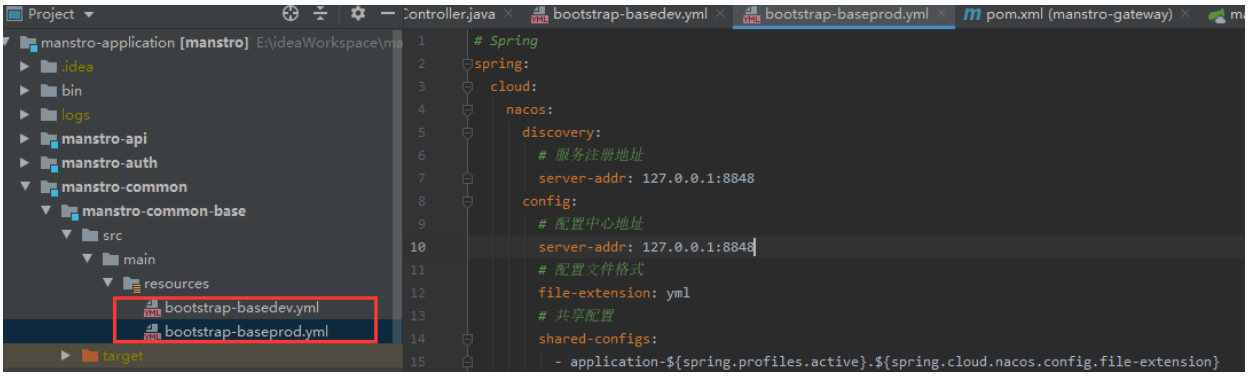
1）nacos配置

确认nacos地址、端口，是否与步骤6的nacos地址一致。

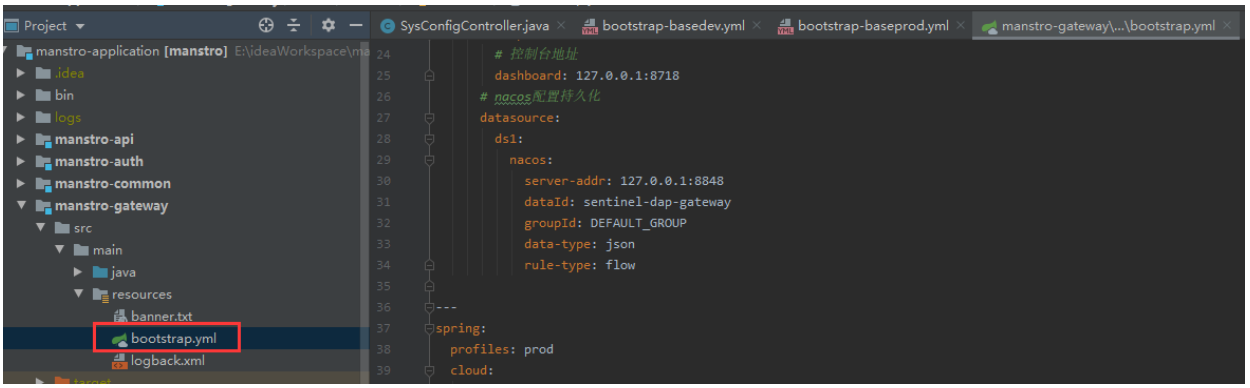
就本基线框架而言，需要确认通用配置和模块配置。

如果配置文件的nacos地址不对，后面附解决办法

通用配置如下图，

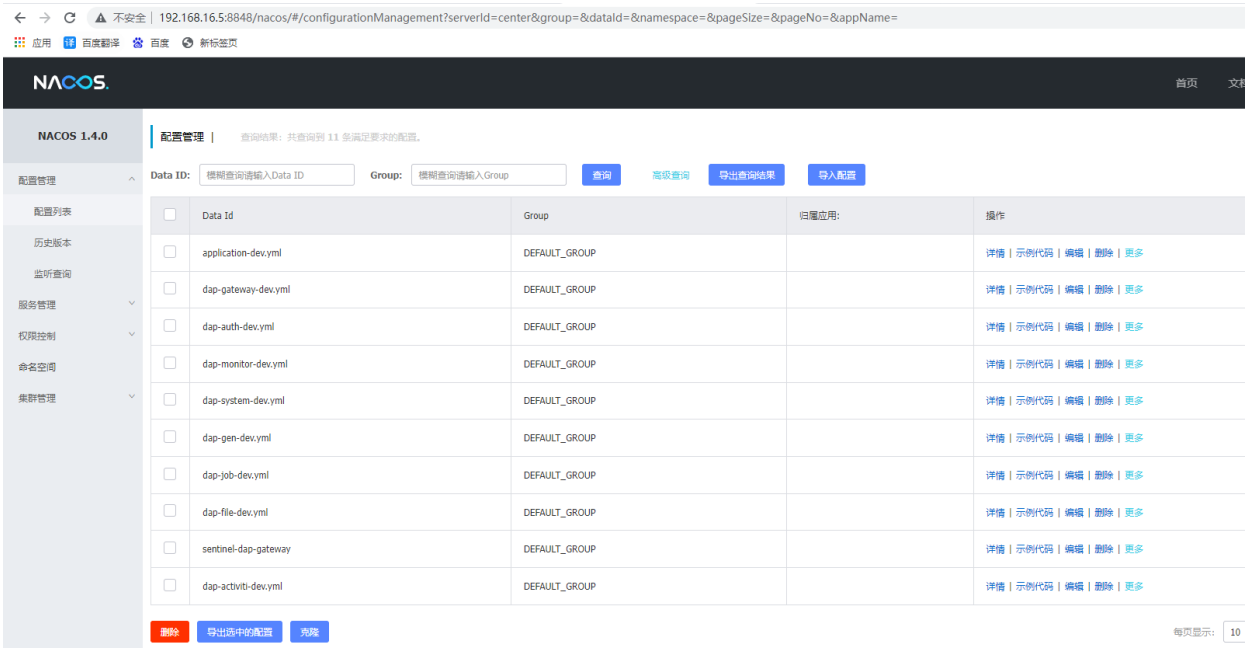


模块配置如下图



2）mysql和redis配置

确认mysql和redis的配置正确。登录第6步创建的nacos，确保每个配置文件配置都正确。



问题：git远端配置文件的nacos地址不对，但是基线代码不能随意修改提交怎么办？

解决办法：通过插件在构建项目之前替换配置文件

解决办法第一步：下载Config File Provider插件

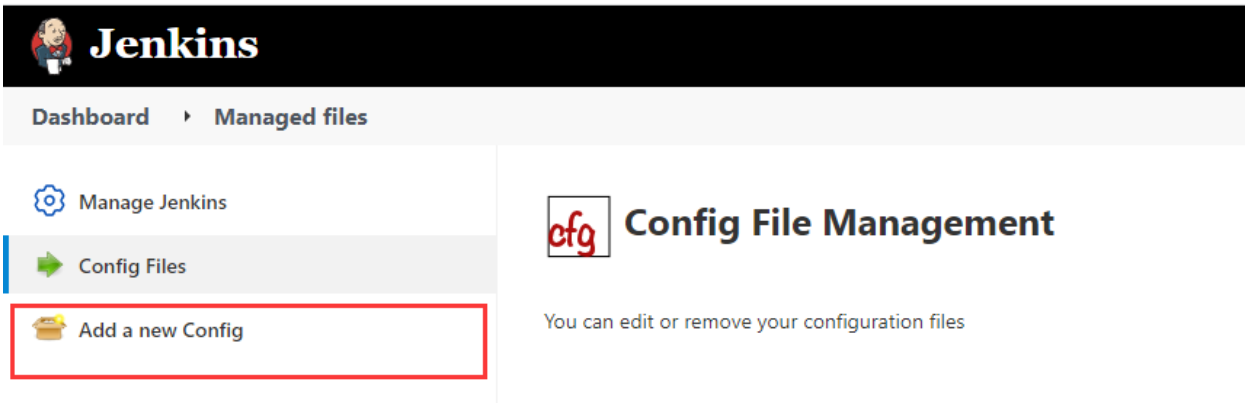
【系统管理】==>【管理插件】==>【可选插件】==>搜索**Config File Provider**插件==>点击 install without restart

这里还是需要重启jenkins的，看安装提示。

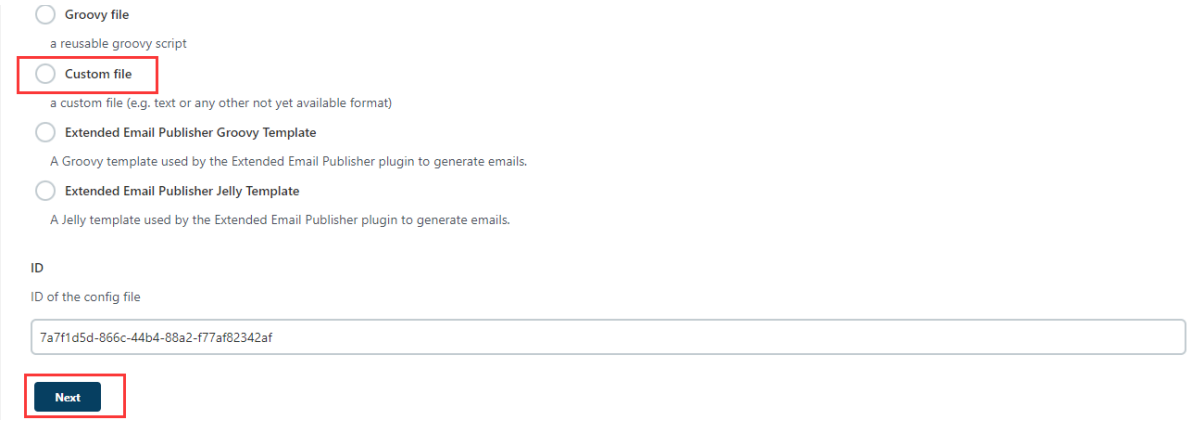
安装成功后，系统配置这里有这个图标

解决办法第二步：创建新的配置文件管理

点击Add a new Config

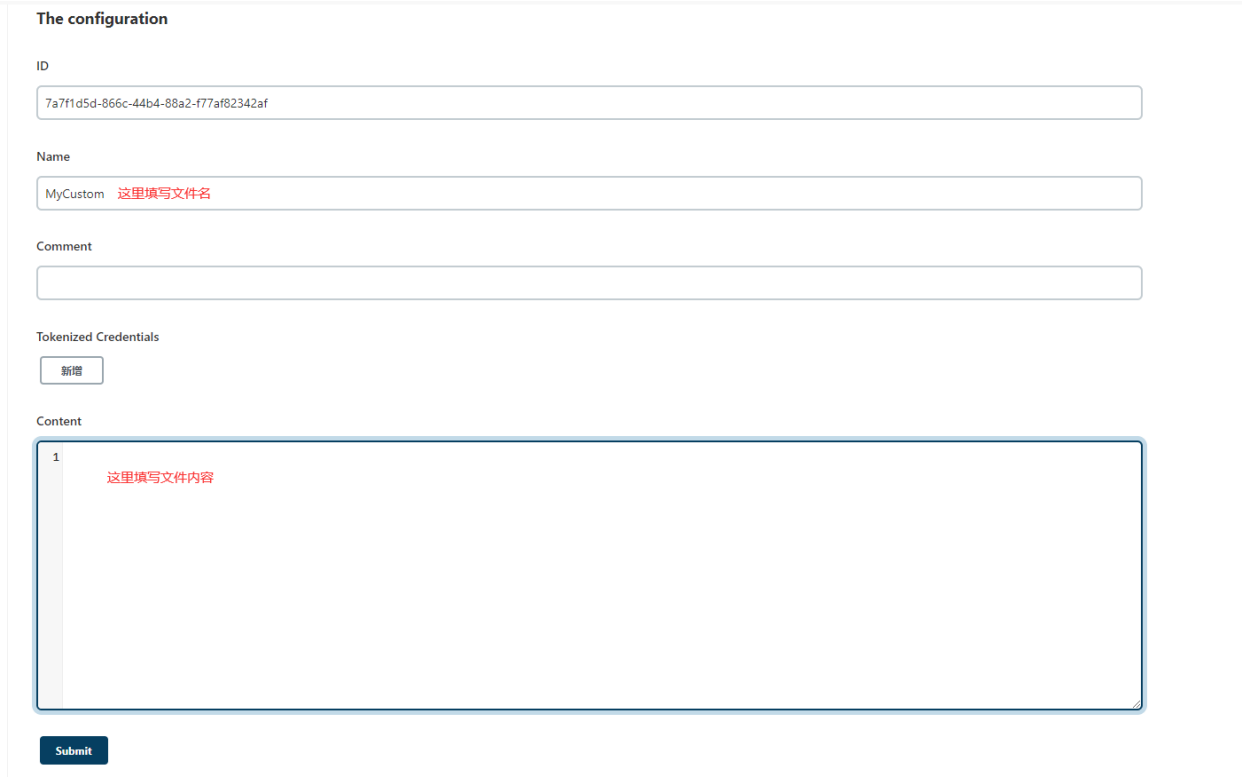


选择Custom file ==> next



填写文件名和文件内容

填写配置文件的名字（这个时候的名字可以跟实际应用时的配置文件名不一致，jenkins调用配置文件覆盖源代码的时候，还会进行更名），配置文件的内容，提交保存



示例1：

Name：manstro-common-bootstrap-basedev.xml

Content：

```
# Spring
spring:
 cloud:
 nacos:
 discovery:
 # 服务注册地址
 server-addr: 192.168.16.5:8848
 # 命名空间 若使用的命名空间为默认的public 则不需要配置
 # namespace: public
```

```
  #群组
  group: DEFAULT_GROUP
  config:
  # 配置中心地址
  server-addr: 192.168.16.5:8848
  # 配置文件格式
  file-extension: yml
  # 命名空间 若使用的命名空间为默认的public 则不需要配置
  # namespace: public
  #群组
  group: DEFAULT_GROUP
  # 共享配置
  shared-configs[0]:
  data-id: application-${spring.profiles.active}.${spring.cloud.nacos.con
fig.file-extension}
  group: DEFAULT_GROUP
  refresh: false
 project:
  # 项目名称
  name: manstro
```

示例2：

Name：manstro-gateway-bootstrap.yml

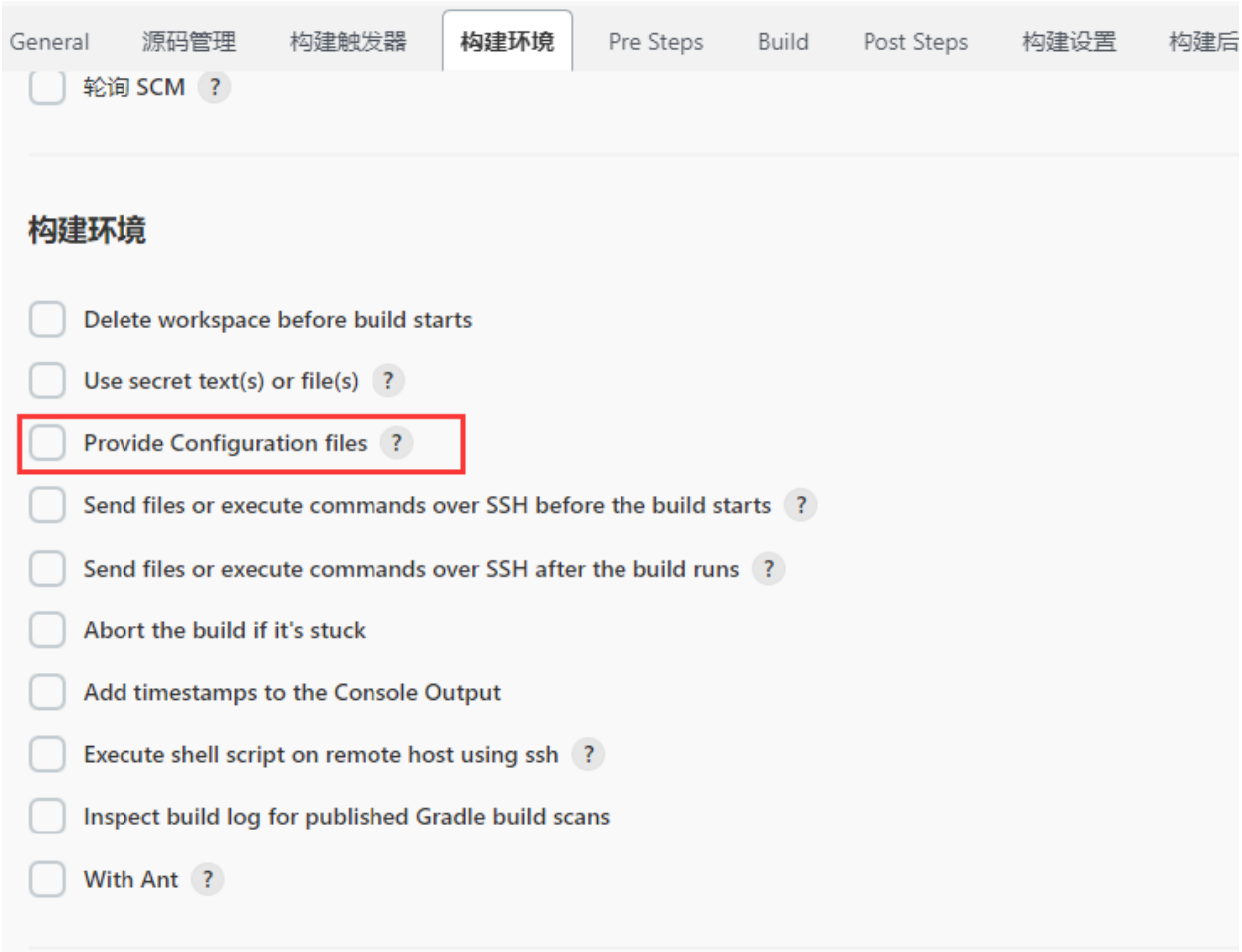
Content：

```
# Tomcat
server:
 port: 8080

# Spring
spring:
 application:
 # 应用名称
 name: dap-gateway
 profiles:
 # 环境配置
 active: @environment@
 include: base@environment@
 main:
 allow-bean-definition-overriding: true
---
```

```yaml
spring:
 profiles: dev
 cloud:
 sentinel:
 # 取消控制台懒加载
 eager: true
 transport:
 # 控制台地址
 dashboard: 192.168.16.5:8718
 # nacos配置持久化
 datasource:
 ds1:
 nacos:
 server-addr: 192.168.16.5:8848
 dataId: sentinel-dap-gateway
 groupId: DEFAULT_GROUP
 data-type: json
 rule-type: flow

---
spring:
 profiles: prod
 cloud:
 sentinel:
 # 取消控制台懒加载
 eager: true
 transport:
 # 控制台地址
 dashboard: 192.168.16.5:8718
 # nacos配置持久化
 datasource:
 ds1:
 nacos:
 server-addr: 192.168.16.5:8848
 dataId: sentinel-dap-gateway
 groupId: DEFAULT_GROUP
 data-type: json
 rule-type: flow
```

解决办法第三步：构建环境

保存成功后，在项目构建配置里构建环境那一栏，勾选provide Configuration files



替换文件：

File：选择新创建的配置文件

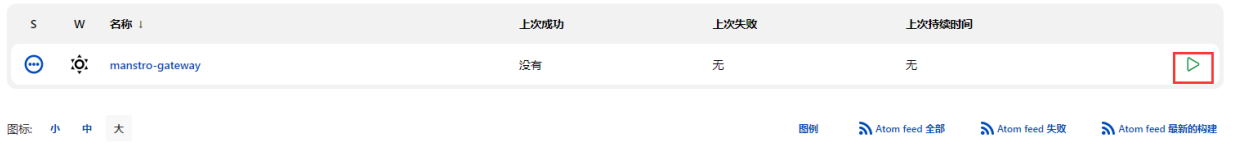
Target：需要替换的 文件目录+文件名+后缀

以下是我需要替换的两个文件路径：

```
manstro-common/manstro-common-base/src/main/resources/bootstrap-basedev.yml
manstro-gateway/src/main/resources/bootstrap.yml
```

6.构建任务测试

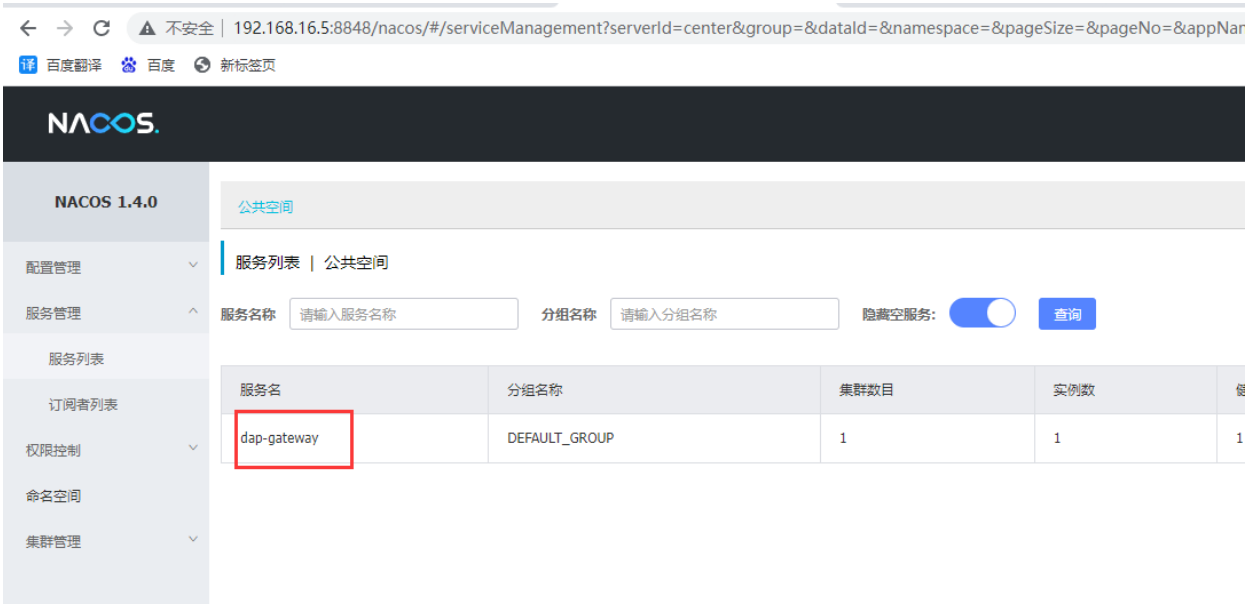
点击按钮构建试试，看看能否顺利



如果构建成功，可以查看容器运行日志

```
docker logs -f manstro-gateway
```

也可以从nacos验证下是否注册成功



7.快速构建其他任务

当构建完成好一个项目，可以快速复制简单修改重要参数就能创建一个新的任务。


输入一个任务名称
manstro-system
» 任务名 ‘manstro-system’ 已存在
构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和任何构建系统来构建你的项目, 甚至可以构建软件以外的系统.
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.
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.
文件夹
创建一个可以嵌套存储的容器。利用它可以进行分组。 视图仅仅是一个过滤器，而文件夹则是一个独立的命名空间， 因此你可以有多个相同名称的的内容，只
如果你想根据一个已经存在的任务创建，可以使用这个选项
复制
manstro-gateway
确定