

jenkins&&docker

3 socroot

```
version: '3'
services:
  jenkins:
    image: harbor.iovhm.com/hub/jenkins/jenkins:2.488-jdk17
    container_name: jenkins
    restart: always
    privileged: true
    user: root
    ports:
      - "8080:8080"
    volumes:
      - ./jenkins_home:/var/jenkins_home
      - ./m2:/root/.m2
      - /var/run/docker.sock:/var/run/docker.sock
      - /usr/bin/docker:/usr/bin/docker
    environment:
      - TZ=Asia/Shanghai
```

docker

> > > Add Credentials

New credentials

类型

Username with password

范围 ?

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

用户名 ?

☐ Treat username as secret ?

密码 ?

ID ?

这个ID很重要, 后面会用到

描述 ?

Create

全局凭据 (unrestricted)

+ Add Credentials

Credentials that should be available irrespective of domain specification to requirements matching.

ID	名称	类型	描述
10b8bc82-8b90-4fbb-9f55-d7e3126b35c0		Username with password	gitlab
docker-huaweicloud		Username with password	docker registry huaweicloud

图标 小 中 大

springboot

sh(''),sh

- sh
- sh(\$){var})

```
pipeline {
  agent any

  //
  environment {
    // docker
    PROJECT_NAME      ="gzxfzd"
```

```

// docker仓库
PROJECT_NS      ="vp-park"
// docker仓库
DOCKER_REGISTRY ="swr.cn-south-1.myhuaweicloud.com"
// 仓库地址docker仓库
// swr.cn-south-1.myhuaweicloud.com/vp-whdev/springboot-demo

}

// 仓库
parameters {

    // 仓库
    choice( name: "BRANCH", choices: [ "development", "release"], description: "仓库")
    // 仓库地址docker仓库
    string( name: "VERSION", defaultValue: "v1", description: "版本")
    // 仓库TAG地址docker仓库
    choice( name: "IMAGE_TAG", choices: [ "test", "prod"], description: "仓库tag")
    // 仓库地址
    // swr.cn-south-1.myhuaweicloud.com/vp-whdev/springboot-demo: v1-latest
    choice(
        name: ' PUSH_SERVICE' ,
        choices: [ ' ALL' , ' jfast-service/jfast-auth' , ' jfast-service/jfast-basic' , ' jfast-
service/jfast-cloud-docking' , ' jfast-devops-center' , ' jfast-gateway' ,
        ' jfast-oa-services' , ' jfast-service/jfast-research-center' ],
        description: ' 仓库地址'
    )
}

tools {
    // 仓库地址mvn仓库
    maven "M3"
    // JDK仓库jdk仓库
    jdk "JDK-8"
}

stages {
    stage('Build') {
        steps {

```

```
// 设置credentialsId为git仓库ID
// 设置url为git仓库地址
git branch: '${BRANCH}', credentialsId: 'donglietao-vpclub-sz-git', url:
'https://g.vpclub.cn/gzxfzhdw/back.git'
```

```
// 设置仓库名称
// 设置pom.xml文件所在目录
// 执行mvn clean compile package -f ./source/pom.xml 编译pom文件
// 设置pom文件所在目录
```

```
dir("./"){
    // 设置仓库名称
    // 设置仓库地址
    // 设置仓库ID
    sh """

    pwd
    java -version
    mvn clean compile package

    """
}
}
```

```
stage("docker"){
    steps{
        script{
            def SERVICE_MAP=[
                "jfast-service/jfast-auth":"jfast-auth",
                "jfast-service/jfast-basic":"jfast-basic",
                "jfast-service/jfast-cloud-docking":"jfast-cloud-docking",
                "jfast-devops-center":"jfast-devops-center",
                "jfast-gateway":"jfast-gateway",
                "jfast-oa-services":"jfast-oa-services",
                "jfast-service/jfast-research-center":"jfast-research-center"
            ]

            // 设置docker仓库ID为docker仓库名称
```

```

        withCredentials([usernamePassword(credentialsId: 'docker-huaweicloud',
passwordVariable: 'docker_password', usernameVariable: 'docker_username')]) {
            def conditionVal= "${PUSH_SERVICE}"
            if(conditionVal=="ALL"){
                for(item in SERVICE_MAP.entrySet()){
                    def serviceName = "${item.value}"
                    def serverPath = "${item.key}"
                    dir("./${serverPath}") {
                        // 目录名称
                        // 目录名称
                        // 目录名称
                        sh """

                        pwd
                        echo ${serviceName}
                        docker login -u ${docker_username} -p ${docker_password}

                        ${DOCKER_REGISTRY}

                        docker build -t

                        ${DOCKER_REGISTRY}/${PROJECT_NS}/${PROJECT_NAME}/${serviceName}:${VERSION}-${IMAGE_TAG} .
                        docker push

                        ${DOCKER_REGISTRY}/${PROJECT_NS}/${PROJECT_NAME}/${serviceName}:${VERSION}-${IMAGE_TAG}

                        """
                    }
                }
            }
            }else{
                def serviceName =SERVICE_MAP[ conditionVal]
                def serverPath = conditionVal
                dir("./${serverPath}") {
                    // 目录名称
                    // 目录名称
                    // 目录名称
                    sh """

                    pwd
                    echo ${serviceName}
                    docker login -u ${docker_username} -p ${docker_password}

                    ${DOCKER_REGISTRY}

                    docker build -t

                    ${DOCKER_REGISTRY}/${PROJECT_NS}/${PROJECT_NAME}/${serviceName}:${VERSION}-${IMAGE_TAG} .
                    docker push

                    ${DOCKER_REGISTRY}/${PROJECT_NS}/${PROJECT_NAME}/${serviceName}:${VERSION}-${IMAGE_TAG}

```

```

pipeline {
    agent any

    // 部署环境
    environment {
        // 部署环境docker环境
        PROJECT_NAME      ="springboot-demo"
        // docker环境
        PROJECT_NS         ="vp-whdev"
        // docker环境
        DOCKER_REGISTRY    ="swr.cn-south-1.myhuaweicloud.com"
        // 部署环境docker环境
        // swr.cn-south-1.myhuaweicloud.com/vp-whdev/springboot-demo

    }

    // 部署环境
    parameters {

        // 部署环境
        choice(name: "BRANCH", choices: ["development", "release"], description: "部署环境")
        // 部署环境docker环境
        string(name: "VERSION", defaultValue: "v1", description: "部署环境")
    }
}

```

```
pipeline {
    agent any

    // 部署环境
    environment {
        // 部署环境docker环境
        PROJECT_NAME      ="springboot-demo"
        // docker环境
        PROJECT_NS        ="vp-whdev"
        // docker环境
        DOCKER_REGISTRY   ="swr.cn-south-1.myhuaweicloud.com"
        // 部署环境docker环境
        // swr.cn-south-1.myhuaweicloud.com/vp-whdev/springboot-demo
    }

    // 部署环境
    parameters {

        // 部署环境
        choice(name: "BRANCH", choices: ["development", "release"], description: "部署环境")
        // 部署环境docker环境
        string(name: "VERSION", defaultValue: "v1", description: "版本")
    }
}
```

```

// 设置TAG为docker
choice(name: "IMAGE_TAG", choices: ["test", "prod"], description: "设置tag")
// 设置仓库地址
// swr.cn-south-1.myhuaweicloud.com/vp-whdev/springboot-demo:v1-latest
}

tools {

    // 设置maven
    maven "M3"
    // 设置jdk
    // jdk "JDK-8"
}

stages {
    stage('Build') {
        steps {

            // Get some code from a gitlab
            // 设置credentialsId为git
            // 设置url, 设置git
            git branch: '${BRANCH}', credentialsId: 'donglietao-vpclub-sz-git', url:
' https://git.whdev.vpclub.cn:10081/park-weihai/backend-api.git'

            // Run Maven on a Unix agent.
            // 设置
            // 设置pom.xml的dir
            // mvn clean compile package -f ./source/pom.xml
            // 设置pom的dir

            dir("./renren-cloud-tenant"){
                sh """

                echo \$(pwd)
                mvn clean compile package

                """
            }
        }
    }
}

```

```

stage("docker"){
    steps{
        // docker build [redacted]Dockerfile
        // [redacted]Dockerfile[redacted]dir [redacted]
        // [redacted]Dockerfile[redacted]dir [redacted]
        dir('./renren-cloud-tenant/park-admin/park-admin-server/') {

            // [redacted]docker[redacted]credentialsId[redacted]docker [redacted]
            withCredentials([usernamePassword(credentialsId: 'docker-huaweicloud',
passwordVariable: 'docker_password', usernameVariable: 'docker_username')]) {

                sh """

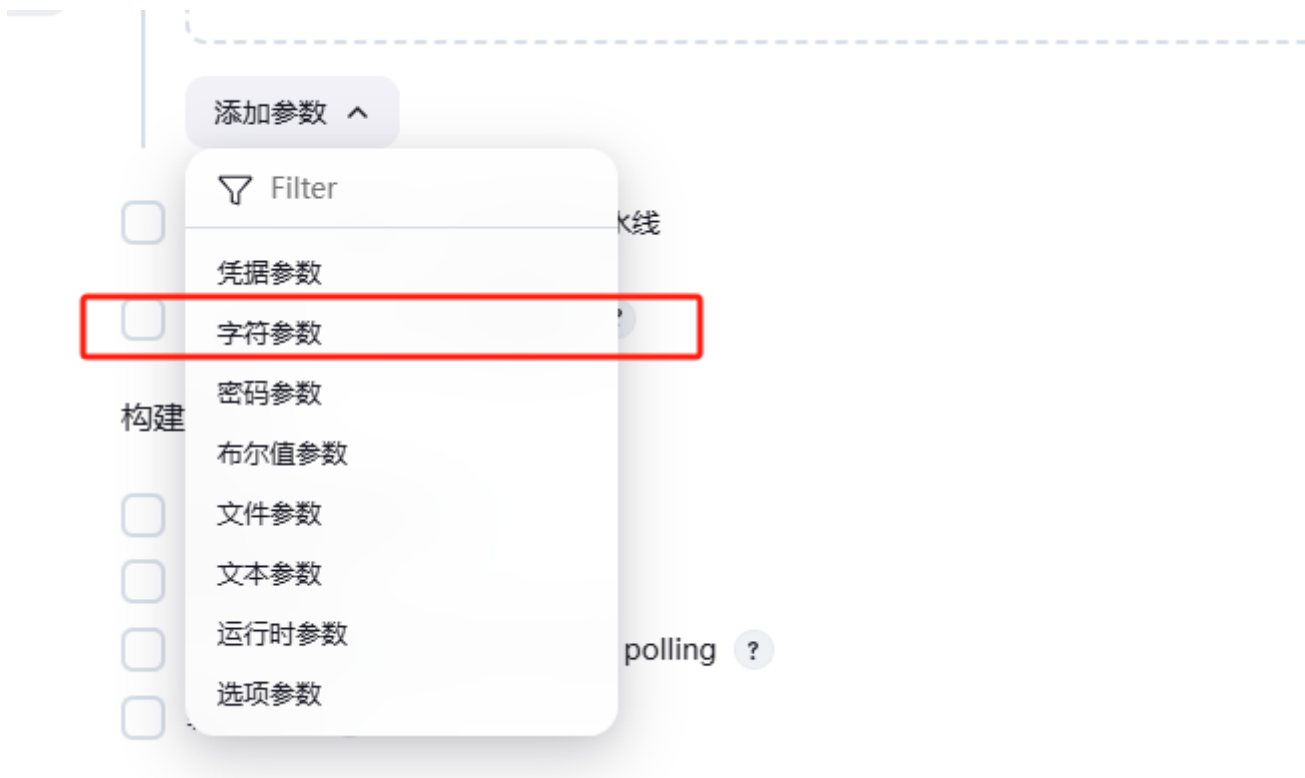
                echo \$(pwd)
                docker login -u ${docker_username} -p ${docker_password}
                ${DOCKER_REGISTRY}
                docker build -t
                ${DOCKER_REGISTRY}/${PROJECT_NS}/${PROJECT_NAME}:${VERSION}-${IMAGE_TAG} .
                docker push
                ${DOCKER_REGISTRY}/${PROJECT_NS}/${PROJECT_NAME}:${VERSION}-${IMAGE_TAG}

                """
            }
        }
    }
}

```

[redacted]

[redacted]Pipeline[redacted]



☒ 参数化构建过程 ? 勾选参数化构建复选框

≡ 字符参数 ?

名称 ?

BRANCH 填写与构建参数对应的环境变量

默认值 ?

dev 填入默认值

描述 ?

代码分支 填入参数描述

纯文本 预览

☐ 清除空白字符 ? 自动删除前后的空格

nodejs Pipeline

```
pipeline {
  agent any

  // 构建环境
  environment {
    // GIT
  }
}
```

```

GIT_URL          = "https://g.vpclub.cn/park/sdyk/front-web-admin.git"
// GIT 配置
GIT_IDENTITY     = "donglietao-vpclub-sz-git"
// 配置docker
PROJECT_NAME     ="nodejs-demo"
// docker 配置
PROJECT_NS       ="vp-whdev"
// docker 配置
DOCKER_REGISTRY ="swr.cn-south-1.myhuaweicloud.com"
// 配置docker
// swr.cn-south-1.myhuaweicloud.com/vp-whdev/nodejs-demo
// DOCKER 配置
DOCKER_IDENTITY = "docker-huaweicloud"
// nodejs配置
NODE_OPTIONS     = "--max-old-space-size=4096"
}

// 配置
parameters {
    // 配置
    choice(name: "BRANCH", choices: ["development", "release"], description: "配置")
    // 配置docker
    string(name: "VERSION", defaultValue: "v1", description: "配置")
    // TAG配置docker
    choice(name: "IMAGE_TAG", choices: ["test", "prod"], description: "配置tag")
    // 配置
    // swr.cn-south-1.myhuaweicloud.com/vp-whdev/nodejs-demo: v1-latest
}

stages {
    stage('Build') {
        steps {
            // Get some code from a gitlab
            // 配置credentialsId配置git
            // 配置url, 配置git
            git branch: "${BRANCH}", credentialsId: "${GIT_IDENTITY}", url: "${GIT_URL}"

            // build

```

```

// [REDACTED]nodejs[REDACTED]nodejs[REDACTED]
nodejs(' node- 18.20' ) {
    sh """

    echo \$(pwd)
    node -v
    npm -v
    npm install --registry=https://registry.npmirror.com
    npm run build:test

    """
}
}
}

stage("docker"){
    steps{

        // docker build [REDACTED]Dockerfile
        // [REDACTED]Dockerfile[REDACTED]dir [REDACTED]
        // [REDACTED]Dockerfile[REDACTED]dir [REDACTED]

        withCredentials([usernamePassword( credentialsId: "${DOCKER_IDENTITY}",
passwordVariable: ' docker_password', usernameVariable: ' docker_username' )]) {

            sh """

            echo \$(pwd)
            docker login -u ${docker_username} -p ${docker_password}
            ${DOCKER_REGISTRY}

            docker build -t
            ${DOCKER_REGISTRY}/${PROJECT_NS}/${PROJECT_NAME}:${VERSION}-${IMAGE_TAG} .
            docker push ${DOCKER_REGISTRY}/${PROJECT_NS}/${PROJECT_NAME}:${VERSION}-${
            IMAGE_TAG}

            """

        }

    }
}

```

```
}  
}  
}
```

nodejs

gyp sass jenkins docker

```
FROM harbor.iovhm.com/hub/node: 14. 21. 3 AS build  
WORKDIR /data  
COPY . / /data  
RUN cd /data && \  
    yarn install --registry=https: //registry.npmmirror.com && \  
    yarn run build:prod && \  
    true
```

```
FROM harbor.iovhm.com/hub/nginx: 1. 14. 2  
COPY --from=build /data/dist/ /usr/share/nginx/html/management
```

- AS build
- --from=build

docker