

弹性微服务 持续集成



腾讯云

【 版权声明 】

©2013–2024 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 】

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

持续集成

- 使用 Coding 部署插件创建持续集成

- 使用 Python 脚本部署应用

持续集成

使用 Coding 部署插件创建持续集成

最近更新时间：2024-02-01 10:08:21

操作场景

弹性微服务应用可以使用 CODING 构建持续集成方案，更多关于 CODING 持续集成功能和使用的说明，请参见 [CODING 持续集成](#)。

本文以 Java 应用为例，介绍如何使用 Coding 以 镜像 及 JAR/WAR 包 方式部署应用至弹性微服务。

准备工作

准备 Demo 应用

DEMO 代码由 [Spring Initialize](#) 自动生成，使用 Gradle 构建，依赖 Spring Web、Actuator 和 Prometheus 等。

在控制台完成首次部署（可选）

首次部署时，有较多的参数需要设置。这部分信息如果全部固化到 CI/CD 流程里，就会显得特别复杂。因此，如果您的业务涉及较多部署配置项，建议您在 [弹性微服务控制台](#) 中完成首次发布，或者通过 Python 部署脚本的方式配置持续集成。

ⓘ 说明：

Coding 构建计划里配置的发布方式应当与 TEM 保持一致，例如：如果弹性微服务应用为镜像部署方式，则 Coding 构建计划内也使用镜像部署。

确认 TCR 镜像信息


如果您需要通过镜像部署应用，需要将镜像推送至 TCR 镜像仓库并部署，因此需要获取相应 TCR 镜像仓库的 docker login 的 Registry。

← test-tcr/nginx

版本管理 镜像构建 仓库信息


基本信息

仓库名称 nginx


仓库地址 .tencentcloudcr.com/test-tcr/nginx

创建时间 2020-06-17 09:37:17

更新时间 2020-06-17 09:37:17


简短描述 

登录腾讯云容器镜像服务 Docker Registry

```
docker login .tencentcloudcr.com --userna... 复制
```


当前临时登录指令有效期为 1 小时，如需长期保持登录状态，请创建并使用 [访问凭证](#)


从 Registry 拉取镜像

```
sudo docker pull .tencentcloudcr.com/test-tcr/nginx:[tag] 复制
```

其中 [tag] 请根据您需要拉取镜像的具体版本镜像替换，如 latest。更多命令说明，请参看官方文档：[docker pull](#)

向 Registry 中推送镜像

```
sudo docker tag [imageId] demo-.tencentcloudcr.com/test-tcr/nginx:[tag] 复制
```

```
sudo docker push .tencentcloudcr.com/test-tcr/nginx:[tag] 复制
```

其中 [imageId] 请替换为您所要推送的实际镜像ID，或使用本地镜像的完整路径，[tag] 请替换为您期待的镜像版本。更多命令说明，请参看官方文档：[docker tag](#)、[docker push](#)

Coding 平台操作

创建 Coding 项目和仓库

1. 创建全功能 DevOps 项目

STEP 1/2
← 选择项目模板



全功能 DevOps 项目
开启一个全功能的 CODING 项目，包含代码托管、项目协同、持续集成、...

🔗 📁 🔔 📄 📁 📁 📁 📁

[更换项目模板](#)

STEP 2/2
填写项目基本信息

项目名称

可以使用中英文、数字、空格组合

项目标识

 /

用于唯一标记项目，也可用于组织项目 URL

项目描述

完成创建 **取消**

2. 创建代码仓库。

TEM DEMO

- 项目概览
- 项目协同
- 代码仓库
- 持续集成
- 持续部署
- 代码扫描 Beta
- 应用管理
- 制品管理
- 测试管理
- 文档管理
- 生态能力
- CoDesign
- 设置
- 项目设置

← 创建代码仓库

普通创建
模板创建

在其他网站已有仓库? [点击导入](#)

所属项目 *

TEM DEMO

仓库类型 *	仓库名称 *	
GIT 仓库	tem_demo	8/100

仓库描述

tem demo

初始化仓库

- 生成 README 文件
- 添加 .gitignore 文件
- 添加分支模型（仓库创建后将根据所选模型创建分支）

是否开源

- 私有仓库（仅对仓库成员可见，仓库成员可访问仓库。）
- 公开仓库（您所在的团队尚未实名认证，不允许公开源代码，[请先进行实名认证。](#)）

3. 添加 remote 仓库，并推送代码。

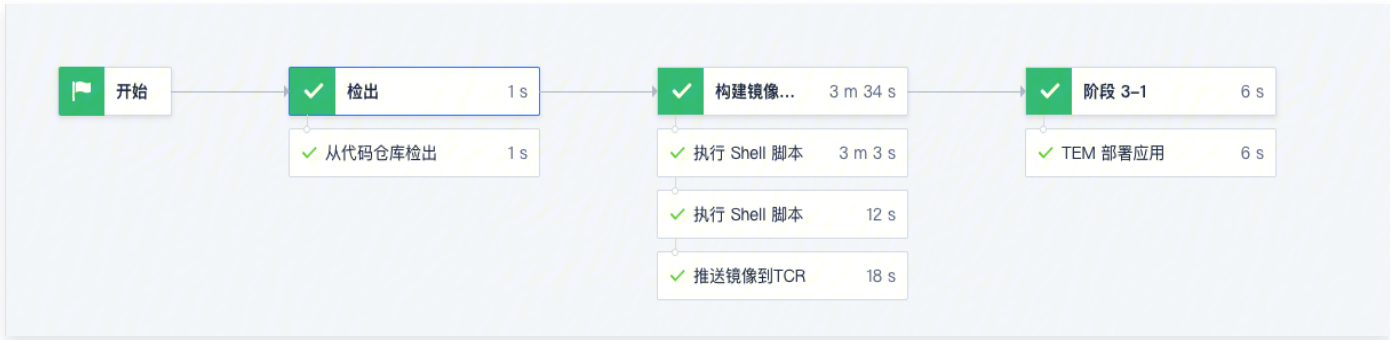
```

git init
git add .
git commit -m "init"
git remote add origin ***/tem-test/tem-demo.git
git push -u origin master
```

构建计划概览

默认 `git push origin master` 后，会触发构建计划。如您有特殊需求，可自行配置构建计划的触发规则。

- 采用镜像方式部署时：



● 采用程序包方式部署时:



配置构建计划

1. 模板选择自定义构建计划。

若没有找到合适的模板，可选择自定义构建过程



自定义构建过程

允许您根据 Jenkinsfile 的规范来随意定制持续集成流水线过程。

2. 在流程配置中，添加构建模块（执行 Shell 脚本）。由于 DEMO 中使用的 gradle 构建，可参考以下命令：

```
(cd /root/.gradle && rm init.gradle)
./gradlew clean build
```




3. 镜像部署时在流程配置中，添加镜像构建模块（执行 Shell 脚本）

```
"docker build -t
${DOCKER_NAMESPACE}/${DOCKER_IMAGE_NAME}:${DOCKER_IMAGE_VERSION}
-f ${DOCKERFILE_PATH} ${DOCKER_BUILD_CONTEXT}"
```

您可自行在环境变量中定义相关变量：

- DOCKER_NAMESPACE：命名空间。
- DOCKER_IMAGE_NAME：镜像仓库名称。
- DOCKER_IMAGE_VERSION：镜像版本号，e.g. \${GIT_COMMIT}。
- DOCKERFILE_PATH：dockerfile 文件位置，e.g. Dockerfile。
- DOCKER_BUILD_CONTEXT：Docker 构建目录，e.g. 使用当前目录。

4. 镜像部署时在流程配置中，添加镜像推送模块（TCR 插件），腾讯云内网目前只支持 TCR（由于镜像拉取是大流量操作，对带宽要求比较高，不建议 NAT 接出公网）。您也可用使用 docker login && docker push 自行完成镜像推送。



5. 在流程配置中，添加 TEM 部署模块。



TEM 部署插件配置

1. 配置通用信息与应用基本信息。

TEM 部署应用（新版）

插件配置 高级配置

插件版本 *

最新版本

secretId * ?

缺少必填字段

secretKey * ?

缺少必填字段

地域 * ?

中国

应用名称 * ?

缺少必填字段

环境名称 * ?

缺少必填字段

- 插件版本：选择最新。
- SecretId/SecretKey：腾讯云的 API 密钥。
- 地域：TEM 的服务地域。
- 应用名称：支持根据名称新建应用，或选择存量应用部署至新环境。
- 环境名称：根据环境名称指定一个存量环境，若无合适环境，请前往 [创建环境](#)。

2. 配置镜像：配置关联的镜像 TAG，如 commitId 等。

镜像部署: 仓库类型 [?](#)

请选择



镜像部署: 仓库地址 [?](#)

镜像部署: 企业版TCR实例ID [?](#)

3. 配置 JAR / WAR 包: 除了配置部署版本以外, 还需要额外配置 JAR/WAR 的文件路径。

Jar 包/War 包部署: Jdk版本 [?](#)

请选择



Jar 包/War 包部署: 程序包文件路径 [?](#)

Jar 包/War 包部署: 程序包版本号 [?](#)

4. 部署配置。

发布触发策略 * [?](#)

自动触发 ▼

发布批次次数 * [?](#)

2

批次间等待时间 (单位: 秒) * [?](#)

60

小批量验证批次的实例数 [?](#)

发布过程中保障的最小可用实例数 [?](#)

-1

○ 发布触发策略:

- 自动触发: 发布全自动执行。
- 全手动触发: 发布不同批次间, 需要用户人工确认才会继续执行。
- 小批量验证后自动触发: 小批量验证完成后, 经人工确认, 后续批次自动触发。

○ 发布批次: 实例的发布批次次数。

○ 批次间等待时间: 发布批次之间的间隔时间, 以秒为单位, 用来减少可能的服务抖动。

○ 发布过程中保障的最小可用实例数:

- -1 代表保证全量实例可用。
- 0 代表不保障发布过程中一定有实例可用。
- n 代表发布过程中保障有n个实例可用。

部署详情

部署操作是异步的, 在部署操作执行完之后, 会返回访问部署详情的链接。

```
49 [2022-01-20 23:31:18] 部署详情: https://console.cloud.tencent.com/tem/release-record?rid=4&service=service-...&ns=env-...&version=version-...
```

手工确认的分批次发布场景 (可选)

1. 增加 Coding 插件 (人工确认)。



2. 继续下一批（执行 Shell 脚本）。

```

pip install tccli
tccli configure set secretId $SECRET_ID secretKey $SECRET_KEY region $REGION
output json
tccli tem ResumeApplication --version 2021-07-01 --ApplicationId $APPLICATION_ID
--EnvironmentId $ENVIRONMENT_ID
    
```

使用 Python 脚本部署应用

最近更新时间：2024-02-02 16:01:11

弹性微服务应用可以使用 Python 脚本来部署。

前提条件

在开始持续集成之前，需要完成下述的准备工作：

1. 保证机器上安装的 Python 版本不低于 3.0 版本，并已安装 PIP 等 Python 包管理工具。
2. 获取腾讯云的 [访问密钥](#) SecretId 和 SecretKey。
3. 在弹性微服务 [创建了环境](#)。
4. 安装脚本所需的依赖。

```
pip install tencentcloud-sdk-python cos-python-sdk-v5
```

部署准备

1. 下载 TEM 部署 Python 脚本，[deploy.py](#)。
2. 根据脚本注释，修改您的部署脚本参数。
3. 运行脚本。

```
python3 deploy.py
```

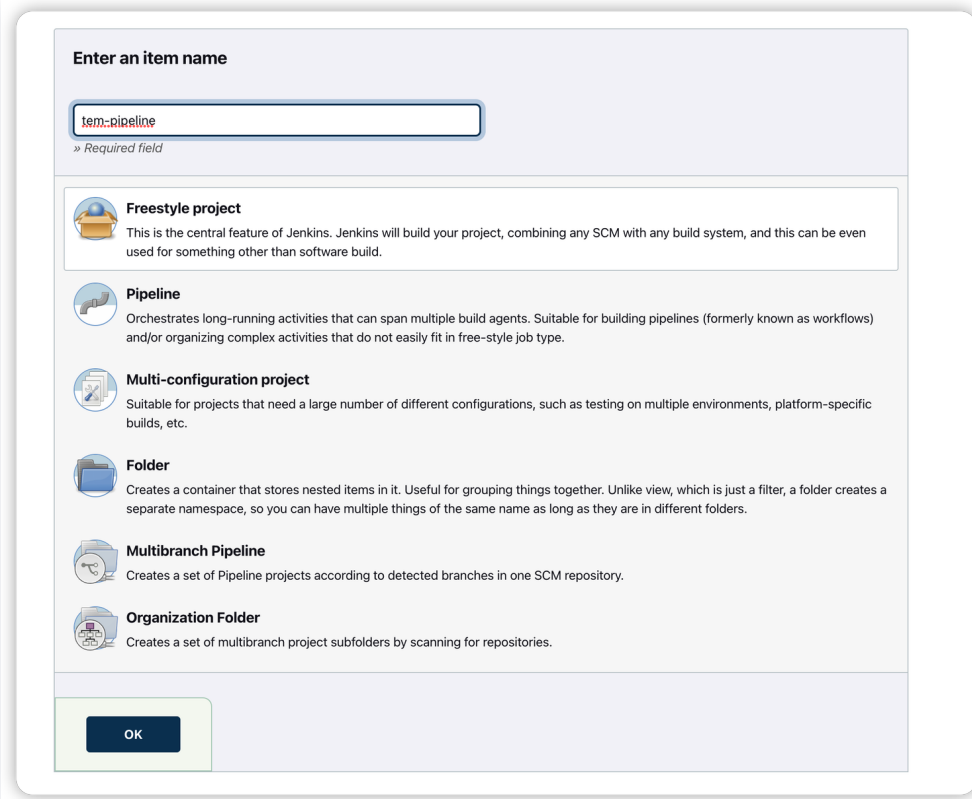
4. 登录 [弹性微服务控制台](#)，在左侧导航栏单击应用管理，进入应用列表页，选择目标应用，单击应用 ID，进入应用详情页，查看部署结果。

使用 Jenkins 创建持续集成

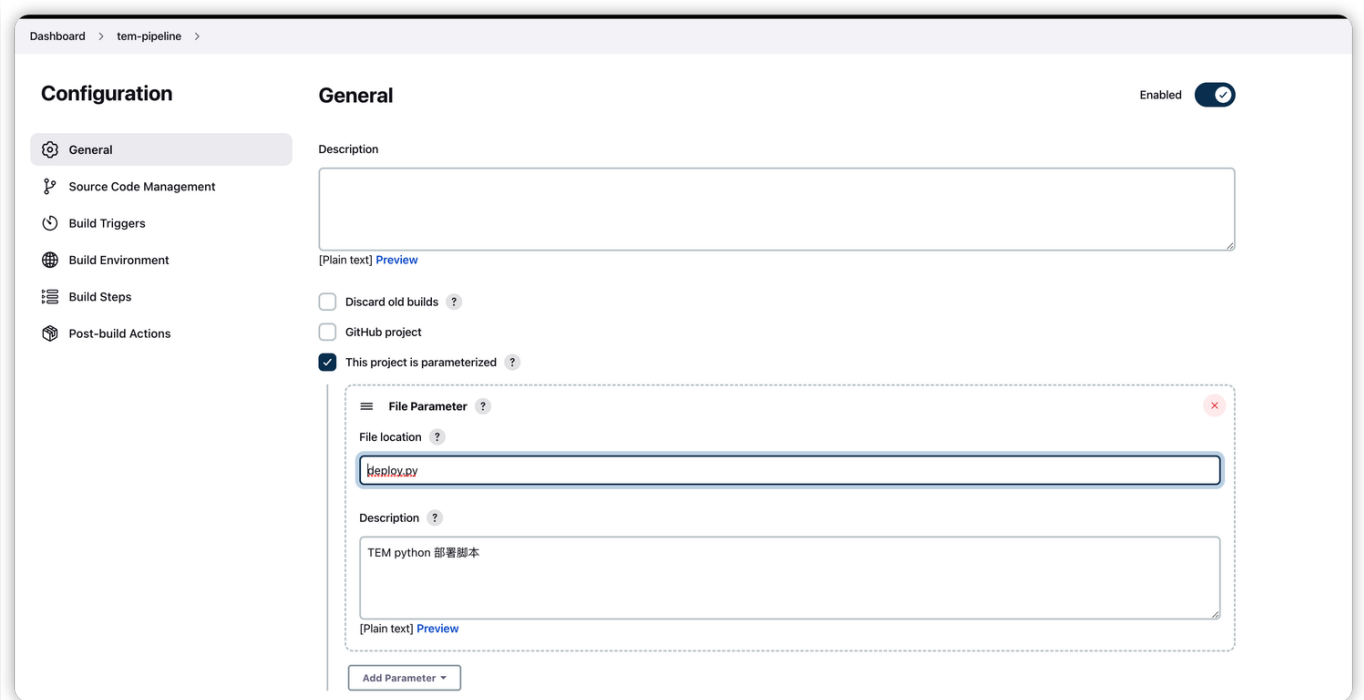
可以使用部署脚本在流水线构建持续集成方案，此处以 Jenkins 为例。

配置 Jenkins

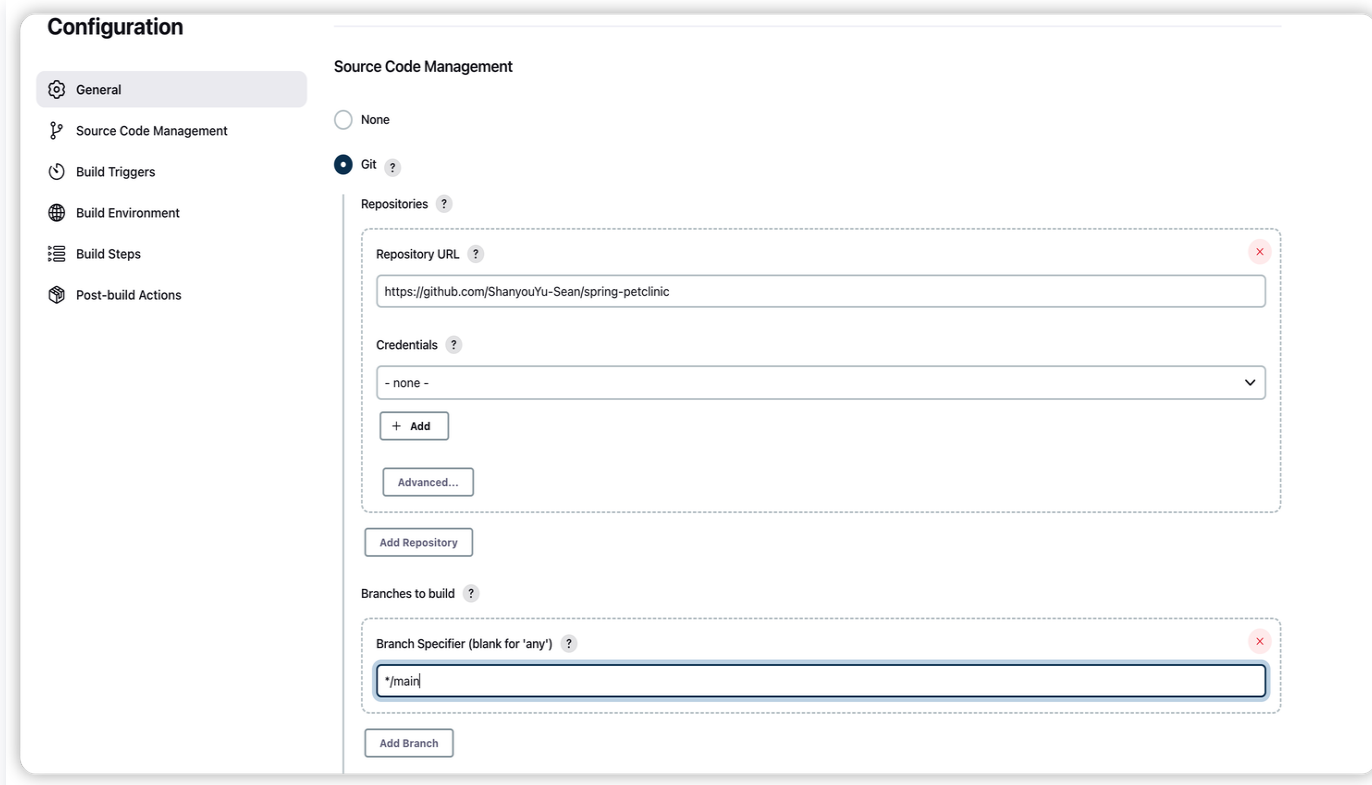
1. 在 Jenkins 首页左侧导航栏中单击新建，创建 Jenkins 任务，并选择构建一个自由风格的软件项目。



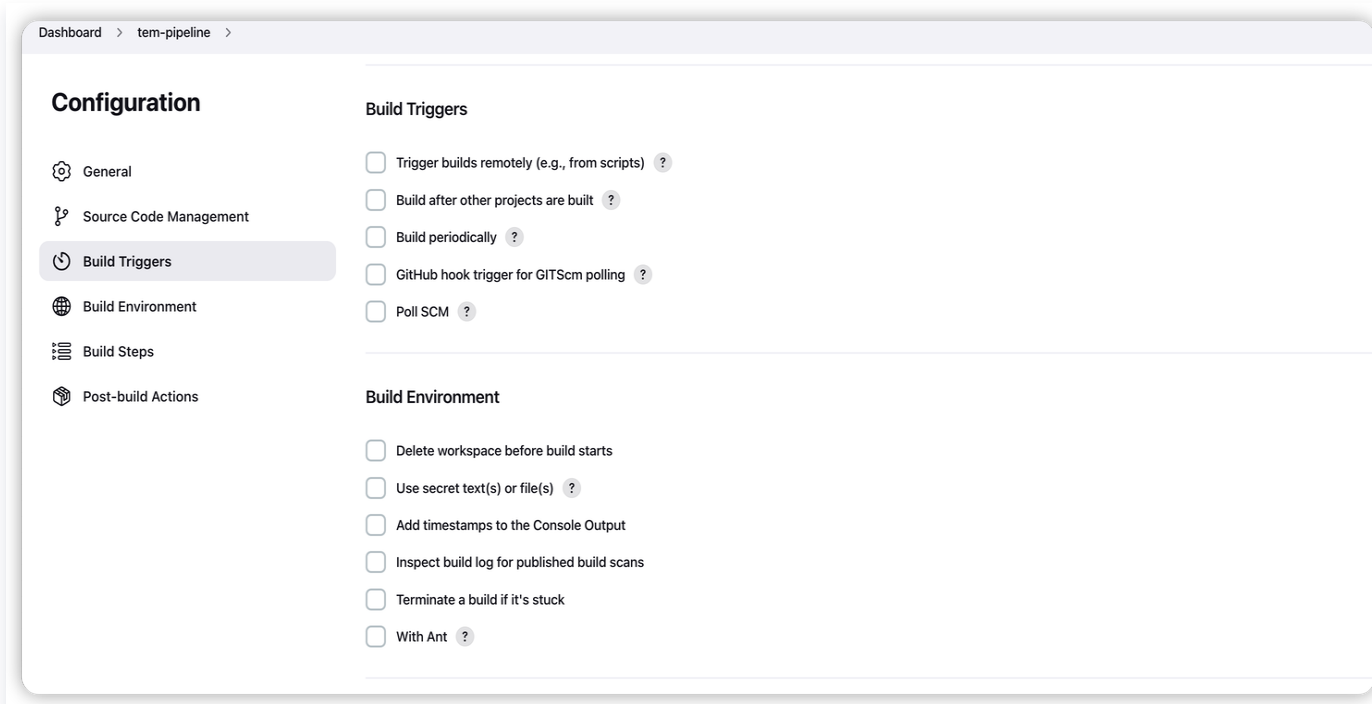
2. 在 General 中选择 This project is parameterized -> File Parameter ，配置部署文件参数。



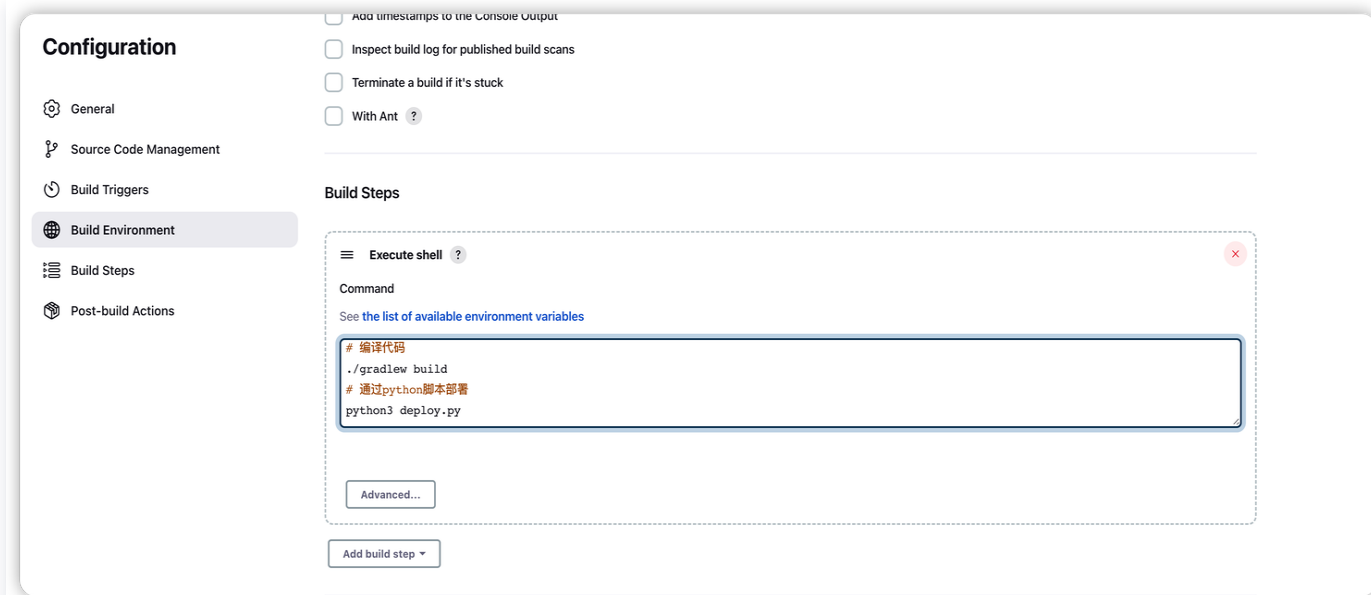
3. 配置您的项目源码。



4. 配置您的项目构建环境和触发器（本文档中暂省略，进行手动触发）。

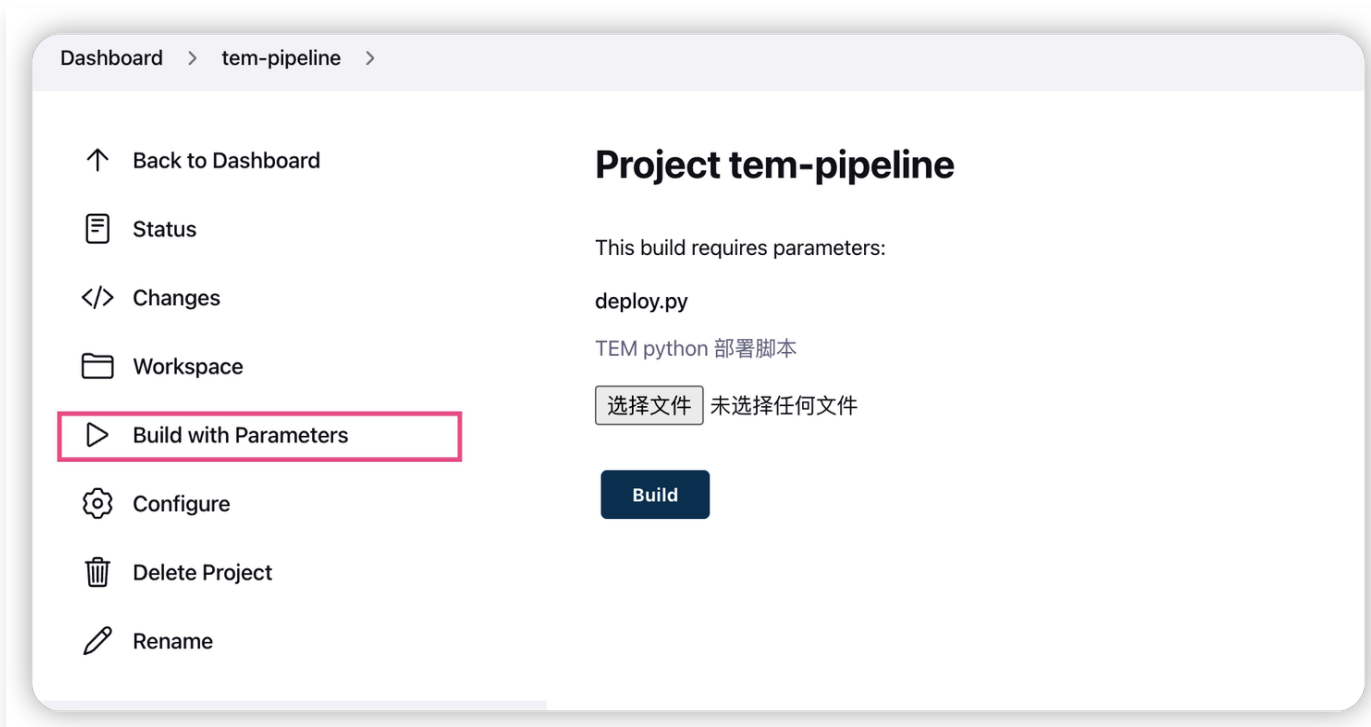


5. 添加构建步骤。

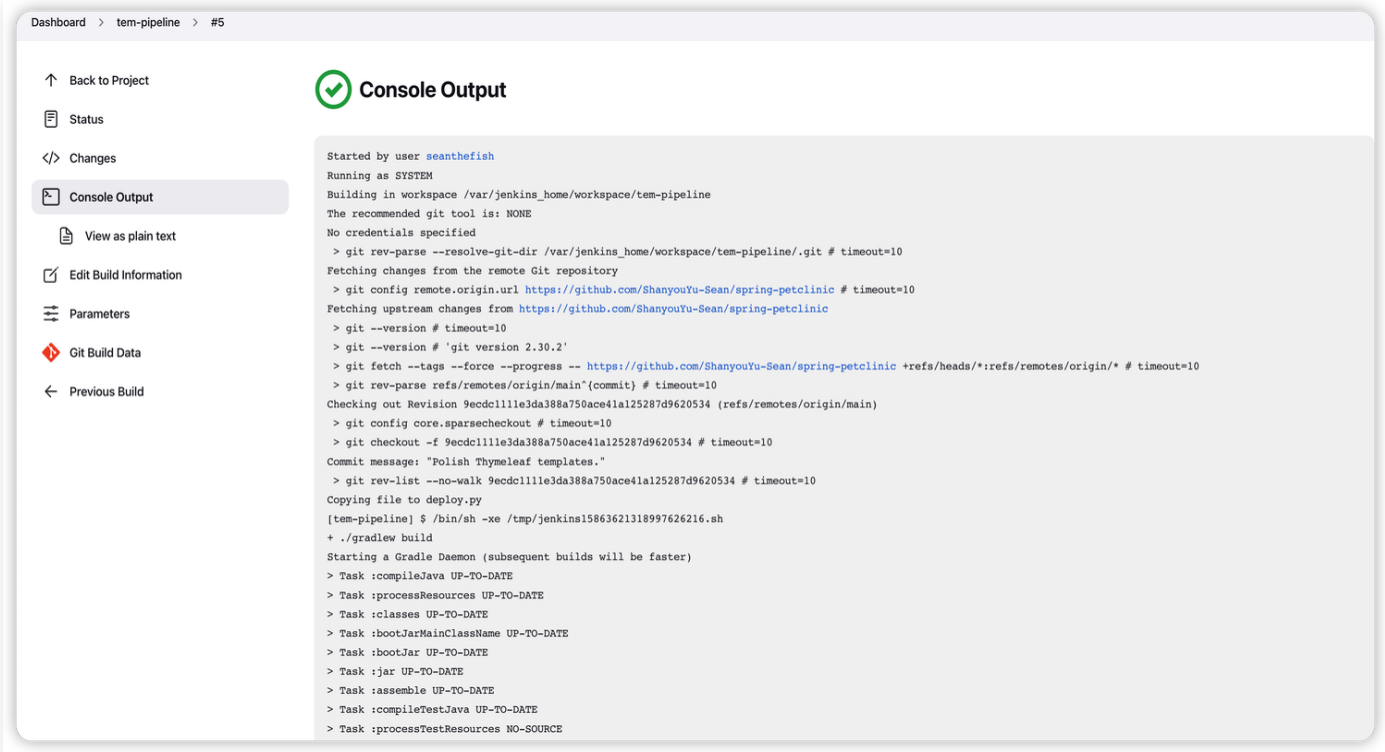


触发构建和部署

1. 选择 Build with Parameters, 上传 **TEM 部署脚本**, 触发构建和部署。



2. 在 Jenkins 中查看部署结果。



3. 在弹性微服务控制台查看部署结果。

