

# **CODING DevOps**







【版权声明】

©2013-2025 腾讯云版权所有

本文档(含所有文字、数据、图片等内容)完整的著作权归腾讯云计算(北京)有限责任公司单独所有,未经腾讯云 事先明确书面许可,任何主体不得以任何形式复制、修改、使用、抄袭、传播本文档全部或部分内容。前述行为构成 对腾讯云著作权的侵犯,腾讯云将依法采取措施追究法律责任。

【商标声明】

# 🕗 腾讯云

及其它腾讯云服务相关的商标均为腾讯云计算(北京)有限责任公司及其关联公司所有。本文档涉及的第三方主体的 商标,依法由权利人所有。未经腾讯云及有关权利人书面许可,任何主体不得以任何方式对前述商标进行使用、复 制、修改、传播、抄录等行为,否则将构成对腾讯云及有关权利人商标权的侵犯,腾讯云将依法采取措施追究法律责 任。

【服务声明】

本文档意在向您介绍腾讯云全部或部分产品、服务的当时的相关概况,部分产品、服务的内容可能不时有所调整。 您所购买的腾讯云产品、服务的种类、服务标准等应由您与腾讯云之间的商业合同约定,除非双方另有约定,否则, 腾讯云对本文档内容不做任何明示或默示的承诺或保证。

【联系我们】

我们致力于为您提供个性化的售前购买咨询服务,及相应的技术售后服务,任何问题请联系 4009100100或 95716。



# 文档目录

持续部署

快速开始

云账号

部署流程管理

部署方式

自动发布 Docker 制品时触发

在构建计划中添加部署阶段

手动提交发布单

# 🔗 腾讯云

# 持续部署 快速开始

最近更新时间: 2025-03-17 15:33:52

#### () 说明:

CODING DevOps 于2025年9月1日起更新 CODING 订购方案,取消原标准版套餐,下线部分功能 (制品安全扫描、测试管理、测试协同、仪表盘、研发度量),新注册团队用户界面无持续部署、应用管理 功能,为确保您的使用权益和资产数据安全,请及时关注并处理,了解更多详情。

## 功能介绍

CODING 持续部署用于把控构建之后的项目发布与部署交付流程。能够无缝对接上游 Git 仓库、制品仓库以实现全 自动化部署。在稳定的技术架构、运维工具等基础上,具备蓝绿发布,灰度发布(金丝雀发布),滚动发布,快速回 滚等能力。

下文将以一个简单的 Demo 项目为例,演示如何使用 CODING 持续部署控制台将应用发布至腾讯云集群。

#### 前置准备

- 开启持续部署设置权限,请参见 权限详情。
- 导入 示例代码库。点击了解如何 导入或关联外部仓库。
- Docker 制品仓库,单击了解如何使用项目中的 Docker 制品仓库。
- 一个可被 CODING 持续部署访问的 Kubernetes 集群,单击了解如何申请 腾讯云标准集群。

#### 🕛 说明:

集群的推荐配置为 8 核 32 GB,需开启公网访问权限。

### 操作步骤

#### 步骤1: 获取并关联云账号

🕛 说明:

因使用了腾讯云容器服务,部署后应用将发布至集群,本示例使用的团队账号已在**团队设置中心 > 第三方应** 用中关联 腾讯云账号 。

单击首页左侧的基础设施,在云账号中绑定腾讯云账号。云账号名称可以自拟,选择地域后将自动获取对应集群。



工作台	基础设施 集群管理 云账号 主机组		≥ 绑定云账号	
山 项目	技名称授素 Q		云账号类别*	
/> 代码仓库	账号名称	账号状态		
● 项目集				
📙 知识管理 beta			腾讯云 TKE Kubernetes 腾讯云	
• 仪表盘		Q	<b>示能是之助</b> *	
〒 全部事项			478 - 1111 支持大小写字母 中朝純和下朝純	
008 日标		未找到相关信息	×107.0.m3 1 497 1 432618 F 4326	
日际			请选择地域*	
MALE 111 贝敦			请选择 ~	
山 研发度量 beta			请选择 TKE 集群*	
11 公开资源			请选择	
□ 全部制品				
E Cloud Studio			自动生成 CODING Docker 仓库访问凭证 ②	
② 应用中心 >			请选择	
🕞 基础设施			允许持续部署管理集群已有资源	
			加贝石柳 设置权限组 (译情) 操作	

在集群中新建 命名空间(Namespace) 用于存储自动生成的制品仓库访问凭证,本文中所使用的集群命名为: cd-demo。

於 腾讯云 总览 云	产品▼					容器服务	8 Q 💯	99+ 备案	工具 → 云审	十 支持 ▼	费用 ▼	<u> </u> 乘 🔻
容器服务	🔶 集群(广州) /	cls-ow54ato4	4(体验版集群)								YAML®	建资源
₽ 概览	基本信息		Namespace								扬	操作指南 ☑
② 集群	节点管理	<b>.</b>						点人关始内田				ch. I
◎ 弹性容器 🔹 🗸	命名空间		新建					多个大键子用	全线 " " 分隔,多1"	过滤标金用回牛	ii Q	\$ ₹
🕰 边缘集群	工作负载	•	名称	状态	描述		创建时间		操作			
分 服务网格	自动伸缩	•	default	Activo			2021-07-27		石坊	<b>\$1日</b> □□□12全		
应用中心	服务与路由	•		Active			14:32:34		HUBK	141 UU MA		
☆ 应用	配置管理	•	kube-node-lease	Active	-		2021-07-27 14:32:32		配额	管理 删除		
回 镜像仓库 *	授权管理	<b>.</b>										
🖸 应用市场	存储	<b>.</b>	kube-public 🗖	Active	-		2021-07-27 14:32:32		配额	會理 删除		
运维中心	组件管理						2021-07-27					
⑦ 集群运维 🛛 🖌	日志		kube-system I	Active	-		14:32:32		日已名积1	雪埋 删际		
🖸 云原生监控			第1页							20 -	条/页 🔹	)
品 容器镜像服务 ☑	事件											
∞ DevOps ⊠												Ģ ₹
□ 快速入门												持

#### 步骤2:配置应用

成功添加云账号后,在部署控制台中单击**创建应用**,填写应用名与选择部署方式。有关于**应用**的更多介绍,详细说明 请参见 应用管理 。



← 应用中心	应用 #2000+						
€ 应用	云账号: 全部 × 关联项目: 全部 × 排序方式: 更新时间倒序 × 授亲。	× 创建应用					
	dev ☴ ▲ 五株号:0 ● 关联项目:7 🐉	应用名 * ③ 填写应用名,创建后不支持变更 器膏方式 *		app ☴ ▲ 五株号: 0 ● 关联项目: 0	 8 # 0		
	flaskapp 🚍 畫 云张号: 0 🐠 关联项目: 2 📑	Cubernetes(TKE) 時讯云弹性伸缩 主机组	 8 A 0	spring-demo 🚍 畫 云张号: 0 🛛 😋 关联项目: 0			
	client 書 ▲ 云张号: 0 ●● 关联项目: 0 ■	描述 请输入描述信息		devops 票 畫 云张号: 0 ●● 关联项目: 0	 8 A 0		
	testing11111 హ 2 云张号: 0 🐽 关联项目: 0 📑	应用权限 ⊙		devops-solution ☴ 畫 云张号: 0 🔹 关联项目: 1			

选择部署到 Kubernetes 集群模板,填写名称与描述后完成创建。

← dev → 部署流程	创建部署流程		
状态 全部 ∞ 任务数量 2 ∞	有山顶方法理 Kukawakaa		创建流程
名称	复制现有流性 Kubernetes		操作
> dev	部署 Helm 应用到 Kubernetes 集群	名称 *	9 启动 编辑 …
	配置 Bake (Manife 部署 (Manifest)	请输入部署流程名称,不超过 50 个字符	
		描述	
	部署 Deployment 和 Service 到 Kubernetes 集群 配置 部署 Deploy 部署 Service	请输入描述	
	部署到 Kubernetes 集群前进行人工确认	部署流程权限 ①	
		▲ 添加用户 + ▲ 添加用户组 + ▲ 添加部门 +	
	并行部署 Deployments 和 Services	成员名称 设置权限组 (详情) 操作	
	配置 部署 Deploy 部署 Service	- 0	
	部署 Deploy 部署 Service		
		暂无数据	
		能认	

#### 步骤3:初始化项目

此步骤主要用于配置持续部署所涉及的项目文件,默认您已完成前置准备中的导入示例代码库及创建 Docker 制品 库。

首先,将本地拟发布的 Docker 制品推送至 CODING 制品仓库,具体操作可参考 Docker。



	70.74
🔶 操作指引	推送
	输入以下推送相关信息,生成推送命令:
配置凭据	本地镜像 tag:
推送	制品名称:
拉取	
焙伤酒加油 🙃	制品版本:
現豚源加速 💋	1. 请在命令行执行以下命令给本地镜像打标签:
	<pre>docker tag <local_image_tag> chasylee-docker.pkg.coding.net/typical/docker/<package>:<v< pre=""></v<></package></local_image_tag></pre>
	2. 请在命令行执行以下命令进行推送:
	<pre>docker pushpkg.coding.net/typical/docker/<package>:<version> Copy</version></package></pre>

推送至制品仓库后,获取制品的拉取地址(即"操作指引" > "拉取"命令中的镜像路径)并填写至代码仓库中 /k8s/deployment.yaml 中的 image 地址。



← k8sDemo ▼ 浏	<b>征</b> 提交 分支 合并请求 版本 对比 设置	○ ●已开源 + 创建代码仓库 -
🛧 k8sDemo	💡 master 👻 🏠 │ 查找文件 🗸 k8s / deployment.yaml	<b></b>
> in gradle/wrapper	<b>文件</b> 历史 6 代码对比 按行查看	⊿ 编辑
🗋 deployment	🚳 主账号 更新文件 deployment.yaml	最后提交 762348efea 🗊 于 1 天前
service.yaml	deployment.yaml   504 Bytes 1 apiVersion: apps/v1	
> 📔 lib	2 kind: Deployment 3 metadata:	
> 🧧 src	4 labels: 5 app: k8sdemo	
🗋 .gitignore	6 name: k8sdemo-deployment 7 namespace: cd-demo	
Dockerfile	8 spec: 9 replicas: 1	
build.gradle	10 selector: 11 matchLabels:	
🗋 gradlew	12   app: k8sdemo     13   template:	
🗋 gradlew.bat	14metadata:15labels:	
🗋 settings.gradle	16     app: k8sdemo       17     spec:       18     imagePullSecrets:       19     - name: coding-registry-cred-252585       20     containers:       21     - image: 'StrayBirds-docker.pkg.coding.net/flask-demo/cd-demo/hello-world'	
	22 name: k8sdemo 23 ports: 24 - containerPort: 8080	

接下来需导入云账号的 imagePullSecrets 至代码仓库中。在基础设施 > 云账号中单击查看详情后,复制名称。





粘贴至代码仓库中的 deployment.yaml 文件中,同时在 namespace 参数一栏中填写在上文中所创建的命名 空间 cd-demo。



← k8sDemo ∽ 浏览	【 提交 分支 合并请求 版本 对比 设置	<ul> <li>● 已开源</li> <li>+ 创建代码仓库</li> </ul>
♠ k8sDemo	😵 master 👻 🛖 ≜ 查找文件 🗸 k8s / deployment.yaml	<b></b>
> 🧧 gradle/wrapper		相大 西松
∨ 📙 k8s		提父
deployment	😝 主账号 更新文件 deployment.yaml	最后提交 762348efea 🗊 于 1 天前
service.yaml	deployment.yaml 504 Bytes	
> 🖿 lib	1 apiVersion: apps/v1 2 kind: Deployment 3 metadata:	
> 📄 src	4 labels: 5 app: k8sdemo	
🗋 .gitignore	6 name: k8sdemo-deployment	
Dockerfile	<pre>/ namespace: cd-demo 8 spec: 9 replicas: 1</pre>	
build.gradle	10 selector:	
🗋 gradlew	12 app: k8sdemo 13 template:	
🗋 gradlew.bat	14 metadata: 15 labels	
Settings.gradle	10     app: k80demo       17     spec:       18     imagePullSecrets:       19     - name: coding-registry-cred-252585       20     containers:       21     - image: 'StrayBirds-docker.pkg.coding.net/flask-demo/cd-demo/hello-world'       22     ports:       23     ports:       24     - containerPort: 8080	

同一层级的 service.yaml 文件中的 namespace 内容也需保持一致。

← k8sDemo - 浏览	2 提交 分支 合并请求 版本 对比 设置	○ •已开源 + 创建代码仓库 -
1 k8sDemo	第 master ▼ 介 当找文件 ∨ k8s / service.yaml	<b></b>
> in gradle/wrapper	<b>文件</b> 历史 ④ 代码对比 按行查看	⊿ 编辑 …
deployment	R 管理员 更新文件 service.yaml	最后提交 01c504d016 🗊 于 3 个月前
🗋 service.yaml	service.yaml 181 Bytes	
> 🖿 lib	2 kind: Service 3 metadata:	
> src	4 name: k8sdemo 5 namespace: cd-demo	
🗋 .gitignore	6 spec: 7 selector:	
Dockerfile	8 app: k8sdemo 9 ports:	
🗅 build.gradle	10 - port: 8080 11 targetPort: 8080	
🗋 gradlew	12 type: LoadBalancer	
🗋 gradlew.bat		
settings.gradle		

## 步骤4:部署流程



在应用中心找到上述步骤 2 创建的应用,点击其名称进入部署流程配置页面,可以为此流程设定:

- 流程的执行选项(在此示例中保持默认即可)。
- 部署 Deployment 阶段以及部署 Service 阶段所需制品。
- 手动或自动触发。

首先配置部署(Manifest)阶段。基础设置选择已绑定的云账号,在 Manifest 来源选择 **CODING 代码库**,填 写相应的路径。

← cd-demo 💆	新署 Deployment
◆ 基础配置                  ● ABIO-world                  ◆ K8s/deployment.yanl             master                 ◆ K8s/service.yanl             master	依赖阶段: 基础配置 ~
	∨ 镜像版本配置

配置部署 Service 阶段时步骤同上,但在文件路径处需选择 k8s/service.yaml 文件。



	➤ 部署 Service	
·····································	<b>部署 (Manifest) 配置</b> 执行选项 通知 描述	
阶段类型: 部署 (Manifest)	Manifest 来源 ① ● 使用制品    输入内容	
	制品来源	
	CODING 代码库	~
	项目	
	演示项目	~
	仓库	
	CD-demo	~
	默认分支或标签 ①	
	master	~
	文件路径	
	k8s/service.yaml	~
	高级配置	
	跳过 SpEL 表达式计算 ⑦	

镜像版本配置默认选择自动获取镜像来源。若设置自定义版本规则,将仅传送特定的 image 版本信息号至集群中。



✔ 镜像版本配置						
Manifest 中的镜像版本默认支持动态替换,即启动部署流程时可以指定版本覆盖 Manifest 中的默认版本。查看 帮助文档 🖸						
镜像来源						
● 自动获取 ○ 上游阶段生成						
镜像名称 hello-world						
镜像地址 ⑦ StrayBirds-docker.pkg.coding.net/demo/docker/hello-world						
高级配置						
🔵 自定义版本规则 ② 🔹 锁定默认版本 ③ 🔹 忽略版本 ③						

#### 步骤5: 触发器配置

完成部署阶段配置后,您可以使用自动化触发器、手动提交发布单执行部署。

#### 自动触发

在基础配置中点选触发器类型,选择 Docker 仓库触发器。当开发人员更新代码仓库并使用 CI 将镜像打包推送至 制品库后,Docker 镜像版本的更新将自动触发部署流程并将应用发布至 Kubernetes(TKE)集群,完成后可 以在应用中心查看并确认应用是否发布成功。





#### 手动提交发布单

若希望通过手动提交发布单的形式触发部署流程,那么可以将应用(例如本范例的 flaskapp)与项目关联。在应用 中心搜索项目名称进行关联:

应用			创建应用
云账号: 全部 ▼ 关联项目: 全部 ▼ 排序方式: 更新时间倒序 ▼	搜索 Q		
<b>flaskapp ☴</b> ▲ 云帐号: 0   ● 关联项目: 0	关联项目	<b>go ☴</b> ▲ 云帐号: 0   ● 关联项目: 0	 13 4 ¢
<b>test ☴</b> ▲ 云帐号: 0   • 关联项目: 1	 8 A Ø		



#### 关联完成后,单击项目中的**持续部署 > Kubernetes** 手动提交发布单。

① 项目概览		🤶 cd–de	emo → 发布单 集群 部署流程		□ 帮助文档	当 新建发布 🔻
<♪ 代码仓库		状态:全部▼	<b>创建人: 全部 →</b> 部署流程: 全部 → 发布时间: 最近十天 → 搜索:	请搜索关键字		
🕑 代码扫描 beta	>					
∞ 持续集成	>	发布单状态	发布单名称	部署流程	创建时间	创建人
♪ 持续部署	~	🕑 成功	gogo	cd-demo	2021-07-27 19:53:02	▶号 主账号
Kubernetes		🗸 成功	20210726-cd-demo-cd-demo	cd-demo	2021-07-26 17:58:48	▶ 主账号
弹性伸缩		🗸 成功	20210726-cd-demo-cd-demo	cd-demo	2021-07-26 17:54:50	▶ 主账号
主机部署		•				
网站托管						
□ 制品管理	>					
文档管理	>					

#### 步骤6:发布完成

发布成功后,可以查看发布的制品及启动参数及阶段执行详情等信息。

础信息	阶段	状态 成功	开始时间 2021-0	7–27 19:53:03
手动触发		耗时 19 秒		
<ul> <li>A ■ 主账号</li> <li>2021-07-27 19:53:02</li> </ul>	部署 Deployment 耗时: 19 秒	阶段详情		
③ 25 秒		状态    脚本名称	启动时间	耗时
制品		✓ 成功 部署 Deployment	2021-07-27 19:53:03	19秒 ^
<ul> <li>StrayBirds-docker.pk g.coding.net/flask-d emo/cd-demo/hello -world latest</li> </ul>		DeployStatus Task Status Artifac ✓ Deployment k8sdemo-deploymen	ct Status it 查看	Yami 内容 跳转查看资源详
k8s/deployment.yaml master		ScalingReplicaSet 3 分钟 以前		
♦ k8s/service.yaml		Scaled up replica set k8sdemo-o	deployment-994479977 to 1	



#### 当需要查看某个资源在集群中的运行状态时,单击**集群**下的工作负载即可查看详情(例如工作负载的 Pod 实例,日 志等信息)。

← cd-demo → 发布单 集群 部	署流程	6	
<b>工作负载 服务</b> 云账号: <b>全部</b> 命名	空间: 全部 🔹 类型: 🕯	:部▼ 状态:全部▼	k8sdemo-deployment-994479977 操作 ▼
名称	命名空间	云账号	基础信息
deployment k8sdemo-deployment V001 StrayBirds-docker.pkg.coding.net/flask-d	cd-demo emo/cd-demo/hello-wo	Go rld:latest 🛛 🛱 Load E	创建时间 2021-07-27 19:53:05 云账号 Go 命名空间 cd-demo 资源类型 replicaSet 控制器 deploymentk8sdemo-deployment
			镜像
			StrayBirds-docker.pkg.coding.net/flask-demo/cd-demo/hello-world:latest ①
			事件
			1 x SuccessfulCreate
			– 以前 Created pod: k8sdemo-deployment-994479977-nmkdx
			LABELS
			app:k8sdemo
			app.kubernetes.io/managed-by:spinnaker app.kubernetes.io/name:cd-demoteam174750
			pod-template-hash:994479977

#### 在腾讯云的容器服务中查看工作负载。



容器服务	← 集群(广州) / cls-	w54ato4(体验版集群)	建资源
<b>器 概</b> 览	基本信息	<b>Deployment</b> #	作指南 🖸
◎ 集群	节点管理		d. 1
💿 弹性容器 🔹 🔻	命名空间	新建 监控 前名空间 cd-demo ▼ Label格式要求: name=value, 多个关键字用竖线 Q	Q ±
▲ 边缘集群	파고그리	- 名称 Labels Selector 运行///// 短Pod 教告 Request/Limits 操作	
↔ 服务网格	⊥作贝载 ● Deployment		
应用中心	<ul> <li>StatefulSet</li> </ul>	CPU:无限制/无 K8sdemo-deployment币 app:k8sdemo、a app:k8sdemo 1/1  限制 更新Pod数量 更新Pod	
☆ 应用	<ul> <li>DaemonSet</li> </ul>	内存: た成制/ た 限制	
◎ 镜像仓库 ◆	<ul> <li>Job</li> </ul>	第1页	•
四 応用市场	- CronJob		
	自动伸缩	·	_
	服务与路由	•	
	配置管理	•	
	授权管理	<b>v</b>	
品 容器镜像服务 C	存储	*	
∞ DevOps 🖾	组件管理		支持
🖸 快速入门			<b>P</b> t
	口心		Ţ.
	事件		文档

# 配置自动化构建流程

若在后期需要频繁开发项目,能否仅依靠代码推送就能够自动完成发布呢?答案是肯定的。经典的持续交付工作流设 计思路如下:

1. 配置持续集成任务,设置监听代码更新后触发。



2. 将项目代码打包成制品后,发布至制品仓库。



品仓库 全部制品	仓库管理				创建制品
▶ ruby Docker 仓库 \ 项目内	<b>ruby 司</b> 类型 Docker 权限 项目内			✿ 设置仓库	代理设置 版本覆盖
apk Generic 仓库   项目内	镜像列表				18.76
▶ build Docker 仓库   公开	发布状态 <b>全部 ▼</b> + 制品属性 ▼ 	提紫制品名称 Q 最新推送版本	最近更新时间 ≑	版本数	操作
daily-sentence Docker 仓库   项目内	ruby 	2 dc0c37fa405025c573807dc673cfad	2021-12-08 11:24:35	1	
electron Generic 仓库 团队内	v1.0 	? dc0c37fa405025c573807dc673cfad	2021-12-08 10:38:17	1	
gwt Generic 仓库 项目内	1-2 个, 共 2 个				每页显示行数 15 👻

3. 持续部署监听到制品版本号更新后,自动发布至集群。

← cd-demo ⊉	▶ 基础配置
↓ 基础配置 # 基础配置 # 新署 Deployment 阶段类型: 部署 (Manif 制品 ★ k8s/deployment.yaml	执行选项 自动触发器 启动参数 通知 描述 st)
master k8s/service.yaml master hello-world 	触发器启用开关 触发器类型 CODING docker 仓库触发器 てCR 个人版仓库触发器 TCR 个人版仓库触发器 TCR 企业版仓库触发器 TCR Helm 仓库触发器 Git 仓库触发器 Git 仓库触发器 Git 仓库触发器 CODING Generic 仓库触发器 CODING Generic 仓库触发器

🕥 腾讯云

# 云账号

最近更新时间: 2023-09-11 16:05:27

云账号是访问云资源的凭证,只有配置了云账号,持续部署控制台才能实现对云资源的部署管理和基础设施管理。目 前支持三种云账号类型:

- 腾讯云 TKE: 使用腾讯云容器服务中的集群。
- Kubernetes: 支持 Kubeconfig 和 Service Account 两个常用凭据。
- 腾讯云账号:即腾讯云 API 密钥。

单击团队首页左侧的基础设施,进入**云账号**管理页,单击右上方的**绑定云账号**。

■ 工作台	基础设施 集群管理 云账号 主机组		3 绑定云账号
□ 项目	按名称搜索 Q		云账号类别*
	账号名称	账号状态	
➡ 知识管理 beta	🐼 adolf	已验证	腾讯云 TKE Kubernetes 腾讯云
• 仪表盘			
── 全部事项			ム版 5 日前 - 支持大小写字母、中划线和下划线
orr 目标			请诜柽他博 *
<b>北</b> 工作负载			·····································
山 研发度量 beta			请选择 TKE 集群 *
公开资源			请选择
<ul> <li>全部制品</li> <li>Cloud Studio</li> </ul>			自动生成 CODING Docker 仓库访问凭证 ⑦
<ul> <li>Cidad Stadio</li> <li>应用中心 &gt;</li> </ul>			请选择
□ 基础设施			允许持续部署管理集群已有资源
			云账号权限 ◎
			▲ 添加用户 + ↓ ▲ 添加用户组 + ↓ 読 添加部门 +
			成员名称 设置权限组 (详情) 操作

# 腾讯云 TKE

1. 云账号类型选择腾讯云 TKE,按照指引完成与云账号名下的集群绑定。若没有集群请前往 腾讯云 TKE 创建集群。



基础设施 集群管理 云账号 主机组		⊗ 绑定云账号
按名称搜索 Q		云账号类别*
账号名称	账号状态	
🔗 adolf	已验证	腾讯云 TKE Kubernetes 腾讯云
		云账号名称。
		支持大小写李母、中划线和下划线
		请选择地域*
		请选择 ~
		请选择 TKE 集群 *
		请选择 ~
		自动生成 CODING Docker 仓库访问凭证 ⑦
		请选择
		允许持续部署管理集群已有资源
		云账号权限 ⊙
		▲ 添加用户 + ) <b>≛</b> * 添加用户组 + )
		成员名称 设置权限组 (详情) 操作

2. 选择拟部署的集群,单击确定后会自动验证该账号名下的集群并完成互联。

# **Kubernetes**

Kubernetes 云账号支持 Kubeconfig 和 Service Account 两种常用凭据。以 Kubeconfig 为例: 登录云计算网页控制台,复制 Kubeconfig,并将 CODING IP 段添加至集群外网访问控制列表(白名单)。





於 購訊云 总览	云产品 🖌   网站备	案 +				ŝ	E
容器服务	← 集群(上海) / cl	s-bjvylwsb(k	(8s-dev)				
<b>器 概</b> 览	基本信息		集群APIServer	言息			
◎ 集群	节点管理	Ŧ	访问地址	https://cls-bjvylwsb.ccs.tencent-cloud	l.com		
② 弹性集群	命名空间		外网访问	こ开启			
岛 边缘集群	工作负载	Ŧ		已放通IP地址:212.64.105.0/24;212	.129.144.0/	/24 🎤	
↔ 服务网格	自动伸缩	Ŧ	内网访问	● 未开启			
应用中心	服务与路由	Ŧ	Kubeconfig	以下kubeconfig文件为当前子账号的ku	beconfig内	容:	
😴 应用	和苦答理	<b>.</b>		apiVersion: v1 clusters:			
回 镜像仓库 ·	心里自注			- cluster:			
凹 应用市场	授权管理	Ŧ		certificate-authority-da LS0tLS1CRUdJTiBDRVJUSUZJQ0F	ata: JRSØtLSØ <sup>.</sup>	tCk1JSU	N5RENDQW
	存储	Ŧ		REFWTVJNdØVRWURWUVFERXdwcmR	KSmwKY20	1bGRHVn	pNQjRYRF

将 Kubeconfig 粘贴到云账号中,选择 Cluster Context 完成云账号添加。

基础设施 集群管理 云账号 主机组		⊗ 绑定云账号
按名称搜索 Q		云账号类别*
账号名称	账号状态	
🐼 adolf	已验证	腾讯云 TKE Kubernetes 腾讯云
		选择认证方式 * ● Kubeconfig ○ Service Account Kubeconfig * apiVersion: v1 clusters: - cluster:
		certificate-authority-data: LSOLISCRULITIBERVIJUSZJOOFURSOILSOICKUSUBBVENDOWVIZOF3SUJBZOIKOVBYVW Z4eh-UPCNDETUE-WRONTOLITISWIZHEFFCKNASVUFNOmNARIBRYEJINTYKOKENTUBERKKONA UCTWOEBETV/VVVEFRoweBVENKONVESITORUEU/LEDEGCGAUNTDBAKIGAVBREISTUBBY OMNOwMADIBRI/E_INTYCOLUMPEDE/K0xeDTEALTDATODAKILUUCOMBLIOSDEET2N#MAkio/wah



## 腾讯云账号

1. 云账号类型选择腾讯云账号,输入云账号名称,并选择区域。支持多选区域,CODING 持续部署将获得勾选区 域的腾讯云资源管理权限。

基础设施 集群管理 云账号 主机组		⊗ 绑定云账号
按名称搜索    Q		云账号类别*
账号名称	账号状态	
S adolf	已验证	腾讯云 TKE Kubernetes 腾讯云
		云账号名称*
		支持大小写字母、中划线和下划线
		请选择区域 ⑦*
		请选择
		SecretID* ⑦如何获取腾讯云 SecretID 7
		请输入
		SecretKey*
		请输入 SecretKey
		<b>元张是如照 ①</b>
		成武名鄉 设置权限组 (译情) 操作

2. 在腾讯云 访问管理控制台 页面中拷贝 API 密钥信息。

🔗 腾讯云	总览	云产品 → 网站备案 十	盟	0	⊠4 ∦	明助 工单	≧▼ 费用▼	👤 Leo 🗸
访问管理		API密钥管理						云 API 使用文档 🖸
計 概览 用户 ∞ 用白網		调用 <mark>腾讯云AP</mark> I时需要签名,云API密钥用于生成签名,查看生成签名算法。 API 密钥是构建腾讯云 API 请求的重要凭证,使用腾讯云 API 可以操作您名下的所有腾讯云资源,为了您的财产 密钥。	和服务安全	, 请妥	善保存和定期」	更换密钥, 当	当您更换密钥后,	,请及时删除旧
云 府广组 □ 策略 □ 角色		<ul> <li>您的 API 密钥代表您的账号身份和所拥有的权限,等同于您的登录密码,切勿泄露他人</li> <li>上次访问时间和上次访问服务为当前访问密钥在 30 日内最近一次访问云 API 的时间和服务(支持记录访问操机 访问记录或不存在访问记录将被展示空</li> </ul>	乍级或资源	及粒度的	的服务,访问服	务级粒度的	服务不作记录)	,超过 30 日的
· · · · · · · · · · · · · · · · · · ·		新建密钥						
<ul><li>(₽) 访问密钥</li><li>• API密钥管理</li></ul>		APPID 密钥	创建	时间	上次访	上次访	状态	操作
		Secretic: ANILO	2020	-03	2020-03	云服务器	已启用	禁用



3. 将拷贝的 SecretID 和 SecretKey 粘贴到对应的文本框,单击确定完成云账号添加。

腾讯云

# 部署流程管理

最近更新时间: 2024-08-02 15:23:01

部署流程是实现持续部署最核心的模块。其优势在于支持阶段以任意的顺序组合,这样的能力让部署流程具备出色的 灵活性、一致性和可重复性。

- 灵活性: 支持串行、并行控制。
- 一致性: 支持多种部署策略, 回滚能力, 确保发布结果符合预期。
- 可重复性: 部署流程可重复执行,阶段可被其他部署流程复制使用。

用户可以配置完全自动化的部署流程,也可以在某些阶段加入手工判断条件。此外部署流程支持多种事件的自动化触发,例如 Webhook 触发、由其他部署流程触发等。

#### 新建部署流程

前往应用中心,单击应用卡片右下方的部署流程按钮。

← 应用中心	应用 体验新版→					创建应用
✿ 应用	云账号:全部▼ 关联项目:全部▼ 排序方式:更新时间倒序▼		Q			
The second second	day =	部署流程	now of =		ann =	
	🔮 云帐号: 0 🔹 关联项目: 7	8 4 0	▲ 云桃号: 0   ∞ 关联项目: 1	8 4 0	app = a 云桃号: 0 🔹 关联项目: 0	8 A Ø
	<b>devops 示</b> ▲ 云帐号: 0 ● 关联项目: 0	 8 A Ø	client 📻 ▲ 云帐号: 0 🛛 🐠 关联项目: 0	 8 A Ø	auto-testing హ ▲ 云帐号: 0 🔹 关联项目: 0	 8 A Ø
	spring-demo ☴ ▲ 云桃号: 0 🛛 🐽 关联项目: 0	 8 # \$	web ☴ ▲ 五帐号: 0 🔹 关联项目: 2		flaskapp ☴ ▲ 五张号:0   ◆ 关联项目:2	 2 A Ø
	testing11111 ☴ ▲ 云桃号: 0 🛛 ● 关联项目: 0	 8 # 0	testing 〒 ▲ 五紙号: 0 🔹 关联项目: 1	 8 A Ø	devopssolution ☴ ▲ 五张号: 0   ◆ 关联项目: 1	 8 A Ø

1. 单击右上角的创建流程。



←         dev ~         部署流程           状态 全部 ~         任务数量 2 ~            26	创建部署流程 复制现有流程 Kubernetes		包建筑理
> dev	部署 Helm 应用到 Kubernetes 集群 配置Bake (Manife 部署 (Manifest)	(1) 请在左侧选择部署流程	启动 编辑 …
	部署 Deployment 和 Service 到 Kubernetes 集群 配置 部署 Deploy 部署 Service		
	部看到 Kubernetes 集群前进行人工确认 Bake (Manife 部		
	并行部署 Deployments 和 Service 配置 部署 Deploy 部署 Service 部署 Deploy 部署 Service		

2. 您可以复制在其他应用中创建的流程,或通过空白流程自行创建。



创建部署流程		创建流程
复制现有流程   Kubernetes   腾讯云弹性伸缩 ————————————————————————————————————		
部署 Helm 应用到 Kubernetes 集群 配置 Bake (Manife 部署 (Manifest)	🕤 请在左侧选择部署流程	启动 编辑 …
部署 Deployment 和 Service 到 Kubernetes 集群 配置 部署 Deploy 部署 Service		
部署到 Kubernetes 集群前进行人工确认		
并行部署 Deployments 和 Services		
部署 Deploy 部署 Service		

# 基础配置

应用的基础配置可以理解为构建整体的初始环节,既可以设置触发条件,也可以配置部署流程的通知方式等。



← GO <u>≉</u>	▶ 基础配置
OLE	

#### 自动触发器

自动触发器支持 CODING Docker 制品仓库、TCR 个人版仓库触发器、Git 仓库触发器等触发条件。

#### 添加部署流程参数

在部署流程配置页面,单击**添加参数**,即可开始填写参数。



← GO <u>#</u>	●■
↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓	执行选项 自动触发器 启动参数 通知 描述
制品	✓ 启动参数 □
	参数名
◆ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■	请输入
可在阶段中配置	心填参数
	参数类型
	字符串    ▼
	默认值
	请输入
	描述信息
	·阿·田/X
	→ 添加启动参数

## 添加阶段

在部署流程配置页面单击加号即可添加新的阶段,右侧列表中支持选择阶段类型。



← GO <u>2</u>	选择阶段	<b>没</b>	搜索阶段名称	Q
●	依赖阶段: Kubern	请选择阶段 。 netes 通用类型		
阶段类 添加阶段 + 可	8	<b>部署 (Manifest)</b> 部署 yaml/json 格式的 Kubernetes manifest 文件		选择
	*	<b>过滤 (Manifest)</b> 过滤 (Manifest)		选择
	<b>(</b>	<b>扩缩容 (Manifest)</b> 对 Kubernetes 对象执行扩缩容		选择
	<b>(</b>	<b>回滚 (Manifest)</b> 回滚至目标版本		选择
	<b>(</b>	<b>删除 (Manifest)</b> 删除 Kubernetes (Manifest)		选择
	<b>(</b>	<b>Bake (Manifest)</b> 使用 Helm Bake manifest 文件		选择
	*	Patch (Manifest) Patch a Kubernetes object in place.		选择

# 执行部署流程

部署流程配置完成后,您可以通过设置好的触发器依提交自动执行,或在持续部署中提交发布单手动触发部署流程。



## 部署流程配置

腾讯云

部署流程支持删除、禁用、锁定、查看历史版本与编辑 JSON 配置。

← app 测试 <u>@</u>		撤销变更	已同步 …
			删除
↓ 基础配置	●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●		禁用
	阶段类型:部署 (Manifest)		锁定
制品			编辑 JSON 配置
🐠 flaskapp			查看历史版本



#### 删除部署流程

设置后,将删除此部署流程。

#### 禁用部署流程

设置后,将禁止任意触发器启动部署流程,包括手动触发。可以选择在团队内整体禁用或仅在项目内禁用。

← app 测试 ⊿			撤销变更	已同步 …
	确认禁用部署流程?	×		删除
↓ 基础配置	● 全局埜田 ○ 顶日内埜田			禁用
				锁定
制品	全局禁用后,任何方式都不能触发部署流程执行。			编辑 JSON 配置
🐠 flaskapp				查看历史版本
	祭用			
		_		

# 锁定部署流程

锁定部署流程后,不能在应用中心对部署流程进行任何修改。

6	← test23 🧕				撤销变更	已同步 …
						删除
	• 基础配置	(+) 适行服	确认锁定部署流程?	×		禁用
		阶段类				锁定
	制品		锁定后,不能在应用中心对部署流程进行任何修改。			编辑 JSON 配置
	~ <b>—</b> -		<b>描述 ②</b> 请输入			查看历史版本
	*					
	暂无制品		锁定取消			



### 查看修订历史

保存新的部署流程配置后,旧版本将会添加到修订历史。您可以在修订历史页对比各版本信息,选择并还原到任意历 史版本。

2021-06-16 10:25:41 (当前版本) pppl: coding 2021-06-15 15:02:05 secset: "", expectedArtifacts": [ { "defaultArtifact": { "customKind": false, "id": "fb25B6 "yersion": "" }, "difaultArtifact": { "customKind": false, "id": "flaskapp", "id": "flaskapp", "ide: "id=: id=: id=: id=: id=: id=: id=: id=:	2021-06-16 10:25:41 (当前版本)  appl: 2021-06-15 15:02:05  besc": "",  expectedArtifact": {     "defaultArtifact": {     "customKind": false,     "id": "8b215!     ",     "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp",     "pkgName": "flaskapp",     "type": "coding_docker/image",     "version": ""     },     "displayName": "flaskapp",     "id": "cflaedbd-     ",     "matchArtifact": {     "customKind": false,     "id": "flaskapp",     "id": "flaskapp",     "ustomKind": false,     "id": "flaskapp",     "ustomKind": false,     "id": "flaskapp",     "id":	2021-06-16 10:25:41 (当前版本) pppl 2021-06-15 15:02:05 lesc": "", "cotingNickname": "flaskapp", "id": "8b215t ", "maeme": "lkkprod-docker.pkg.coding.net/cd-demo/flaskapp", "perference": "lkprod-docker.pkg.coding.net/cd-demo/flaskapp", "type": "coding_docker/image", "type": "coding_docker/image", "type": "laskapp", "type": "flaskapp", "type": "flaskapp", "type: "type: "flaskapp", "type: "flaskapp", "type: "flaskapp", "type: "type: type: ty	2021-06-16 10:25:41 (当前版本) ll: 2021-06-15 15:02:05 ir", weetedArtifacts": [ "defaultArtifact": { "customKind": false, "id": "8b215t "name": "lhkprod-docker.pkg.coding.net/cd-demo/cd-demo/flaskapp", "parentType": "docker", "pkgName": "flaskapp",	ion": "flaskapp", ckname": "主账号", ",
<pre>ppl: odif 2021-06-15 15:02:05 esc": "", xpectedArtifacts": [ { "defaultArtifact": { "ustomKind": false, "id": "8b215L "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp", "perentType": "docker", "ykgName": "flaskapp", "type": "coding_docker/image", "version": "" }, "displayName": "flaskapp", "id": "flaskapp", "type": "codiag_docker/image", "version": "" }, "displayName": "flaskapp", "id": "flaskapp", "id"</pre>	<pre>ppl: odii 2021-06-15 15:02:05 esc": "", xpectedArtifacts": [ { "defaultArtifact": { "customKind": false, "id": "8b215! "name": "lhkprod-docker.pkg.coding.net/cd-demo/cd-demo/flaskapp", "parentType": "docker", "kypVame": "flaskapp", "type": "coding_docker/image", "version": "" }, "displayName": "flaskapp", "type: "coding_docker/image", "version": "" }, "displayName": "flaskapp", "id": "cflaedbd- ", "matchArtifact": { "customKind": false, "displayName": "flaskapp", "id": "folsece "displayName": "flaskapp", "id": "cflaedbd- ", "matchArtifact": { "customKind": false, "displayName": matchArtifact": { "customKind": false, "displayName": false, "displayName": false, "displayName":</pre>	<pre>ppl: odi 2021-06-15 15:02:05 esc": "", xpectedArtifacts": { "defaultArtifact": { "ustomKind": false, "id": "fbz7bd-docker.pkg.coding.net/cd-demo/flaskapp", "parentType": "locker", "ustomKind": falskapp", "reference": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp" "ype": "coding_docker/image", "version": "" }, "displayName": "flaskapp", "id": "flasedbd- ", "matchArtifact": { "ustomKind": false, "id": "fflasebbe- ", "matchArtifact": { "ustomKind": false, "id": "fflasebbe- ", "matchArtifact": { "ustomKind": false, "id": "fflasebbe- ", "matchArtifact": { "ustomKind": false, "id": "fflasebbe- ", "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp", "atchArtifact": { "ustomKind": false, "id": "fflasebbe- ", "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp", "atchArtifact": { "ustomKind": false, "id": "fflasebbe- ", "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp", "docker] "ustomKind": false, "id": "fflasebbe- ", "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp", "docker] "ustomKind": false, "id": "fflasebbe- ", "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp", "docker] "ustomKind": false, "id": "fflasebbe- ", "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp", "docker] "docker] "ustomKind": false, "id": "fflasebbe- ", "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp", "docker] "ustomKind": false, "id": "fflasebbe- ", "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp", "docker] "docker] "ustomKind": false, "id": "fflasebbe- ", "ustomKind": false, "id": "fflasebbe- ", "ustomKind": false, "id": "fflasebb- ", "ustomKind": false, "id": "fflasebb- ", "ustomKind": false, "id": falseb- "id": falseb- "id": falseb- "id": falseb-</pre>	<pre>"applicati "applicati "codingNid" "defaultArtifacts": [ "defaultArtifact": { "customKind": false, "id": "8b215t "name": "lhkprod-docker.pkg.coding.net/cd-demo/cd-demo/flaskapp", "parentType": "docker", "pkgName": "flaskapp",</pre>	ion": "flaskapp", ckname": "主账号", ",
<pre>dould esc": "", mesc": "", "desc": ", "desc": ",</pre>	<pre>could could c</pre>	<pre>dual """, "", "", "", "", "", "", "", "", "</pre>	<pre>could a could a c</pre>	
<pre>xxpectedArtifacts": [ {     "defaultArtifact": {     "customKind": false,     "id": "Bb215i</pre>	<pre>xpectedArtifacts:: [ {     "defaultArtifact:: {         "customKind": false,         "id": "8b215!         ",         "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp",         "parentType": "docker",         "pkgName": "flaskapp",         "reference": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp"         "type": "coding_docker/image",         "version": ""     },     "displayName": "flaskapp",     ",     "matchArtifact": {         "customKind": false,         ",         "matchArtifact:: {         "customKind": false,         "id": "efapsee         "         "idsection"         "type: "coding_docker/image",         "version": ""     },     "displayName": "flaskapp",     ",     "matchArtifact": {         "customKind": false,         "idsection"         "ustomKind": false,         "idsection"         "ustomKind": false,         "idsection"         "ustomKind": false,         "idsection"         "ustomKind": false,         "usto</pre>	<pre>xxpectedArtifacts": [ {     "defaultArtifact": {         "customKind": false,         "id": "8b215t         ",         "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp",         "parentType": "docker",         "pkgName": "flaskapp",         "reference": "lhkprod-docker.pkg.coding.net/cd-demo/cd-demo/flaskapp"         "type": "coding_docker/image",         "version": ""         },         "displayName": "flaskapp",         "id": "flaskapp",</pre>	<pre>vectedArtifacts": [ "defaultArtifact": {     "customKind": false,     "id": "8b215t     ",     "name": "lhkprod-docker.pkg.coding.net/cd-demo/cd-demo/flaskapp",     "parentType": "docker",     "pkgName": "flaskapp",     "</pre>	
<pre>{   "defaultArtifact": {     "customKind": false,     "id": "8b215t     ",     "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp",     "parentType": "docker",     "pkgName": "flaskapp",     "reference": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp"     "type": "coding_docker/image",     "version": ""     },     "displayName": "flaskapp",     "id": "flaskapp",     "id: "cflaedbd-     ",     "matchArtifact": {         "customKind": false,         "id: "fla5686-         ",         "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp",     "jestion": ""     }     "defaultArtifact": {         "customKind": false,         "id: "flo25686-         ",         "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp",         "parentType": "docker",     } } </pre>	<pre>{     "defaultArtifact": {         "customKind": false,         "id": "8b215t         ",         "name": "lhkprod-docker.pkg.coding.net/cd-demo/cd-demo/flaskapp",         "parentType": "docker",         "pkgName": "flaskapp",         "reference": "lhkprod-docker.pkg.coding.net/cd-demo/cd-demo/flaskapp"         "type": "coding_docker.pkg.coding.net/cd-demo/cd-demo/flaskapp"         "type": "coding_docker.jkg.coding.net/cd-demo/cd-demo/flaskapp"         "type": "coding_docker.jkg.coding.net/cd-demo/jkg.coding.net/cd-demo/jkg.coding.net/cd-demo/jkg.coding.net/cd-demo/jkg.coding.net/cd-demo/jkg.coding.net/cd-demo/jkg.coding.net/cd-demo/jkg.coding.net/jkg.coding.net/jkg.coding.net/jkg.coding.net/jkg.coding.net/jkg.coding.net/jkg.coding.net/jkg.coding.net/jkg.coding.net/jkg.coding.net/jkg.coding.net/jkg.coding.n</pre>	<pre>{     "defaultArtifact": {         "customKind": false,         "id": "8b215!         ",         "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp",         "preference": "lhkprod-docker.pkg.coding.net/cd-demo/cd-demo/flaskapp"         "type": "coding_docker.pkg.coding.net/cd-demo/cd-demo/flaskapp"         "type": "coding_docker.pkg.coding.net/cd-demo/flaskapp"         "type": "flaskapp",         "version": ""     },         "displayName": "flaskapp",         "id": "flaskapp",         "id": "flaskapp",         "id": "flaskapp",         ",         "matchArtifact": {         "customKind": false,         "id": "flo25666-         ",         "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp",         "         "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp",         "         "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp",         "         "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp",         "         "         "</pre>	<pre>"defaultArtifact": {     "customKind": false,     "id": "8b215t     ",     "name": "lhkprod-docker.pkg.coding.net/cd-demo/cd-demo/flaskapp",     "parentType": "docker",     "pkgName": "flaskapp",</pre>	
<pre>"defaultArtifact": {     "customKind": false,     "id": "8b215!     ",     "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp",     "parentType": "docker",     "pkgName": "flaskapp",     "reference": "lhkprod-docker.pkg.coding.net/cd-demo/cd-demo/flaskapp"     "type": "coding_docker/image",     "version": ""     },     "displayName": "flaskapp",     ",     "matchArtifact": {         "customKind": false,         "id": "cflaedbd-         ",         "matchArtifact": {         "customKind": false,         "id": "ff025686-         ",         "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp",         "parentType": "docker",         "docker",         "state = ",         "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp",         "parentType": "docker",         "state = ",         "state = ",</pre>	<pre>"defaultArtifact": {     "customKind": false,     "id": "8b215}     ",     "name": "lhkprod-docker.pkg.coding.net/cd-demo/cd-demo/flaskapp",     "parentType": "docker",     "pkgName": "flaskapp",     "reference": "lhkprod-docker.pkg.coding.net/cd-demo/cd-demo/flaskapp"     "type": "coding_docker/image",     "version": ""     },     "idsplayName": "flaskapp",     "idisplayName": "flaskapp",     "id": "cflaedbd-     ",     "matchArtifact": {         "customKind": false,         "."         "tforefeence": "</pre>	<pre>"defaultArtifact": {     "customKind": false,     "id": "8b215t ",     "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp",     "parentType": "docker",     "pkgName": "flaskapp",     "reference": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp"     "type": "coding_docker/image",     "version": ""     },     "displayName": "flaskapp",     "id": "cflaedbd-",     ",     "matchArtifact": {         "customKind": false,         "id": "ff025686-",         ",         "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp",         "id": "ff025686-",         ",         "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp",         "         "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp",         "         "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp",         "         "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp",         "         "         "</pre>	<pre>"defaultArtifact": {     "customKind": false,     "id": "8b215t     ",     "name": "lhkprod-docker.pkg.coding.net/cd-demo/cd-demo/flaskapp",     "parentType": "docker",     "pkgName": "flaskapp",</pre>	
<pre>"customKind": false, "id": "8b215i ", "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp", "parentType": "docker", "pkgName": "flaskapp", "reference": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp" "type": "coding_docker/image", "version": "" }, "displayName": "flaskapp", "id": "cflaedbd-", "matchArtifact": { "customKind": false, "id": "ff025686-", "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp", "parentType": "docker",</pre>	<pre>"customKind": false, "id": "8b215t ", "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp", "parentType": "docker", "pkgName": "flaskapp", "reference": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp" "type": "coding_docker/image", "type": "coding_docker/image", "version": "" }, "displayName": "flaskapp", "id": "cflaedbd-", "matchArtifact": { "customKind": false, "'d". "f602F000 "</pre>	<pre>"customKind": false, "id": "8b215t ", "name": "lhkprod-docker.pkg.coding.net/cd-demo/claskapp", "parentType": "docker", "pkgName": "flaskapp", "reference": "lhkprod-docker.pkg.coding.net/cd-demo/cd-demo/flaskapp" "type": "coding_docker/image", "version": "" }, "displayName": "flaskapp", "id": "cflaedbd- ", "matchArtifact": { "customKind": false, "id": "ff025686- ", "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp",</pre>	<pre>"customKind": false, "id": "8b215t"", "name": "lhkprod-docker.pkg.coding.net/cd-demo/cd-demo/flaskapp", "parentType": "docker", "pkgName": "flaskapp",</pre>	
<pre>"id": "8b215t ", "name": "lhkprod-docker.pkg.coding.net/cd-demo/cd-demo/flaskapp", "parentType": "docker", "pkgName": "flaskapp", "reference": "lhkprod-docker.pkg.coding.net/cd-demo/cd-demo/flaskapp" "type": "coding_docker/image", "version": "" }, "displayName": "flaskapp", "id": "cflaedbd- ", "matchArtifact": { "customKind": false, "id": "ff025686- ", "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp", "parentType": "docker",</pre>	<pre>"id": "8b215t ", "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp", "parentType": "docker", "pkgName": "flaskapp", "reference": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp" "type": "coding_docker/image", "type": "coding_docker/image", "version": "" }, "displayName": "flaskapp", "id": "cflaedbd-"", "matchArtifact": { "customKind": false, "tdu. "ff02f66e."</pre>	<pre>"id": "8b215t ", "name": "lhkprod-docker.pkg.coding.net/cd-demo/cd-demo/flaskapp", "parentType": "docker", "pkgName": "flaskapp", "reference": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp" "type": "coding_docker/image", "version": "" }, "displayName": "flaskapp", "id": "cflaedbd- ", "matchArtifact": { "customKind": false, "id": "ff025686- ", "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp",</pre>	<pre>"id": "8b215t ", "name": "lhkprod-docker.pkg.coding.net/cd-demo/cd-demo/flaskapp", "parentType": "docker", "pkgName": "flaskapp",</pre>	
<pre>"name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp",     "parentType": "docker",     "pkgName": "flaskapp",     "reference": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp"     "type": "coding_docker/image",     "version": ""     },     "displayName": "flaskapp",     "id": "flaedbd-</pre>	<pre>"name": "lhkprod-docker.pkg.coding.net/cd-demo/cd-demo/flaskapp",     "parentType": "docker",     "pkgName": "flaskapp",     "reference": "lhkprod-docker.pkg.coding.net/cd-demo/cd-demo/flaskapp"     "type": "coding_docker/image",     "version": ""     ,     "displayName": "flaskapp",     "id": "cflaedbd-</pre>	<pre>"name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp",     "parentType": "docker",     "pkgName": "flaskapp",     "reference": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp"     "type": "coding_docker/image",     "version": ""     },     "displayName": "flaskapp",     "idisplayName": flaskapp",     "idisplayName": flaskapp",     "idisplayName": flaskapp",     "idisplayName": flaskapp",     "idisplayName": flaskapp     ",     "matchArtifact": {         "customKind": false,         ",         "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp",         "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp",         "displayName": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp",         "displayName":</pre>	<pre>"name": "lhkprod-docker.pkg.coding.net/cd-demo/cd-demo/flaskapp", "parentType": "docker", "pkgName": "flaskapp",</pre>	
<pre>"parentType": "docker", "pkgName": "flaskapp", "reference": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp" "type": "coding_docker/image", "version": "" }, "displayName": "flaskapp", "id": "cflaedbd-", "matchArtifact": { "customKind": false, "id": "ff025686-", "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp", "parentType": "docker",</pre>	<pre>"parentType": "docker", "pkgName": "flaskapp", "reference": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp" "type": "coding_docker/image", "version": "" }, "displayName": "flaskapp", "id": "cflaedbd-"", "matchArtifact": { "customKind": false, "'du. "ff02fcee.""</pre>	<pre>"parentType": "docker", "pkgName": "flaskapp", "reference": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp" "type": "coding_docker/image", "version": "" }, "displayName": "flaskapp", "id": "cflaedbd- ", "matchArtifact": { "customKind": false, "id": "ff025686- ", "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp",</pre>	<pre>"parentType": "docker",     "pkgName": "flaskapp",</pre>	
<pre>"pkgName": "flaskapp", "reference": "lhkprod-docker.pkg.coding.net/cd-demo/cd-demo/flaskapp" "type": "coding_docker/image", "version": "" }, "displayName": "flaskapp", "id": "flaedbd- ", "matchArtifact": { "customKind": false, "id": "ff025686- ", "name": "lhkprod-docker.pkg.coding.net/cd-demo/cd-demo/flaskapp", "parentType": "docker",</pre>	<pre>"pkgName": "flaskapp", "reference": "lhkprod-docker.pkg.coding.net/cd-demo/cd-demo/flaskapp" "type": "coding_docker/image", "version": "" }, "displayName": "flaskapp", "id": "cflaedbd-", "matchArtifact": { "customKind": false, "'du", "ff02666</pre>	<pre>"pkgName": "flaskapp", "reference": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp" "type": "coding_docker/image", "version": "" }, "displayName": "flaskapp", "id": "cflaedbd- ", "matchArtifact": { "customKind": false, "id": "ff025686- ", "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp",</pre>	"pkgName": "flaskapp",	
<pre>"reference": "lhkprod-docker.pkg.coding.net/cd-demo/cd-demo/flaskapp" "type": "coding_docker/image", "version": "" }, "displayName": "flaskapp", "id": "cflaedbd- ", "matchArtifact": {     "customKind": false,     "id": "ff025686- ",     "name": "lhkprod-docker.pkg.coding.net/cd-demo/cd-demo/flaskapp",     "parentType": "docker",</pre>	<pre>"reference": "lhkprod-docker.pkg.coding.net/cd-demo/cd-demo/flaskapp" "type": "coding_docker/image", "version": "" }, "displayName": "flaskapp", "id": "cflaedbd- ", "matchArtifact": {     "customKind": false,     ""displayName": "flaskapp" """"""""""""""""""""""""""""""""""</pre>	<pre>"reference": "lhkprod-docker.pkg.coding.net/cd-demo/cd-demo/flaskapp" "type": "coding_docker/image", "version": "" }, "displayName": "flaskapp", "id": "cflaedbd- ", "matchArtifact": {     "customKind": false,     "id": "ff025686- ", "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp",</pre>		
<pre>"type": "coding_docker/image", "version": "" }, "displayName": "flaskapp", "id": "cflaedbd- ", "matchArtifact": {     "customKind": false,     "id": "ff025686- ",     "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp",     "parentType": "docker",</pre>	<pre>"type": "coling_docker/image",     "version": "" }, "displayName": "flaskapp", "id": "cflaedbd- ", "matchArtifact": {     "customKind": false,     "tury "#f625626</pre>	<pre>"type": "coding_docker/image", "version": "" }, "displayName": "flaskapp", "id": "cflaedbd- ", "matchArtifact": {     "customKind": false,     "id": "ff025686- ",     "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp",</pre>	"reference": "lhkprod-docker.pkg.coding.net/cd-demo/cd-demo/flaskapp"	
<pre>version: }, "displayName": "flaskapp", "id": "cflaedbd", "matchArtifact": { "customKind": false, "id": "ff025686", "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp", "parentType": "docker",</pre>	<pre>version : }, "displayName": "flaskapp", "id": "cflaedbd- ", "matchArtifact": {     "customKind": false,     "id": "false,     "jd": "</pre>	<pre>version: }, "displayName": "flaskapp", "id": "cflaedbd</pre>	"type": "coding_docker/image",	
<pre>/, "displayName": "flaskapp", "id": "cflaedbd- ", "matchArtifact": {     "customKind": false,     "id": "ff025686- ",     "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp",     "parentType": "docker",</pre>	<pre>// "displayName": "flaskapp", "id": "cflaedbd- ", "matchArtifact": {     "customKind": false,     "'d": "false,     "'d": "false,</pre>	<pre>',   "displayName': "flaskapp",   "id": "cflaedbd- ",   "matchArtifact": {     "customKind": false,     "id": "ff025686- ",     "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp",</pre>	version :	
<pre>"id": "cflaedbd- ", "matchArtifact": {     "customkind": false,     "id": "ff025686- ",     "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp",     "parentType": "docker",</pre>	"id": "cflaedbd- ", "matchArtifact": { "customKind": false,	<pre>"id": "cflaedbd- ", "matchArtifact": {     "customKind": false,     "id": "ff025686- ",     "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp",</pre>	J, "disnlavAlame", "flaskann"	
<pre>"matchArtifact": {     "customKind": false,     "id": "ff025686- ",     "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp",     "parentType": "docker",</pre>	<pre>"matchArtifact": {     "customKind": false,     "identification": fal</pre>	<pre>"matchArtifact": {     "customKind": false,     "id": "ff025686-     ",     "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp",</pre>	"id": "cflaedhd- ".	
<pre>"customKind": false, "id": "ff025686-"", "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp", "parentType": "docker",</pre>	"customKind": false,	<pre>"customKind": false, "id": "ff025686</pre>	"matchArtifact": {	
<pre>"id": "ff025686-""", "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp", "parentType": "docker",</pre>		"id": "ff025686-"", "name": "lhkprod-docker.pkg.coding.net/cd-demo/flaskapp",	"customKind": false.	
<pre>"name": "lhkprod-docker.pkg.coding.net/cd-demo/cd-demo/flaskapp", "parentType": "docker",</pre>	-00022011 : 01	"name": "lhkprod-docker.pkg.coding.net/cd-demo/cd-demo/flaskapp",	"id": "ff025686-	
"parentType": "docker",	"name": "lhkprod-docker.pkg.coding.net/cd-demo/cd-demo/flaskapp",		"name": "lhkprod-docker.pkg.coding.net/cd-demo/cd-demo/flaskapp",	
	"parentType": "docker",	"parentType": "docker",	"parentType": "docker",	
"pkoName": "flaskapp".			"pkgName": "flaskapp",	
	"pkgName": "flaskapp",	"pkqName": "flaskapp",		
	<pre>"name": "lhkprod-docker.pkg.coding.net/cd-demo/cd-demo/flaskapp", "parentType": "docker",</pre>	"parentType": "docker",	<pre>"matchArtifact": {     "customKind": false,     "id": "ff025686-     ",     "name": "lhkprod-docker.pkg.coding.net/cd-demo/cd-demo/flaskapp",     "parentType": "docker",</pre>	
"pkoName": "flaskapp".			"pkgName": "flaskapp",	
	"pkgName": "flaskapp",	"pkgName": "flaskapp",		~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

关闭

### 编辑 JSON 配置

在部署控制台中所做的任何更改最终都会以 JSON 格式文件保存,直接编辑部署流程的 JSON 内容可以为部署流 程添加新属性或自定义 UI 界面尚未显示的配置项。

▲ 注意:

Х

此种方式允许用户将在文本框内自由编辑部署流程,但有可能会破坏部署流程的可用性,我们提供了从修订 历史中恢复到任意指定版本的能力。

#### 编辑 JSON 配置

🕥 腾讯云

1	4	Martin Lawris Martin Lawris Martin Lawris Martin Lawris Martin Lawris
2	「 "codingNickname": "主账号",	With a straight of the straigh
3	"desc": "",	
4	<pre>"expectedArtifacts": [</pre>	Contraction of the Contraction o
5	{	
6	<pre>"defaultArtifact": {</pre>	Apres 1
7	"customKind": false,	
8	"id": "8b215b5c ",	
9	"name": "lhkprod-docker.pkg.coding.net/cd-demo/cd-demo/flaskapp",	
10	"parentType": "docker",	
11	"pkgName": "flaskapp",	A REAL PROPERTY
12	<pre>"reference": "lhkprod-docker.pkg.coding.net/cd-demo/cd-demo/flaskapp",</pre>	
13	<pre>"type": "coding_docker/image",</pre>	Construction of the second sec
14	"version": ""	
15	· · · · · · · · · · · · · · · · · · ·	Annual
16	"displayName": "flaskapp",	
17	"id": "cflaedbd ",	
18	<pre>"matchArtifact": {</pre>	
19	"customKind": false,	
20	"id": "ff025 ",	
21	"name": "lhkprod-docker.pkg.coding.net/cd-demo/cd-demo/flaskapp",	
22	"parentType": "docker",	
23	"pkgName": "flaskapp",	
24	"type": "coding_docker/image"	
25	3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3	
26	"useDefaultArtifact": true,	
27	"usePriorArtifact": false	
28	) }	

取消 保存变更



# 部署方式 自动发布 Docker 制品时触发

最近更新时间: 2023-09-11 16:05:28

CODING 持续部署的一大优势在于能够便捷的集成上下游产品为工作流,下文将演示如何通过三个步骤实现**持续集** 成任务推送制品 > 制品仓库镜像更新 > 触发部署流程这一基础的自动化流水线配置。

### 操作步骤

#### 步骤1: 应用与项目关联

部署流程控制台中的**应用**需提前与项目相关联。前往应用中心,单击应用中的**关联项目图标**,选择持续集成配置所在 的项目并进行关联。

应用 体验新版→					创建应用
云账号: 全部 ▼ 关联项目: 全部 ▼ 排序方式: 更新时间倒序 ▼		Q			
dev 📻		new-cd 📻		app 〒	
▲ 云帐号: 0 🔹 关联项目: 7	8 A Ø	2 云张号: 0 🔹 关联项目: 1	8 # 0	▲ 云张号: 0   ● 关联项目: 0	8 A Ø
flaskapp ☴		auto-testing		spring-demo ☴	
▲ 云张号: 0   ● 关联项目: 2	关联项目		8 A Ø	▲ 云张号: 0   ● 关联项目: 0	8 A Ø
<b>client ☴</b>		web 〒		devops ☴	
▲ 云张号: 0   ● 关联项目: 0	5 A 0	▲ 云帐号: 0   ● 关联项目: 2	5 # \$	▲ 云帐号: 0   ● 关联项目: 0	8 A Ø
testing11111 〒		testing 〒		devops-solution హ	
2 云帐号: 0 ◆ 关联项目: 0	8 A 0	▲ 云张号: 0 🔹 关联项目: 1	8 # \$	畫 云张号: 0   ● 关联项目: 1	8 A Ø

## 步骤2: 配置持续集成

此步骤使用持续集成将制品推送至制品仓库。您可以通过持续集成计划模板创建,或直接编写 Jenkinsfile 手动增加此阶段。



🗲 选择构	<b>]建计划</b> 模版									自定义构	建过程
构建计划是持续	使集成的基本单元	元,在这里你可以	央速创建一个构建计划	创,更多内容可以	到构建计划详情	中进行配置。	查看帮助文档	Z			
全部	团队模版	编程语言	Serverless	镜像仓库	制品库	部署	基础	API 文档		请输入模版关键字进行搜	索 <b>Q</b>
	<b>构建镜像并</b> 基于源代码变	• <b>推送到 TCR 企</b> 变更自动触发镜像 <sup>4</sup>	<b>业版</b> 沟建,并推送镜像至铜	容器镜像服务TCR	企业版		<b>构建镜</b> 基于源代	<b>像并推送至 TCR</b> <sup>·</sup> 记码变更自动触发镜值	<b>个人版(容器服</b> <sub>象构建,并推送镜(</sub>	<b>务–镜像仓库)</b> 像至容器镜像服务TCR个人	版
•	<b>CODING D</b> 将一个构建完	<b>Docker 镜像推</b> 说 記毕的 Docker 镜f	<b>生</b> 象推送到当前项目下的	勺 Docker 制品库	中						
若没有找到合适	适的模版,可选择	译自定义构建过程									
	<b>自定义构建</b> 允许您根据 。	<b>过程</b> Jenkinsfile 的规范	可来随意定制持续集成	流水线过程。							

#### 在持续集成流程中手动增加此阶段:

← flask-docker ☑    基础信息    流程配置    触发规则   变量与缓存   通知提醒	■ 前往最新构建 操作 > 立即构建
静态配置的 Jenkinsfile ⑦ 图形化编辑器 文本编辑器	♦ 环境变量 丢弃修改 保存
5-1       构建 Docker 镜像       +       +       #送到 CODING D       +       +       #         ↓       执行 Shell 脚本       +       +       #       +       #       +       #       #       +       +       #       #       +       #       #       #       +       #<	插件配置       高级配置         Pipeline 脚本 *         1 docker.withRegistry(         2         "\${env.CCI_CURRENT_WEB_PROTOCOL}://\${env.CODING_D         3         "\${env.CODING_ARTIFACTS_CREDENTIALS_ID}"         4 ) {       {         5         docker.image("\${CODING_DOCKER_IMAGE_NAME}:\${env.DOC         6 }

Jenkinsfile 参考:





"\${env.couing_AkiifAci5_ckedeniiAl5_iD}"
<pre>docker.image("\${CODING_DOCKER_IMAGE_NAME}:\${env.DOCKER_IMAGE_VERSION}"). push()</pre>

#### 步骤3: 根据制品镜像版本触发

前往持续部署中的应用部署流程,单击**基础配置**中的触发器启用开关。此处选择通过 CODING Docker 制品更新 触发,将监听关联项目中制品版本号。若持续集成将制品推送至制品仓库时,将自动触发部署流程;选择**自定义**能够 监听其他项目的制品仓库更新情况。

除了通过 CODING Docker 制品更新触发,您还可以通过 Git 仓库或定时器触发此部署流程。



# 在构建计划中添加部署阶段

最近更新时间: 2023-09-11 16:05:28

在持续集成中触发部署时请提前前往应用中心,将应用与项目关联。

应用 体验新版→					创建应用
云账号:全部 → 关联项目:全部 → 排序方式:更新时间倒序 →		Q			
dev 🚍		new-cd 🚍		app 🚍	
🛓 云帐号: 0 🛛 🔹 关联项目: 7	8 A 0	💄 云帐号: 0 🛛 🔹 关联项目: 1	8 A Ø	💄 云帐号: 0 🛛 🐟 关联项目: 0	8 A Ø
flaskapp 🚍		auto-testing 🚍		spring-demo 🚍	
🛓 云帐号: 0 🛛 🐟 关联项目: 2	关联项目	💄 云帐号: 0 🛛 🔹 关联项目: 0	8 A Ø	💄 云帐号:0 🛛 🔹 关联项目:0	8 A Ø
client 🚍		web 🚍		devops 🚍	
🛓 云帐号: 0 🛛 🔹 关联项目: 0	8 A Ø	💄 云帐号: 0 🛛 🔹 关联项目: 2	8 A Ø	🛓 云帐号: 0 🛛 🐟 关联项目: 0	8 A Ø
testing11111 🚍		testing 🚍		devops-solution 🚍	
🛓 云帐号: 0 🛛 🐟 关联项目: 0	8 A Ø	💄 云帐号: 0 🛛 🔹 关联项目: 1	8 A Ø	💄 云帐号: 0 🛛 🔹 关联项目: 1	8 A Ø

本文给出了两种设置方法,您可以按照需求有选择性阅读。

- 直接使用构建计划模板
- 在已有构建计划中添加部署阶段

# 使用构建计划模板

请在应用中心关联涉及到相关集群的云账号,详情请参见 云账号 。 单击项目内左侧产品栏**持续集成**,右上角创建构建计划,选择**部署**分类下的**推送到 Kubernetes** 模板 。



按照模板提示选择相应的制品仓库、远端集群地址等信息,完成后勾选创建后触发构建。

腾讯云

🔗 腾讯云	
-------	--

2	构建 Docker 镜像	1
	Docker 镜像名称 *	stage('构建镜像并推送到 CODING Docker 制品库') {     steps {         start {
	my-docker-image	<pre>script {     docker.withRegistry(         "\${CCI_CURRENT_WEB_PROTOCOL}://\${CODING_DOCKER_REG_HOST}",</pre>
	Dockerfile 文件位置 *	<pre>"\${CODING_ARTIFACTS_CREDENTIALS_ID}" ) {     def dockerImage = docker.build("\${CODING DOCKER IMAGE NAME}:\${DOCKER IMAGE VERSION}".</pre>
	Dockerfile	<pre>dockerImage.push() }</pre>
	Docker 构建目录 *	} } }
		stage('部署到远端 Kubernetes 集群') {     steps {         cdPenlov([
	Docker 镜像版本 *	<pre>deployType: 'PATCH_IMAGE', application: "\${CCI_CURRENT_TEAM}",</pre>
	分支名-修订版本号 (\${GIT_LOCAL_BRANCH:-branch}-\$▼	<pre>pipelineName: "\${PROJECT_NAME}-\${CCI_JOB_NAME}-\${CD_CREDENTIAL_INDEX}", image: "\${CODING_DOCKER_REG_HOST}/\${CODING_DOCKER_IMAGE_NAME}:\${DOCKER_IMAGE_VERSION}", cloudAccountName: "\${CD_ACCOUNT_NAME}",</pre>
3	推送到 CODING Docker 制品库	namespace: "\${CD_NAMESPACE_NAME}", manifestType: "\${CD_MANIFEST_TYPE}", manifestName: "\${CD_MANIFEST_NAME}",
	Docker 制品库 *	<pre>containerName: "\${CD_CONTAINER_NAME}", credentialId: "\${CD_CREDENTIAL_ID}", personalAccessToken: "\${CD_PERSONAL_ACCESS_TOKEN}".</pre>
	🔹 cd-demo 🔹	1) }
	· · · · · · · · · · · · · · · · · · ·	} }
4	th, Eransian Lapounces with	
	集群 *	
	Go	
	命名空间 *	
	demo	

设置完成后,运行持续构建计划即可完成自动发布。

# 添加部署阶段

在此方法中您可以在构建计划 > 流程配置中使用编辑器或填写命令添加部署阶段。

# 图形化编辑器

在已有构建计划设置中添加部署阶段,填写镜像地址、集群与命名空间等关键信息。



🗲 flask-docker 🗈 🛛	基础信息 <b>流程配置</b>	触发规则 变量与缓存 通知提醒	2 前往最新构建 操作 ∨ ● 立即构建
静态配置的 Jenkinsfile ⑦ 图形化编辑器	器 文本编辑器		♦ 环境变量 丢弃修改 保存
		6	× 镜像更新
-1 构建 Docker 镜像+	→ 5–1 推送到 CODING D	+ 増加阶段	<b>插件配置</b> 高级配置
执行 Shell 脚本	品 镜像更新		将上游构建的 Docker 镜像部署到 Kubernetes 集群。 查看完整帮助文档 🖸
+	+		镜像 ⑦ *
+ 增加并行阶段	快速筛选 Q		cd–demo
			集業 *
	<♪ 代码管理 ▶		жит 
	文件操作		Go
	器 发布部署 ▶		命名空间 *
	▶ 收集报告		请选择命名空间
	品 流程控制 ▶		资源类型 ★
	☑ 安全 →		又前/天王 注述短次历光期
	品 持续部署 ▶	全部 官方插件 团队插件	頃远拝資源尖型    ▼
	△ 质量管理 • •	镜像更新	资源名称 *
	器 其他 ▶		请选择资源名称
	器 腾讯云插件		Pod 容器 ⑦ *
	器 编译 ▶		请选择 Pod     ▼
	器 消息通知 ▶		

#### Jenkinsfile 参考:

```
stage('部署到远端 Kubernetes 集群') {
    steps {
        cdDeploy([
            deployType: 'PATCH_IMAGE',
            application: "${CCI_CURRENT_TEAM}",
            pipelineName:
    "${PROJECT_NAME}-${CCI_JOB_NAME}-${CD_CREDENTIAL_INDEX}",
            image:
    "${CODING_DOCKER_REG_HOST}/${CODING_DOCKER_IMAGE_NAME}:${DOCKER_IMAGE_VE
    RSION}",
        cloudAccountName: "${CD_ACCOUNT_NAME}",
        namespace: "${CD_NAMESPACE_NAME}",
        manifestType: "${CD_MANIFEST_TYPE}",
        manifestName: "${CD_CONTAINER_NAME}",
        containerName: "${CD_CONTAINER_NAME}",
        c
```



credentialId: "\${CD\_CREDENTIAL\_ID}",
 personalAccessToken: "\${CD\_PERSONAL\_ACCESS\_TOKEN}",
 ])
}



# 手动提交发布单

最近更新时间: 2025-04-14 17:03:52

## 新建发布单

在应用的部署流程中添加人工确认步骤,确保通过发布单发布时是经二次确认的,通过权限控制把控发布质量。

Cd-dem	no 🖉	Ø	选择阶段 依赖阶段:	✿ 部署 Service →	搜索阶段名称	Q
•	· 部署 Service	请选择阶段	Kubern	etes 通用类型		
	阶段类型:部署 (Manifest)	阶段类型:	FQ	预置条件检查 在执行下一阶段前检查预置条件		选择
			2	<b>部署流程</b> 选择其他部署流程作为本部署流程的阶段		选择
			 (23)	<b>自定义变量</b> 下游阶段可以在表达式中通过 Key 值引用自定义变	を量	选择
				<b>绑定部署流程制品</b> 在另一个部署流程直找并绑定制品		选择
			G	<b>等待</b> 等待一定的时间段后继续执行		选择
			2	<b>人工确认</b> 在继续执行前等待人工确认		选择
			ŗ	Entity Tags Applies entity tags to a resource.		敬请期待!

1. 要使用发布单功能,必须将 应用与项目关联。

2. 关联完成后,进入项目中**持续部署**找到该应用,单击发布单进入发布单管理页面。



Kubernetes 应用部署	
您可以提交发布单部署 Kubernetes 应用	,并在部署成功后查看应用信息。更多内容查看帮助文档 🖸
□ 请输入应用名称	Q
aloe = 最近一次发布于 2023-09	<b>flaskapp</b> 最近一次发布于 2023-08-1
✔ 发布单	━ 发布单 💦 集群

3. 单击新建发布单后,可以运行已有应用及部署流程。

🔶 aloe 🗸	发布单 集群 部署流程		🛚 帮助文档	新建发布 ▼
状态:全部▼	创建人: 全部 ▼ 部署流程: 全部 ▼ 发布时间: 最近十天 ▼ 搜索• 关键词搜索			快速发布
				新建发布单
发布单状态	发布单名称	部署流程	创建时间	创建人
🕏 成功	20230905-aloe-触发器-tcr个人版	触发器tcr个人版	2023-09-05 10:25:30	0
😑 已终止	2222	主机组部署	2023-09-04 17:35:08	k coding

# 快速发布

若不希望在团队设置复杂的权限限制,需要直接体验持续部署功能,可以使用**快速发布**功能。您无需在控制台中配置 部署流程即可将镜像发布至集群中,适用于部署流程更加灵活复杂的场景,例如说临时镜像变更等突发场景,无需快 速将制品发布至集群中。



#### 快速发布配置

快速发布适用于简单发布场景,如需配置更灵活复杂的部署流程,请前往部署控制台 🖸

制品类型	● 镜像 ── Manifest 文件 ── Helm 包	
仓库类型	CODING Docker 仓库	~
镜像仓库	cd-demo	~
镜像名称	hello-world	~
镜像版本	latest	~

#### 制品配置说明

- 当制品类型选择镜像时, 仓库类型支持 CODING Docker 仓库、Tencent TCR 企业版和 Tencent TCR 个人版。
  - 1. Tencent TCR 企业版说明 🖸
  - 2. Tencent TCR 个人版说明 <sup>[2]</sup>
- 当制品类型为 Manifest 文件时,需要输入 Manifest 文件在代码仓库中的相对路径,例如: k8s/deployment.yaml

发布完成后可以在持续部署中查看发布详情。



