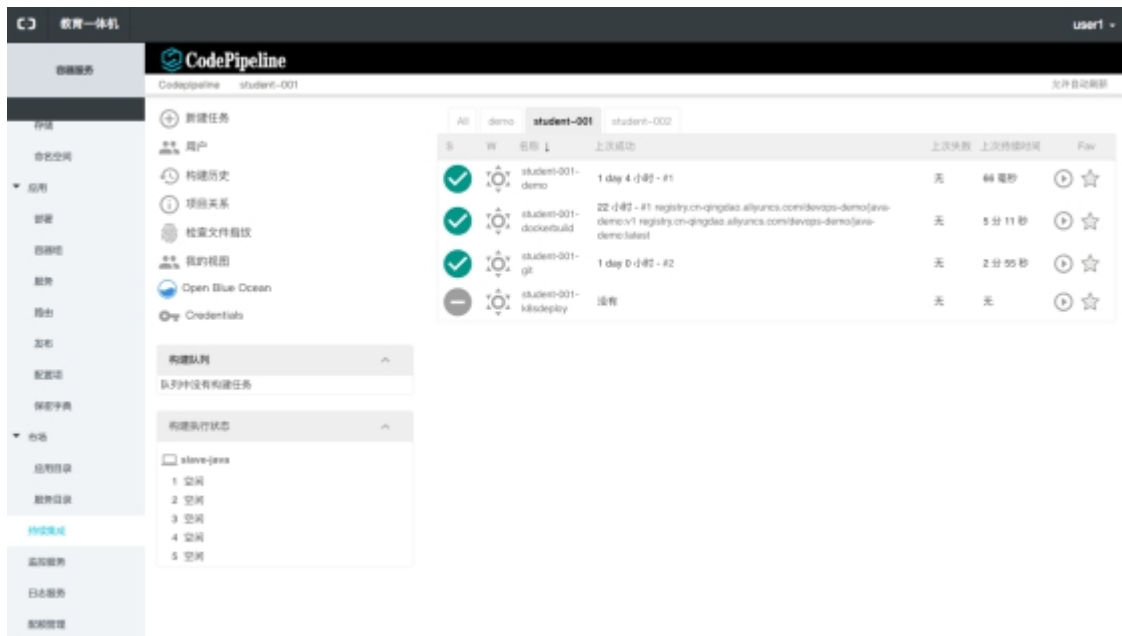


实验课程7-4 使用CodePipeline部署应用到容器集群

本节课程会演示如何使用CodePipeline自动拉取docker镜像并部署应用到容器集群中。

1、新建名为student-001-k8sdeploy的项目



2、在java-demo仓库中新增deployment.yaml文件

```
$ cd java-demo/ $ tree |— deployment.yaml |— Dockerfile |— pom.xml |— README.md |— src |— main |— webapp |— index.jsp |— WEB-INF |— web.xml
```

在当前目录下新建deployment.yaml文件，代码如下：

```
apiVersion: extensions/v1beta1 # API版本
kind: Deployment # 资源类型, Deployment
metadata: # 元数据
  labels: # 元数据, 标签列表
    name: java-demo-codepipeline # 元数据, deployment的标签名称
    name: java-demo-codepipeline # 元数据, deployment的名字
    namespace: student-001 # 元数据, deployment的命名空间
spec: # deployment中容器模板的详细定义
  replicas: 2 # pod的副本数
  template:
    metadata:
      labels:
        app: java-demo-codepipeline
    spec:
      containers:
        - name: java-demo-codepipeline # 容器名称
          image: registry.cn-qingdao.aliyuncs.com/devops-demo/java-demo:v1 # 容器使用的镜像
          imagePullPolicy: Always # 获取镜像的策略
          ports:
```

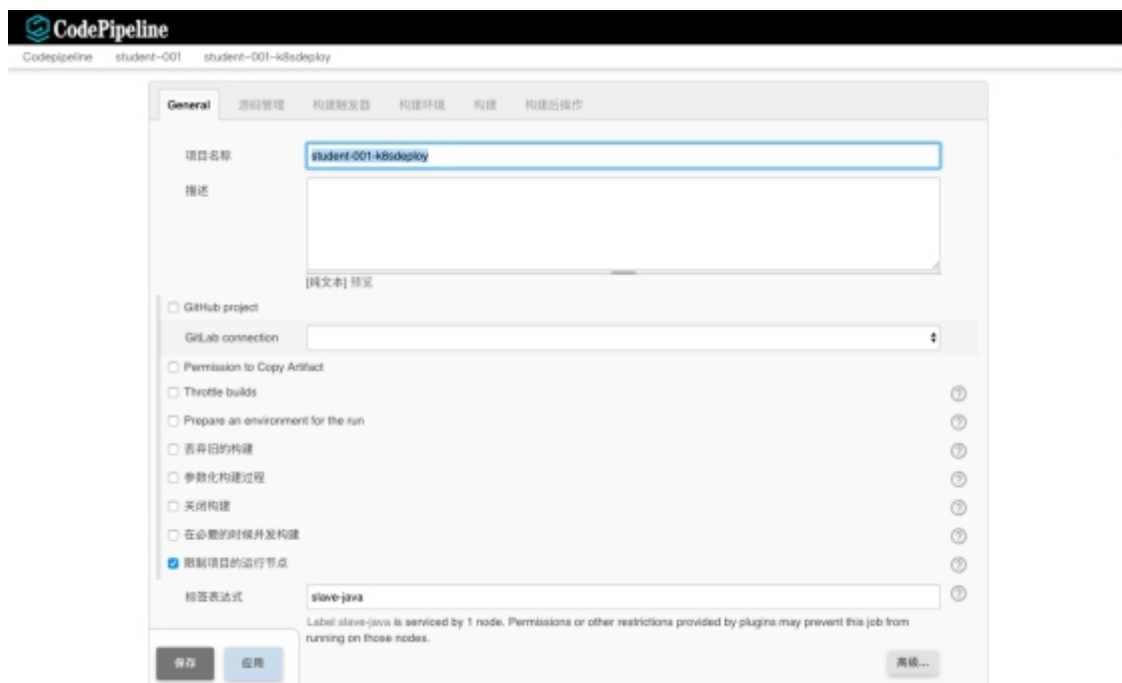
```
- containerPort: 8080 # 容器要暴露的端口
imagePullSecrets:
- name: regsecret
```

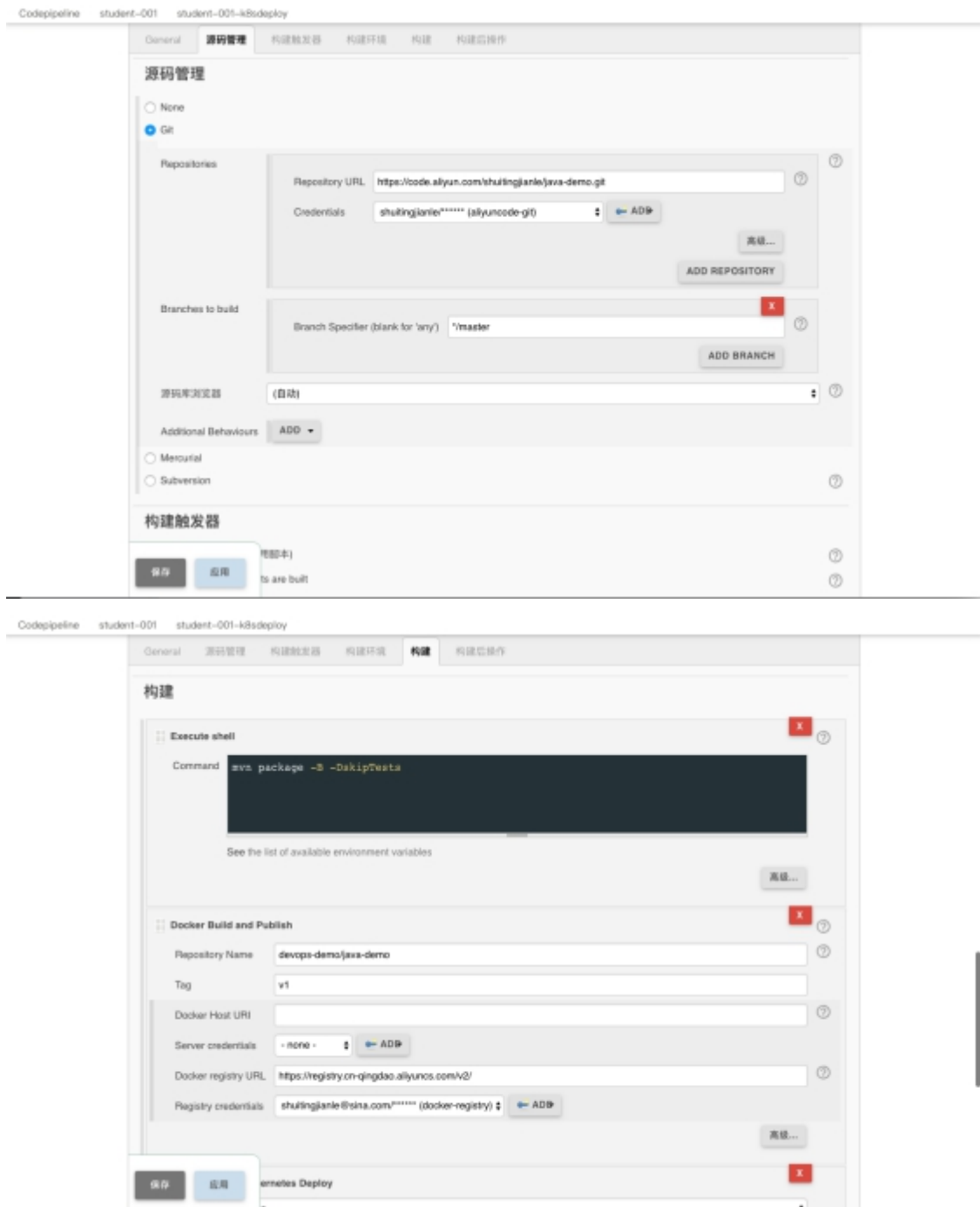
```
---
apiVersion: v1 # API版本
kind: Service # 资源类型, service
metadata: # 元数据
  labels: # 元数据, 标签列表
    name: java-demo-codepipeline # 元数据, service的标签名称
    name: java-demo-codepipeline # 元数据, service的名字
  namespace: student-001 # 元数据, service的命名空间
spec:
  ports:
  - port: 80 # 提供给容器内部应用访问的端口号
    targetPort: 8080 # pod上应用监听的端口
    name: java-demo-codepipeline
  selector:
    app: java-demo-codepipeline # 应用选择
  type: NodePort # 向外部用户暴露端口的方式
```

将修改后的源码提交并推送到远程仓库。

3、配置构建节点、源码管理、构建命令、docker镜像构建

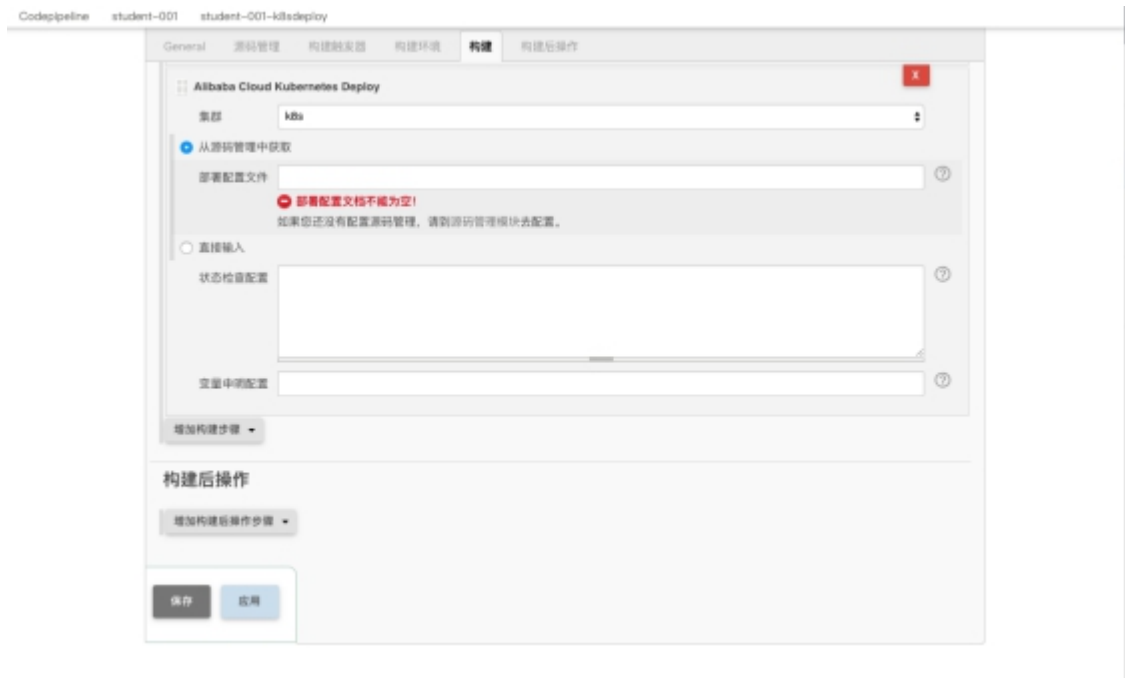
这部分与项目student-001-dockerbuild的配置相同





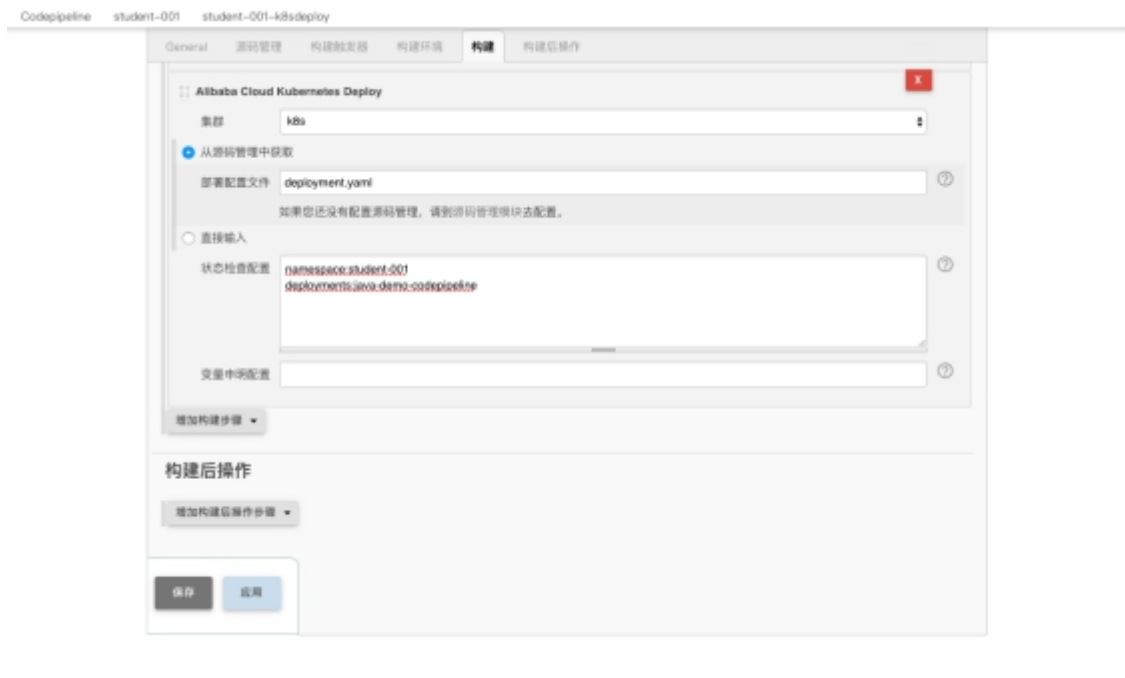
4、配置部署应用到容器集群

在 构建 区域，增加构建步骤 选择 Alibaba Cloud Kubernetes Deploy并配置：



- (1) 集群 -> 下拉列表中选择要部署应用的Kubernetes集群
- (2) 从源码管理中获取 -> 填写deployment.yaml的路径 或者选择 直接输入 -> 可以复制deployment.yaml文件的内容直接输入
- (3) 状态检查配置 -> 检查要部署的资源是否真正Available
- (4) 变量声明配置 -> 可以替换部署模板中的变量

配置完成后如下:



保存。

5、构建并查看日志

```
Codepipeline student-001 student-001-k8sdeploy #! registry.cn-qingdao.aliyuncs.com/devops-demo/java-demo:v1 registry.cn-qingdao.aliyuncs.com/devops-demo/java-demo:latest

Start status check.
#Change namespace to: student-001
##Exec status check for: java-demo-codepipeline
kubectl get deployments --kubeconfig=/home/jenkins/workspace/student-001-k8sdeploy/.kubeconfig --namespace=student-001
NAME DESIRED CURRENT UP-TO-DATE AVAILABLE AGE
java-demo-codepipeline 2 2 2 0 9s
##Exec status check for: java-demo-codepipeline
kubectl get deployments --kubeconfig=/home/jenkins/workspace/student-001-k8sdeploy/.kubeconfig --namespace=student-001
NAME DESIRED CURRENT UP-TO-DATE AVAILABLE AGE
java-demo-codepipeline 2 2 2 2 30s
=====
addresses:
- address: 192.168.0.250
- address: 1zm5ehafp00thnas8zsib3z
addresses:
- address: 192.168.0.251
- address: 1zm5ehafp00thnas8zsib5z
addresses:
- address: 192.168.0.249
- address: 1zm5ehafp00thnas8zsib6z
addresses:
- address: 192.168.0.10
- address: 1zm5ehafp00thnas8zsib7z
addresses:
- address: 192.168.0.247
- address: 1zm5ehafp00thnas8zsib8z
addresses:
- address: 192.168.0.248
- address: 1zm5ehafp00thnas8zsib9z
Finished: SUCCESS
```

6、查看端口并访问服务

查看端口:

概览

集群

节点

存储

命名空间

应用

部署

容器组

服务

路由

发布

配置项

命名空间

市场

应用目录

服务目录

服务列表

集群 myk8s 命名空间 student-001

| 名称 | 类型 | 创建时间 | 地址IP | 内部端点 | 外部端点 | 操作 |
|------------------------|----------|---------------------|---------------|---|------|-------|
| java-demo-codepipeline | NodePort | 2018-11-01 10:41:27 | 172.19.15.92 | java-demo-codepipeline:80 TCP java-demo-codepipeline:32066 TCP | - | 更新 删除 |
| java-demo-service | NodePort | 2018-11-01 10:51:59 | 172.19.13.90 | java-demo-service:80 TCP java-demo-service:37536 TCP | - | 更新 删除 |
| java-demo-v1 | NodePort | 2018-11-01 14:57:52 | 172.19.10.47 | java-demo-v1:80 TCP java-demo-v1:35447 TCP | - | 更新 删除 |
| java-demo-v2 | NodePort | 2018-11-01 15:38:52 | 172.19.11.214 | java-demo-v2:80 TCP java-demo-v2:39989 TCP | - | 更新 删除 |

访问服务:

```
⏪ ⏩ 🔍 不安全 | console.myk8s.aliyuncs.local:32066/demo/ ☆ ❏

Hello CodePipeline!
```