云效

快速入门

快速入门

云效快速入门

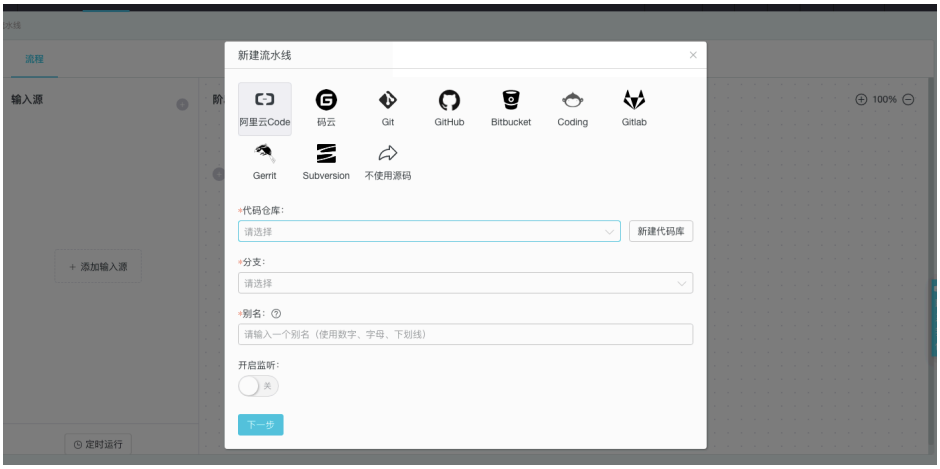
创建企业

如果你还没有加入任何企业，系统会引导你先创建一个企业，点击这里查看详情。

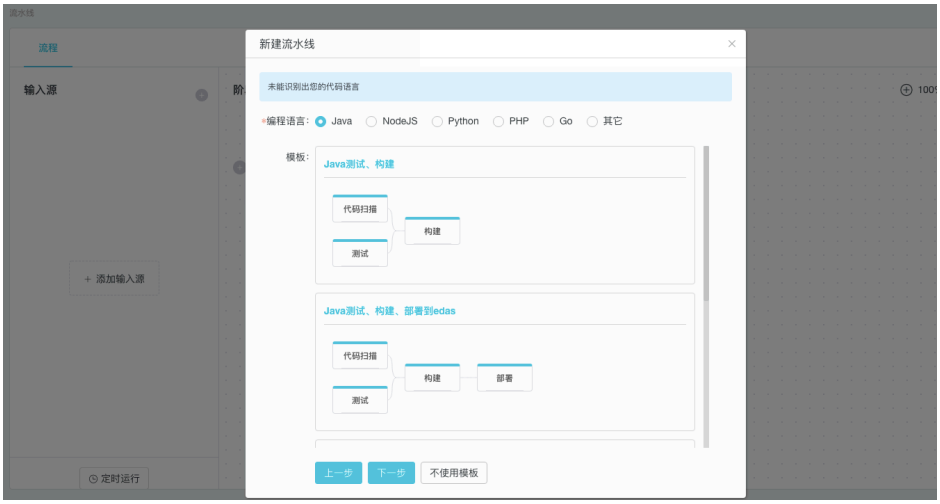
开始持续交付

在吊顶栏，点击“首页”菜单。在页面中，点击“开始持续交付”，向导会引导你快速创建流水线体验从开发到交付：

新建流水线时，选择配置当前流水线使用的代码源：



输入代码源后，选择相应的流水线模版即可快速开始持续交付过程：



点击演示环境的地址，可以看到实际效果：

开始项目管理

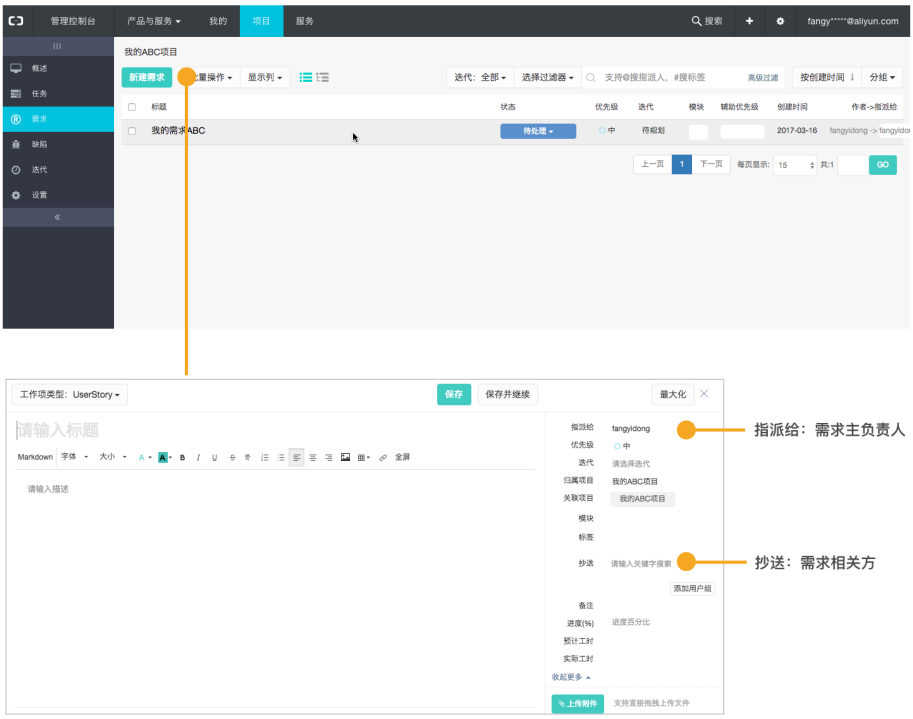
在吊顶栏，点击“首页”菜单。在页面中，点击“开始项目管理”，向导会引导你开始Devops项目管理：

录入需求

录入需求是比较简便的操作，只要点项目左边“需求”TAB，就可以发现新建需求的按钮，点一下按钮，然后填上需求的标题和正文，再点“保存”即可。

提示：

- 需求正文可直接按CTRL+V进行贴图
- 需求正文支持markdown方式编辑



创建迭代

迭代一般由ScrumMaster来创建和管理。ScrumMaster主要职责制定最佳工作模式，协调团队开发和跟进解决blocker，并保护团队避免受到外部干扰。

在项目里点左侧“迭代”TAB可创建迭代：

规划迭代内容

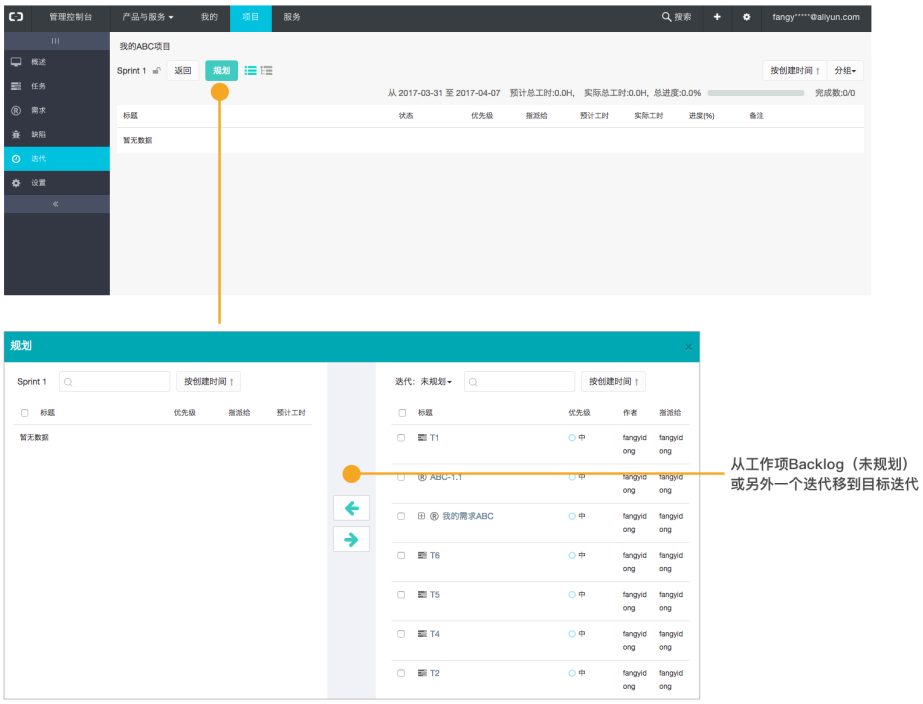
每个迭代具体要排期哪些内容，Product Owner定优先级，研发团队根据需求估算、团队速率和可承受并发度等确定能做多少内容。

在RDC里面，把工作项（需求、任务、缺陷）规划进迭代有3种方式：

在工作项详情页，找到“迭代”字段，选择目标迭代

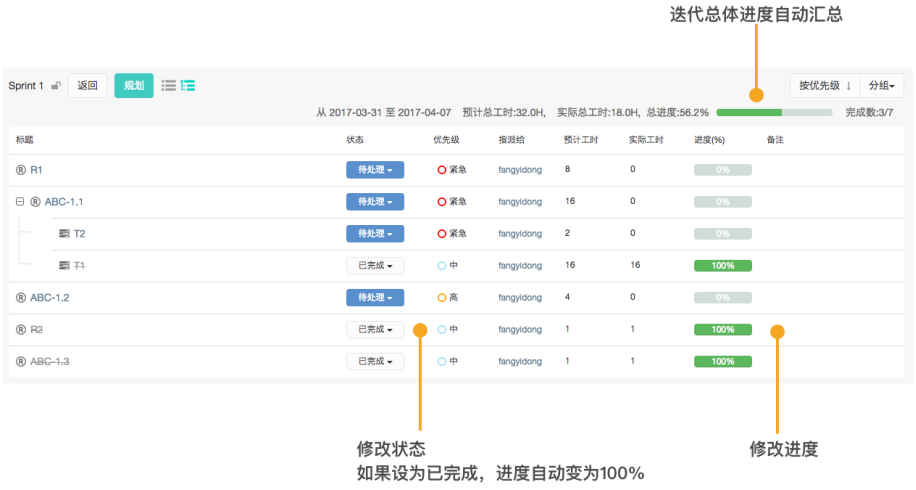
在工作项列表页，直接在迭代列点击选中目标迭代

在迭代里面，点“规划”按钮，可批量把工作项拉入迭代



迭代进度跟进

研发负责的工作项完成后，把状态设为已完成，进度自动更新为100%，迭代总体进度会自动进行重新计算:



体验从开发到交付

流水线

从左边栏“流水线”菜单项点击查看流水线。

流水线把从开发到交付的各项工作串起来。自由模式的默认配置的流水线体现的流程是，取master分支的最新点，构建打包。随后，把包部署到日常环境（常用来做集成测试），测试通过后，再部署到预发环境，进而部署到正式环境，也就是交付上线。

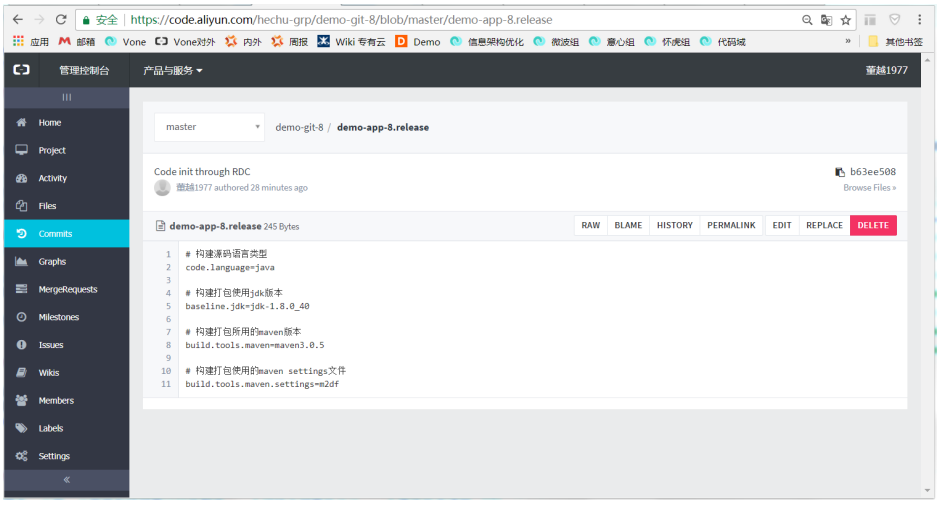
在一站式方案创建过程中，已经把构建和日常环境部署这两步配置好了。可以跑通，并可以看到demo程序运行起来的网页效果。

可以尝试修改Git库的master分支的代码并推送上来。然后再次点击启动流水线，完成日常环境部署，观察demo程序展示内容的变化。

要想投入真实使用场景，需要进一步配置好预发和正式环境。下面我们概要介绍构建和部署的配置方法。

配置构建的方法

如何构建，是由源代码根目录下的一个名为“<应用名>.release”的构建配置文件决定的。这个文件由若干属性名-属性值对儿组成。

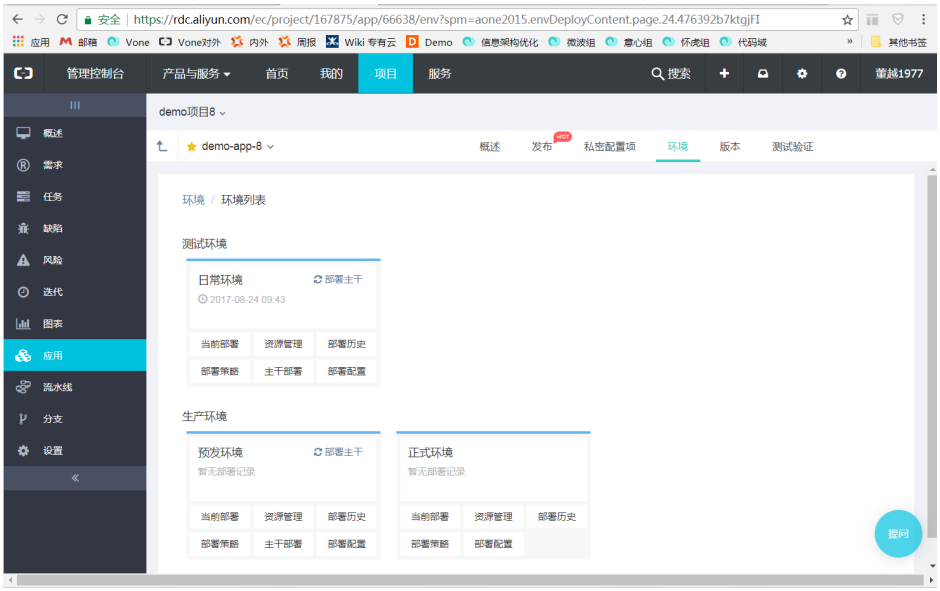


详细介绍见自定义构建配置。

配置部署的方法

先介绍RDC的一个关键概念：应用。应用是指被部署运行的程序，一个可独立部署的单元。一个源代码库通常对应一个应用，该应用被部署到不同的环境中，比如日常环境（通常用来做集成测试）、预发环境、正式环境。

应用运行的每个环境上，要部署到哪些机器，部署用什么脚本等，是配置在应用的这个环境里的。具体路径是，在左边栏点击“应用”菜单项进入应用，然后浏览“环境”这个菜单项下的内容。



需要首先由企业管理员购买阿里云的ECS服务器，并关联到企业的机器池，各应用各环境才能配置为使用这些机器。

关于部署配置的详细介绍：部署配置。

小结

本文介绍了如何通过向导快速搭建一套一站式研发协作环境，包含从提出需求，到代码实现，到构建、部署、测试，并最终发布上线所需的各个工具和功能。随后介绍了如何开始使用这套一站式环境，以及如何进一步配置它，满足企业实际研发交付场景的需求。

本文只是基本功能的简单介绍，更多功能，更多详细内容，欢迎继续阅读。

点这里立即体验

基于Istio的Kubernetes蓝绿发布

基于Istio的Kubernetes蓝绿发布

本文将向读者介绍如何在云效中发布基于Istio的应用程序，同时利用云效提供的蓝绿发布能力更安全的发布和验证应用。

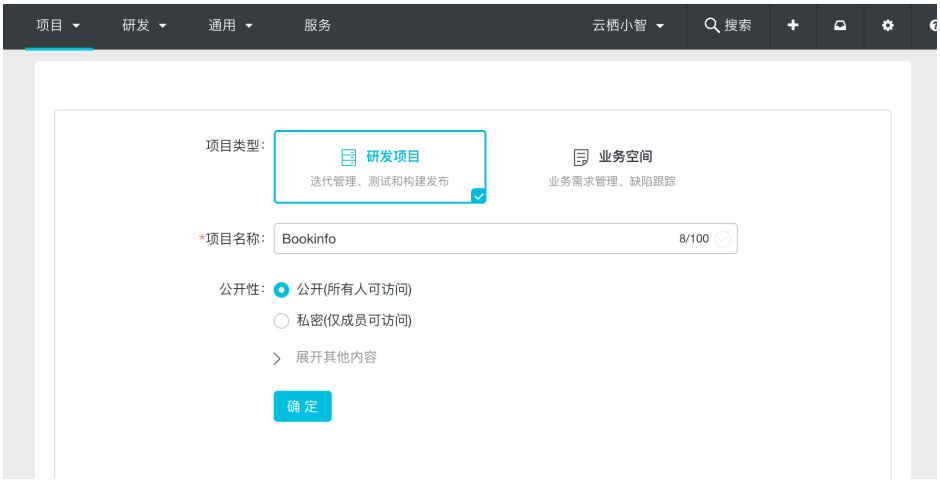
本文采用Istio官方的Bookinfo实例程序，源码地址：https://gitee.com/moo/bookinfo.git

前提条件

- 在阿里云容器服务Kubernetes中创建集群，并导入到当前企业中
- Kuberntes集群安装Istio组件
- 开通阿里云镜像仓库服务用于托管容器镜像

创建项目

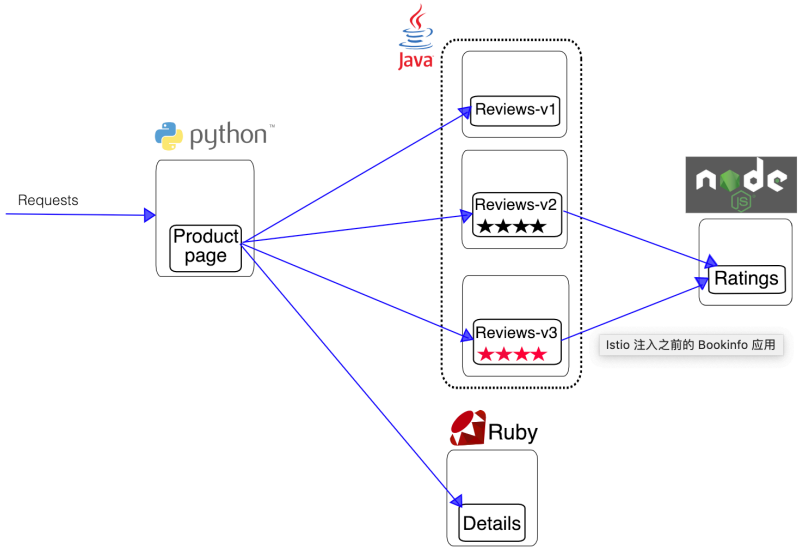
在云效中我们采用**项目**的概念来管理一组相关的应用，并且对项目提供了如敏捷看板，测试，文档等服务。通过项目可以端到端的管理应用的整个交付周期：



这里我们为Bookinfo创建一个独立的项目，创建完成后就可以进入到该项目的主页：

创建应用

Bookinfo应用是一个典型的微服务应用程序，其主要结构如下所示：



Bookinfo示例程序组要由Productpage,Reviews,Details以及Ratings4个部分组成，同时使用了Python,Java，Ruby以及Node四种不同的技术栈。

创建Productpage应用

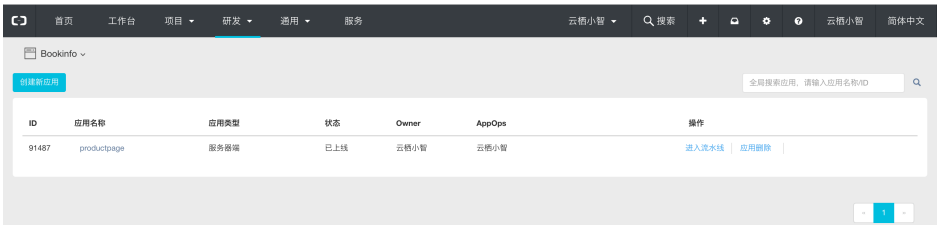
首先，我们在Bookinfo项目下创建应用productpage，如下所示：

由于示例应用源码是托管到码云(Gitee)中，我们需要完成码云授权，并关联已有源码：

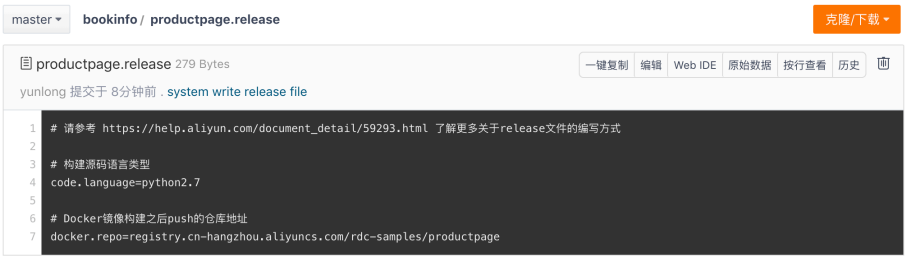
关联已有代码后，我们需要为当前应用选择相应的应用模板，当前Productpage应用采用Python进行开发，并且使用Kubernetes进行部署：

在完成编程语言以及部署选项的设置后，用户需要定义应用时如何构建的，由于需要将应用部署到Kubernetes中，这里我们勾选Docker构建选项，并且定义了当前应用镜像发布的镜像仓库为rdc-samples/productpage。云效会自定在项目中生成productpage.release文件，该文件定义了Productpage是如何构建的：

下一步，预览并创建应用即可。



查看源码，可以看到云效自动创建的release文件：https://gitee.com/moo/bookinfo/blob/master/productpage.release：

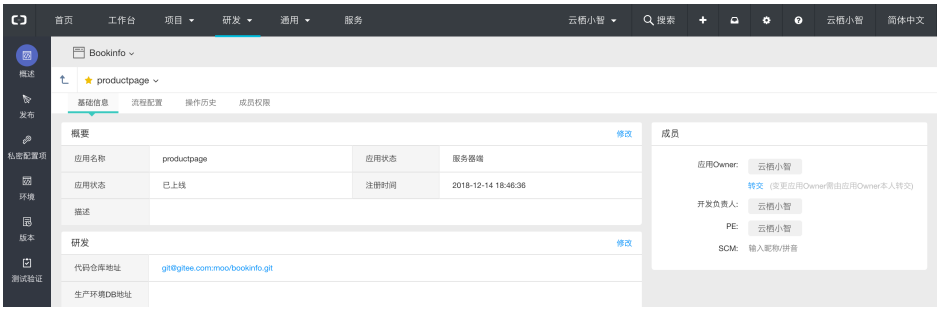


由于Productpage应用是在项目的src/productpage路径下，这里需要手动修改productpage.release文件的内容，并制定Productpage应用的Dockerfile文件路径，完整配置如下所示：

```
# 构建源码语言类型
code.language=python2.7

# Docker镜像构建之后push的仓库地址
docker.file=src/productpage/Dockerfile
docker.repo=registry.cn-hangzhou.aliyuncs.com/rdc-samples/productpage
```

进入到productpage应用，可以看到在云效中应用会包含应用的发布流水线，配置管理，环境管理以及版本等能力：



进入到应用的发布页面可以看到云效为应用自动创建的持续交付流水线。触发流水线构建，在构建阶段云效会根据productpage.release文件定义的内容对项目进行打包以及镜像构建，并且将镜像推送到阿里云镜像仓库服务：

不过由于，我们还未进行日常，预发以及正式环境的部署配置，因此当流水线运行到日常部署阶段，会出现如下错误信息：

根据提示，我们进入到日常环境的部署配置页面，这里我们为日常环境创建了名为dev的命名空间，并且为该命名空间启用了Istio的自动注入能力：

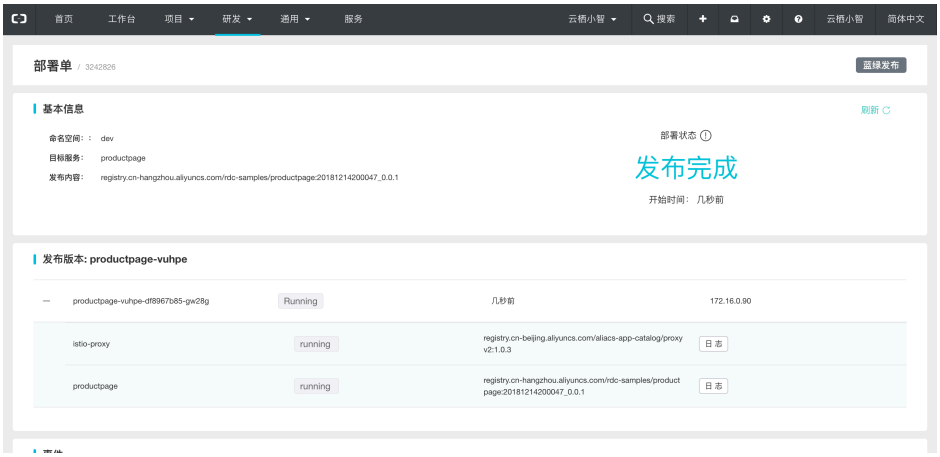
```
kubectl create namespace dev
kubectl label namespace dev istio-injection=enabled
```

选择集群，命名空间并且新建productpage服务：

这里需要指定Productpage服务的端口以及相应的容器端口，在bookinfo中productpage应用监听的是9080端口，完成配置后如下所示，对于启用了Istio支持的集群，云效会自动打开蓝绿部署选项：

完成配置后，回到流水线页面重新触发日常环境部署动作，由于是第一次部署Productpage服务，云效会直接完成服务的初始化动作：

查看发布单，可以查看具体的资源信息，这里云效自动创建了productpage的Pod实例，并且可以看到自动注入的istio-proxy容器：



部署单发布完成后，查看当前集群dev命名空间下的所有资源如下所示：

```
$ kubectl -n dev get all --selector='app=productpage'
NAME READY STATUS RESTARTS AGE
pod/productpage-vuhpe-df8967b85-gw28g 2/2 Running 0 3m

NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
service/productpage ClusterIP 172.19.0.131 <none> 9080/TCP 54m

NAME DESIRED CURRENT UP-TO-DATE AVAILABLE AGE
deployment.apps/productpage-vuhpe 1 1 1 1 3m

NAME DESIRED CURRENT READY AGE
replicaset.apps/productpage-vuhpe-df8967b85 1 1 1 3m
```

云效会为应用版本创建相应的DestinationRules以及VirtualService资源，并且在第一次部署完成后自动将路由规则定向到当前部署版本。

查看DestinationRules如下所示：

```
#$ kubectl -n dev get destinationrules productpage -o yaml
apiVersion: networking.istio.io/v1alpha3
kind: DestinationRule
metadata:
creationTimestamp: 2018-12-14T11:59:07Z
generation: 1
name: productpage
namespace: dev
```

```
resourceVersion: "341256082"
selfLink: /apis/networking.istio.io/v1alpha3/namespaces/dev/destinationrules/productpage
uid: a8fe5fc8-ff97-11e8-bbda-de3c7d18d080
spec:
host: productpage
subsets:
- labels:
version: vuhpe
name: vuhpe
```

查看VirtualService如下所示：

```
#$ kubectl -n dev get virtualservice productpage -o yaml
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
creationTimestamp: 2018-12-14T11:59:07Z
generation: 1
name: productpage
namespace: dev
resourceVersion: "341256093"
selfLink: /apis/networking.istio.io/v1alpha3/namespaces/dev/virtualservices/productpage
uid: a90d8198-ff97-11e8-bbda-de3c7d18d080
spec:
hosts:
- productpage
http:
- route:
- destination:
host: productpage
subset: vuhpe
```

使用Istio Gateway访问应用

为了能够访问Productpage应用，用户需要在dev命名空间下部署Gateway资源如下所示：

```
apiVersion: networking.istio.io/v1alpha3
kind: Gateway
metadata:
name: bookinfo-gateway
namespace: dev
spec:
selector:
istio: ingressgateway
servers:
- hosts:
- '*'
port:
name: http
number: 80
protocol: HTTP
```

将以上内容保存到bookinfo-gateway.yaml中，并通过Kubectl在集群中创建：

```
$ kubectl -n dev create -f bookinfo-gateway.yaml
gateway.networking.istio.io/bookinfo-gateway created
```

创建完Gateway之后，需要手动绑定Productpage服务和Gateway绑定：

```
$ kubectl -n dev edit virtualservice productpage
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
creationTimestamp: 2018-12-14T11:59:07Z
generation: 1
name: productpage
namespace: dev
resourceVersion: "341256093"
selfLink: /apis/networking.istio.io/v1alpha3/namespaces/dev/virtualservices/productpage
uid: a90d8198-ff97-11e8-bbda-de3c7d18d080
spec:
# 添加gateways节点绑定bookinfo-gateway
gateways:
- bookinfo-gateway
hosts:
# 添加需要监听的域名
- productpage.yunxiao.com
- productpage
http:
- route:
- destination:
host: productpage
subset: vuhpe
```

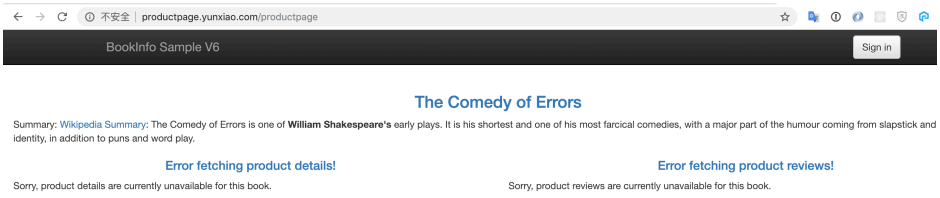
在完成VirtualService完成Gateway绑定后，用户就可以通过集群的istio Gateway访问应用，可以通过以下命令获取Istio Gateway的外网地址：

```
$ export INGRESS_HOST=$(kubectl -n istio-system get service istio-ingressgateway -o
jsonpath='{.status.loadBalancer.ingress[0].ip}')
$ export INGRESS_PORT=$(kubectl -n istio-system get service istio-ingressgateway -o
jsonpath='{.spec.ports[?(@.name=="http")].port}')
```

此时只需要在本地添加DNS设置，即可通过Gateway访问ProductPage应用：

```
182.92.244.178 productpage.yunxiao.com
```

打开浏览器访问http://productpage.yunxiao.com/productpage，如下所示：

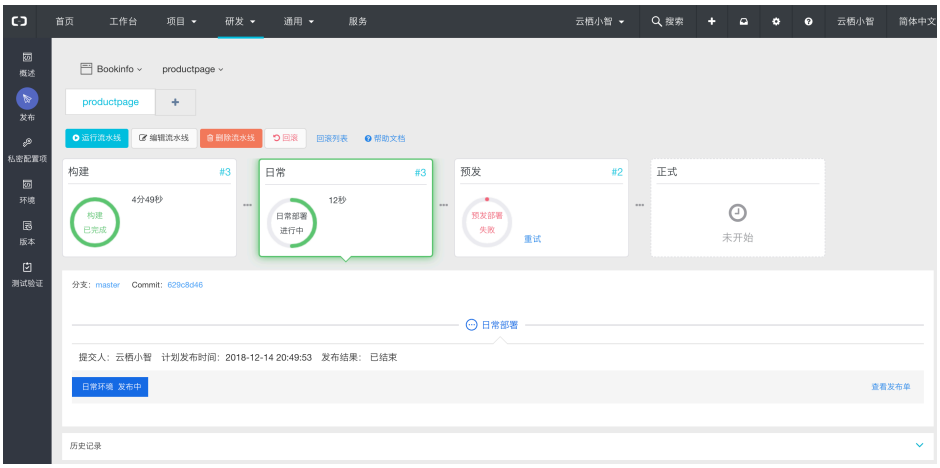


不过由于目前我们只部署了Productpage应用，因此当前页面会提示“Error fetching product details!”和“Error fetching product reviews!”的错误信息。

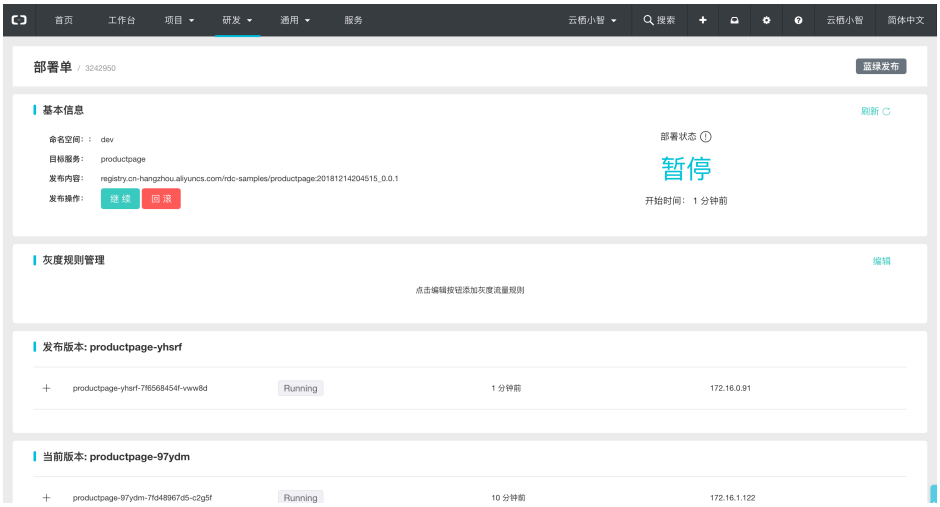
使用蓝绿发布

在Productpage应用第一次发布时由于集群中并不存在任何资源，因此云效会直接创建资源并且将部署设置为成功。在第二次部署应用时，由于底层资源已存在，再次运行流水线，会触发正式的蓝绿部署流程。修改productpage.html文件如下所示：

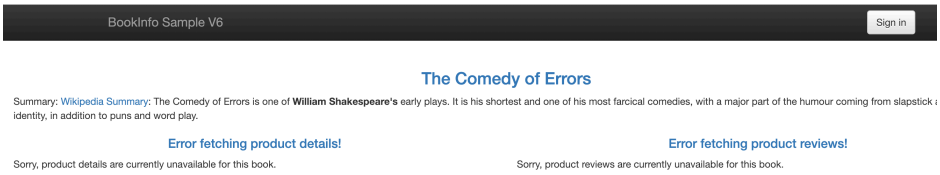
并重新触发流水线，此时当进入到部署阶段后部署任务会进入等待状态：



点击查看发布单按钮，进入发布单页面，可以看到云效为当前服务创建了两个Deploymet版本：



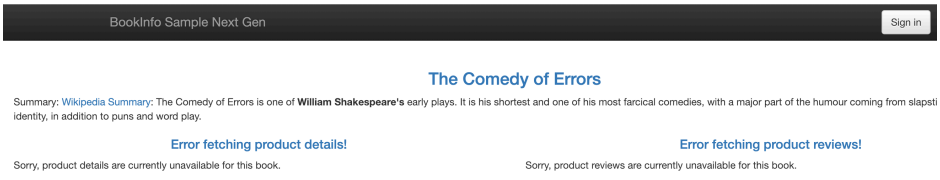
在默认情况下，当前应用的所有流量依然全部指向旧版的应用实例，访http://productpage.yunxiao.com/productpage可以验证当前应用的流量情况:



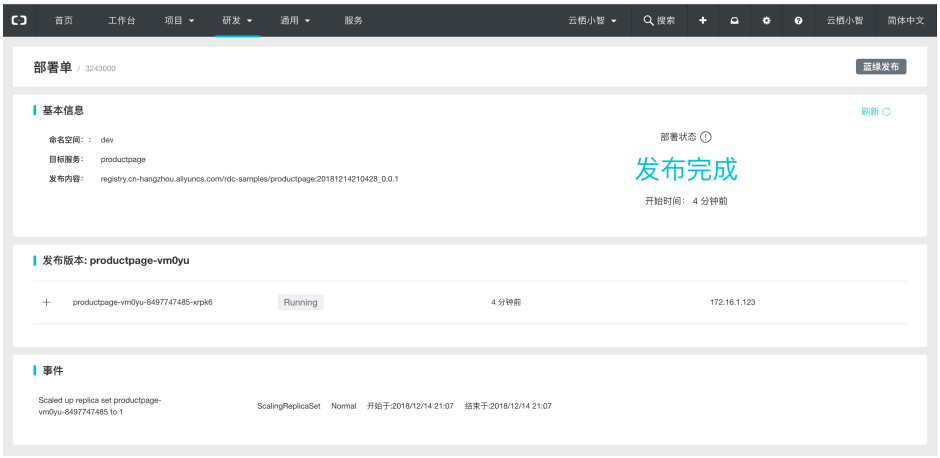
在部署暂停时如果希望某些流量能够进入到新版应用，例如，某些特定的URL或者是HTTP Header中包含特定值的请求。那可以直接在发布单中编辑灰度规则即可。例如，我们希望所有/productpage的请求都直接进入新版，那如下所示，添加一条灰度规则即可：

预览生成的YAML并下发规则：

在灰度规则更新成功后，此时如果再次访问应用，那流量则会进入到新版的ProductPage中：



在确认新版应用能够按照预期工作后，在发布单点击继续按钮，即可完成本次发布过程。发布完成后云效会自动移除旧版应用实例，并且将所有流量规则指向新版本。



小结

在这部分中，我们完成了对bookinfo中productpage应用的发布，通过使用云效可以让用户几乎0成本的将应用接入到Istio模式，并且使用蓝绿部署模式安全的对软件进行升级和发布。

Java入门

Java应用部署到ECS

本文档会帮助您在云效创建一个 Java Spring Boot 的代码库，并部署到阿里云 ECS 服务器。

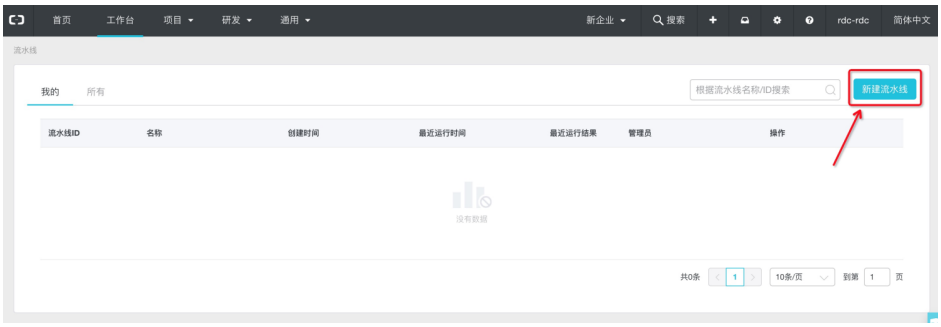
创建企业

首次进入云效，会提示您创建企业。输入企业名称，点击【立即创建】。

创建流水线

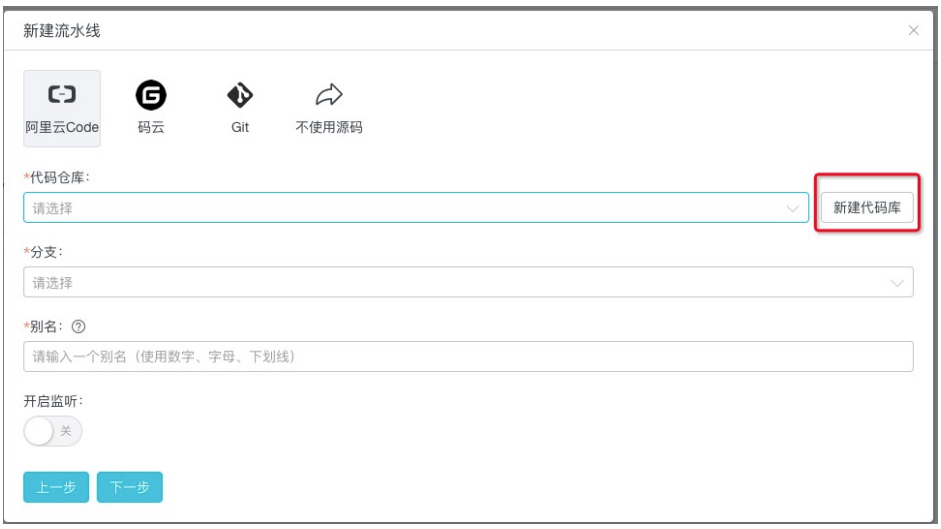
进入企业后，从页面顶栏点击【研发】->【流水线】，进入流水线列表。

点击右上角【新建流水线】，进入流水线创建向导页面。



新建代码库

在源码设置页面中，选择阿里云Code，点击代码仓库右侧【新建代码库】。



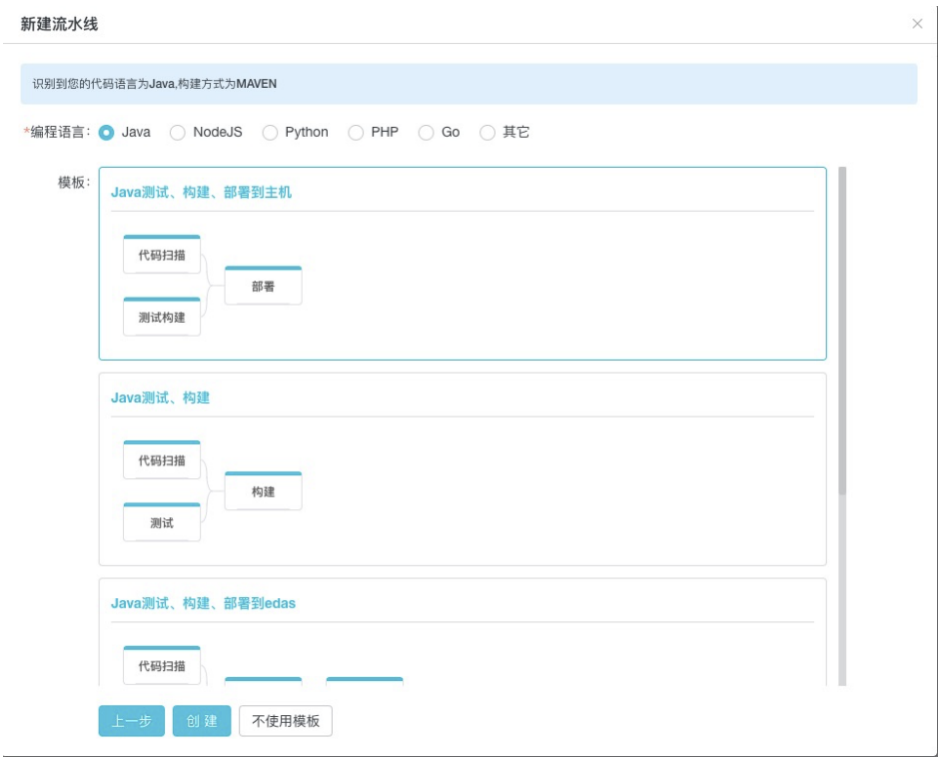
在弹窗中选择或创建代码组，输入新建代码仓库名称，点击【下一步】。

在代码模板中选择 Java 和 spring-boot ，取消勾选【生成Dockerfile】，点击【确认】。您可以点击代码模板图标下的预览按钮查看即将生成的代码库文件内容。

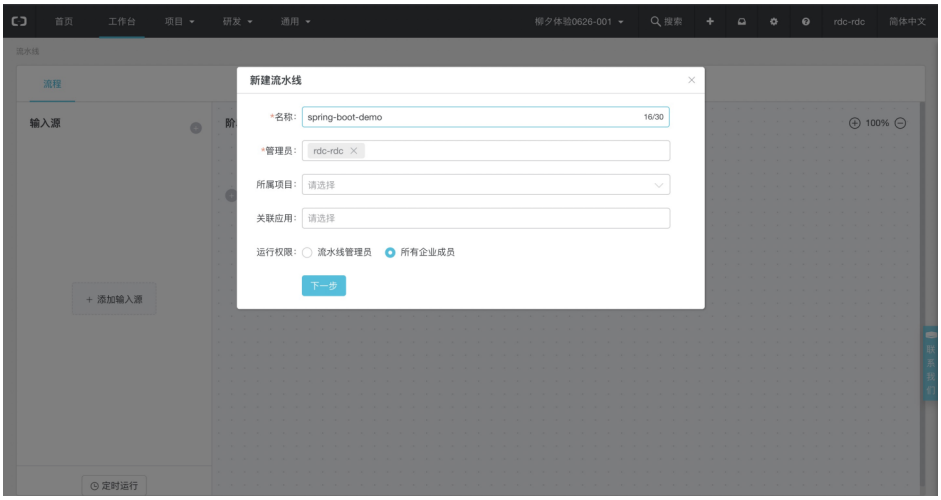
代码库创建完成后会恢复到代码库选择页面并回填刚才创建代码库的信息，为了在代码提交时候触发持续集成，打开【开启监听】，然后点击【下一步】。

编辑流水线

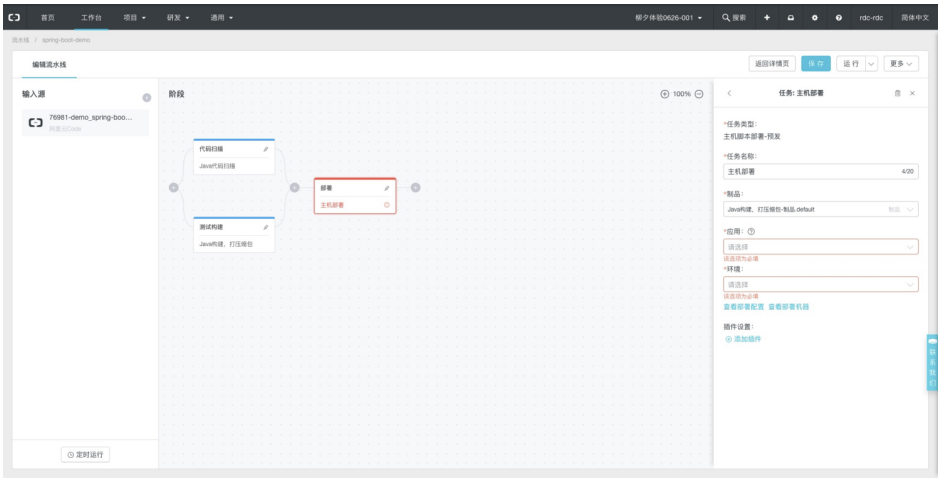
云效会识别代码库语言并推荐相应流水线模板，使用默认置顶选中的【Java构建，测试，部署到主机】流水线模板，然后点击【创建】。



填写流水线名称，点击【下一步】。

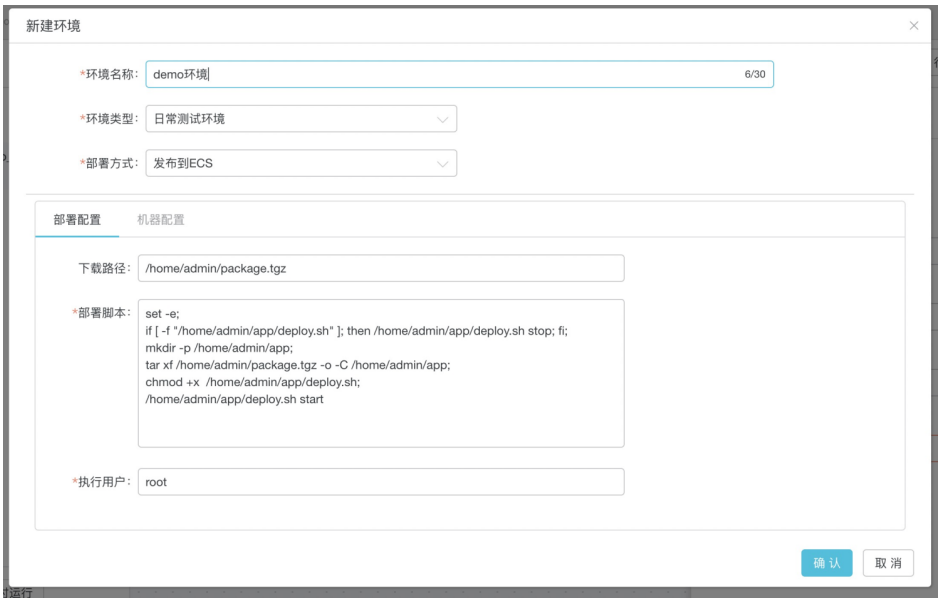


部署配置。点击阶段【部署】阶段进行部署配置，在任务列表中选择【主机部署】，选择由构建环节产出的待部署【制品】。



新建应用。在【应用】下拉框中选择【新建应用】填写应用名称，点击【确认】。

新建环境。在【环境】下拉框中选择【新建环境】，填写环境名称，点击【机器配置】。

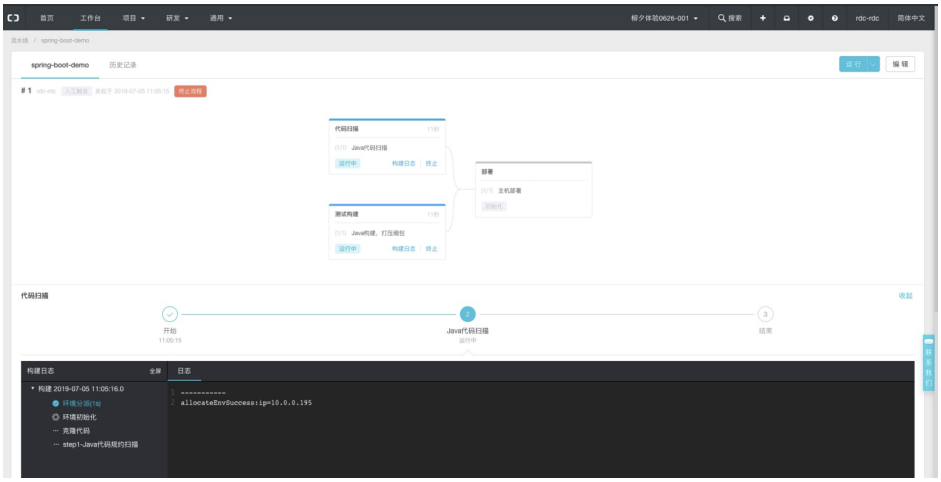


机器配置。点击【使用临时模板机器】，稍等片刻会获得一台用于测试的机器。

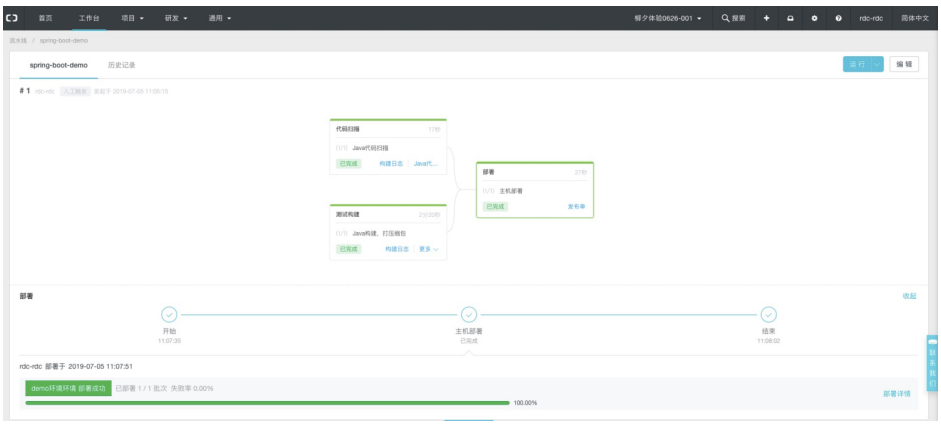
获取临时机器成功后点击【刷新】按钮，会在机器配置页面左边看到刚才创建的机器。选中并添加到右侧，点击【确认】完成环境创建。

运行流水线

点击右上角的【运行】，即可保存并触发流水线。

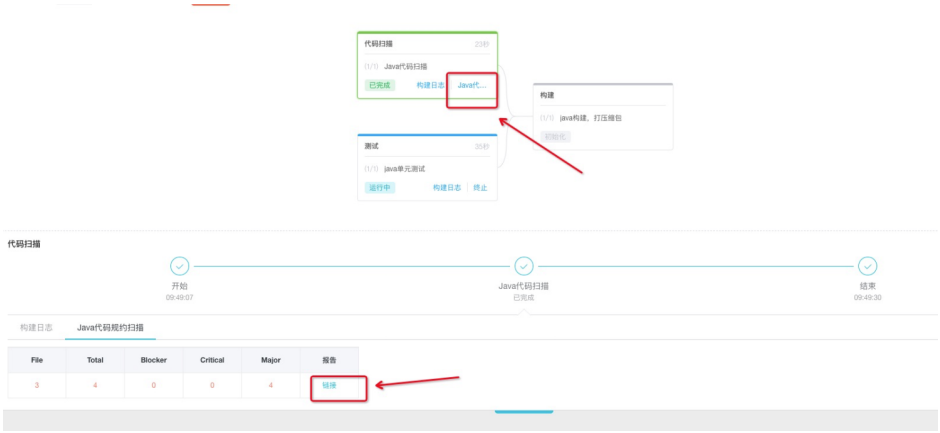


运行成功。

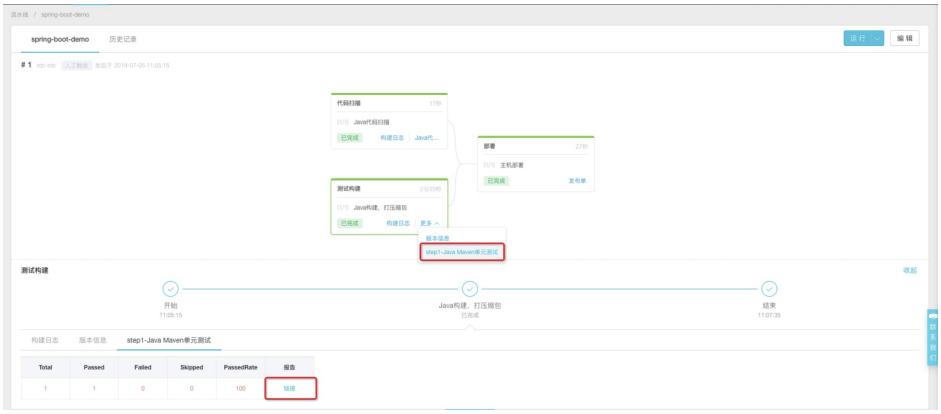


查看测试报告

Java 规约扫描。点击阶段里的点击链接可以查看报告预览，点击预览中的【链接】可以查看完整报告。



点击测试构建阶段里的【更多】，通过【版本信息】可以查看和下载用于部署的软件包，点击【单元测试】可以展开测试结果预览，点击预览里的【链接】可以查看测试报告详情。



查看部署结果

点击【部署详情】可以查看部署单。在部署单中可以查看每台机器的部署情况和日志。

访问 ECS 的公网 IP 的 8080 端口，可以看到服务运行起来了。

Greetings from Spring Boot!

Java应用部署到Kubernetes

本文档会帮助您在云效创建一个 Spring Boot 的代码库，并部署到云效提供的使用 Kubernetes 集群。

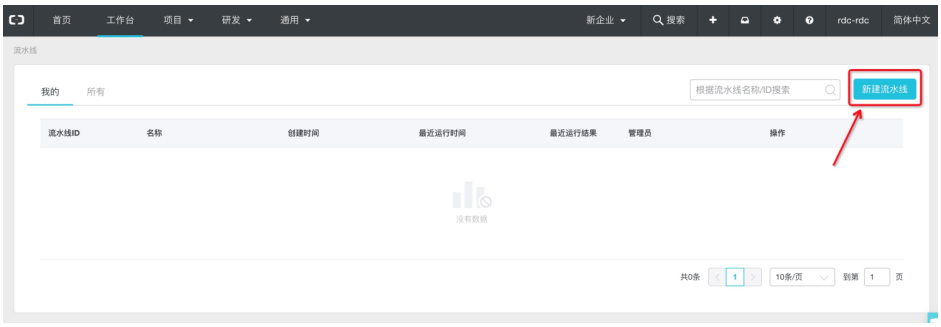
创建企业

首次进入云效，会提示您创建企业。输入企业名称，点击【立即创建】。

创建流水线

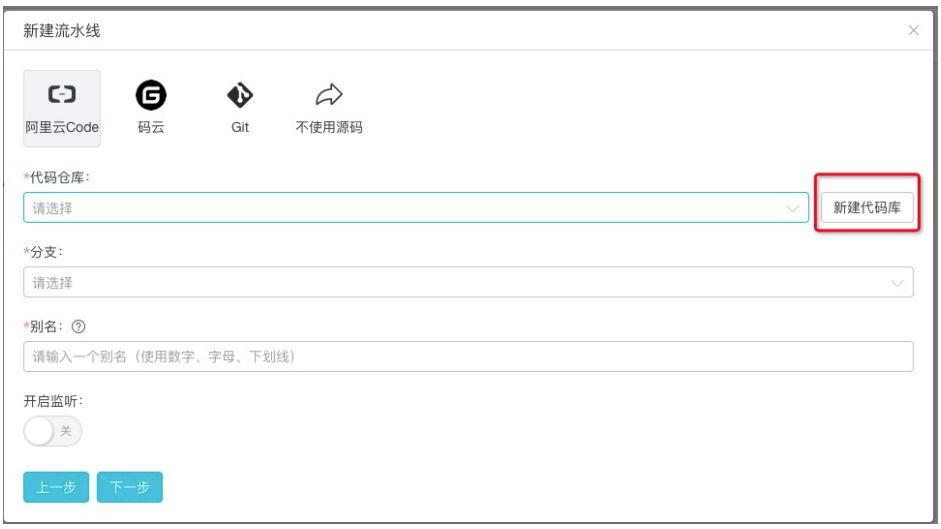
进入企业后，从页面顶栏点击【研发】->【流水线】，进入流水线列表。

点击右上角【新建流水线】，进入流水线创建向导页面。



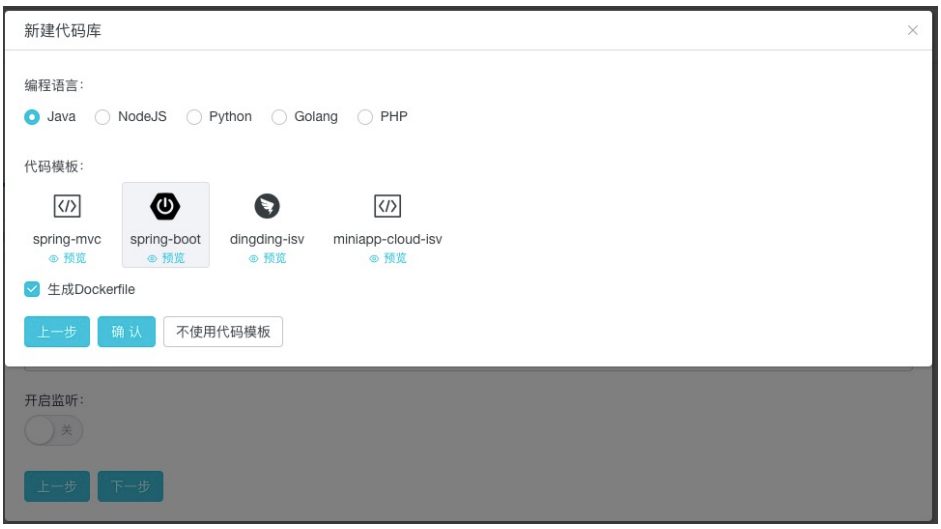
新建代码库

在源码设置页面中，选择阿里云Code，点击代码仓库右侧【新建代码库】。



在弹窗中选择或创建代码组，输入新建代码仓库名称，点击【下一步】。

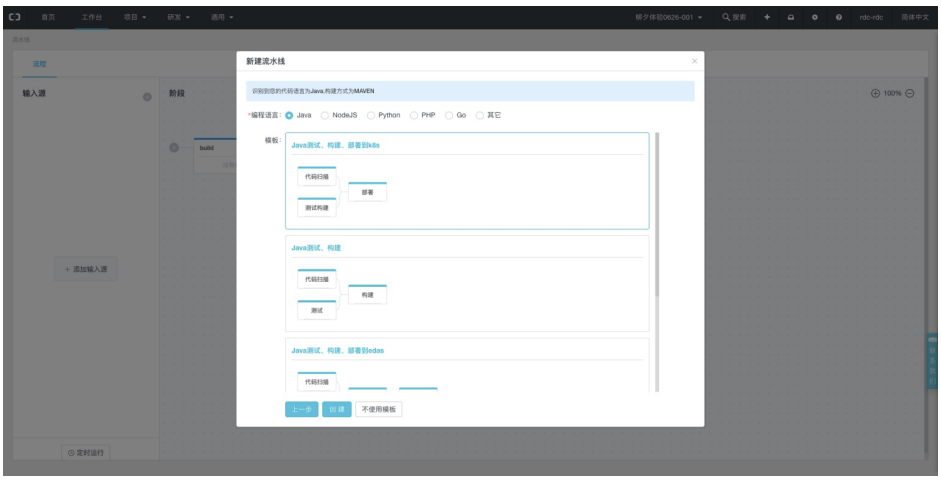
在代码模板中选择 Java 和 spring-boot ，勾选【生成Dockerfile】，点击【确认】。您可以点击代码模板图标下的预览按钮查看即将生成的代码库文件内容。



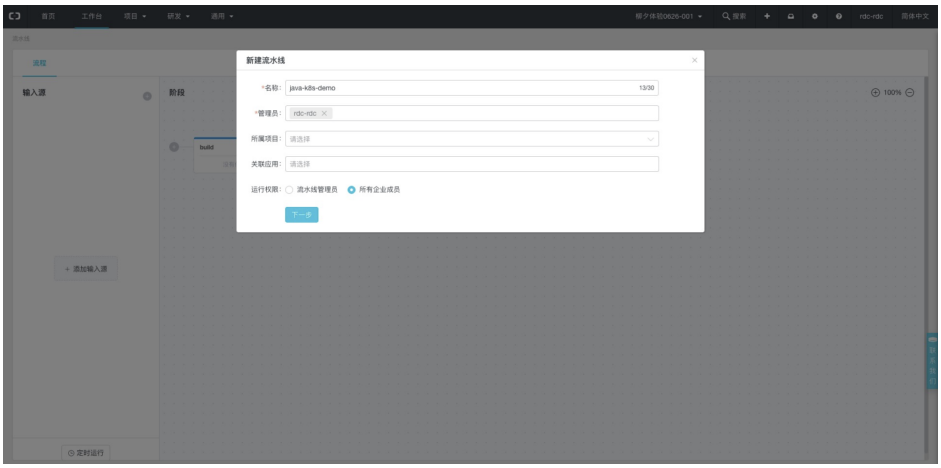
为了在代码提交时候触发持续集成，打开【开启监听】，然后点击【下一步】。

编辑流水线

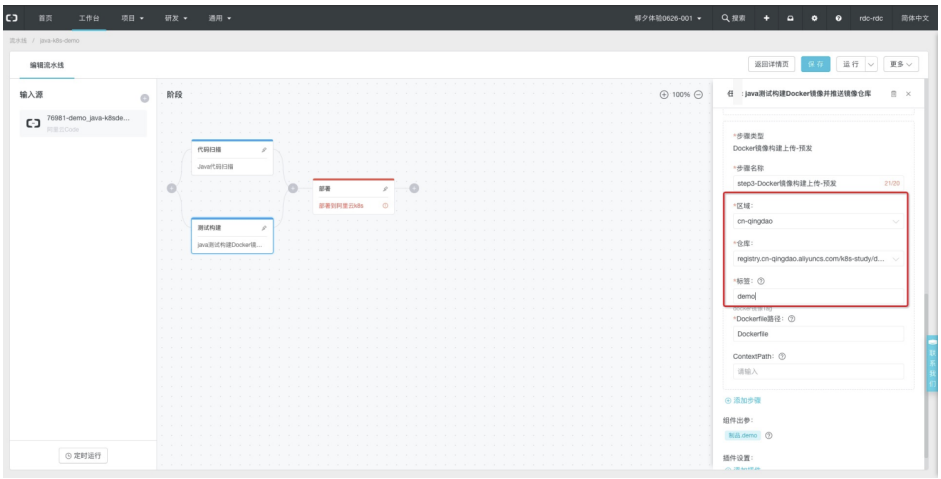
云效会识别代码库语言并推荐相应流水线模板，使用默认置顶选中的【Java测试、构建、部署到k8s】流水线模板，然后点击【创建】。



填写流水线名称，点击【下一步】。

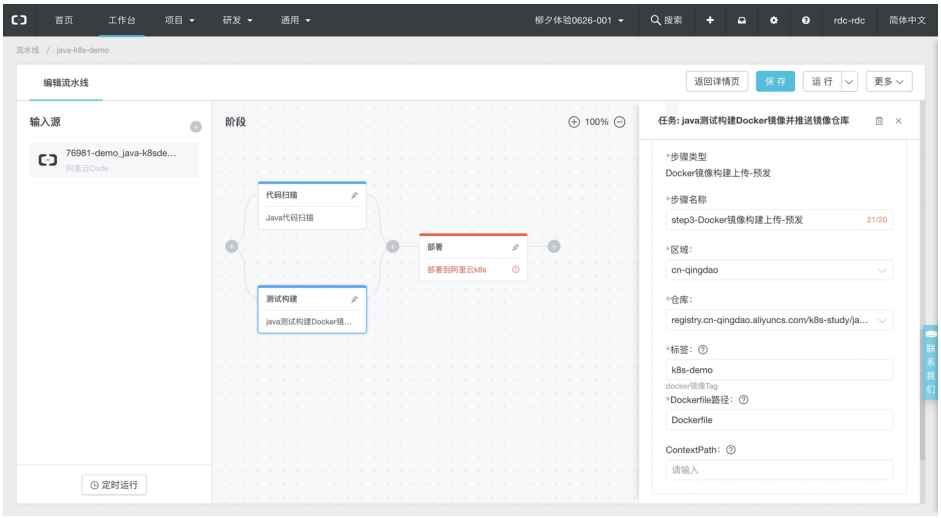


完善测试构建阶段配置。点击【测试构建】阶段，在右侧浮层中点击【java测试构建Docker镜像并推送镜像仓库】任务，在任务编辑页面中点击【请前往授权绑定】的链接完成 RAM 授权。然后回到流水线配置页面选择【区域】。

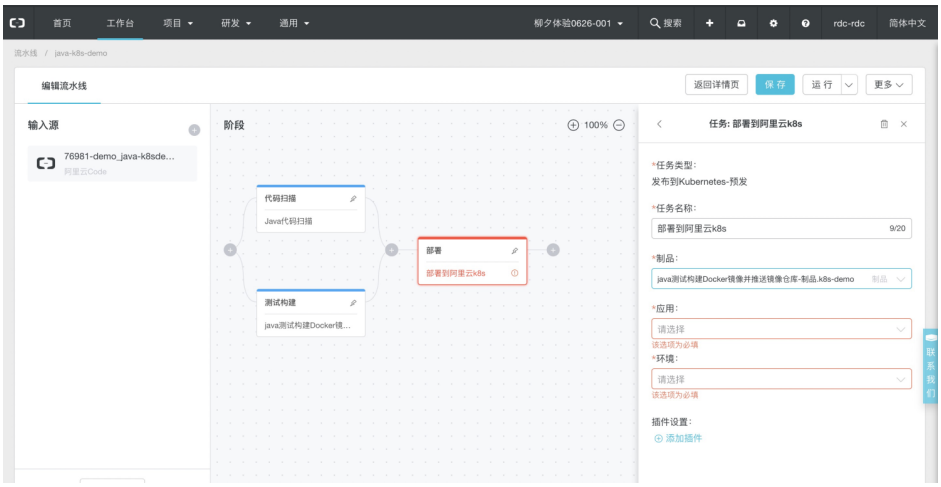


在【仓库】的下拉框中点击【新建镜像仓库】，填写好信息之后，点击【确认】，窗口会自动关闭，并把新创建好的仓库地址回填到流水线中。

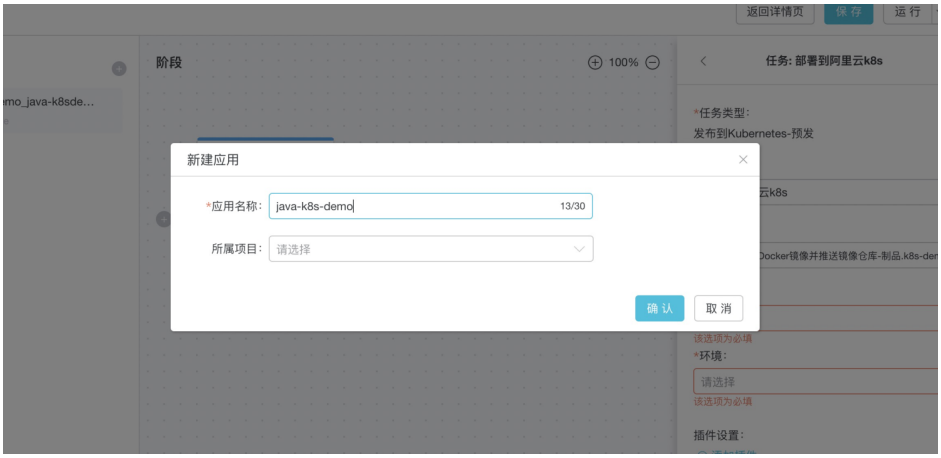
填写【标签】，使用 ${TIMESTAMP} 作为标签，便于版本管理。



进行部署配置。在流水线的部署任务中选择【制品】，即在构建阶段产生的那个镜像。



新建应用。在【应用】下拉框中点击【新建应用】，输入应用名并点击【确认】。



新建环境。在【环境】下拉框中点击【新建环境】，进入到 Kubernetes 集群配置页面。

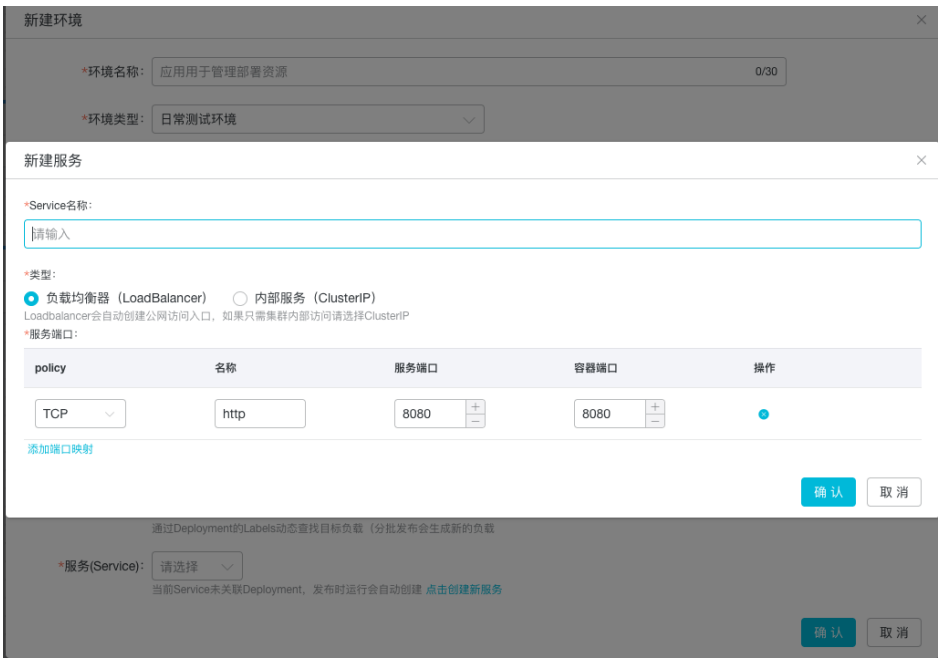
创建免费集群

在新建环境的弹框中进行 Kubernetes 部署配置。点击【免费试用云效提供的集群】来获取一个临时的免费集群，该集群会自动配置到您的企业中，在 5 小时后会失效并从企业中删除。

在试用免费集群的弹窗中记得点击【点此复制kubeconfig】，并进行本地配置，否则将无法从本地访问集群。

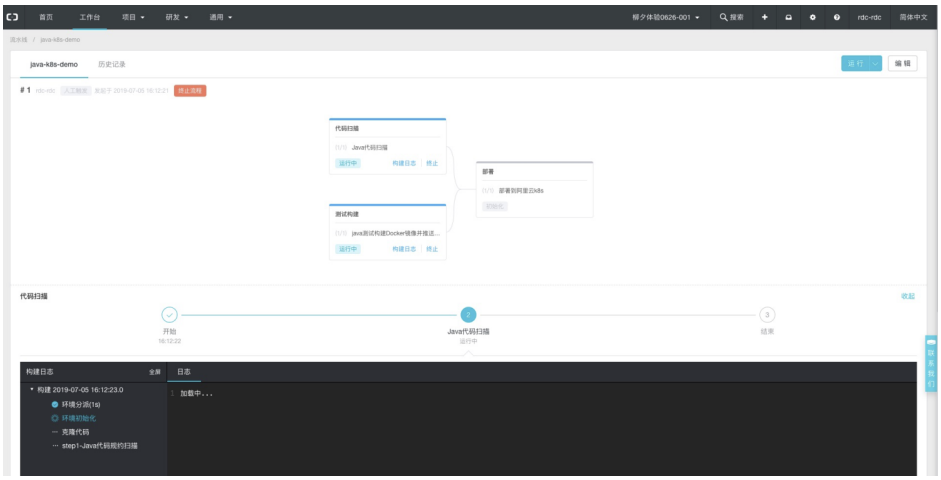
关闭弹框，选择集群、命名空间，选择滚动发布，并点击【点击创建新服务】的链接来创建 Service。

输入 Service 名称，选择负载均衡器类型，服务端口和容器端口都设置为 8080，然后点击【确认】。弹框关闭后，再点击创建环境的弹框的【确认】，环境即创建成功，并回填到流水线配置中。

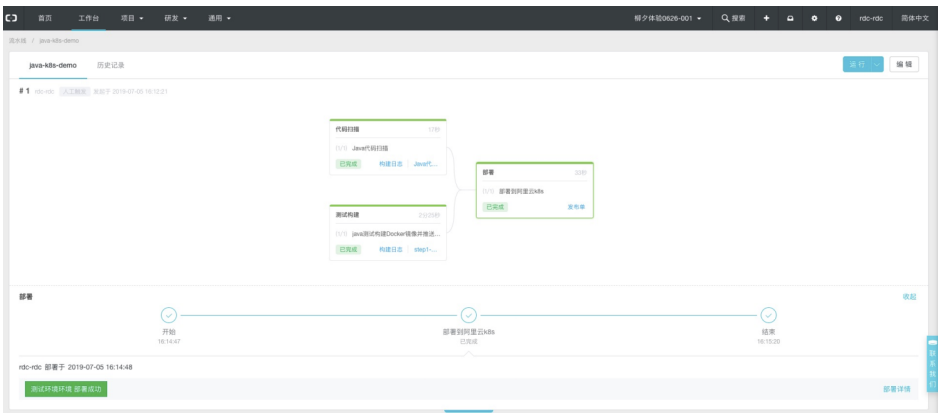


运行流水线

点击流水线编辑页面右上角【运行】，流水线会自动保存并开始运行。

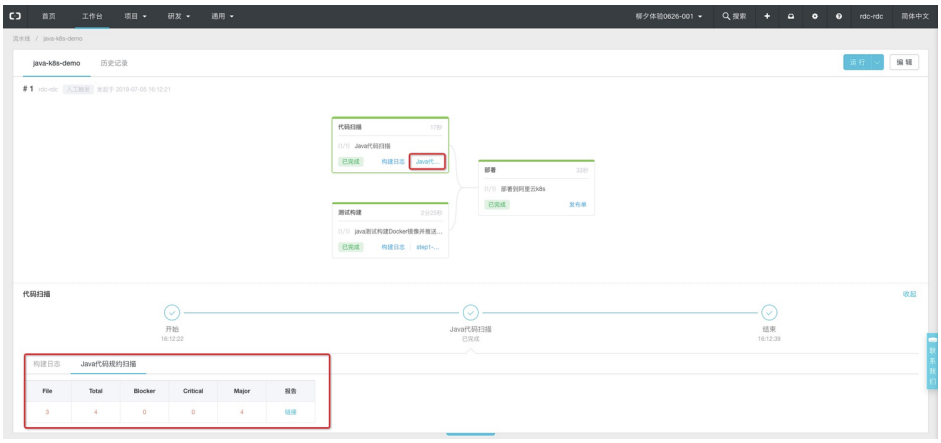


运行成功。

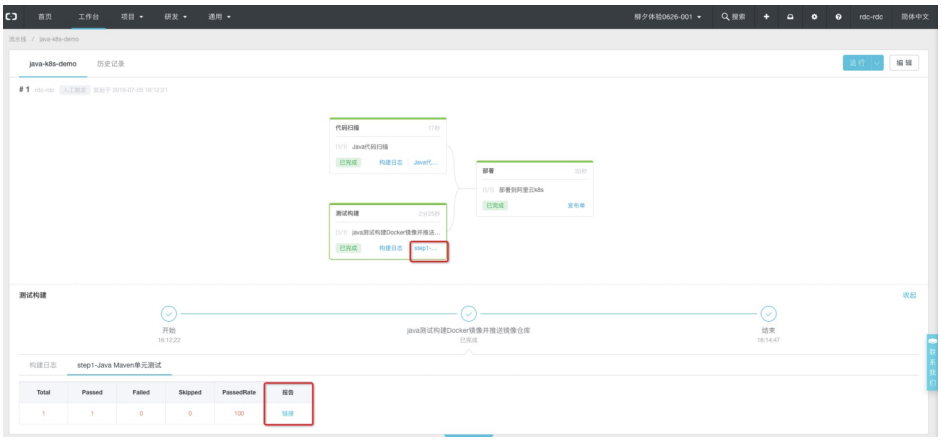


查看测试报告

Java 规约扫描。点击阶段里的点击链接可以在页面下方查看报告预览，预览中点击【链接】可以查看完整报告。

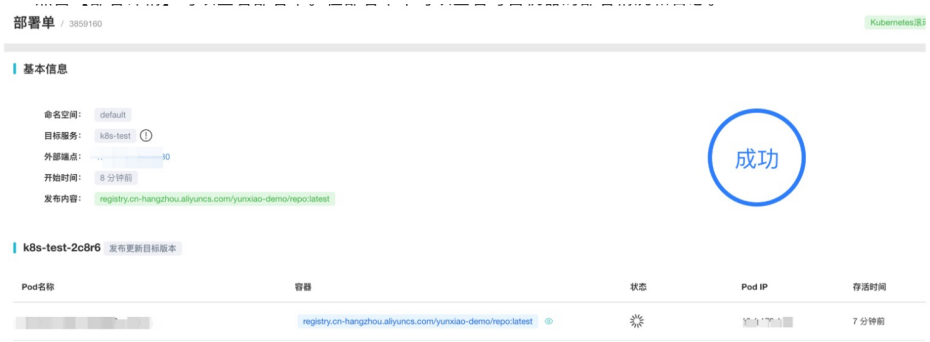


单元测试报告。点击【单元测试】可以展开测试结果预览，点击预览里的【链接】可以查看测试报告详情。

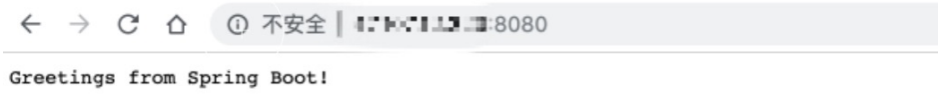


查看部署结果

点击【部署详情】可以查看部署单。在部署单中可以查看每台机器的部署情况和日志。



由于 Service 类型是 LoadBalancer，所以会自动生成一个公网IP来暴露这个 Service。显示在部署单的外部端点部分。点击【外部端点】地址，可以看到服务运行起来了。




```
Greetings from Spring Boot!
```

NodeJS入门

NodeJS应用部署到ECS

本文档会帮助您在云效创建一个 NodeJS Express 的代码库，并部署到阿里云 ECS 服务器。

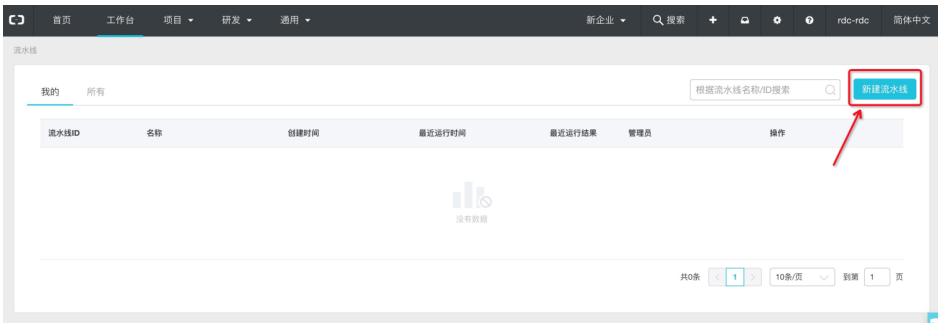
创建企业

首次进入云效，会提示您创建项目。

创建流水线

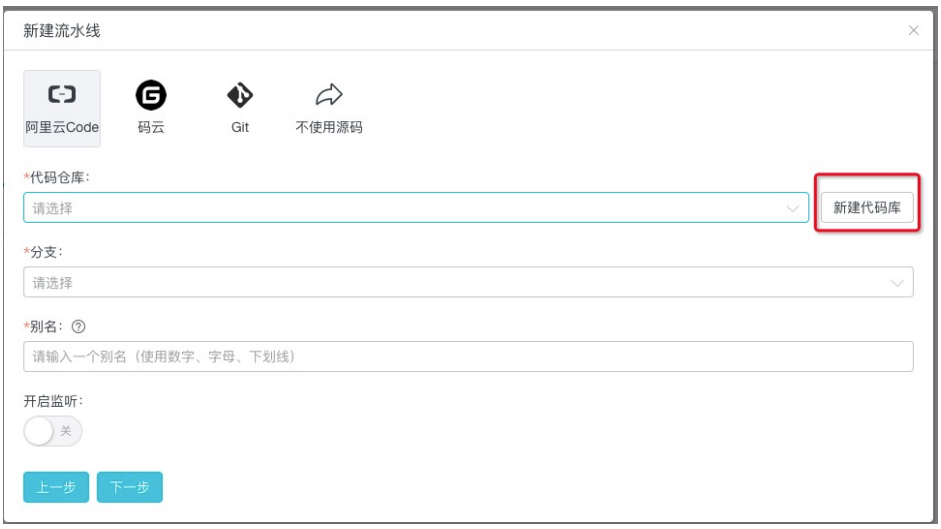
进入企业后，从页面顶栏点击【研发】->【流水线】，进入流水线列表。

点击右上角【新建流水线】，进入流水线创建向导页面。

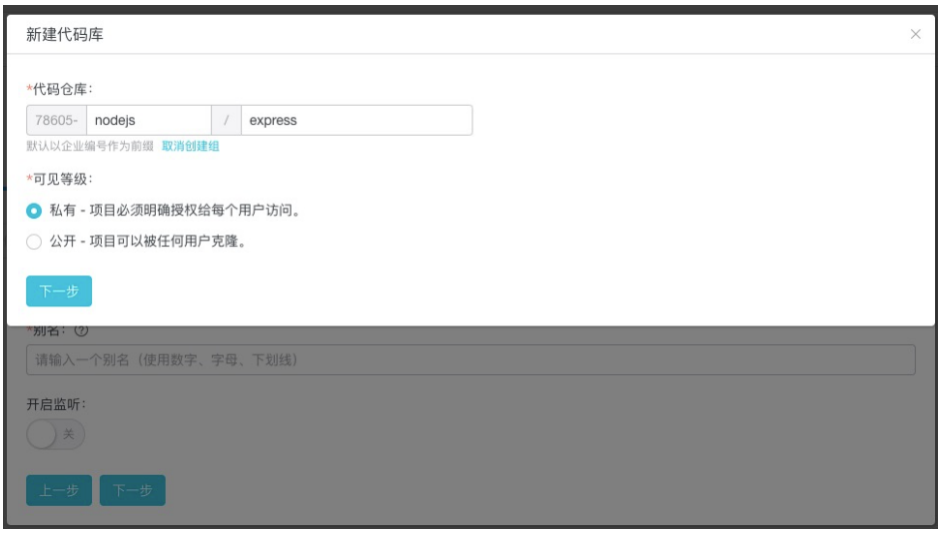


新建代码库

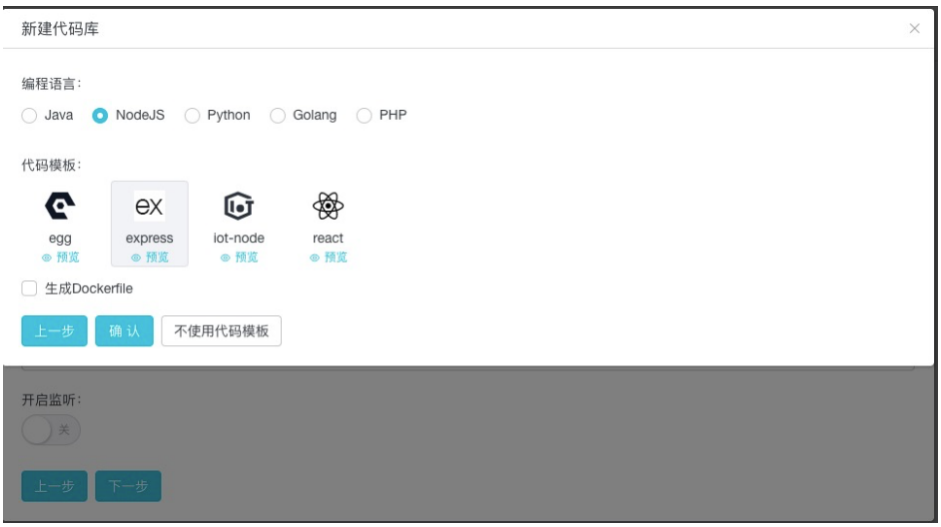
在源码设置页面中，选择阿里云Code，点击代码仓库右侧【新建代码库】。



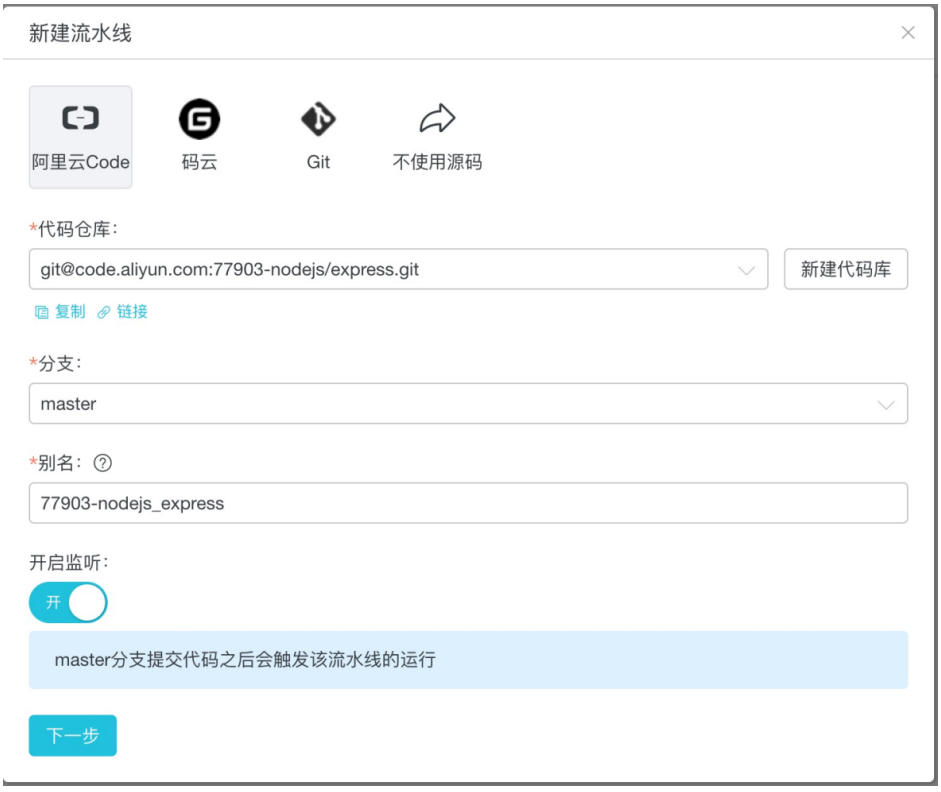
在弹窗中选择或创建代码组，输入新建代码仓库名称，点击【下一步】。



在代码模板中选择 NodeJS 和 express ，取消勾选【生成 Dockerfile】，点击【确认】。您可以点击代码模板图标下的预览按钮查看即将生成的代码库文件内容。

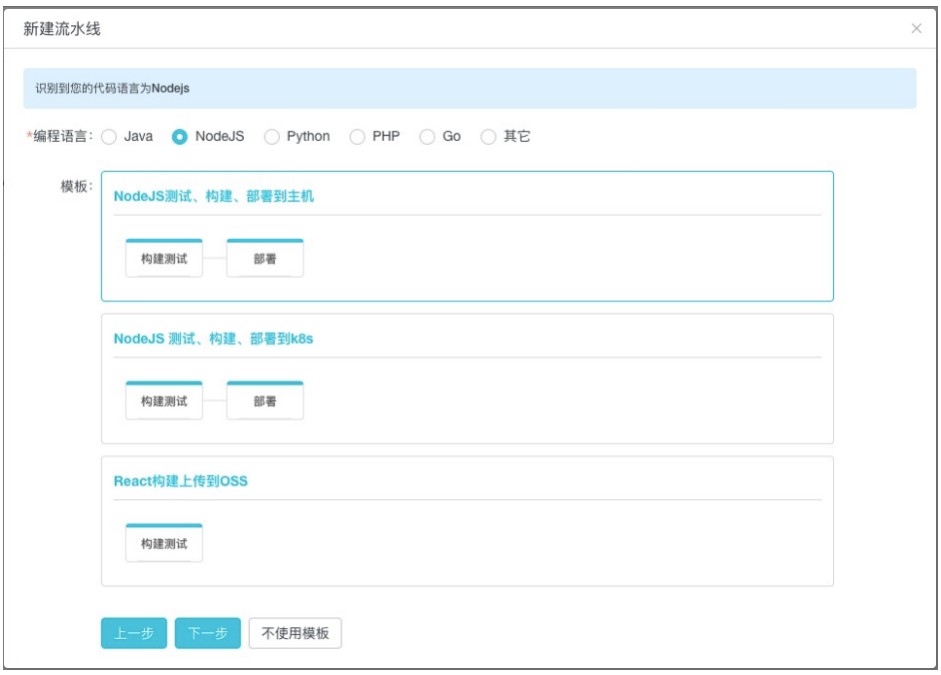


代码库创建完成后会恢复到代码库选择页面并回填刚才创建代码库的信息，为了在代码提交时候触发持续集成，打开【开启监听】。然后点击【下一步】。

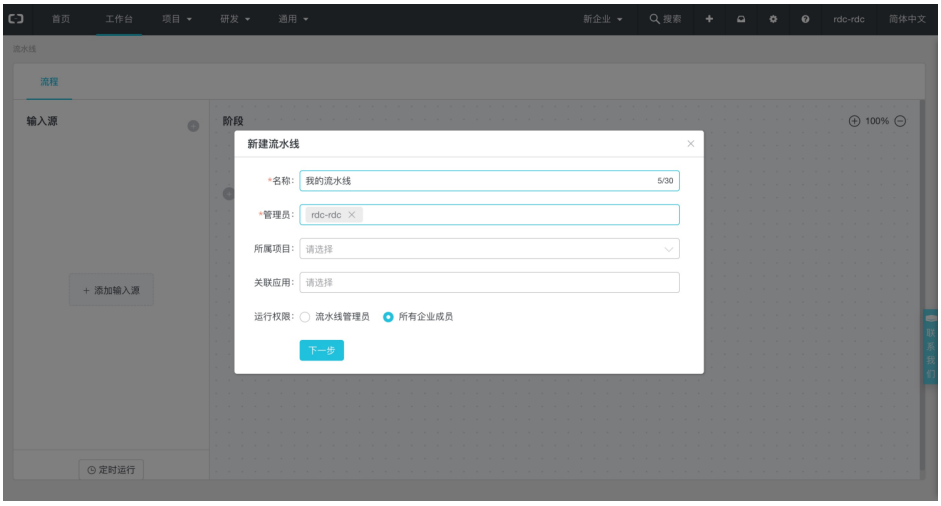


编辑流水线

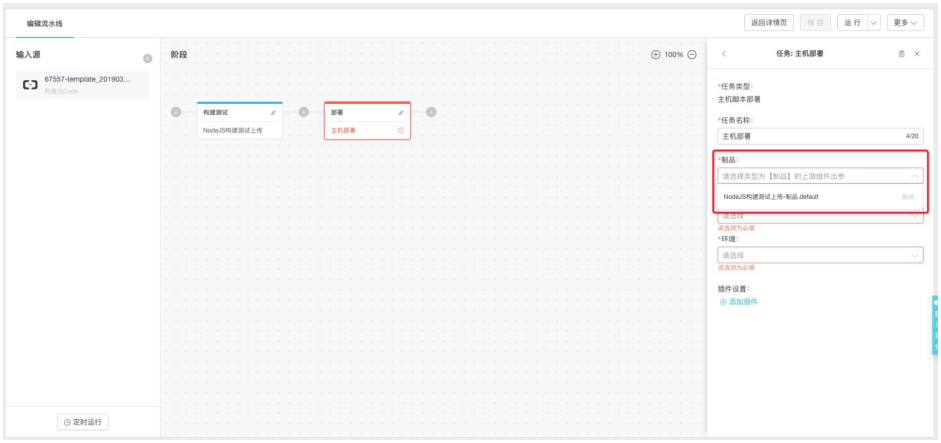
云效会识别代码库语言并推荐相应流水线模板，使用默认置顶选中的【NodeJS测试、构建、部署到主机】流水线模板，然后点击【创建】。



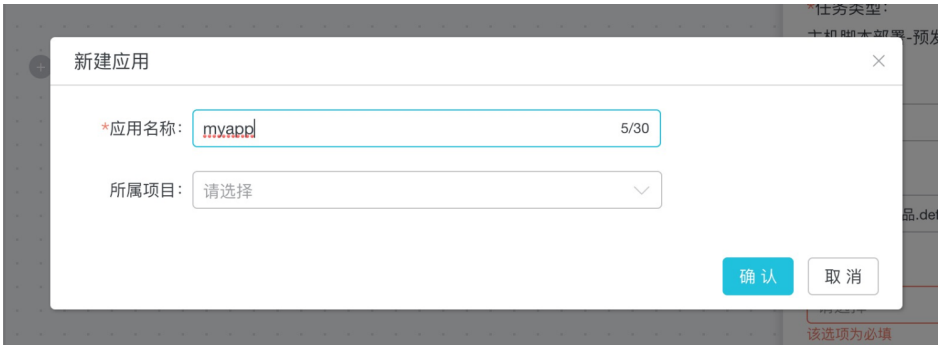
填写流水线名称，点击【下一步】。



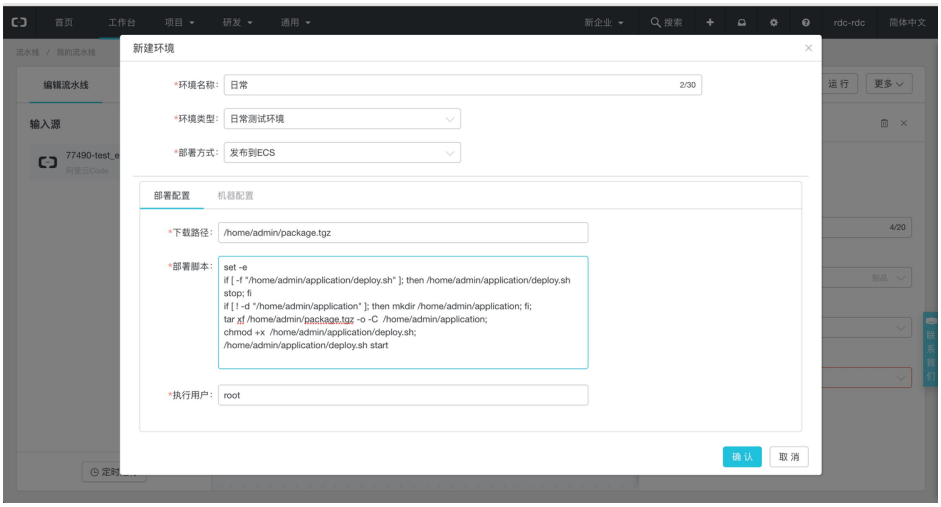
部署配置。点击【部署】阶段进行部署配置。选择由构建环节产出的待部署【制品】。



新建应用。在【应用】下拉框中选择【新建应用】填写应用名称，点击【确认】。



新建环境。在【环境】下拉框中选择【新建环境】，填写环境名称，点击【机器配置】。



进行机器配置。点击【使用临时模板机器】，稍等片刻会获得一台用于测试的机器。

获取临时机器成功后点击【刷新】按钮，会在机器配置页面左边看到刚才创建的机器。选中并添加到右侧，点击【确认】完成环境创建。

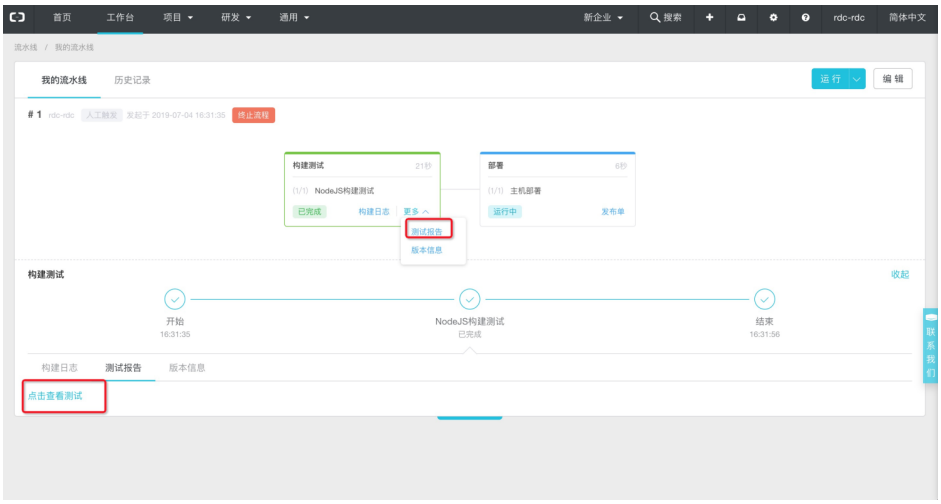
运行流水线

点击右上角的【运行】，即可保存并触发流水线。

运行成功。

查看测试报告

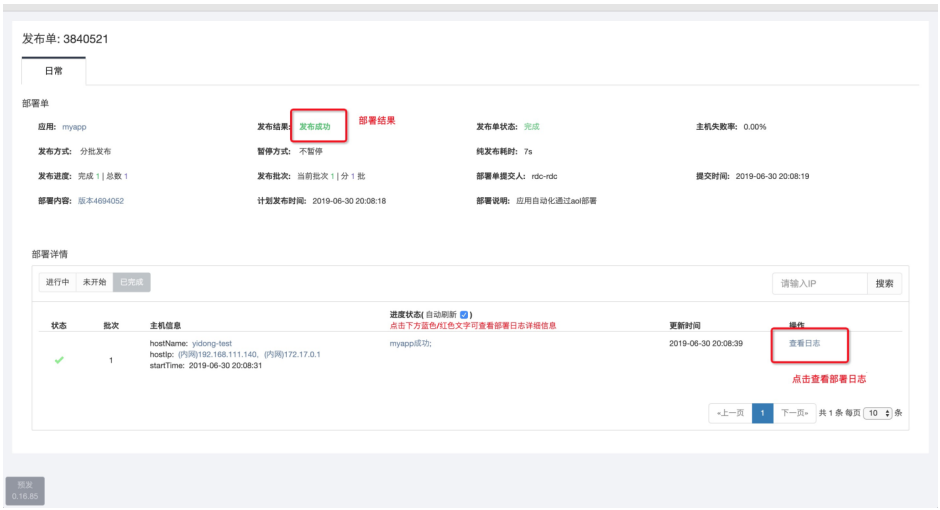
点击阶段中的【测试报告】按钮可以查看测试报告。点击【版本信息】可以查看和下载用于部署的软件包。



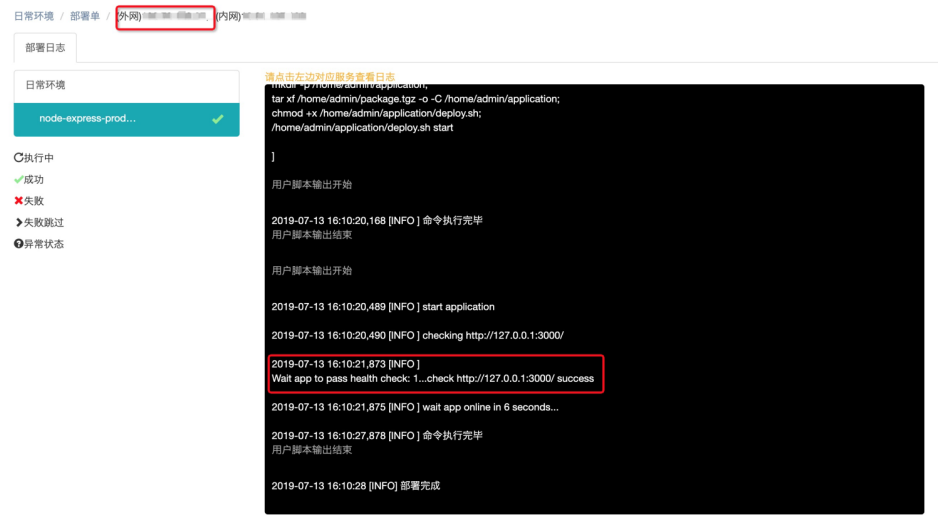
查看部署结果

点击阶段选框，下方视图为阶段详情信息，右下角【打开发布单】用于查看发布单。

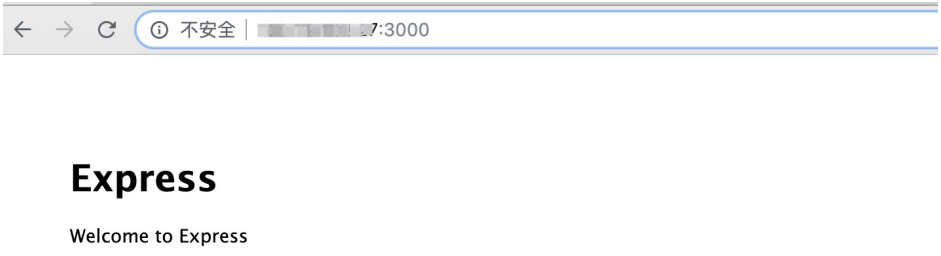
发布单页面，点击【查看日志】查看部署日志。



看到 check success 表示应用已经在 3000 端口启动成功，这个时候访问服务器公网 IP:3000 即可看到应用页面。临时机器的公网 IP 在部署日志页面左上角可以看到。



访问ECS的公网 IP 的 3000 端口，可以看到服务运行起来了。



NodeJS应用部署到Kubernetes

本文档会帮助您在云效创建一个 NodeJS express 的代码库，并部署到云效提供的使用 Kubernetes 集群。

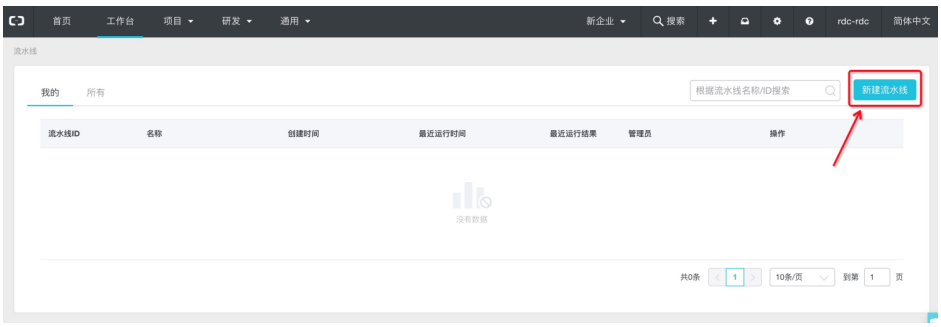
创建企业

首次进入云效，会提示您创建项目。

创建流水线

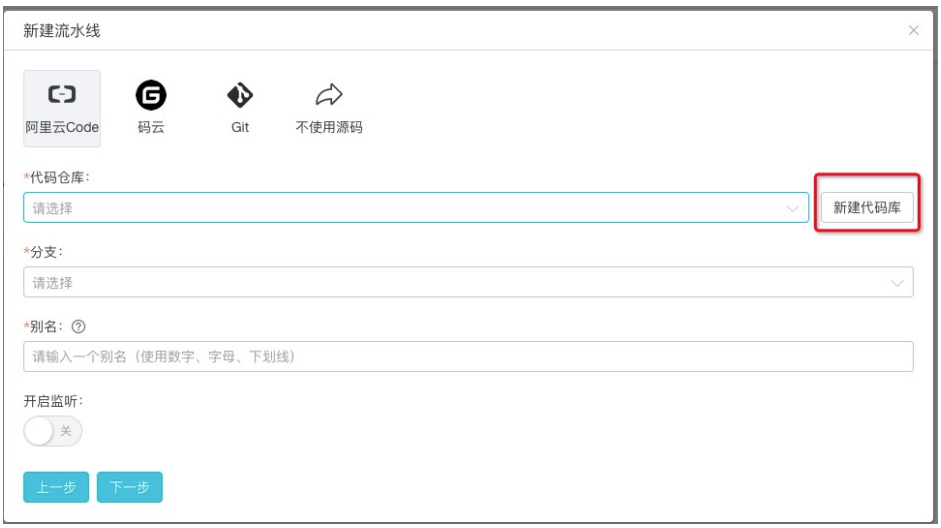
进入企业后，从页面顶栏点击【研发】->【流水线】，进入流水线列表。

点击右上角【新建流水线】，进入流水线创建向导页面。

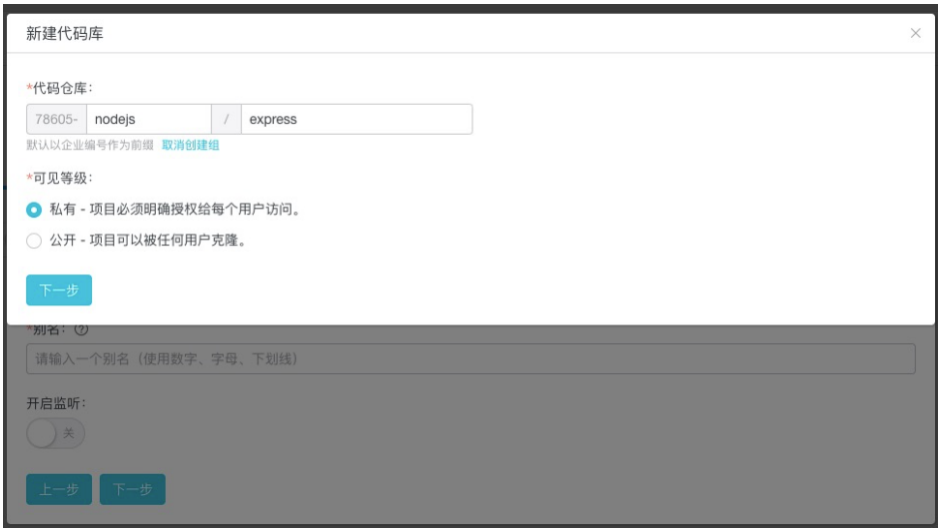


新建代码库

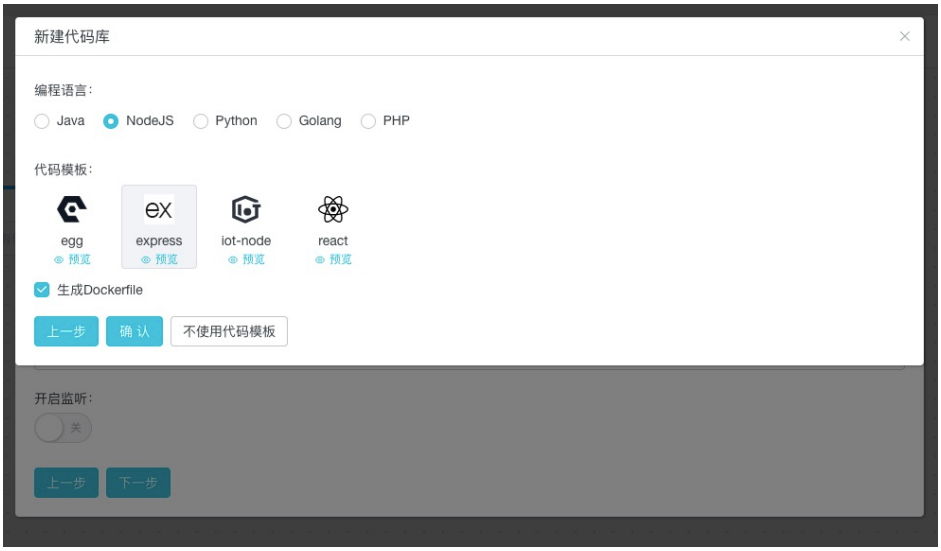
在源码设置页面中，选择阿里云Code，点击代码仓库右侧【新建代码库】。



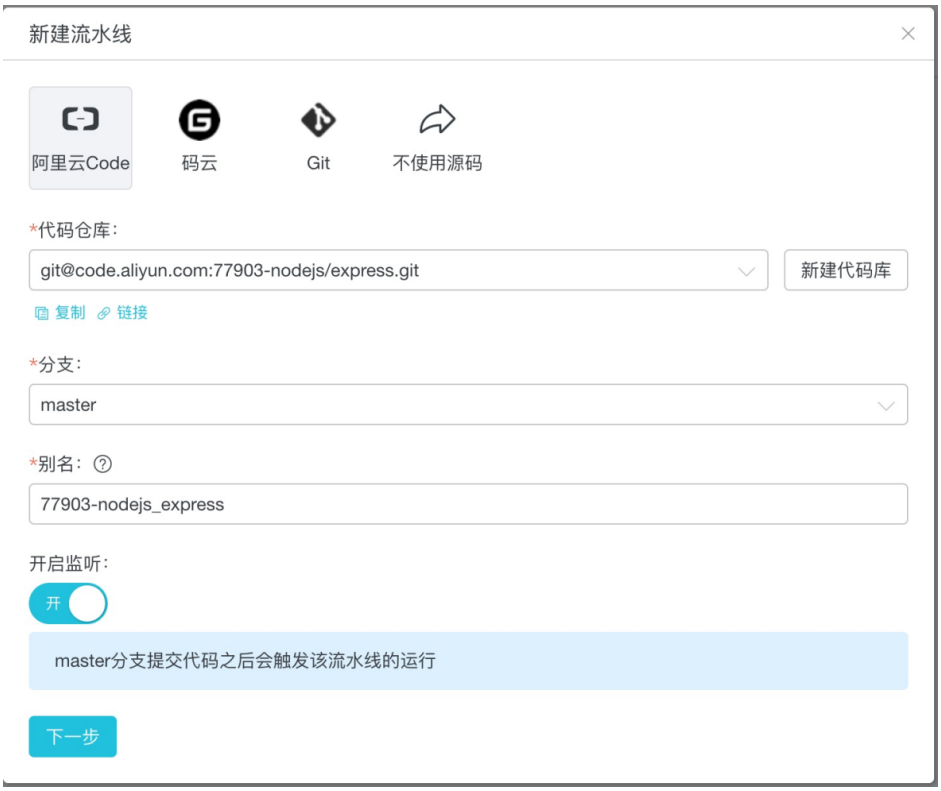
在弹窗中选择或创建代码组，输入新建代码仓库名称，点击【下一步】。



在代码模板中选择 NodeJS 和 express ，勾选【生成 Dockerfile】，点击【确认】。您可以点击代码模板图标下的预览按钮查看即将生成的代码库文件内容。



为了在代码提交时候触发持续集成，打开【开启监听】，然后点击【下一步】。

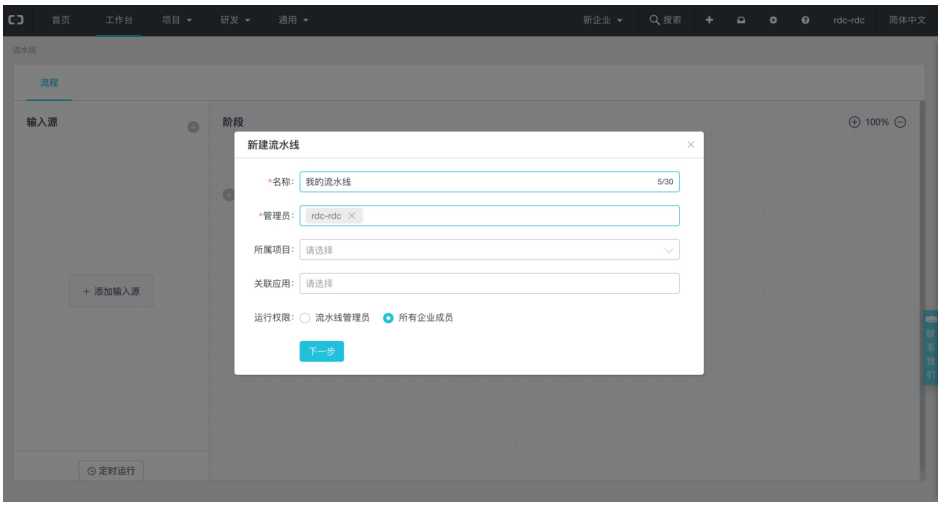


编辑流水线

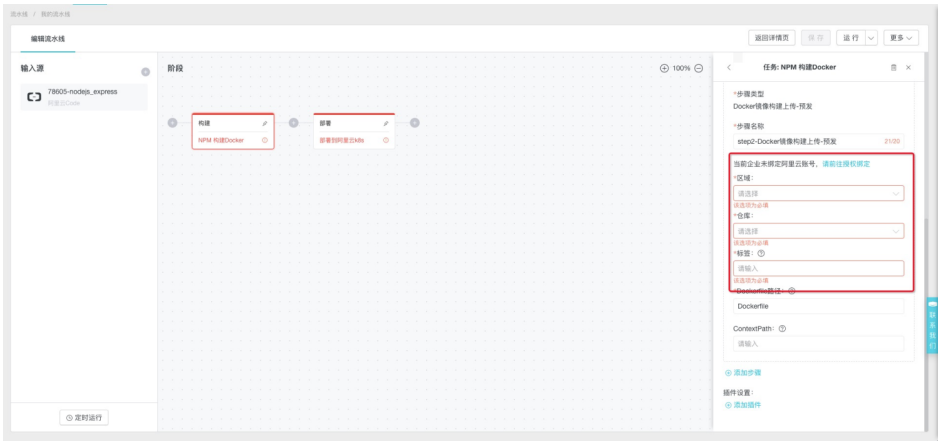
云效会识别代码库语言并推荐相应流水线模板，使用默认置顶选中的【NodeJS构建，测试，部署到

k8s】流水线模板，然后点击【创建】。

填写流水线名称，点击【下一步】。

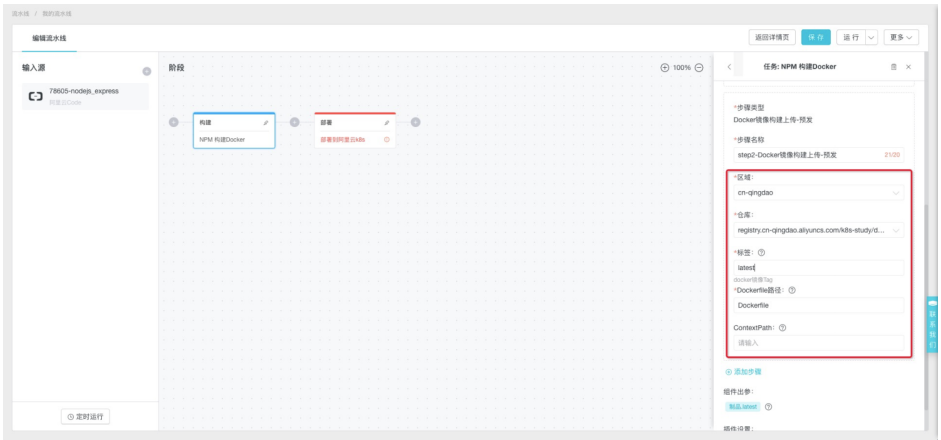


完善测试构建阶段配置。点击【测试构建】阶段，在后侧浮层中点击【NodeJS测试、镜像构建】任务，在任务编辑页面中点击【请前往授权绑定】的链接完成 RAM 授权。然后回到流水线配置页面选择【区域】。

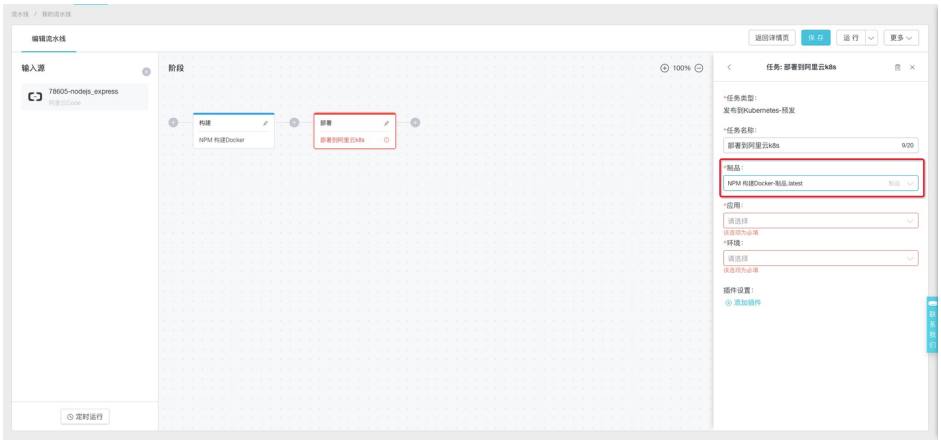


在【仓库】的下拉框中点击【新建镜像仓库】，填写好信息之后，点击【确认】，窗口会自动关闭，并把新创建好的仓库地址回填到流水线中。

填写【标签】，使用 ${TIMESTAMP} 作为标签，便于版本管理。



在【部署】阶段进行部署配置。在流水线的部署任务中选择【制品】，即在构建阶段产生的那个镜像。



在【应用】下拉框中点击【新建应用】，输入应用名，点击【确认】。

在【环境】下拉框中点击【新建环境】，进入到 Kubernetes 集群配置页面。

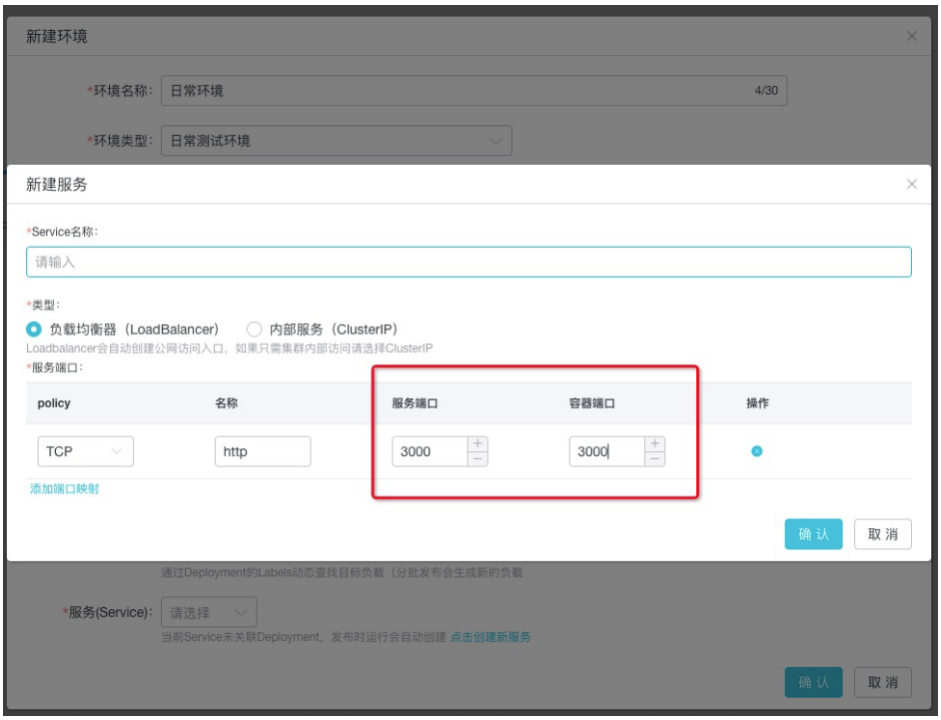
创建免费集群

在【新建环境】的弹框中进行 Kubernetes 部署配置。点击【免费试用云效提供的集群】来获取一个临时的免费集群，该集群会自动配置到您的企业中，并在5小时后会失效并从企业中删除。

在试用免费集群的弹窗中记得点击【点此复制kubeconfig】，并进行本地配置，否则将无法从本地访问集群。

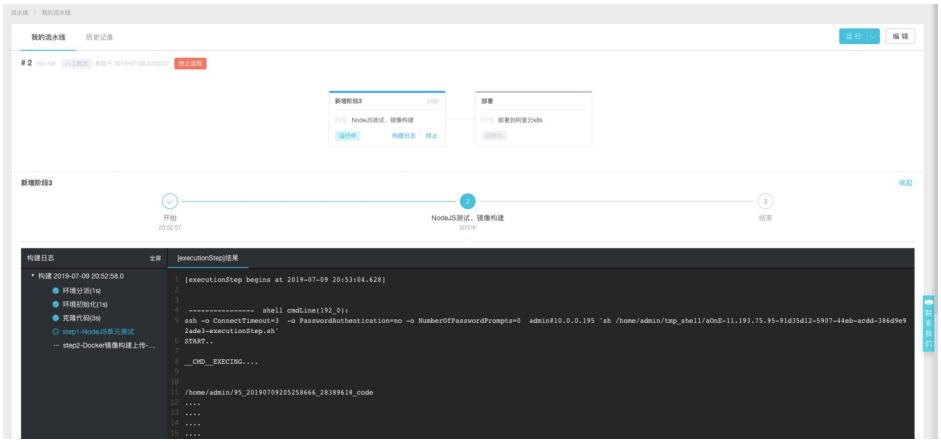
关闭弹框，选择集群、命名空间，选择滚动发布，并点击【点击创建新服务】的链接来创建service。

输入 Service 名称，选择负载均衡器类型，服务端口和容器端口都设置为3000，然后点击【确认】。弹框关闭后，再点击创建环境的弹框的【确认】，环境即创建成功，并回填到流水线配置中。

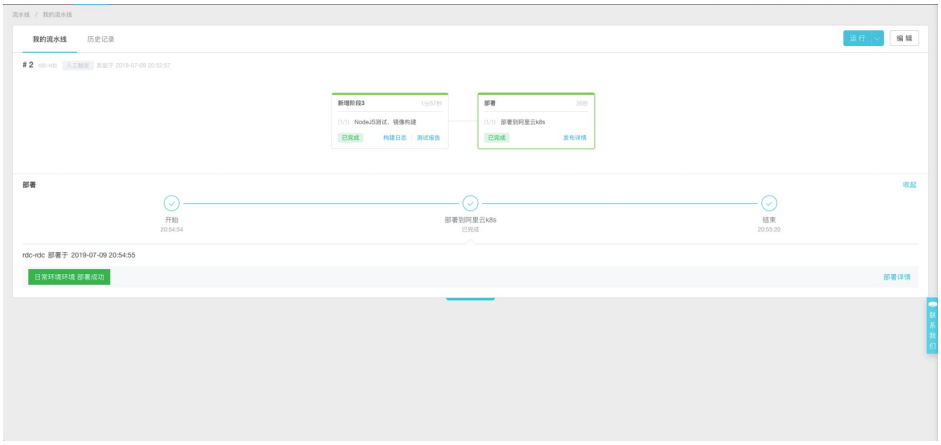


运行流水线

点击流水线编辑页面右上角【运行】，流水线会自动保存，并开始运行。



运行成功。

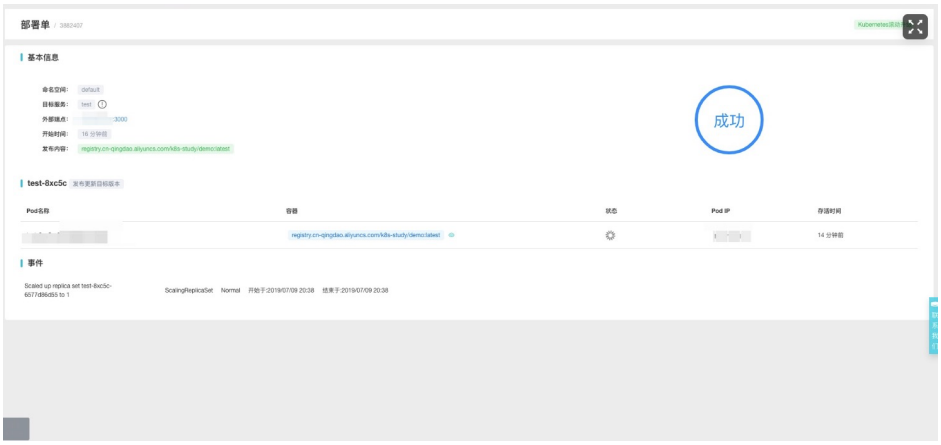


查看测试报告

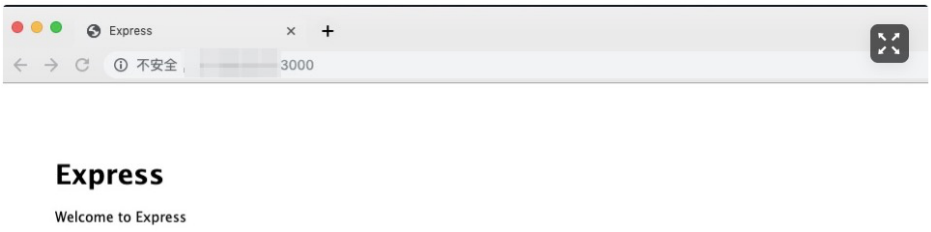
点击构建测试阶段【测试报告】可以展开测试结果预览，点击预览里的【点击查看测试】可以查看测试报告详情。

查看部署结果

点击【部署详情】可以查看部署单。在部署单中可以查看每台机器的部署情况和日志。



由于 Service 类型是 LoadBalancer，所以会自动生成一个公网IP来暴露这个 Service。显示在部署单的外部端点部分。点击【外部端点】地址，可以看到服务运行起来了。



React应用部署到OSS

本文档会帮助您在云效创建一个 React 的代码库，并发布到阿里云 OSS 对象存储服务。

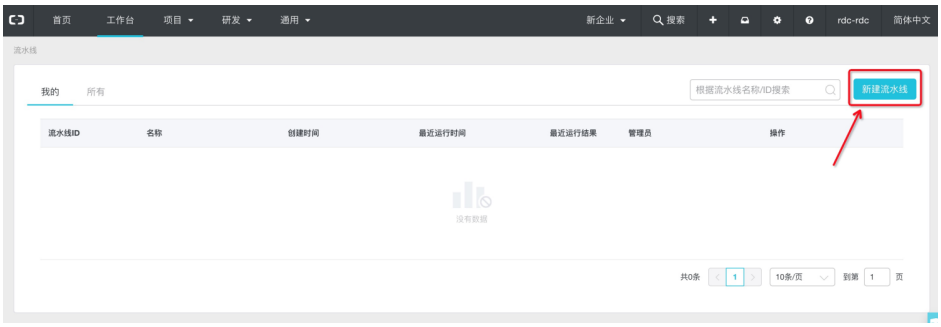
创建企业

首次进入云效，会提示您创建企业。

创建流水线

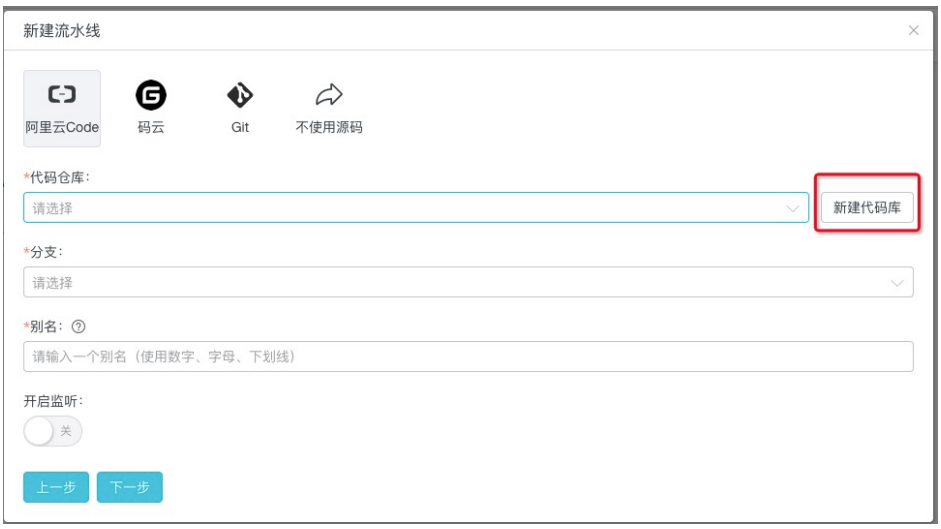
进入企业后，从页面顶栏点击【研发】->【流水线】，进入流水线列表。

点击右上角【新建流水线】，进入流水线创建向导页面。

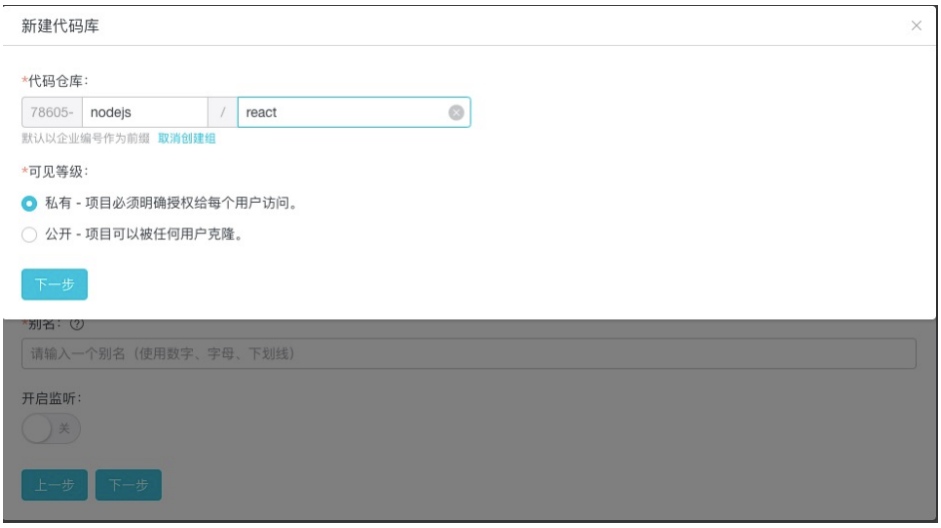


新建代码库

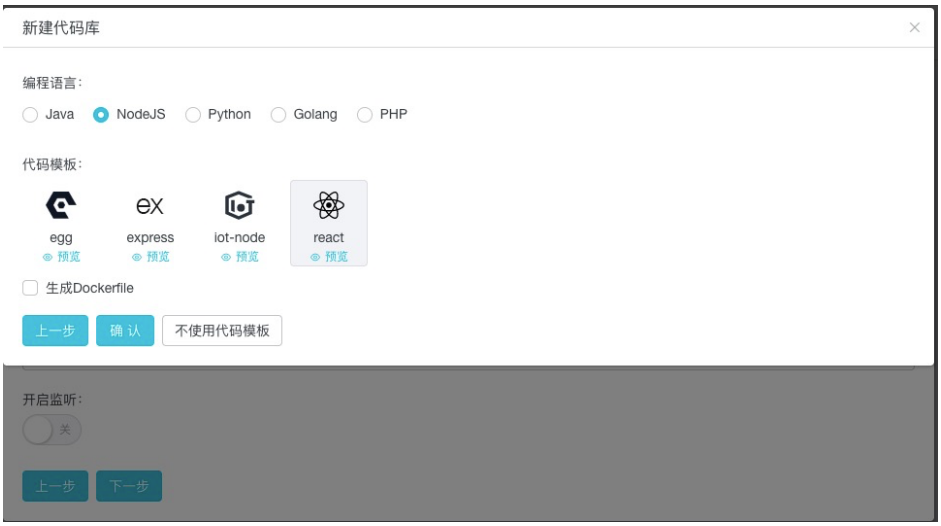
在源码设置页面中，选择阿里云Code，点击代码仓库右侧【新建代码库】。



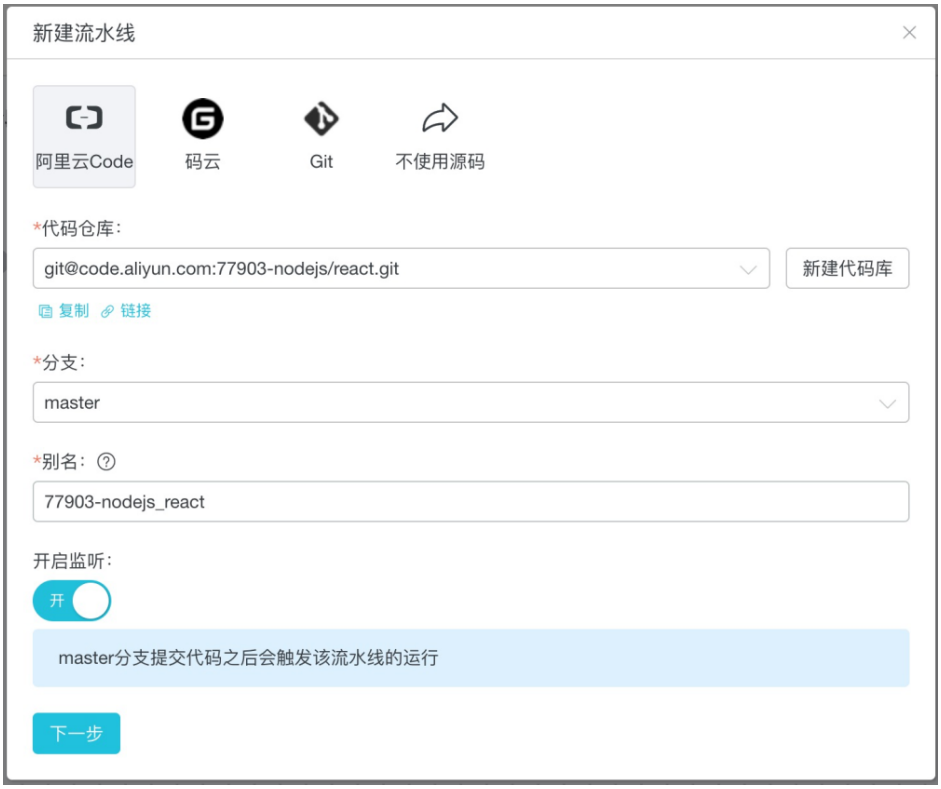
在弹窗中选择或创建代码组，输入新建代码仓库名称，点击【下一步】。



在代码模板中选择 NodeJS 和 React ，点击【确认】。



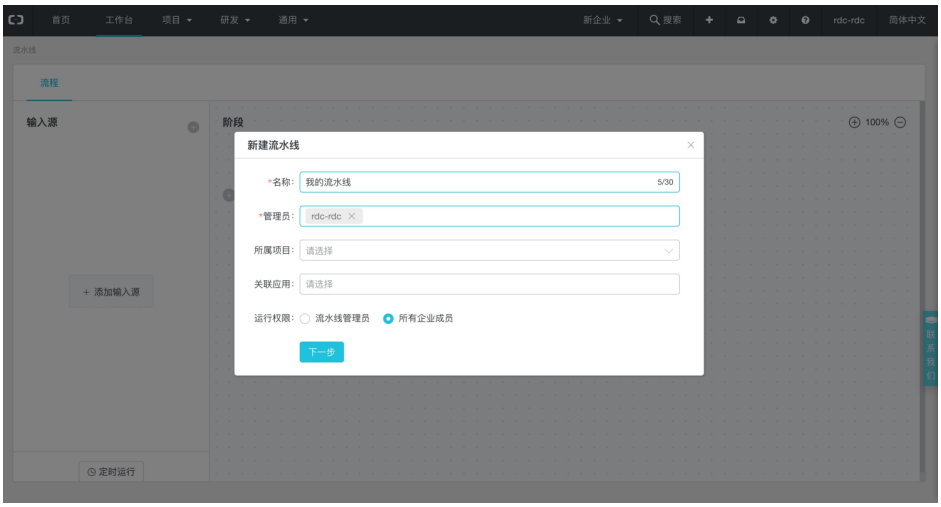
为了在代码提交时候触发持续集成，开启监听，然后点击【下一步】。



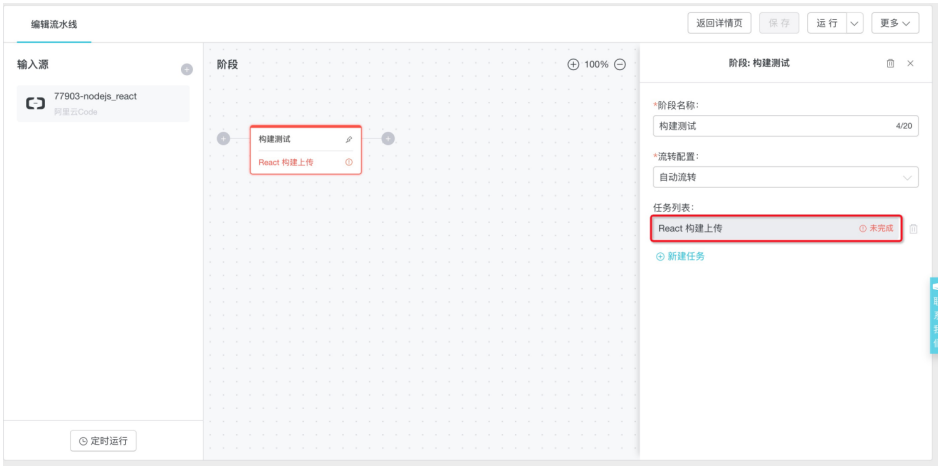
编辑流水线

云效会识别代码库语言并推荐相应流水线模板，使用默认置顶选中的【React 构建上传到 Oss】流水线模板，然后点击【创建】。

填写流水线名称，点击【下一步】。



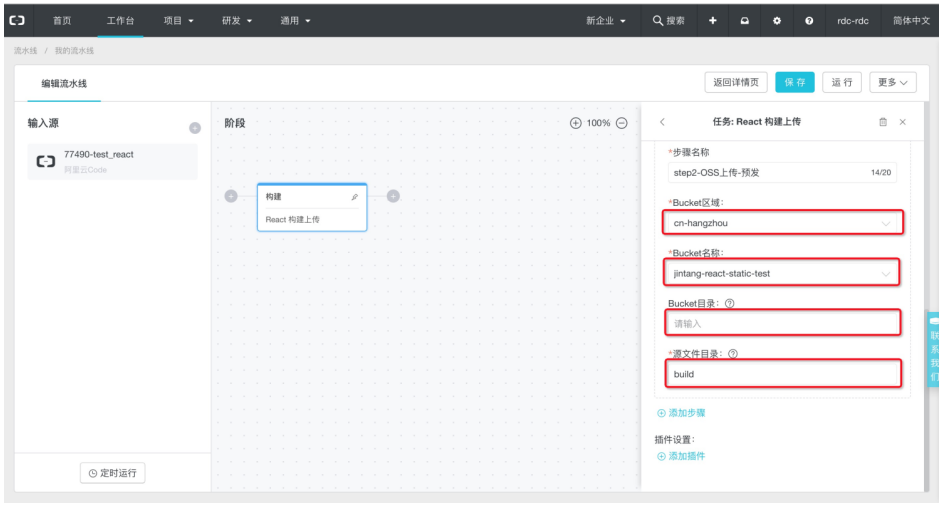
阶段编辑。点击【构建测试】阶段，在右侧浮层中点击【React 构建上传】任务。



授权 Oss，点击【前往授权绑定】进入 RAM 授权界面。

打开账号绑定开关，点击确定。

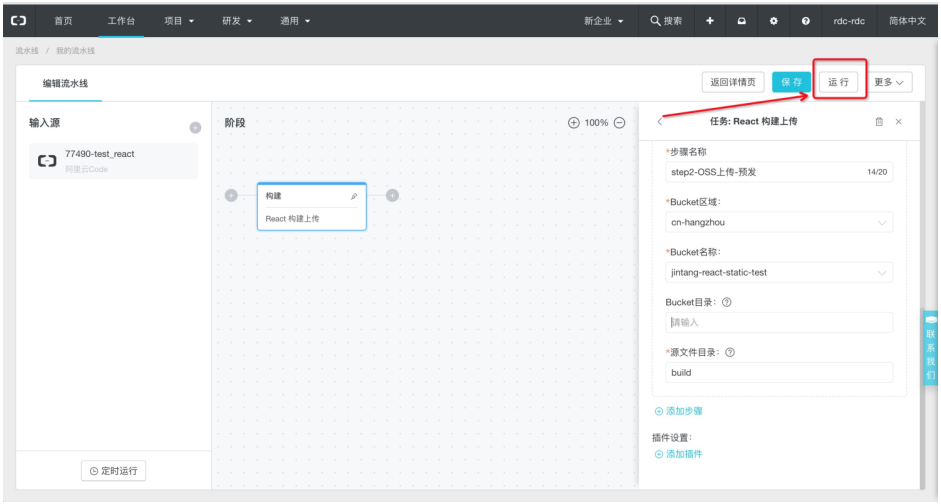
配置静态文件上传。回到流水线编辑页面，选填下图所示四个方框。 如下图所示，即将代码库中构建生成的 build 目录下的所有文件上传至 hangzhou 区域的 jintang-react-static-test bucket 的根



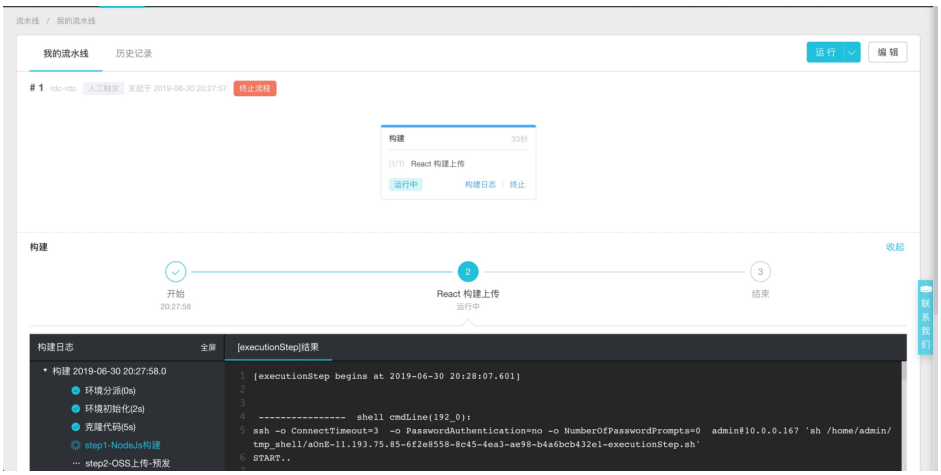
路径下。

运行流水线

点击流水线编辑页面右上角【运行】，流水线会自动保存并开始运行。

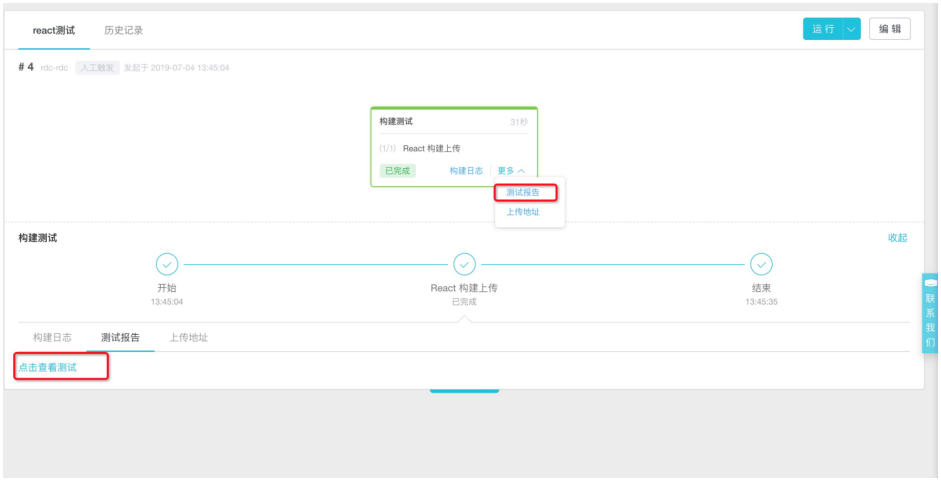


流水线运行中。



查看测试报告

点击【构建测试】阶段中的【测试报告】按钮可以查看测试报告。



查看部署结果

流水线运行成功后，点击阶段中【更多】->【上传地址】可以查看上传到目标 bucket 的文件。

Golang入门

Golang应用部署到ECS

本文档会帮助您在云效创建一个 Golang Gin 的代码库， 并发布到阿里云 ECS 服务器。

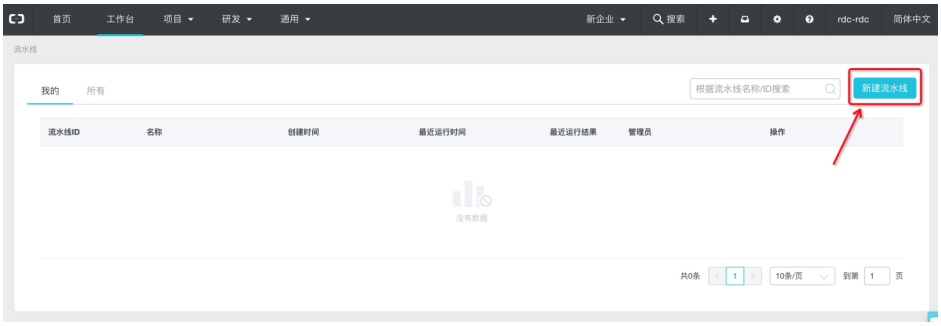
创建企业

首次进入云效，会提示您创建企业。

创建流水线

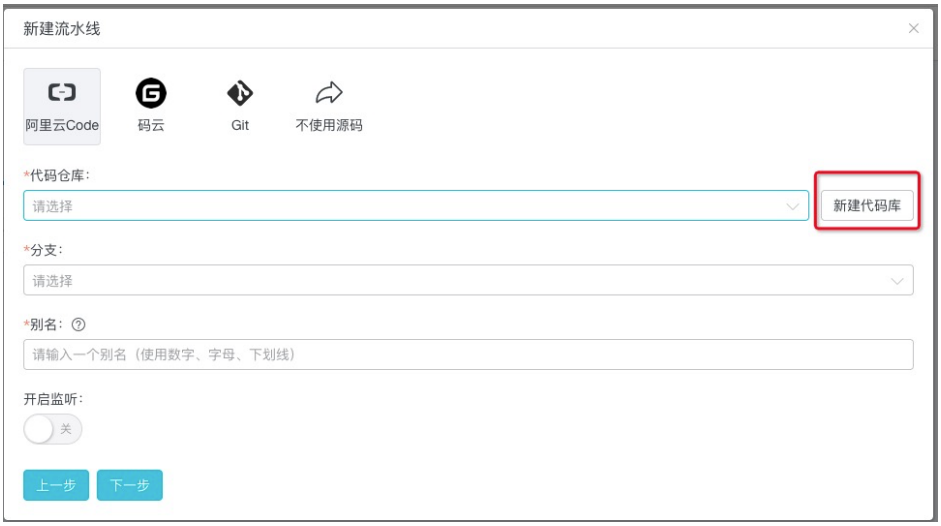
进入企业后，从页面顶栏点击【研发】->【流水线】，进入流水线列表。

点击右上角【新建流水线】，进入流水线创建向导页面。

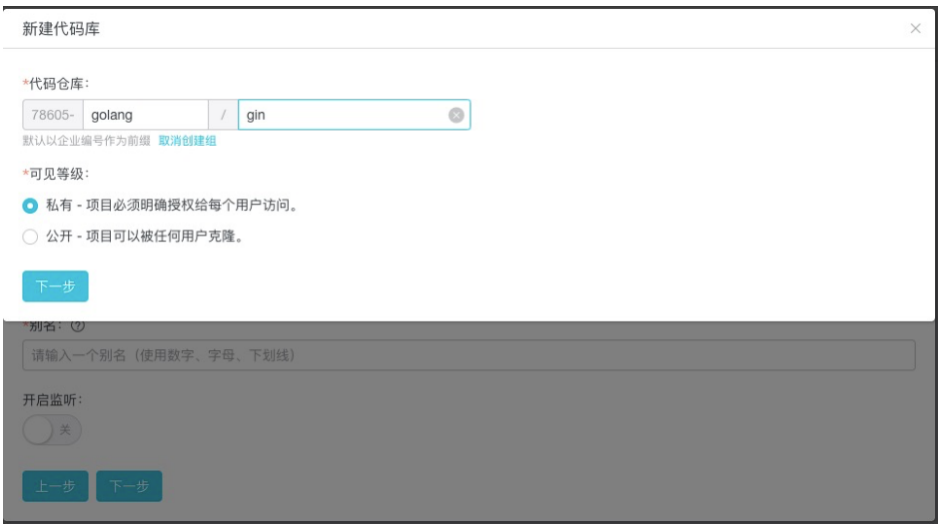


新建代码库

在源码设置页面中，选择阿里云Code，点击代码仓库右侧【新建代码库】。

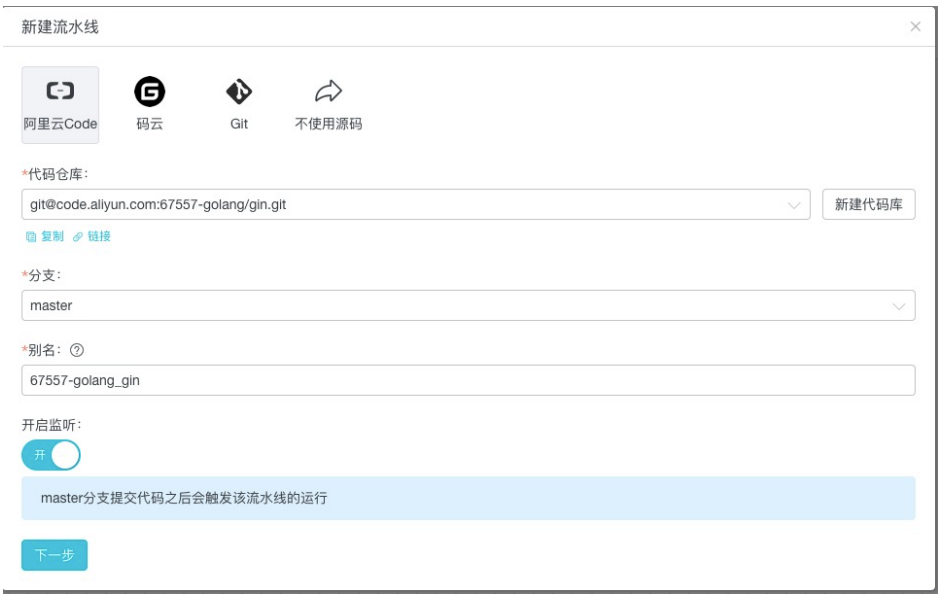


在弹窗中选择或创建代码组，输入新建代码仓库名称，点击【下一步】。



在代码模板中选择 Golang 和 golang-gin-web ，取消勾选【生成Dockerfile】，点击【确认】。您可以点击代码模板图标下的预览按钮查看即将生成的代码库文件内容。

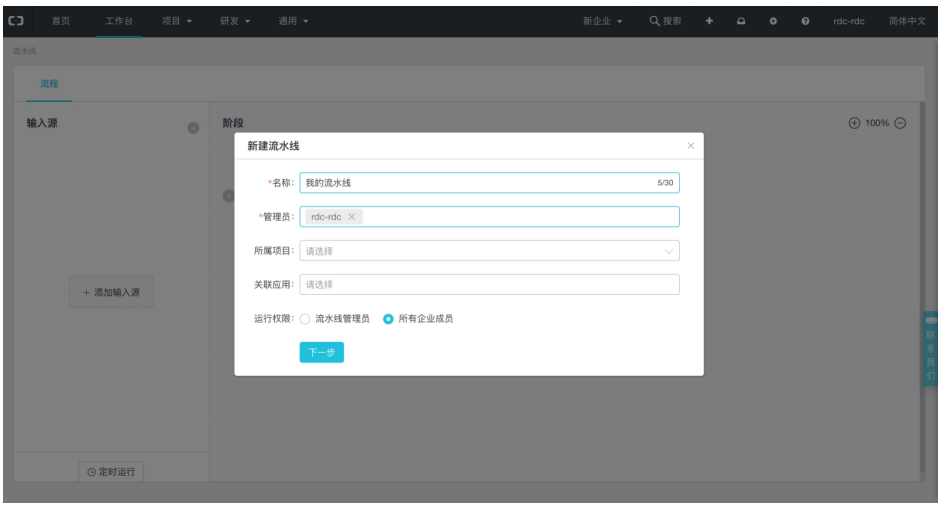
代码库创建完成后会恢复到代码库选择页面并回填刚才创建代码库的信息，为了在代码提交时候触发持续集成，打开【开启监听】。然后点击【下一步】。



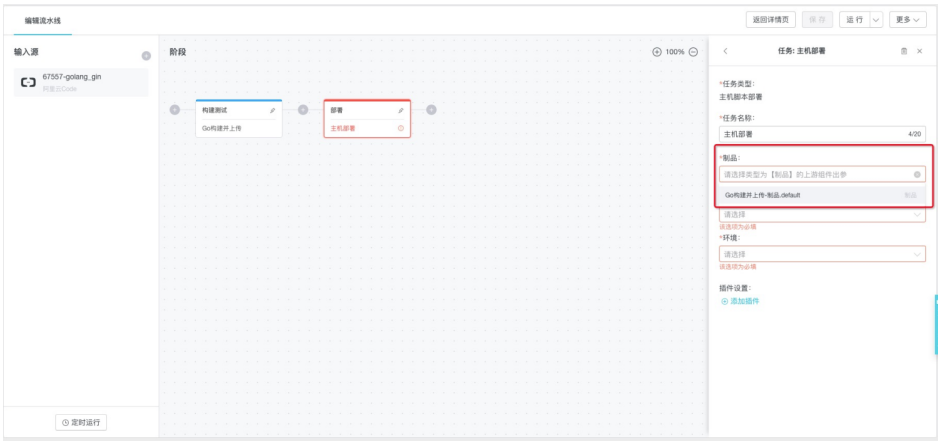
编辑流水线

云效会识别代码库语言并推荐相应流水线模板，使用默认置顶选中的【Go测试、构建、部署到主机】流水线模板，然后点击【创建】。

填写流水线名称，点击【下一步】。

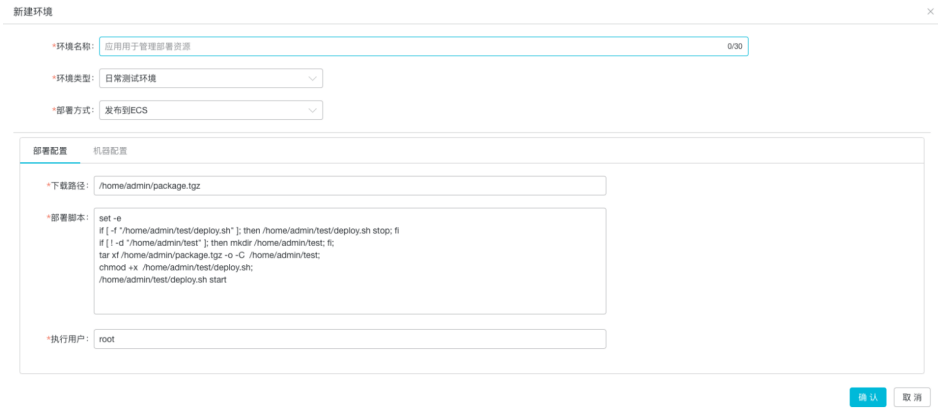


部署配置。点击阶段【部署】进行部署配置。在任务列表中选择【主机部署】，选择由构建环节产出的待部署【制品】。

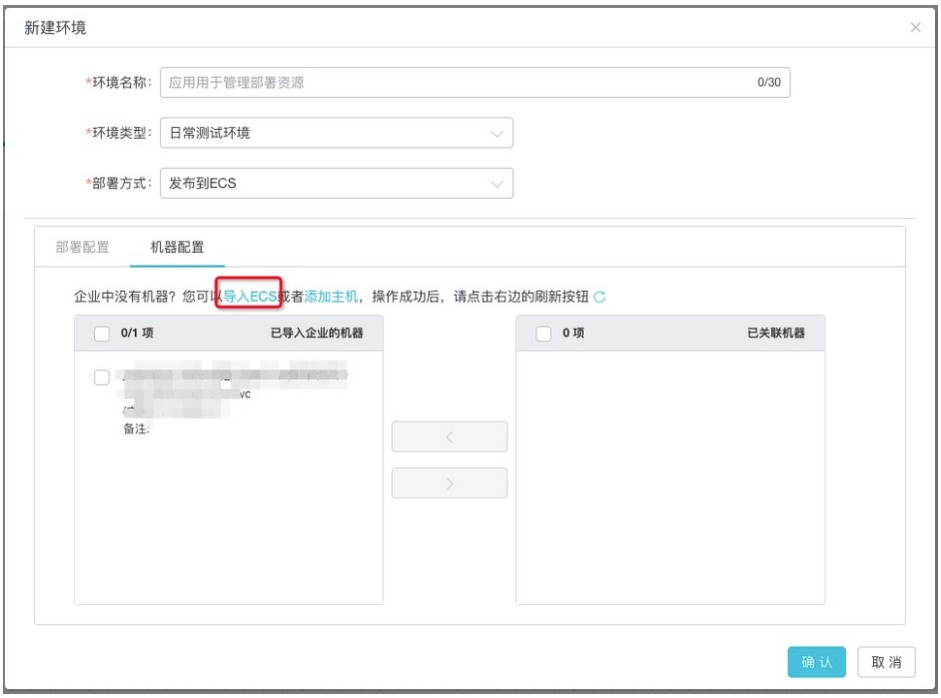


新建应用。在【应用】下拉框中选择【新建应用】填写应用名称，点击【确认】。

新建环境。在【环境】下拉框中选择【新建环境】，填写环境名称，点击【机器配置】。



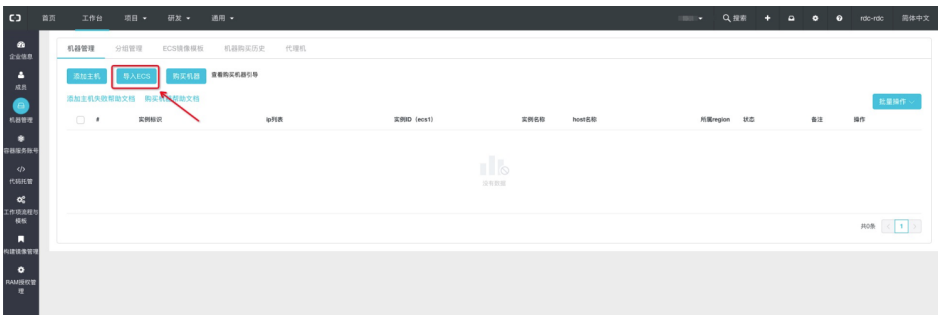
进行机器配置，点击【导入ECS】链接，跳转到导入 ECS 界面。



导入 ECS 机器

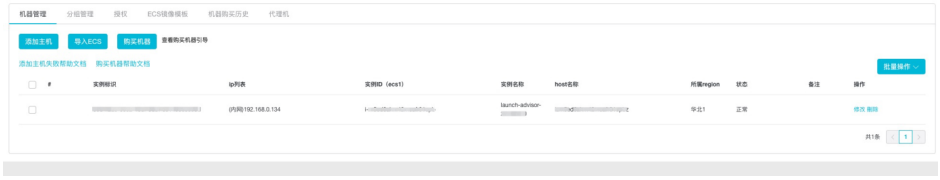
1. 申请一台阿里云ECS服务器，具体方式参见 ECS 文档，请确保该ECS环境上预装了 go 并有公网IP，或者有SLB与之关联，以便进行功能验证。

   在机器管理页面，点击【导入 ECS】。

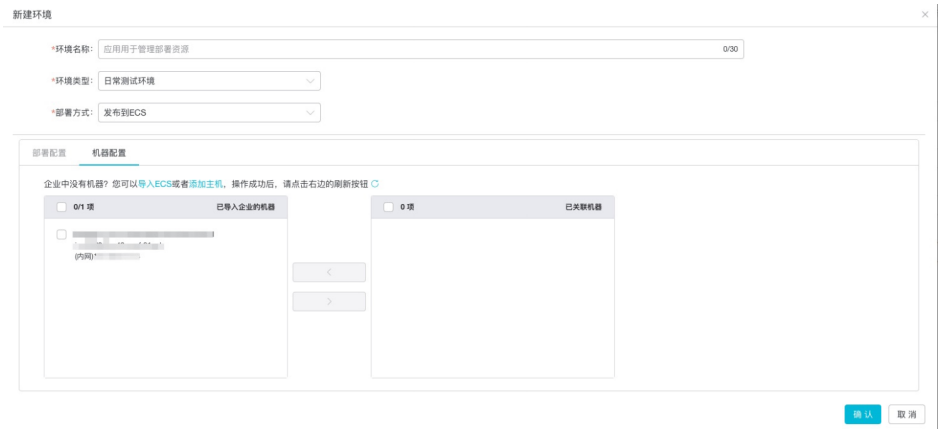


在弹出框中选择新建 ECS 所在区域，选择实例，点击【导入】。

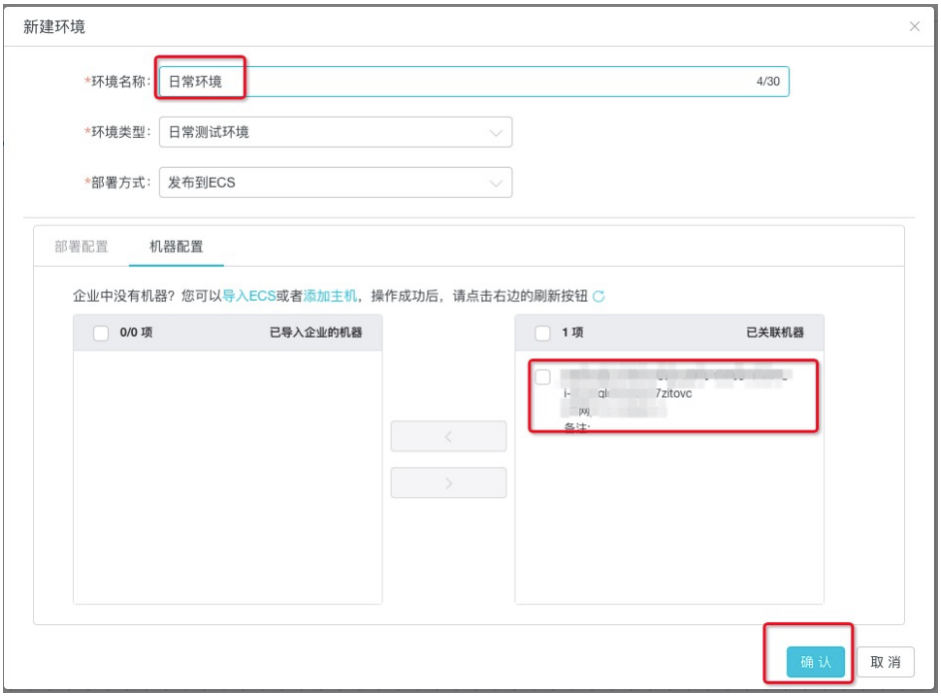
等待大概1分钟左右，刷新页面，可以看到机器状态为【正常】。



机器导入后回到【机器配置】页面，点击【刷新】按钮，可以看到刚才导入的机器。

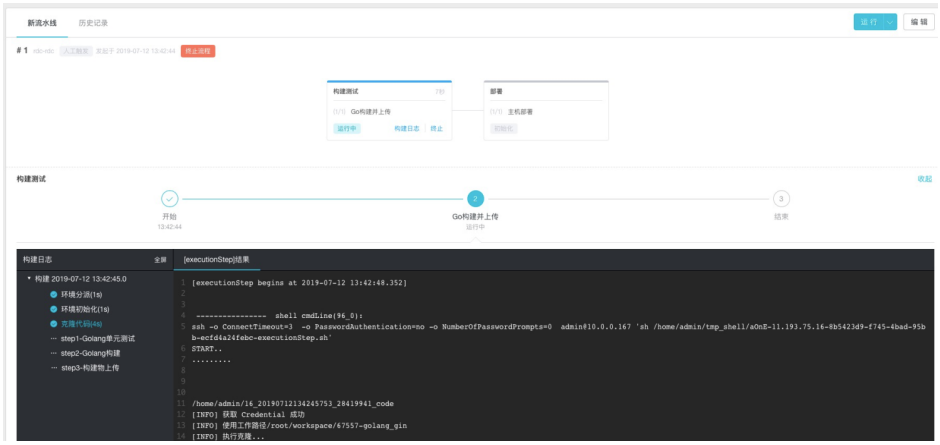


选择机器，并将其加入到右侧列表，填写【环境名称】然后点击【确认】保存。



运行流水线

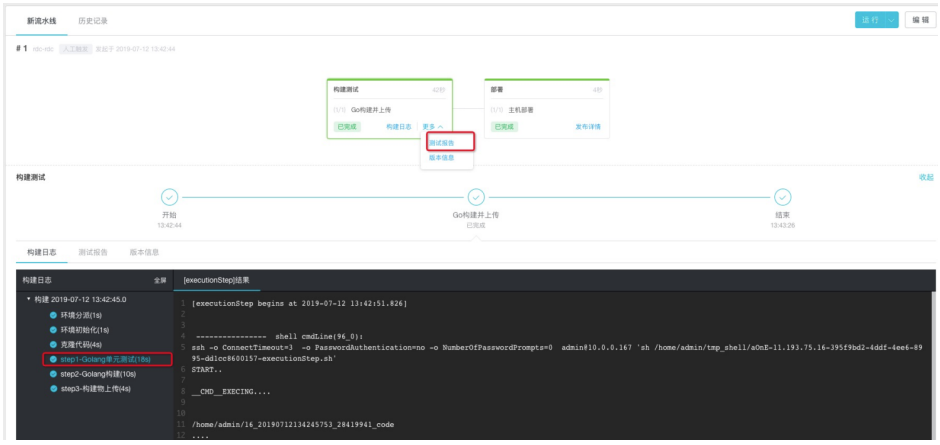
此时流水线已经编辑完成，点击右上角的【运行】，即可保存并触发流水线。



运行成功。

查看测试报告

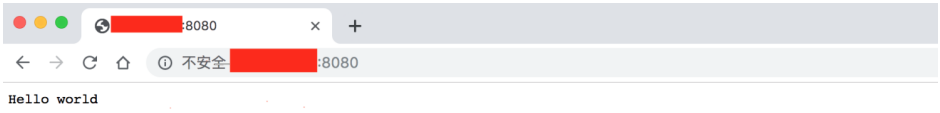
选中【构建测试】阶段，可以查看单元测试运行日志和【测试报告】。



查看部署结果

点击【发布详情】-> 【打开发布单】可以查看部署单。在部署单中可以查看每台机器的部署情况和日志。

访问 ECS 的公网 IP 的 8080 端口，可以看到服务运行起来了。



Golang应用部署到Kubernetes

本文档会帮助您在云效创建一个 Golang Gin 的代码库，并部署到云效提供的使用 Kubernetes 集群。

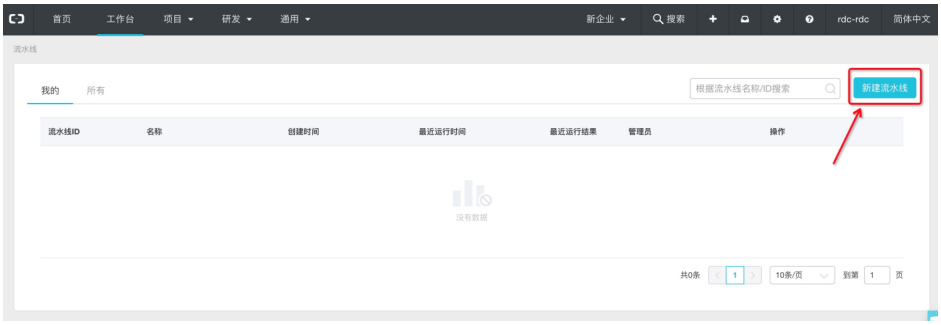
创建企业

首次进入云效，会提示您创建企业。

创建流水线

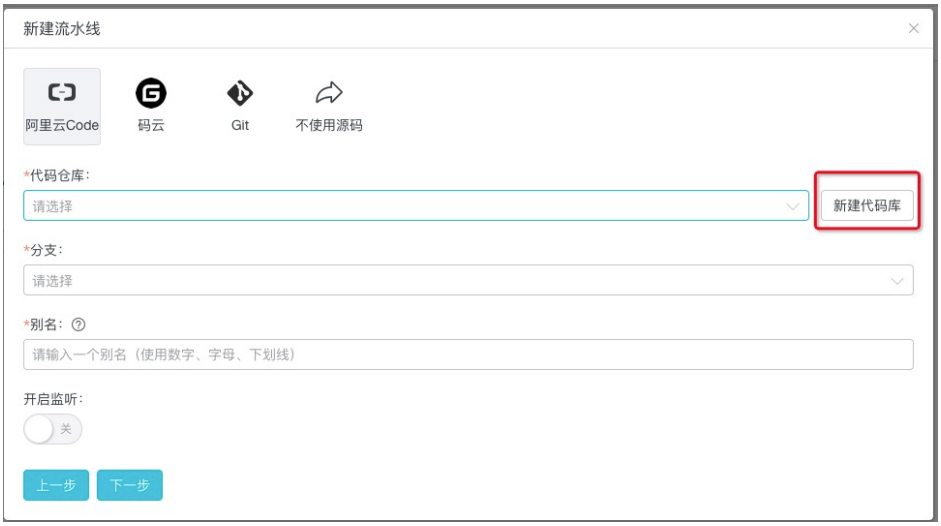
进入企业后，从页面顶栏点击【研发】->【流水线】，进入流水线列表。

点击右上角【新建流水线】，进入流水线创建向导页面。

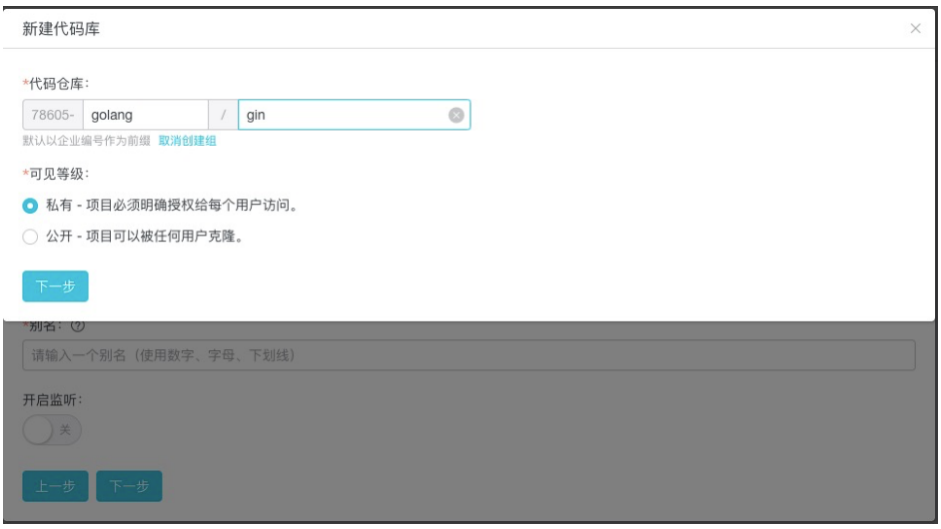


新建代码库

在源码设置页面中，选择阿里云Code，点击代码仓库右侧【新建代码库】。

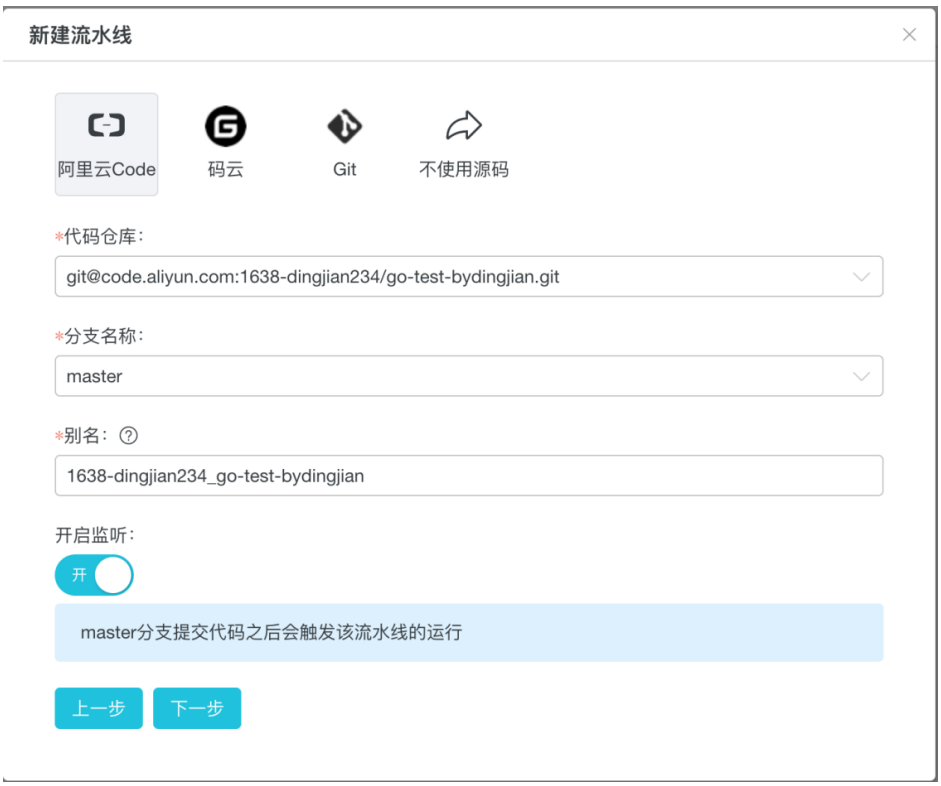


在弹窗中选择或创建代码组，输入新建代码仓库名称，点击【下一步】。



在弹框中 Golang 和 golang-gin-web，选中【生成Dockerfile】,然后点击【确认】。您可以点击代码模板图标下的预览按钮查看即将生成的代码库文件内容。

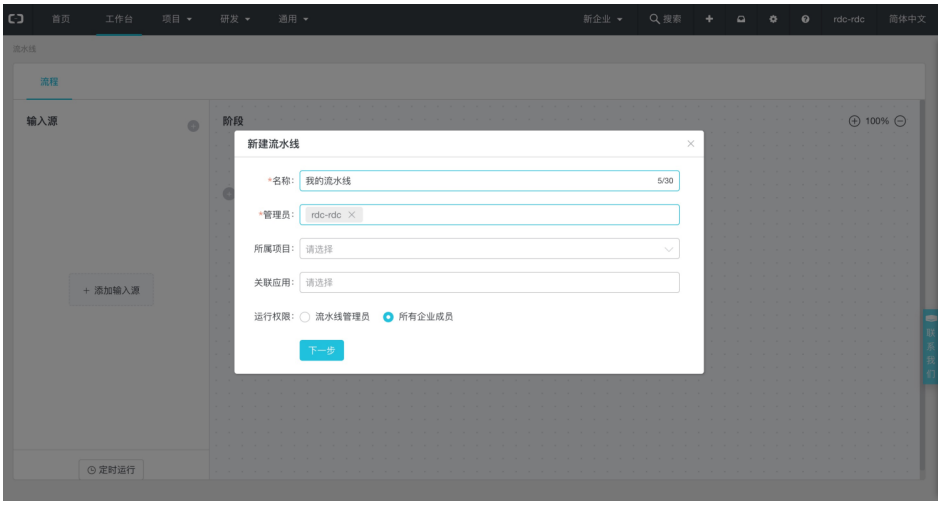
为了在代码提交时候触发持续集成，打开【开启监听】。然后点击【下一步】。



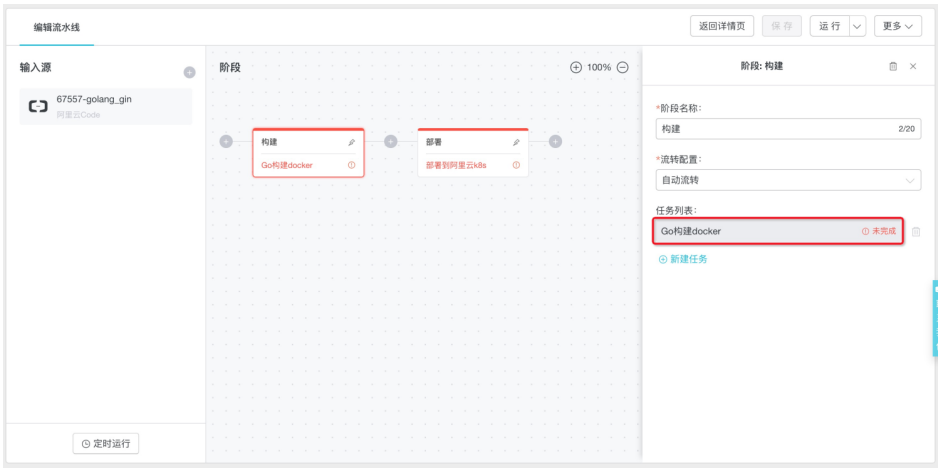
编辑流水线

云效会识别代码库语言并推荐相应流水线模板，使用默认置顶选中的【Go测试、构建、部署到主机】流水线模板，然后点击【创建】。

填写流水线名称，点击【下一步】。

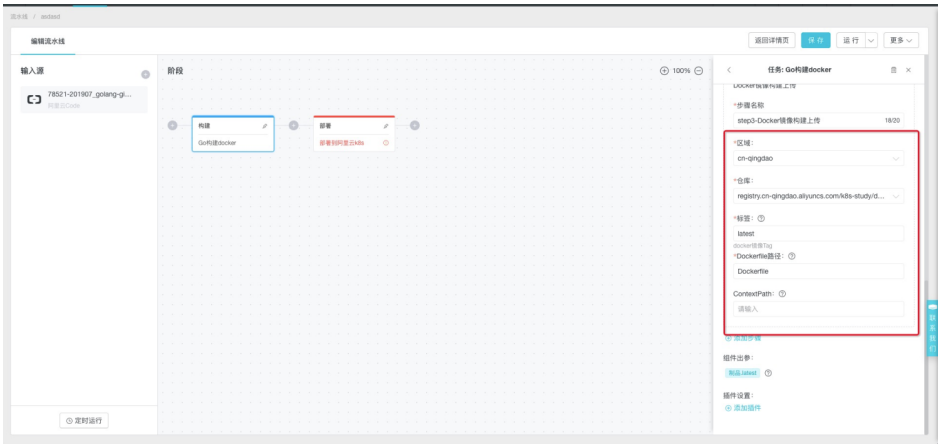


完善测试构建阶段配置。点击【构建】阶段，在右侧浮层中点击【Go构建docker】任务，在任务编辑页面中点击【请前往授权绑定】的链接完成 RAM 授权。然后回到流水线配置页面选择【区域】。

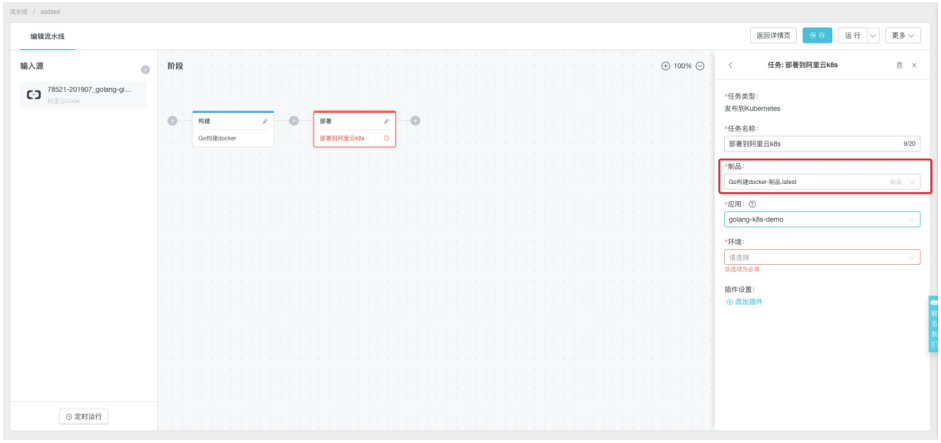


在【仓库】的下拉框中点击【新建镜像仓库】，填写好信息之后，点击【确认】，窗口会自动关闭，并把新创建好的仓库地址回填到流水线中。

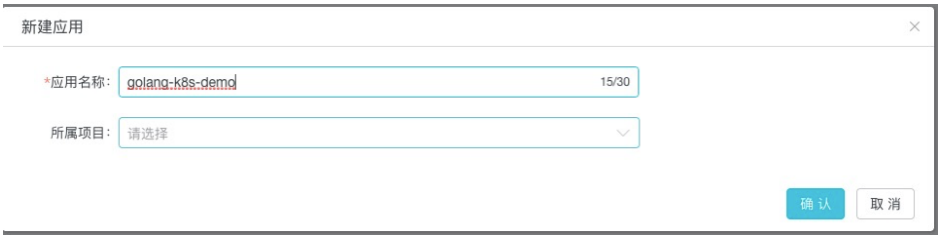
填写【标签】，使用 ${TIMESTAMP} 作为标签，便于版本管理。



进行部署配置。在流水线的部署任务中选择【制品】，即在构建阶段产生的那个镜像。



在【应用】下拉框中点击【新建应用】，输入应用名，点击【确认】。



新建环境。在【环境】下拉框中点击【新建环境】，进入到 Kubernetes 集群配置页面。

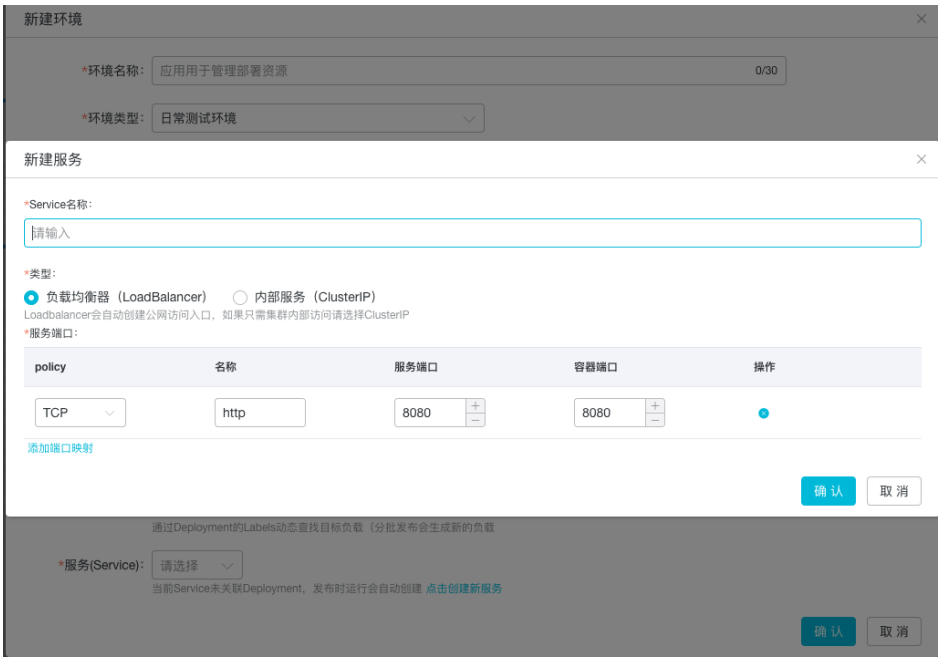
创建免费集群

在【新建环境】的弹框中进行 Kubernetes 部署配置。点击【免费试用云效提供的集群】来获取一个临时的免费集群，该集群会自动配置到您的企业中，在5小时后会失效并从企业中删除。

在使用免费集群的弹窗中记得点击【点击复制kubeconfig】，并进行本地配置，否则将无法从本地访问集群。

关闭弹框，选择集群、命名空间，选择滚动发布，并点击【点击创建新服务】的链接来创建Service。

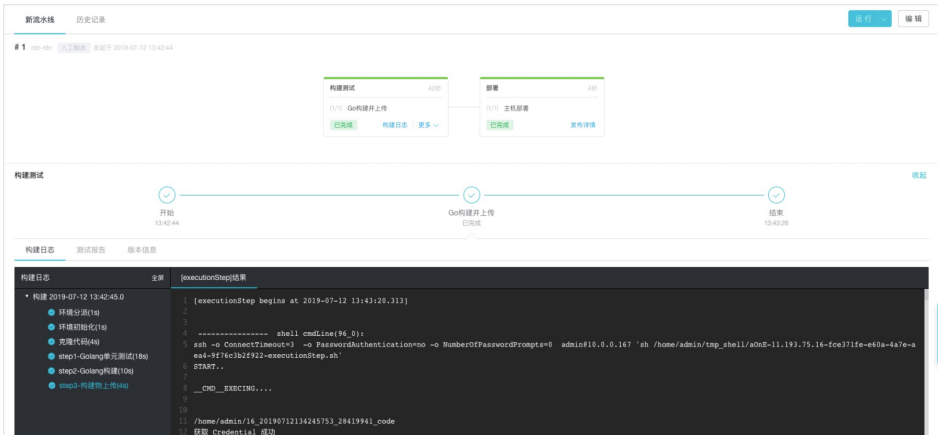
输入 Service 名称，选择负载均衡器类型，服务端口和容器端口都设置为 8080，然后点击【确认】。弹框关闭后，再点击创建环境的弹框的【确认】，环境即创建成功，并回填到流水线配置中。



运行流水线

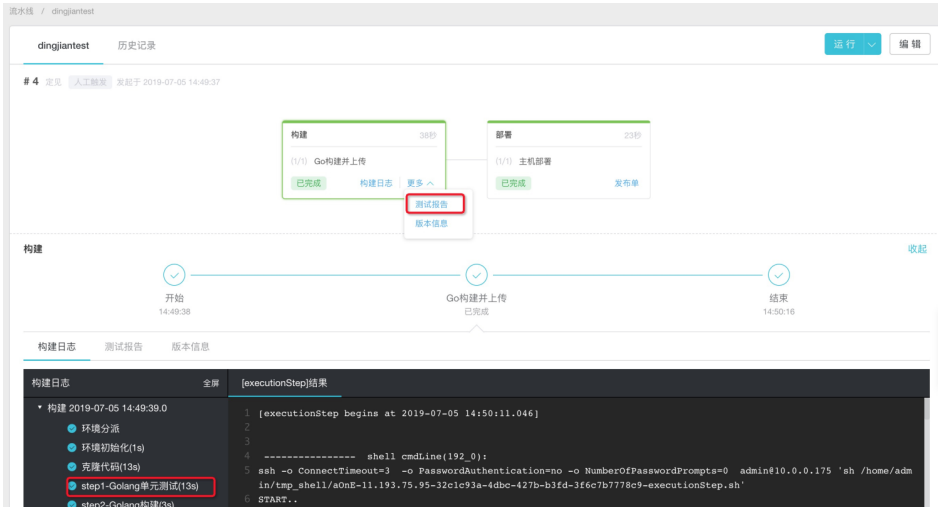
点击流水线编辑页面右上角【运行】，流水线会自动保存并开始运行。

运行成功。



查看测试报告

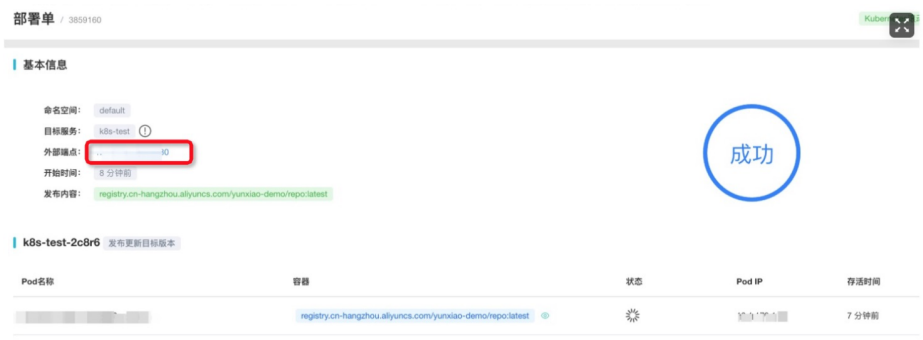
点击构建阶段【更多】-> 【测试报告】可以展开测试结果预览，点击预览里的【链接】可以查看测试报告详情。【版本信息】里的下载链接可以用于下载软件包。



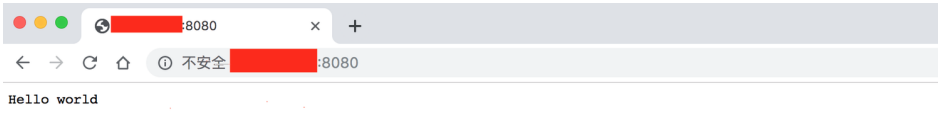
查看部署结果

点击【发布详情】-> 【打开发布单】可以查看部署单。在部署单中可以查看每台机器的部署情况和日志。

在部署单中可以看到该服务所对应的路由的地址。点击【外部端点】可以访问到该应用。



由于 Service 类型是 LoadBalancer，所以会自动生成一个公网IP来暴露这个 Service。显示在部署单的外部端点部分。访问【外部端点】地址，可以看到服务运行起来了。



Python入门

Python应用部署到ECS

本文档会帮助您在云效创建一个 Python3 Flask 的代码库，并部署到阿里云 ECS 服务器。

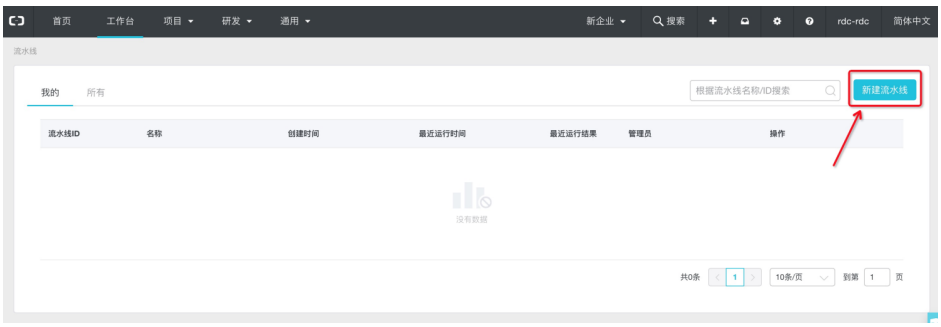
创建企业

首次进入云效，会提示您创建企业。

创建流水线

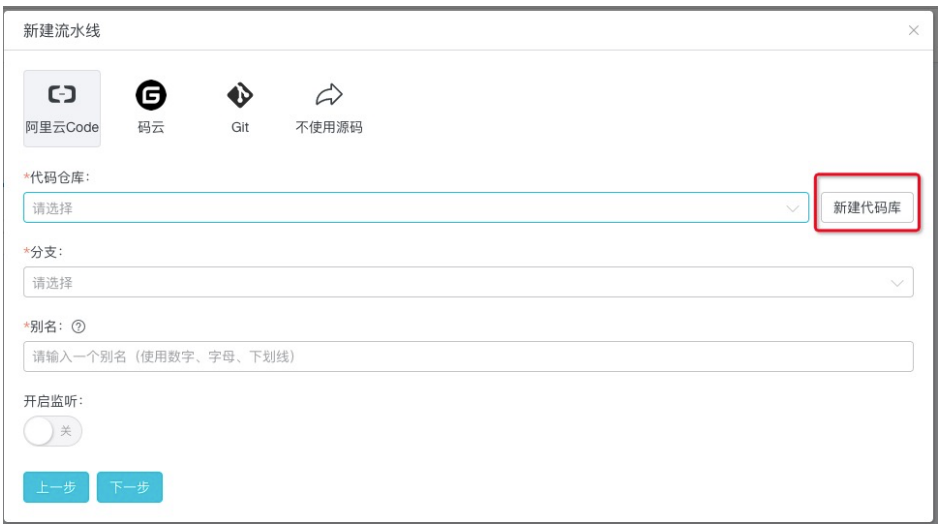
进入企业后，从页面顶栏点击【研发】->【流水线】，进入流水线列表。

点击右上角【新建流水线】，进入流水线创建向导页面。

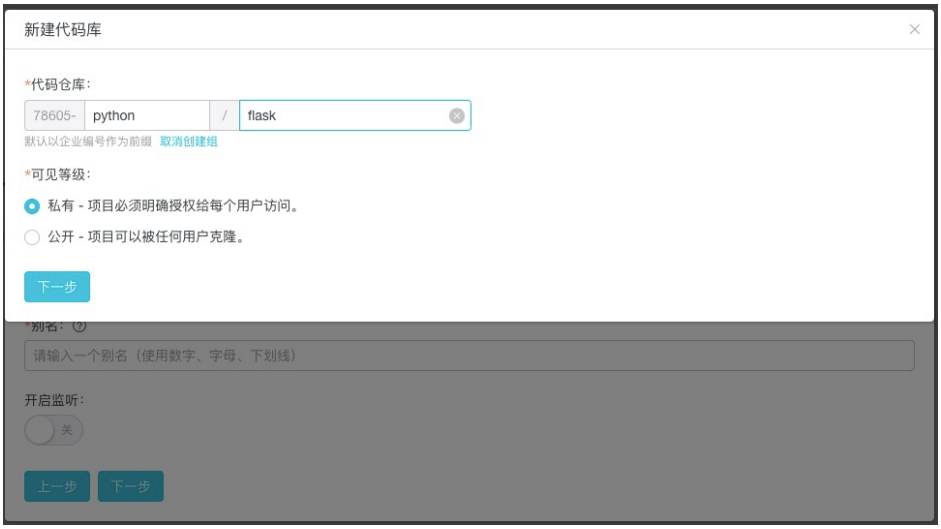


新建代码库

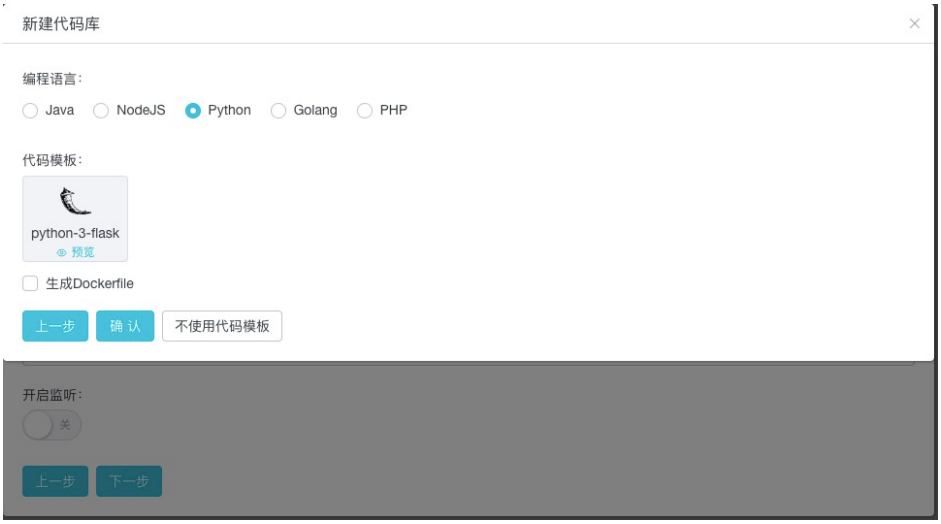
在源码设置页面中，选择阿里云Code，点击代码仓库右侧【新建代码库】。



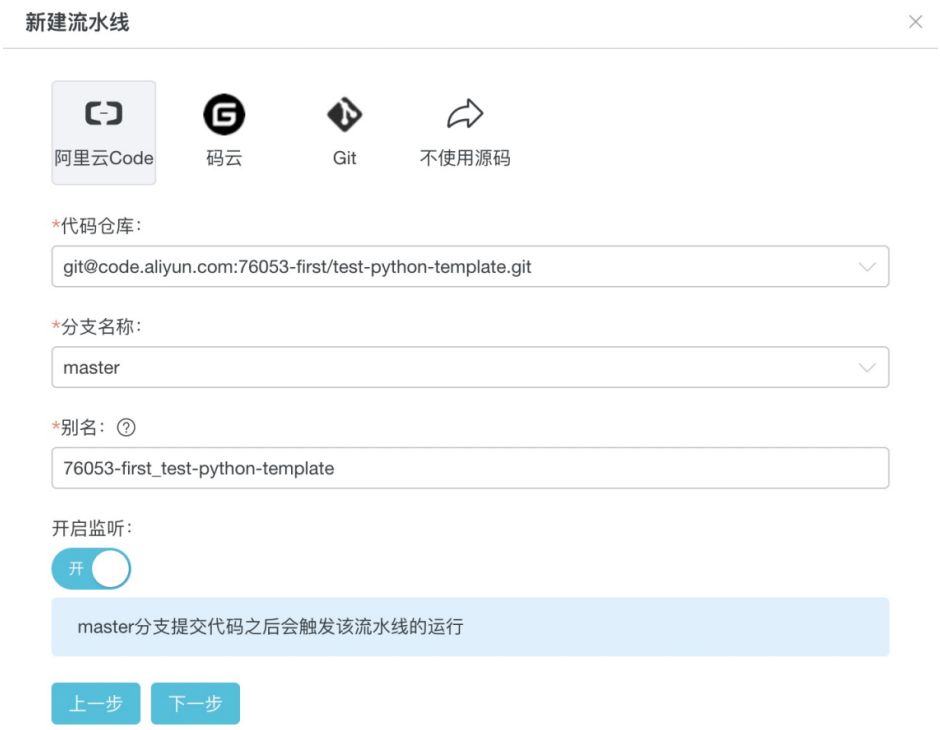
在弹窗中选择或创建代码组，输入新建代码仓库名称，点击【下一步】。



在代码模板中选择 Python 和 python-3-flask ，取消勾选【生成Dockerfile】，点击【确认】。您可以点击代码模板图标下的预览按钮查看即将生成的代码库文件内容。



代码库创建完成后会恢复到代码库选择页面并回填刚才创建代码库的信息，为了在代码提交时候触发持续集成，打开【开启监听】。然后点击【下一步】。

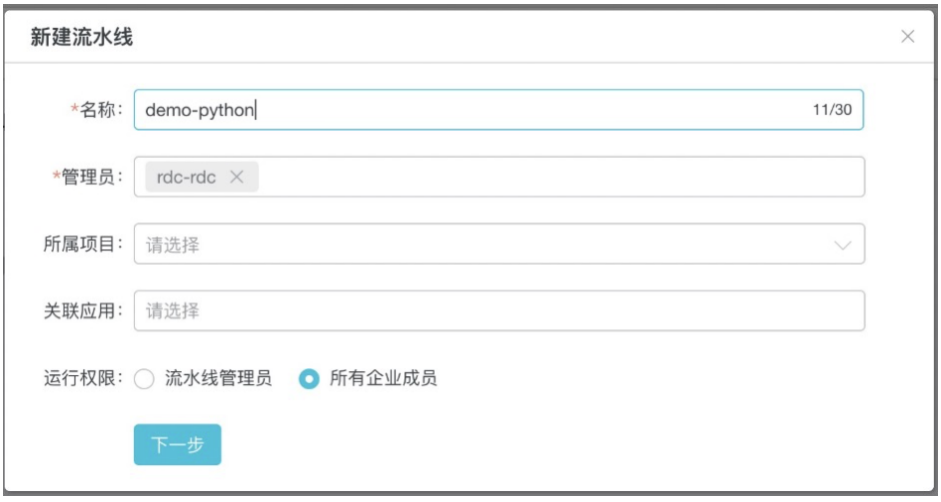


编辑流水线

云效会识别代码库语言并推荐相应流水线模板，使用默认置顶选中的【Python测试，获取代码版本

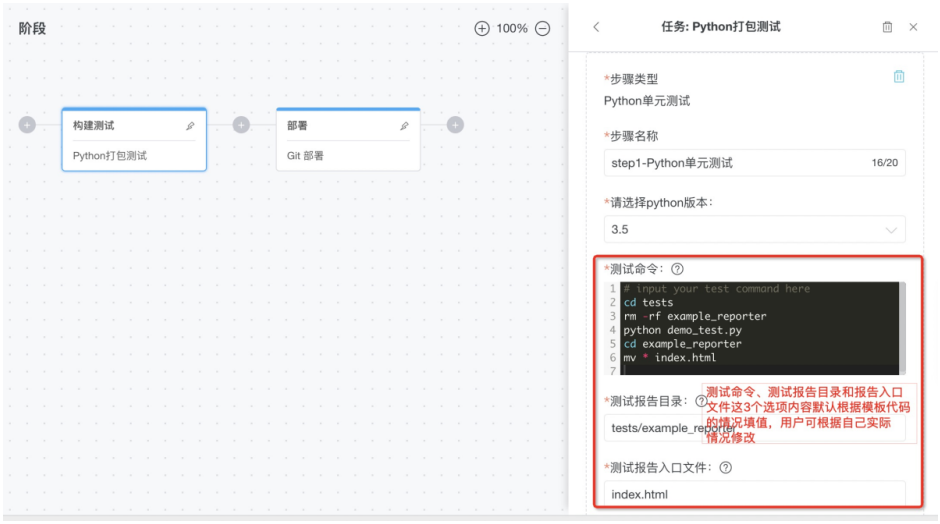
信息，部署到ECS】流水线模板，然后点击【创建】。

填写流水线名称，点击【下一步】。

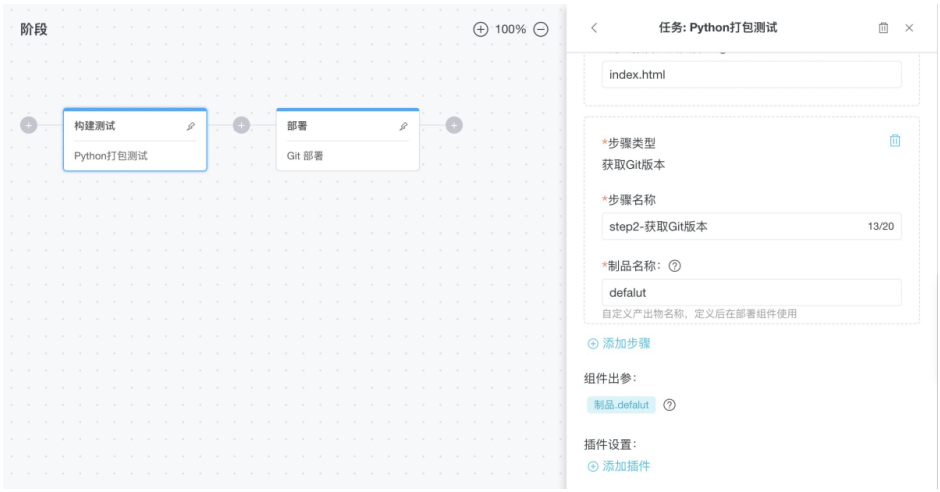


进入流水线页面，构建测试任务包含两个步骤：

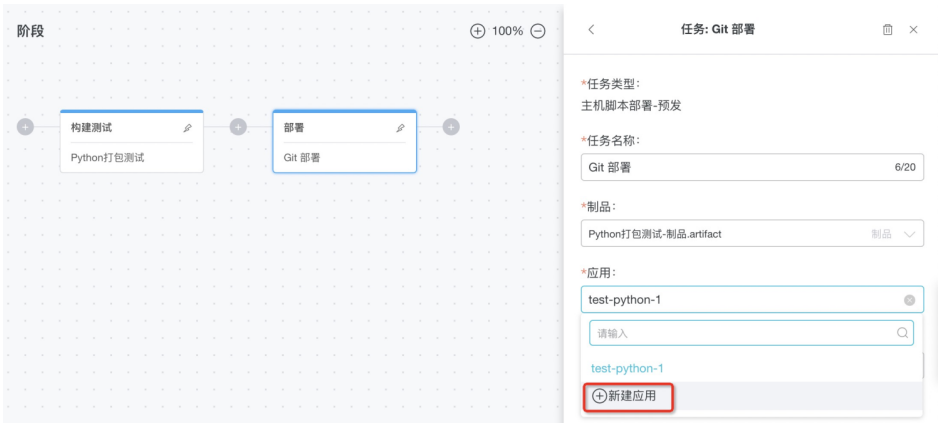
- Python单元测试；
- 获取Git版本。



部署配置。点击阶段【部署】进行部署配置。在任务列表中选择【主机部署】，选择由获取版本环节产出的待部署【制品】。



新建应用。在【应用】下拉框中选择【新建应用】填写应用名称，点击【确认】。



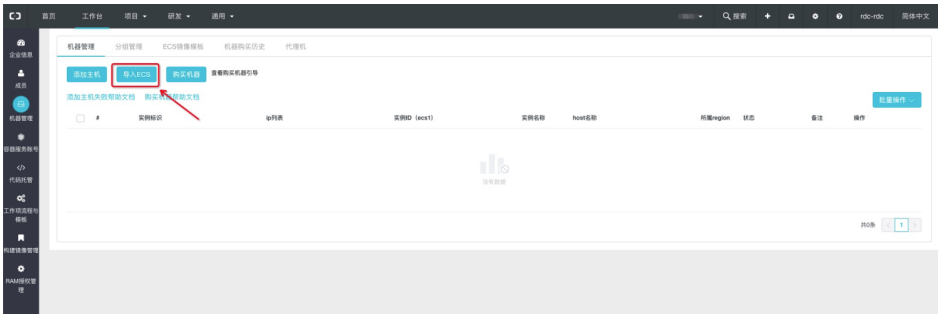
新建环境。在【环境】下拉框中选择【新建环境】，填写环境名称，点击【机器配置】。

进行机器配置，点击【导入ECS】链接，跳转到导入 ECS 界面。

导入 ECS 机器

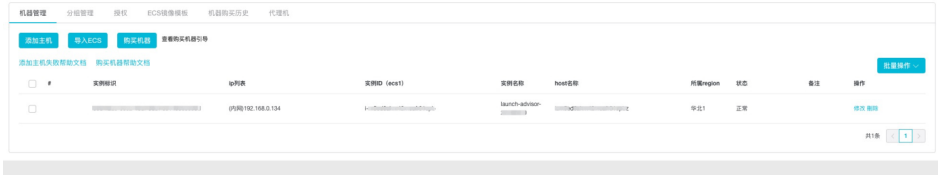
1. 申请准备好阿里云ECS，具体方式参见 ECS 文档，请确保该ECS环境上预装了 python、pip 并有公网IP，或者有SLB与之关联，以便进行功能验证。

   在机器管理页面，点击【导入 ECS】。

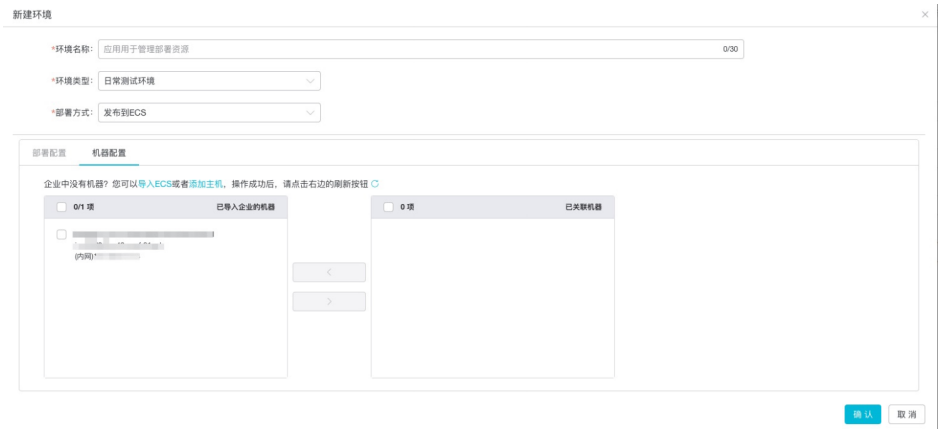


   在弹出框中选择新建 ECS 所在区域，选择实例，点击【导入】。

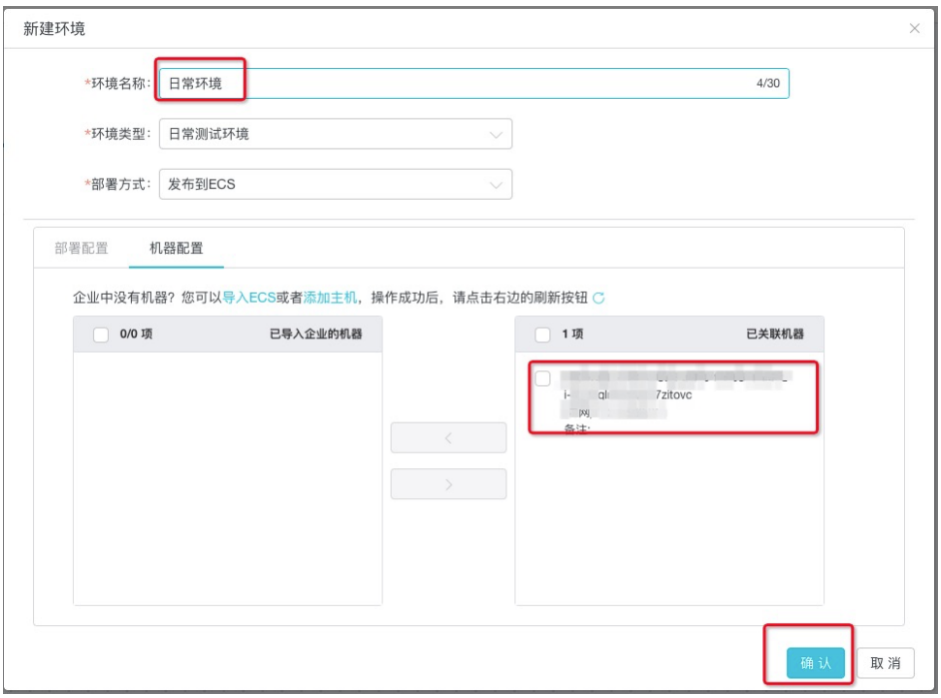
等待大概1分钟左右，刷新页面，可以看到机器状态为【正常】。



机器导入后回到【机器配置】页面，点击【刷新】按钮，可以看到刚才导入的机器。



选择机器，并将其加入到右侧列表，填写【环境名称】然后点击【确认】保存。



运行流水线

此时流水线已经编辑完成，点击右上角的【运行】，即可保存并触发流水线。

查看测试报告

选中【构建测试】阶段，可以查看单元测试运行日志和【测试报告】。

查看部署结果

访问 ECS 的公网 IP 的8080端口，可以看到服务运行起来了。

Hello World

Python应用部署到Kubernetes

本文档会帮助您在云效创建一个 Python3 Flask 的代码库，并部署到云效提供的使用 Kubernetes 集群。

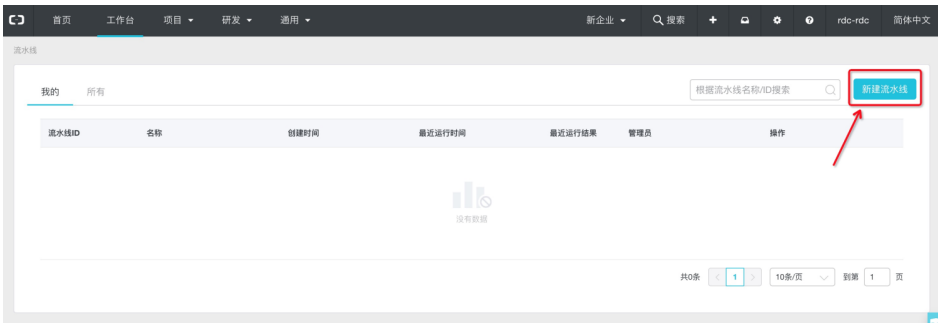
创建企业

首次进入云效，会提示您创建企业。

创建流水线

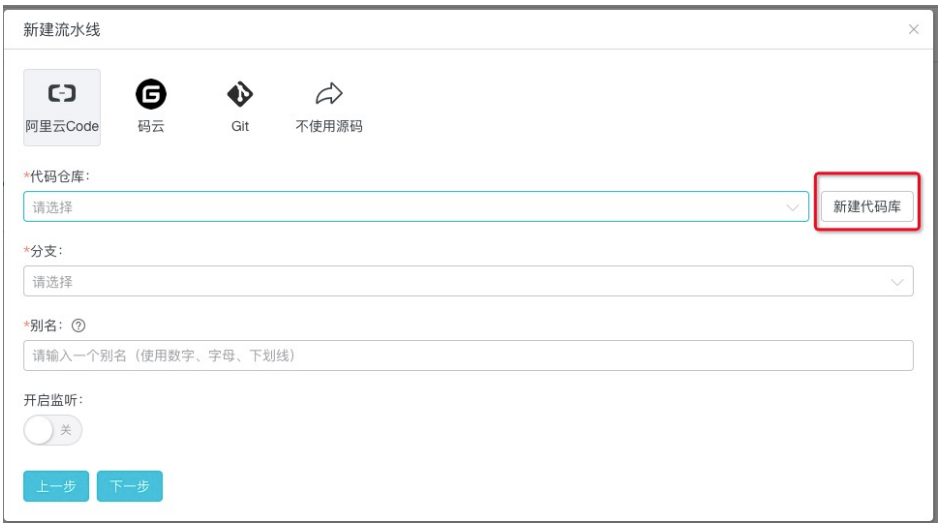
进入企业后，从页面顶栏点击【研发】->【流水线】，进入流水线列表。

点击右上角【新建流水线】，进入流水线创建向导页面。

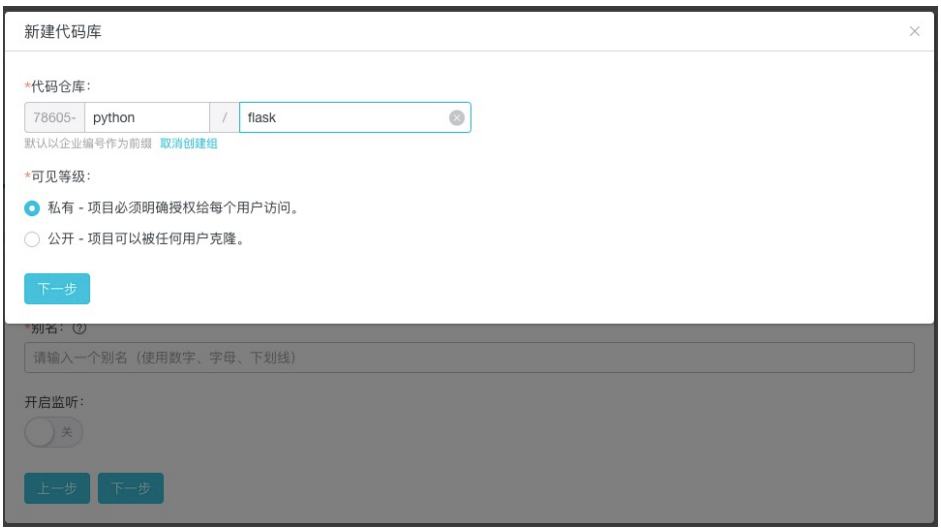


新建代码库

在源码设置页面中，选择阿里云Code，点击代码仓库右侧【新建代码库】。

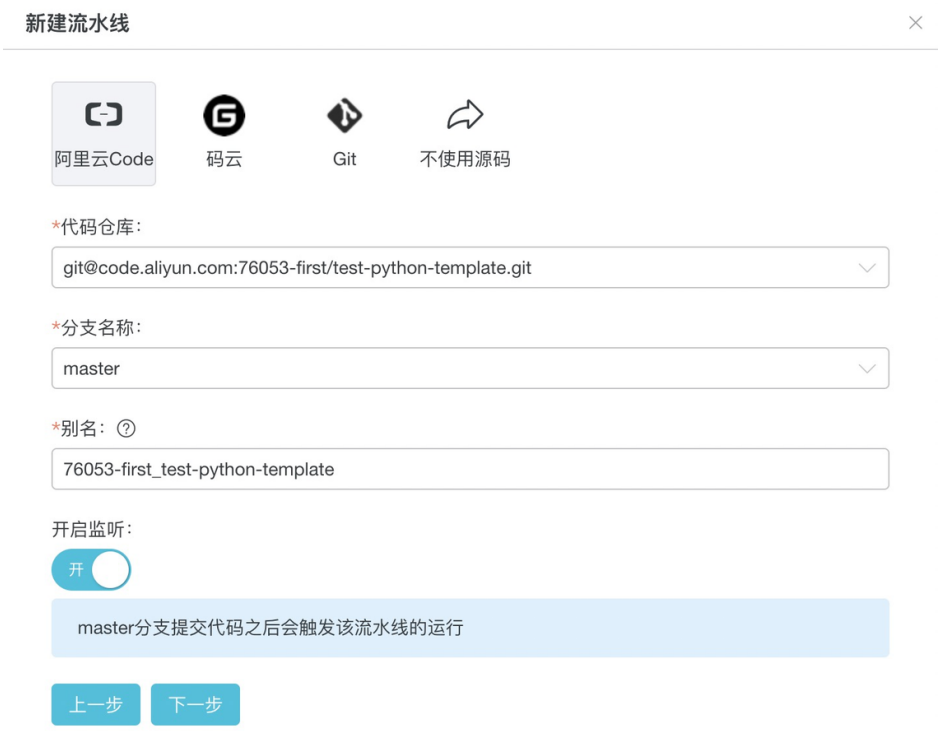


在弹窗中选择或创建代码组，输入新建代码仓库名称，点击【下一步】。



在弹框中选择 Python 语言和 python-3-flask 模板，勾选【生成Dockerfile】，然后点击确认。

为了在代码提交时候触发持续集成，打开【开启监听】。然后点击【下一步】。



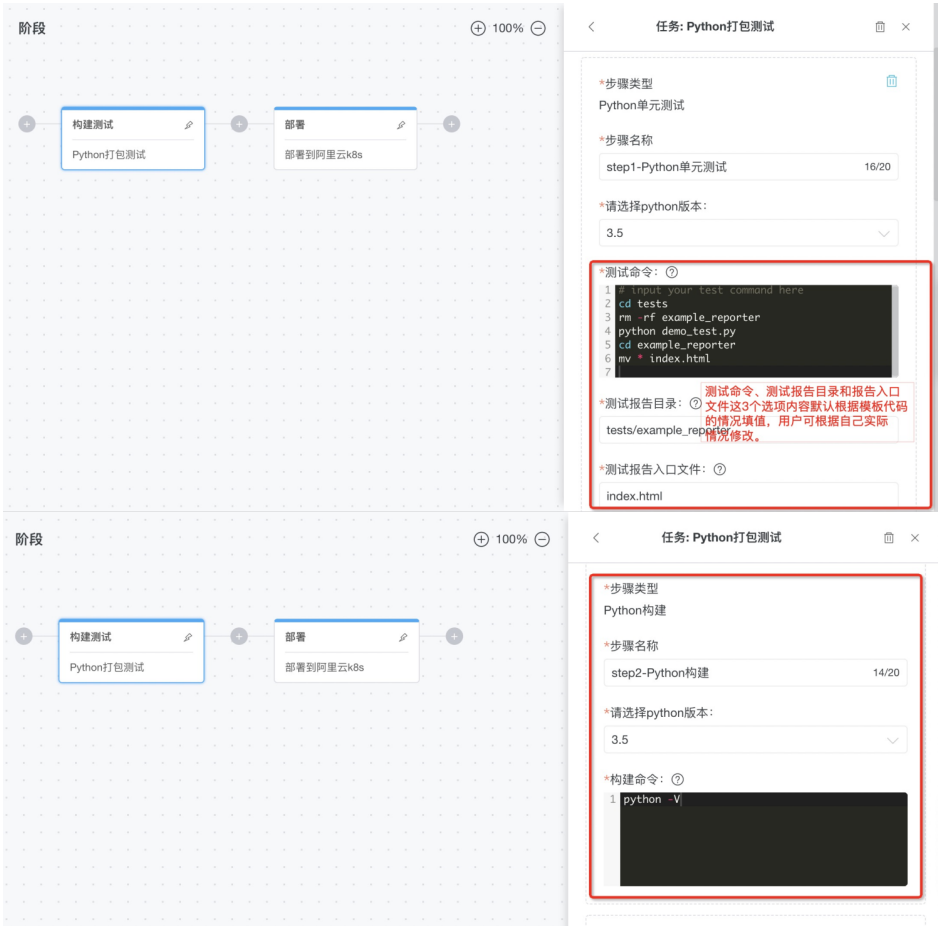
编辑流水线

云效会识别代码库语言并推荐相应流水线模板，使用默认置顶选中的【Python测试、构建部署到k8s】流水线模板，然后点击【创建】。

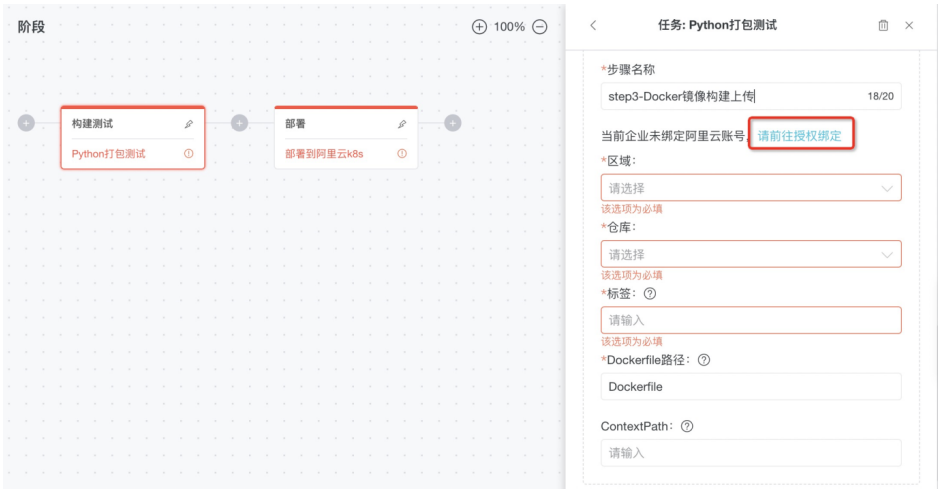
填写流水线名称，点击【下一步】。

进入流水线编辑页面，编辑构建任务。构建测试阶段分为3个步骤:

- Python单元测试。
- 自定义构建需要操作的执行命令。
- 镜像构建以及上传。

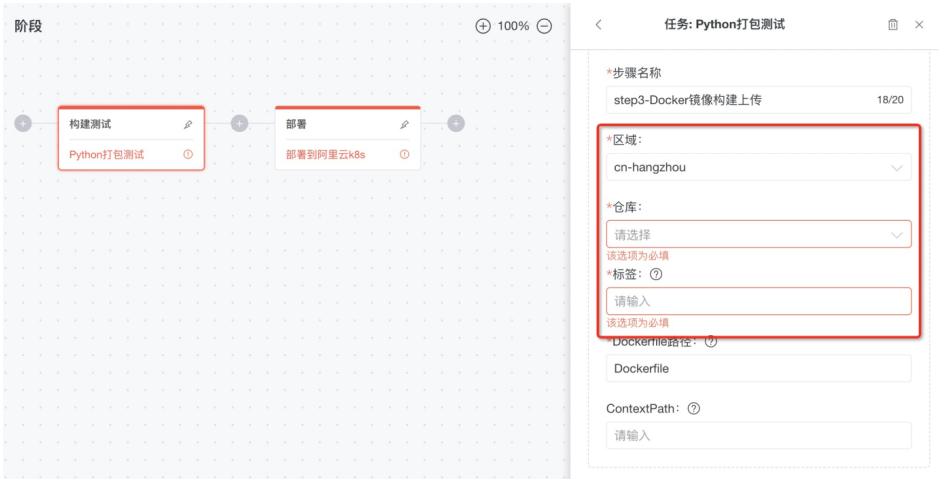


完善测试构建阶段配置。点击【测试构建】阶段，在右侧浮层中点击【Python打包测试】任务，在任务编辑页面中点击【请前往授权绑定】的链接进入 RAM 页面。



RAM 授权。打开账号绑定开关，点击确定。

回到流水线配置页面选择【区域】，在【仓库】的下拉框中点击【新建镜像仓库】，填写好信息之后，点击【确认】，窗口会自动关闭，并把新创建好的仓库地址回填到流水线中。填写【标签】，使用 ${TIMESTAMP} 作为标签，便于版本管理。



进行部署配置。在流水线的部署任务中选择【制品】，即在构建阶段产生的那个镜像。

新建应用。在【应用】下拉框中点击【新建应用】，输入应用名，点击【确认】。

新建环境。在【环境】下拉框中点击【新建环境】，进入到 Kubernetes 集群配置页面。

创建免费集群

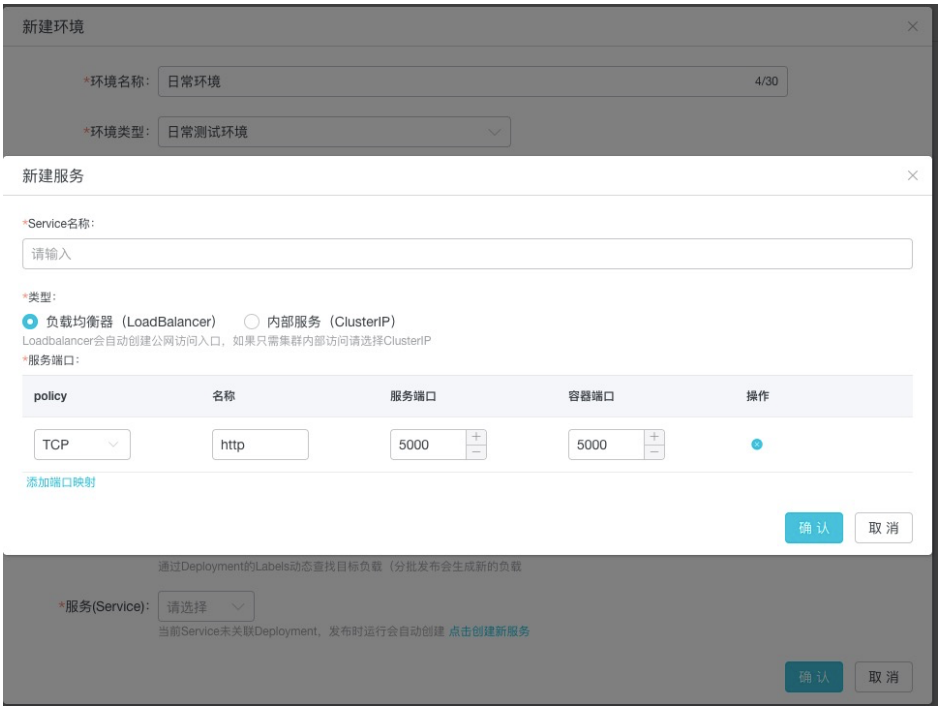
在【新建环境】的弹框中进行Kubernetes部署配置。点击【免费试用云效提供的集群】来获取一个临时的免费集群，该集群会自动配置到您的企业中，在 5 小时后会失效并从企业中删除。

在使用免费集群的弹窗中记得点击【点此复制kubeconfig】，并进行本地配置，否则将无法从本地访问集群。

关闭弹框，选择集群、命名空间，选择滚动发布，并点击【点击创建新服务】的链接来创建 Service。

输入 Service 名称，选择负载均衡器类型，服务端口和容器端口都设置为5000，然后点击确认。弹

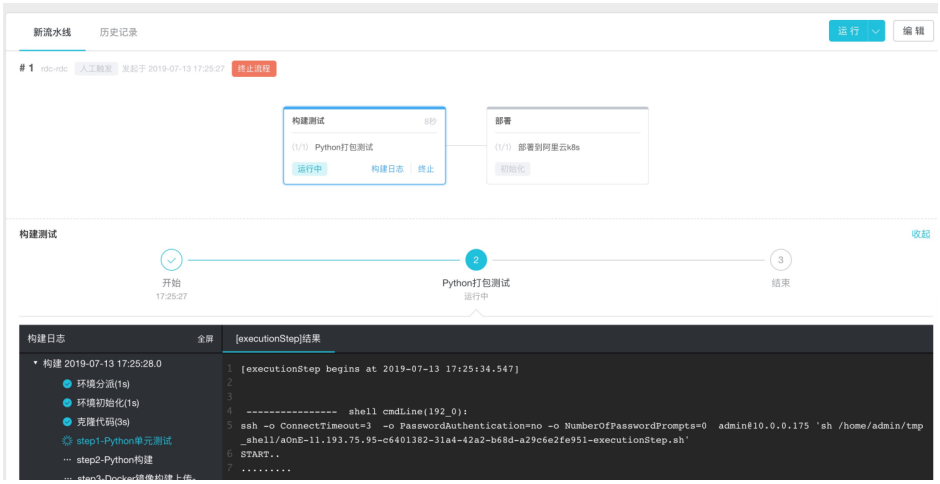
框关闭后，再点击创建环境的弹框的确认，环境即创建成功，并回填到流水线配置中。



运行流水线

点击流水线编辑页面右上角【运行】，流水线会自动保存，并开始运行。

流水线运行中。



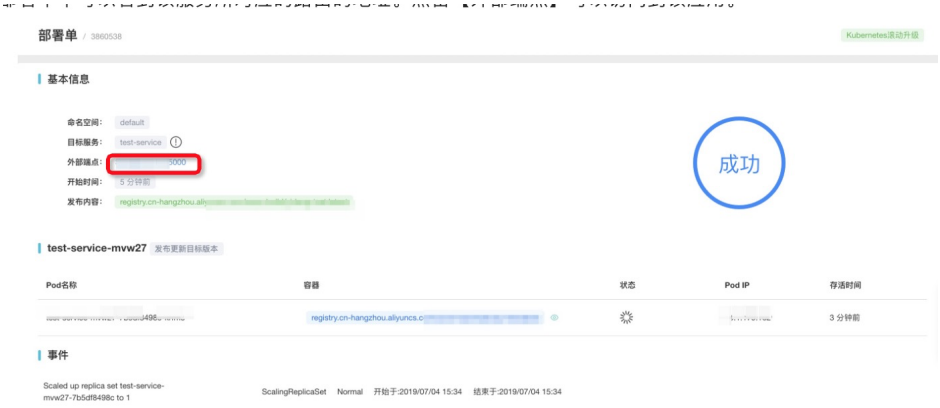
查看测试报告

点击【构建测试】阶段的【测试报告】可以展开测试结果预览，点击预览里的链接可以查看测试报告详情。

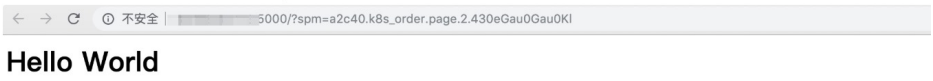
查看部署结果

点击【部署】阶段【发布详情】-> 【打开发布单】可以查看发布单。在发布单中可以查看每台机器的部署情况和日志。

在部署单中可以看到该服务所对应的路由的地址。点击【外部端点】可以访问到该应用。



由于 Service 类型是 LoadBalancer，所以会自动生成一个公网IP来暴露这个 Service。显示在部署单的外部端点部分。访问【外部端点】地址，看到服务已经运行起来了。



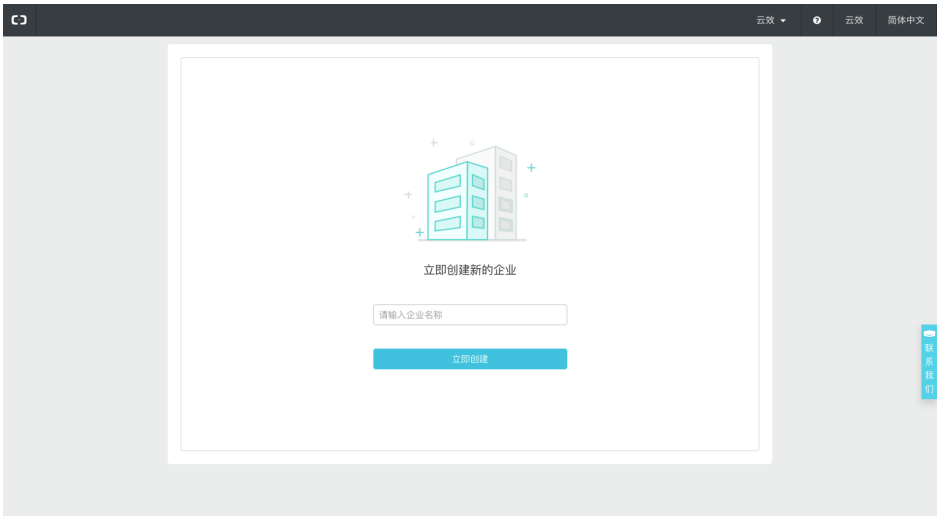
Php入门

PHP应用部署到ECS

本文档会帮助您在云效创建一个 PHP Laravel 的代码库，并部署到阿里云 ECS 服务器。

创建企业

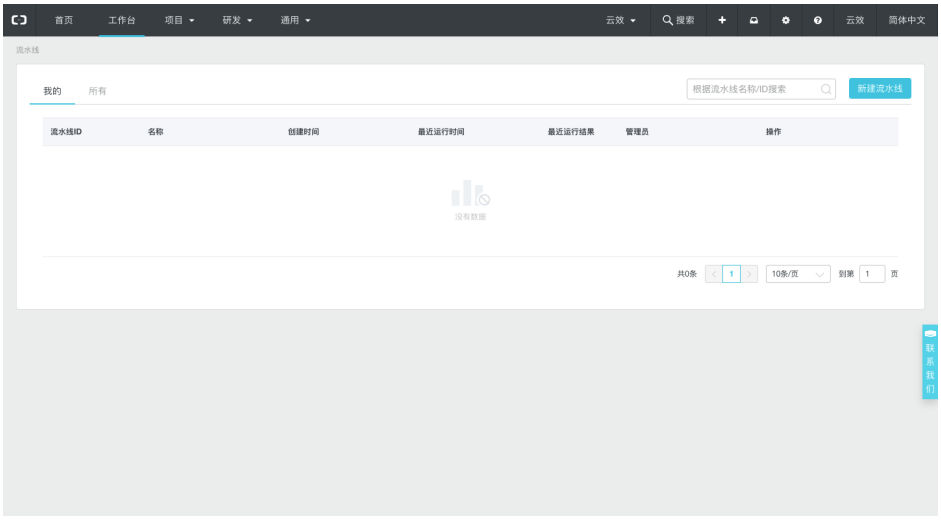
首次进入云效会提示您创建企业。



创建流水线

进入企业后，从页面顶栏点击【研发】->【流水线】，进入流水线列表。

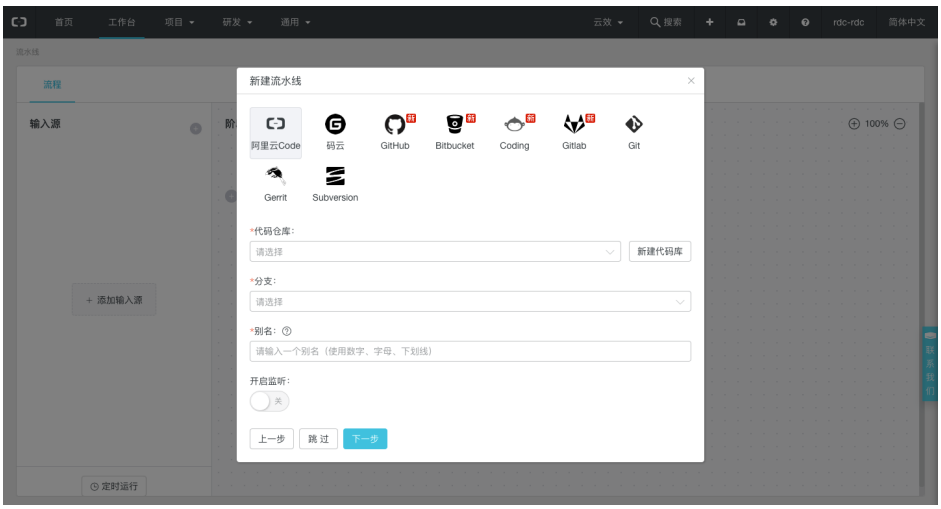
点击右上角【新建流水线】，进入流水线创建向导页面。



选择流水线模板

选择编程语言 【PHP】 和模板 【PHP测试、构建、部署到ECS】。

点击 【下一步】，进入代码源配置。



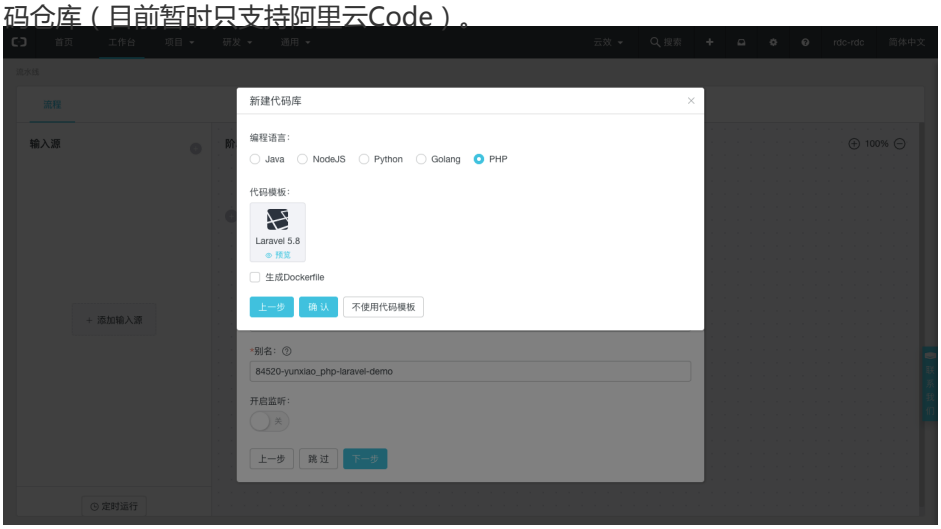
新建代码库

点击【新建代码库】，出现新建代码库对话框。

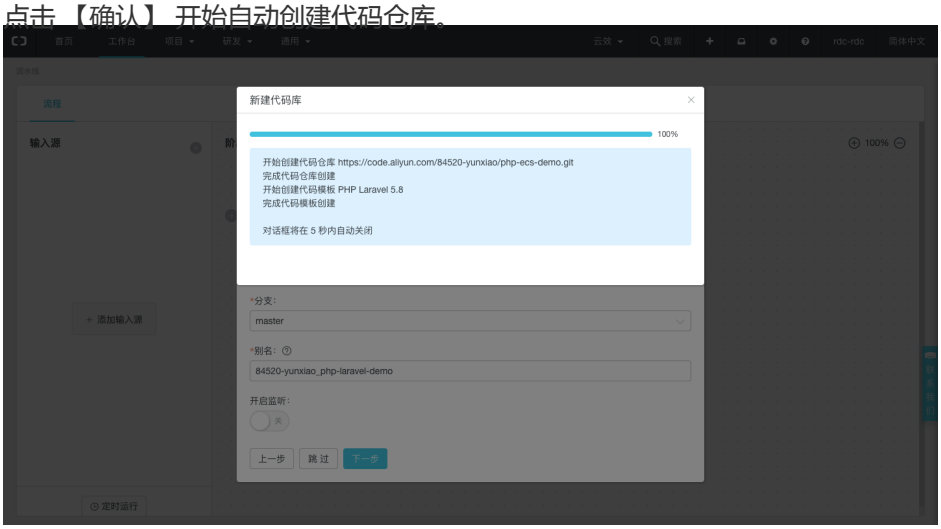
在下拉列表选择具体新建仓库的所属分组 和 仓库名称，配置可见等级等。

如果代码仓库配置提示【没有组】，可以点击下方提示【创建组】，直接创建新代码分组仓库。

点击【下一步】，进入代码模板选择，可以使用云效Laravel代码模板，直接在阿里云Code创建新代码仓库（目前暂时只支持阿里云Code）。



点击【确认】开始自动创建代码仓库。

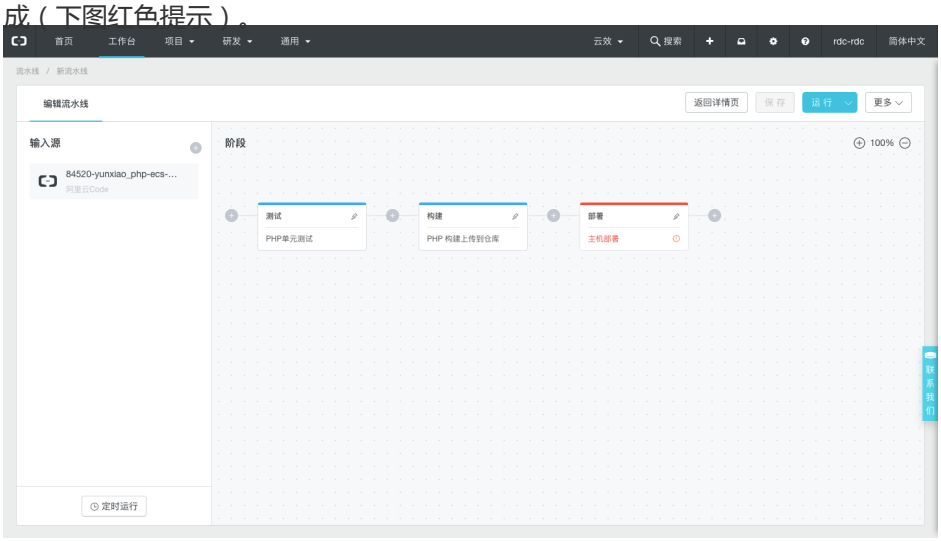


大约15秒，完成后自动关闭新建代码库对话框，代码源信息自动填入到表单里。

填写基本信息

在配置好代码源之后，点击下一步填写基本信息。

点击创建，这个时候已经完成了创建，但是在第一次创建的时候可能会遇到部分流水线组件配置未完成（下图红色提示）。

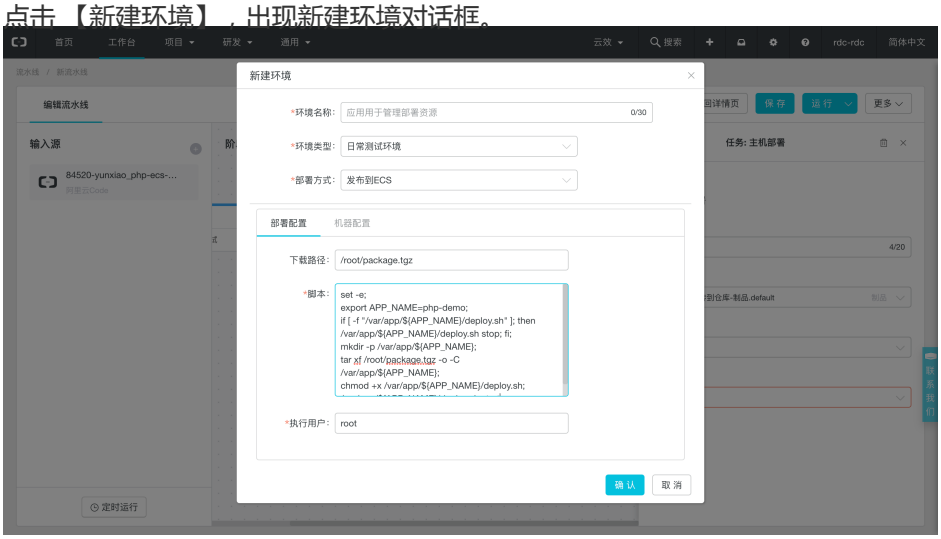


点击未完成的红色卡片，在右侧拉出的菜单里完成信息录入。

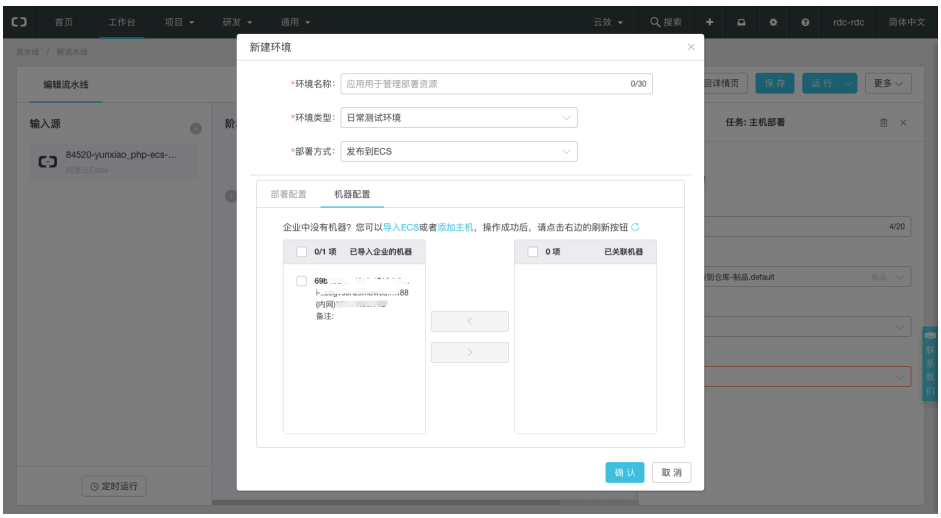
编辑流水线

在主机部署的组件里配置环境信息。

点击【新建环境】，出现新建环境对话框。



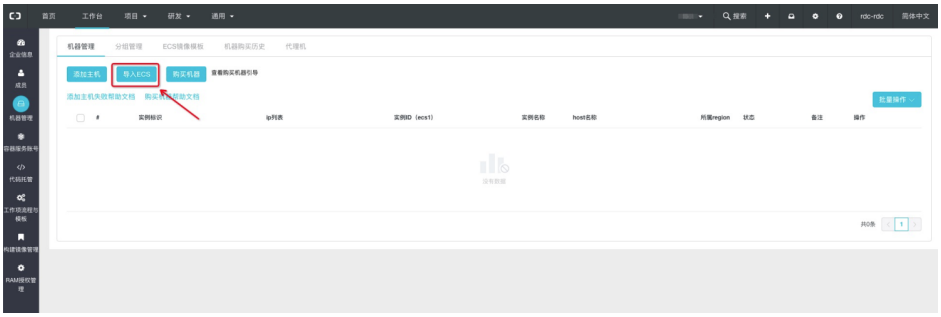
点击新建环境对话框里 【机器配置】，进入机器配置表单，点击【导入ECS】,跳转到导入 ECS 界面



导入 ECS 机器

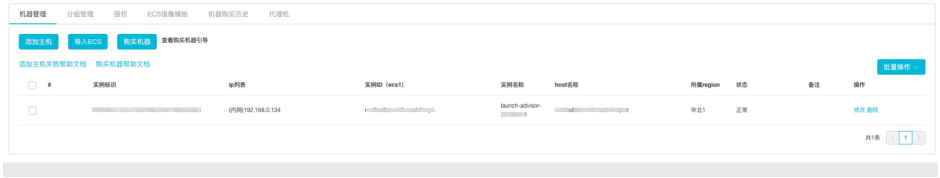
1. 申请准备好阿里云ECS，具体方式参见 ECS 文档，请确保该ECS环境上预装了 PHP 环境并有公网 IP，或者有 SLB 与之关联，以便进行功能验证。

   在机器管理页面，点击【导入 ECS】。

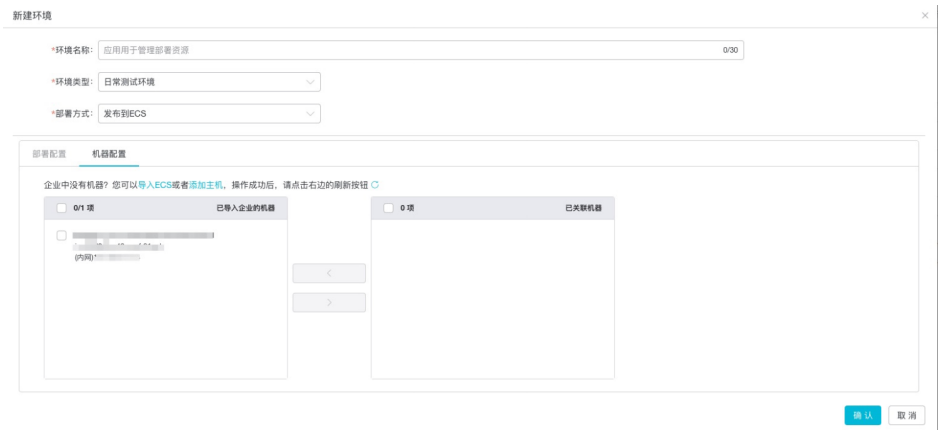


   在弹出框中选择新建 ECS 所在区域，选择实例，点击【导入】。

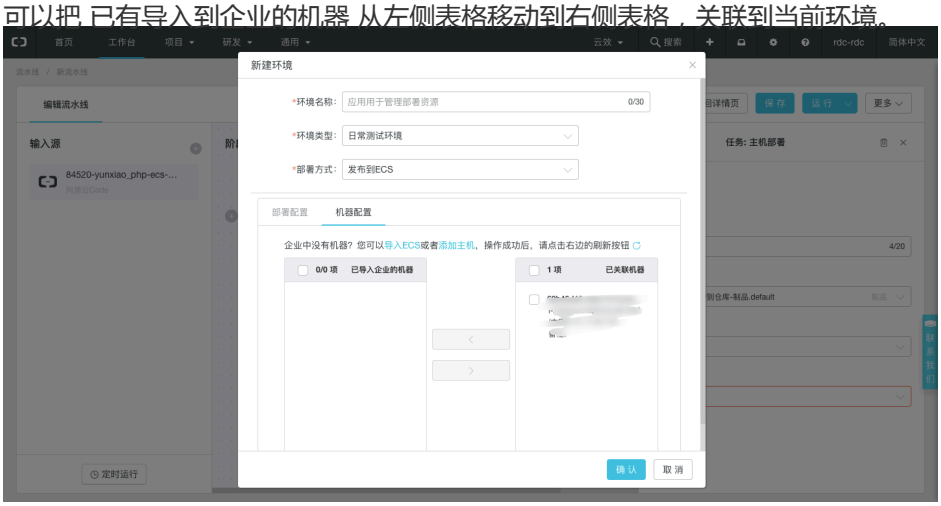
等待大概1分钟左右，刷新页面，可以看到机器状态为【正常】。



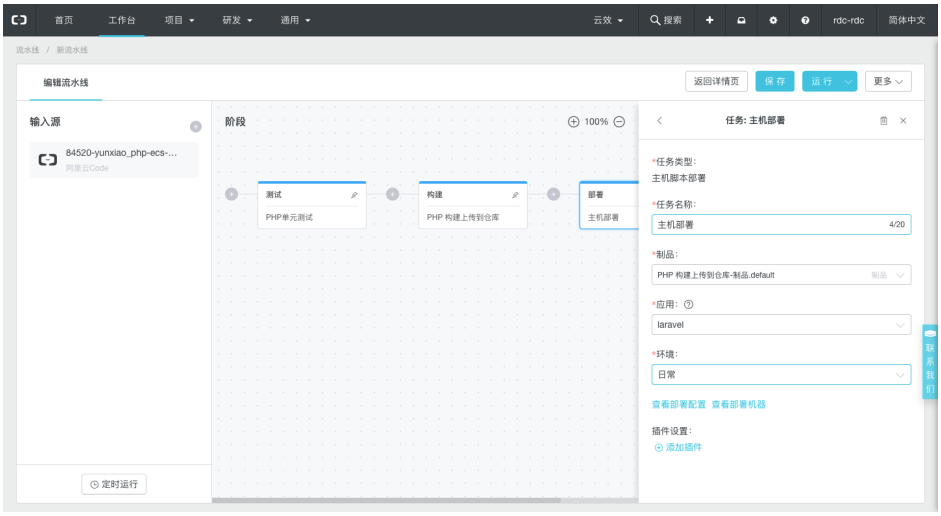
机器导入后回到【机器配置】页面，点击【刷新】按钮，可以看到刚才导入的机器。



可以把已有导入到企业的机器从左侧表格移动到右侧表格，关联到当前环境。

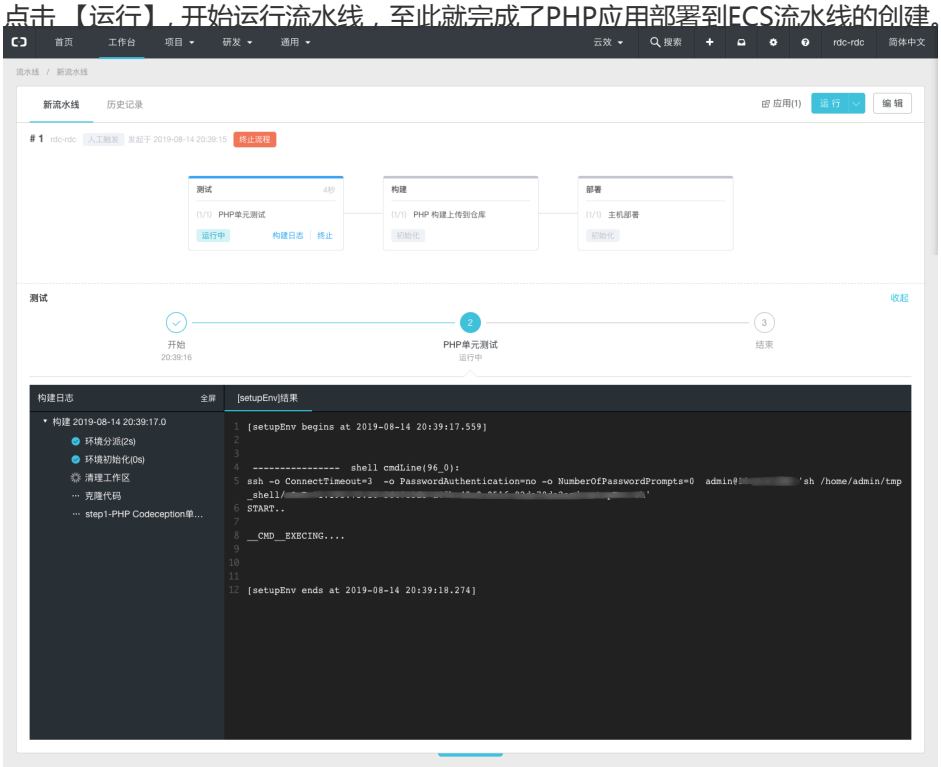


点击 【确认】，返回流水线编辑页面。

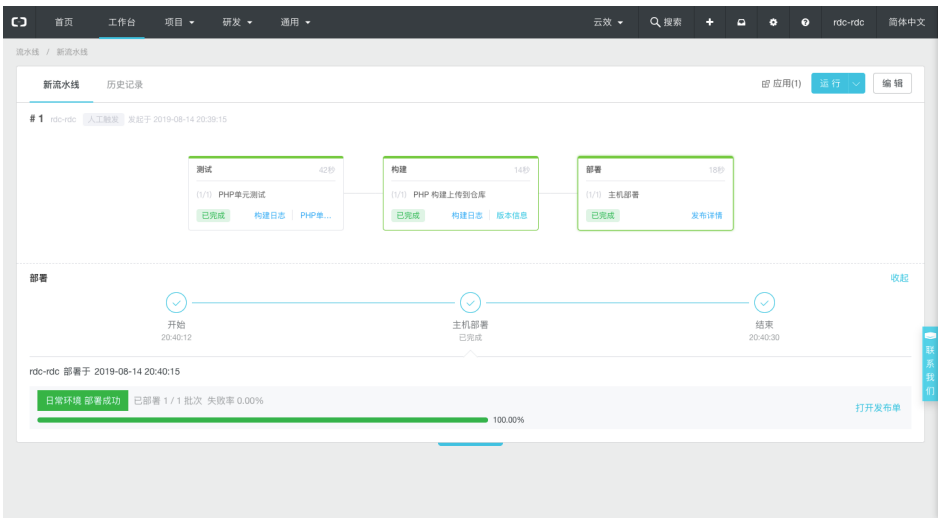


运行流水线

点击【运行】，开始运行流水线，至此就完成了PHP应用部署到ECS流水线的创建。



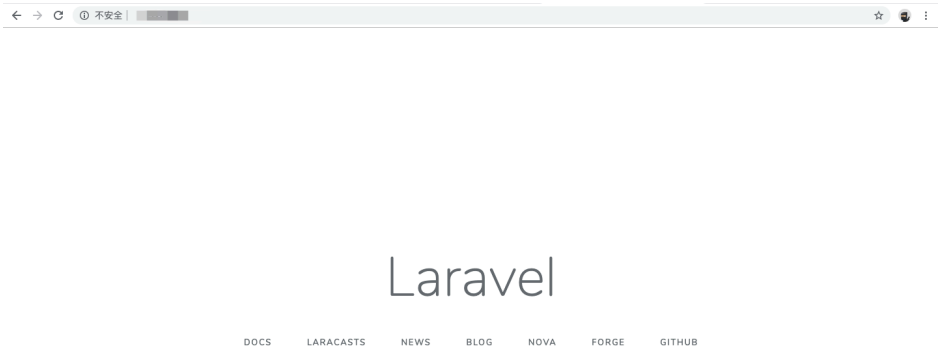
当流水线运行完成后，一个PHP应用就部署到了ECS上。



查看部署结果

点击 打开发布单，可以看到部署详情。

通过浏览器访问 Laravel，完成。


不安全
Laravel
DOCS
LARACASTS
NEWS
BLOG
NOVA
FORGE
GITHUB