# CODING DevOps

# 常见问题

腾讯云

【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

# 常见问题
# DevOps 相关

最近更新时间：2024-06-11 17:43:21

## 什么是 DevOps?

DevOps 是软件开发（Development）和运营（Operations）的结合。代表着重视**软件开发人员（Dev）**和 **IT 运维技术人员（Ops）**之间沟通合作的文化；旨在透过自动化**软件交付**和**架构变更**的流程，使得构建、 测试、发布软件的过程能够更加地快捷、频繁和可靠。Gartner 咨询公司认为 DevOps 代表了 IT 文化的变化趋势。DevOps 可以很好地解释为人们一起工作以快速构思、构建和交付安全软件。DevOps 实践使软件开发（dev）和运营（ops）团队能够通过自动化、协作、快速反馈和迭代改进来加速交付。
源于软件开发的敏捷方法，DevOps 流程扩展了以更快、更迭代的方式构建和交付应用程序的跨职能方法。在采用 DevOps 开发流程时，您正在决定通过在开发周期的所有阶段鼓励更具协作性的环境来改进应用程序的流程和价值交付。

## DevOps 核心原则是什么?

DevOps 方法包括指导应用程序开发和部署的有效性和效率的四个关键原则。

- **软件开发生命周期的自动化**。这包括自动化测试、构建、发布、开发环境的配置以及其他可能减慢软件交付过程或将人为错误引入软件交付过程的手动任务。
- **协作与沟通**。一个好的 DevOps 团队有自动化，但一个伟大的 DevOps 团队也有有效的协作和沟通。
- **持续改进和减少浪费**。从自动化重复性任务到观察性能指标以寻找减少发布时间或平均恢复时间的方法，高绩效 DevOps 团队经常寻找可以改进的领域。
- **通过简短的反馈循环高度关注用户需求**。通过自动化、改进的沟通和协作以及持续改进，DevOps 团队可以花点时间专注于真正的用户真正想要什么，以及如何将其提供给他们。

通过采用这些原则，组织可以提高代码质量、缩短上市时间并进行更好的应用程序规划。

## DevOps 发展会经历哪几个阶段?

随着 DevOps 的发展，它的复杂性也随之增加。这种复杂性是由两个因素驱动的：

- **组织正在从单体架构转向微服务架构**。随着 DevOps 的成熟，组织每个项目需要越来越多的 DevOps 工具。
- **更多项目和每个项目更多工具的结果是项目工具集成数量呈指数级增长**。这需要改变组织采用 DevOps 工具的方式。

这种演变经历了以下四个阶段：

### 第 1 阶段：自带 DevOps

在 Bring Your Own DevOps 阶段，每个团队都选择了自己的工具。当团队因为不熟悉其他团队的工具而试图一起工作时，这种方法会导致问题。

### 第 2 阶段：一流的 DevOps

为了应对使用不同工具的挑战，组织进入了第二阶段，即一流的 DevOps。在此阶段，组织对同一组工具进行标准化，并为 DevOps 生命周期的每个阶段使用一个较为合适的工具。它帮助团队彼此协作，但问题随后变成了通过每个阶段的工具移动软件更改。

### 第 3 阶段：自己动手 DevOps

为了解决这个问题，组织采用了自己动手（DIY）DevOps，在他们的工具之上和之间构建。他们执行了大量定制工作以将他们的 DevOps 单点解决方案集成在一起。然而，由于这些工具是独立开发的，没有考虑到集成，因此它们永远不会很合适。对于许多组织而言，维护 DIY DevOps 是一项重大工作，会导致更高的成本，因为工程师需要维护工具集成，而不是致力于核心软件产品。

### 第4阶段：DevOps 平台

单一应用程序平台方法可改善团队体验和业务效率。DevOps 平台取代了 DIY DevOps，允许在整个 DevOps 生命周期的所有阶段实现可见性和控制。
通过授权所有团队（开发、运营、IT、安全和业务）跨端到端统一系统协作规划、构建、保护和部署软件，DevOps 平台代表了实现完整 DevOps 的潜力。

## 什么是 DevOps 平台?

DevOps 将人类孤岛聚集在一起，DevOps 平台对工具做同样的事情。许多团队使用不同的工具集合开始他们的 DevOps 之旅，所有这些工具都必须维护，其中许多没有或不能集成。DevOps 平台将工具集中在一个应用程序中，以实现无与伦比的协作、可见性和开发速度。DevOps 平台是现代软件应该如何以可重复的方式创建、保护、发布和监控。真正的 DevOps 平台意味着团队可以更快地迭代并共同创新，因为每个人都可以做出贡献。

## 为什么 DevOps 很重要?

DevOps 很重要，因为它有可能通过更快地响应业务需求来帮助组织从竞争对手中脱颖而出。DevOps 是一种新的更好的软件构建方式，它改进了端到端的协作，不仅在开发和运营团队之间，而且还与安全（有时称为 DevSecOps）、测试（质量保证或 QA）等学科协作、版本控制和跨团队协作功能，例如 ChatOps。DevOps 带来更好的软件产品和更成功的实施。
DevOps 的核心是一种实践。它的构想前提是，当软件开发团队真正协作并进行持续集成和持续交付（CI / CD）时，应用程序和服务交付组织才能更好发挥作用。这意味着在每次迭代结束时，无论多短，软件都可以投入生产——即使它并非每次都部署在生产中。
大型组织经历 DevOps 转型以解决企业软件创建的常见和基本问题。当开发人员创建新软件时，他们会在开发人员环境中对其进行编码和测试，该环境是离线的，可以让他们在不危及他们的业务、政府机构、医疗或教育机构的情况下排除错误、调整代码和完善需求。
然而，当需要将新软件或代码部署到真实环境中时，问题就出现了，因为开发人员环境与不断发展的生产环境并不完全相同。这可能会导致一些真正的胃灼热甚至心碎。部署失败可能导致需要花费大量时间和金钱来修复的问题。从历史上看，由于将大量更改汇总到不经常发布的版本中，这个问题变得更糟。
运营团队负责确保产品在生产中可靠地运行，并进行适当的检查和平衡以确保可靠的部署。操作和开发团队之间可能会出现摩擦，试图迭代并尽快将他们的代码更改投入生产。

# 团队管理相关

最近更新时间：2024-11-01 15:36:21

## 如何锁定或解锁成员？

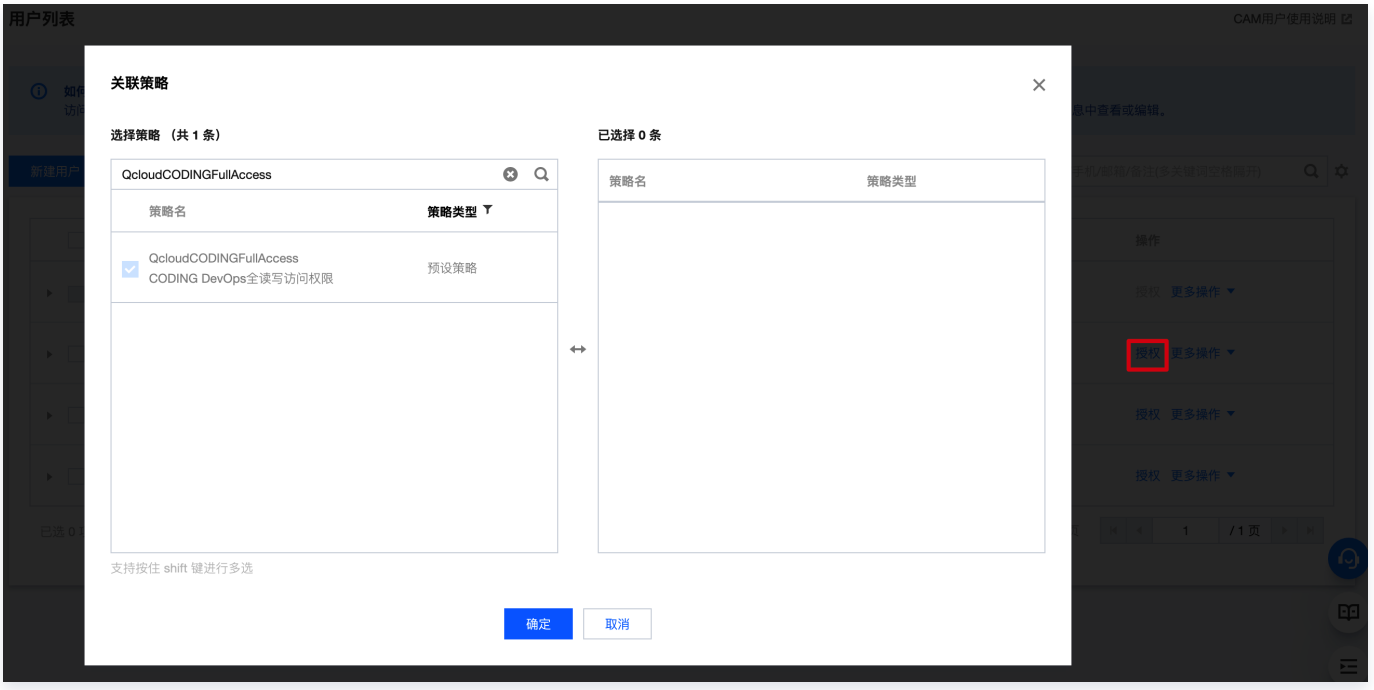
此操作会禁止成员访问团队资源与登录团队。单击左下角的**团队设置中心**，前往**全局设置 > 组织与成员 > 成员管理**中的团队成员列表，选择需要锁定的用户，单击右侧的 ··· 并选择**锁定成员**。被锁定的用户会有锁定标记，如需解锁，执行**解除成员**操作即可。


## 账号被锁的原因及解决办法是什么？

- 如果被锁定的是**团队负责人**，那么需前往 工单中心 提交解锁申请。
- 如果被锁定的是**团队成员**，以下是可能的原因以及解锁办法：

  1.1 登录团队时连续 5 次输入密码错误。请联系团队负责人或管理员，在**设置中心 > 成员管理**中进行解锁。

  1.2 被锁成员的账号类型为腾讯云子账号，缺乏 `QcloudCODINGFullAccess` 角色权限。请联系主账号所有者，前往 访问管理 进行角色策略授权。



## 如何退出团队？

团队成员单击左下角头像下拉框的**个人账户设置 > 个人账户**即可看到退出团队选项。

> **说明：**
> 团队负责人不能退出团队。


## 如何注销团队？

团队负责人单击首页左下角的**团队设置中心**，单击**基本信息 > 注销团队**进入操作页面。

## 如何更换腾讯云账号绑定的 CODING 团队?

当新用户在腾讯云控制台上访问 CODING DevOps 时，腾讯云会为新用户新建一个团队。如果需要将新用户与一个已有团队绑定，使其成为访问腾讯云控制台时默认提供的团队地址，可以按照如下步骤在 CODING 控制台进行操作。

1. 在新团队的**团队设置中心 > 全局设置 > 第三方应用**解除与腾讯云的绑定。如果没有设置密码，请先在**个人账户设置**中绑定邮箱及密码，再进行解绑操作。


2. 登录需要绑定的团队，在已有团队的**团队设置中心 > 全局设置 > 第三方应用**绑定腾讯云，具体操作请参见 凭据授权 。


## 如何转让团队负责人？

> **说明：**
> 若账号已关联腾讯云账号，则不支持此功能。请参见 绑定腾讯云 进行解绑。

团队负责人可以进入**团队设置中心** > **全局设置** > **基本信息**中将团队转让给相应成员。


## 团队域名是否支持修改?

域名绑定后标准版的团队域名不支持修改，高级版、旗舰版的团队域名支持由团队负责人通过自定义团队域名进行修改。在**团队设置中心 > 全局设置 > 自定义域名**页面单击右上角的**自定义域名**，在**自定义域名**对话框中填写域名后单击**创建**完成自定义团队域名的修改。


## 团队成员减少后没有自动降档?

团队成员减少后，系统并不会自动调低目前高级版团队的所属档位。若确认不再需要更多档位，请 提交工单 由客服为您调低档位。工单信息需包括：

- 团队名称
- 团队域名
- 更新后的档位数量
- 申请原因

## 如何退款?

付款后，暂不支持退款，请在确认购买所需的服务后再进行支付。若遇到下单错误等特殊情况请及时联系 官方人员 。

## 如何查看服务到期时间?

团队负责人或管理员单击左下角头像下拉处的**服务订购**，在服务概览右侧中可以查看预估到期时间。


## 如何获取发票或合同?

您可以在腾讯云官网的 费用中心 下载发票或合同。

## 绑定腾讯云账号时失败怎么办?

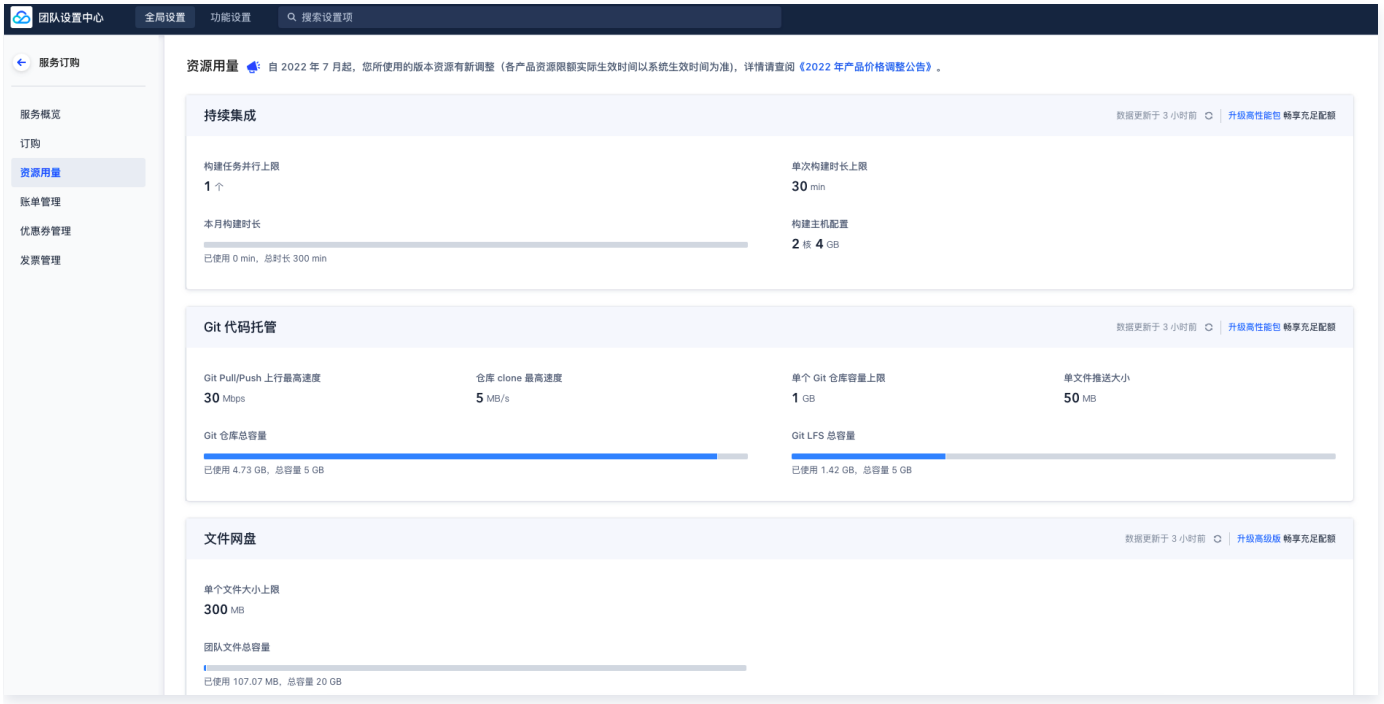
**问题描述**：在第三方应用中绑定腾讯云账号时提示**当前凭据无可用账号**。


**解决办法**：您可能之前使用过腾讯云 Serverless 或某项容器服务，而这些服务调用了 CODING 能力，从而系统自动生成了 CODING 团队并完成两者间绑定。您需要前往腾讯云 控制台 ，登录该自动创建的团队。


补全信息后前往**全局设置 > 第三方应用**解绑腾讯云。然后再登录自己的 CODING 团队，重新前往团队设置中心绑定腾讯云账号。

## 如何查看团队资源用量?

团队资源用量包含持续集成并发数、代码仓库容量与文件网盘等用量。单击首页左下角的**团队设置中心 > 资源用量**进行查看。

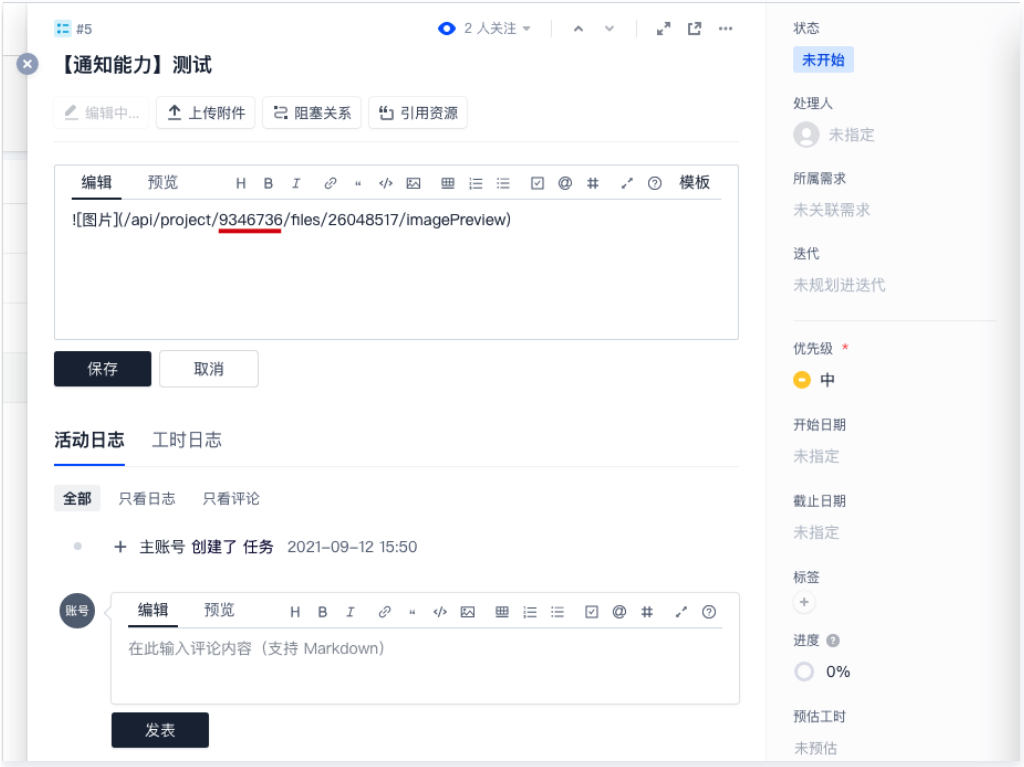


# 项目管理相关

最近更新时间：2024-11-01 15:36:22

## 在事项中粘贴图片失败怎么办?

问题详情：粘贴图片地址后，图片显示裂开，不可用。
造成此问题的原因可能是您没有图片的访问权限，常见于直接复制其他项目的图片地址。在图片地址中可以查看项目 ID。若无法确认本项目的 ID，粘贴新的图片后将显示本项目 ID。您可以直接复制图片后重新粘贴至事项，而避免直接复制图片地址。




## 故事燃尽图中理想线为什么一直为 0?

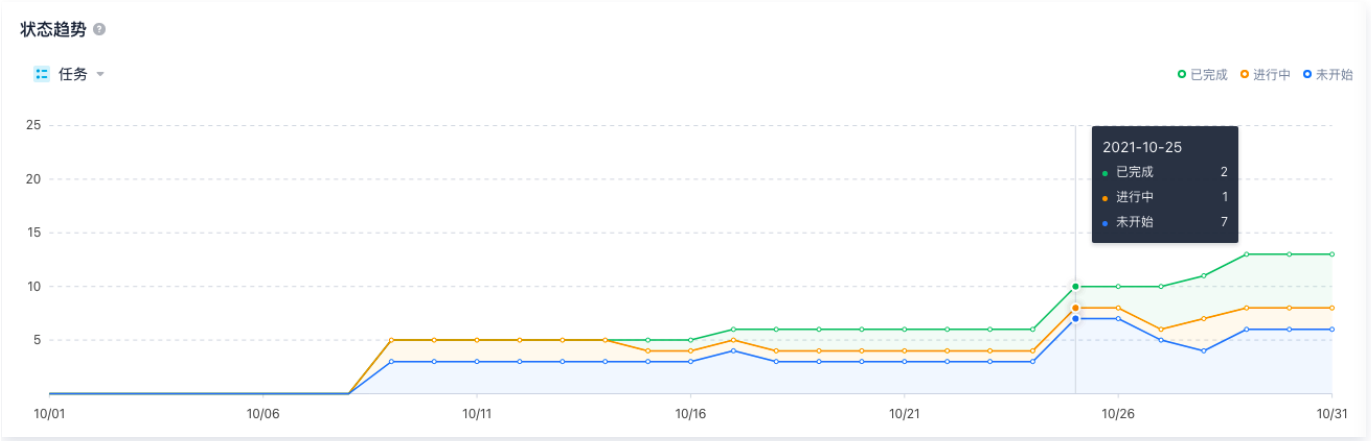
蓝色线为实际线，灰色线为理想线。
例如在下图中，实际线不降反增，而理想线却一直为 0。


理想线的统计逻辑为：绘制原点以**迭代开始时故事点总数**开始计算。此图说明从迭代的开始那一刻算起，迭代中的事项故事点总数为 0，即使后续修改了迭代中事项的故事点，也无法改变理想线的初始值。您可以通过重新调整故事点与迭代开始周期修改起始点。

理想线通常为一条线性下降线，因为将自动平均分配所有的故事点至完整迭代周期中，迭代结束后也意味着已燃尽所有故事点。实践线的统计逻辑为当天内事项故事点的总数，故有可能为一条折线。

## 如何查看状态趋势图?

状态趋势图采用堆积面积图进行数据展示，例如在此图中所有的事项总数为 10，故占有 10 单位面积。未开始的事项数为 7，占据了 7 单位蓝色面积；进行中事项数为 1，仅占据 1 单位黄色面积，它们之间的关系相互独立。




## 如何将项目迁移至其他团队?

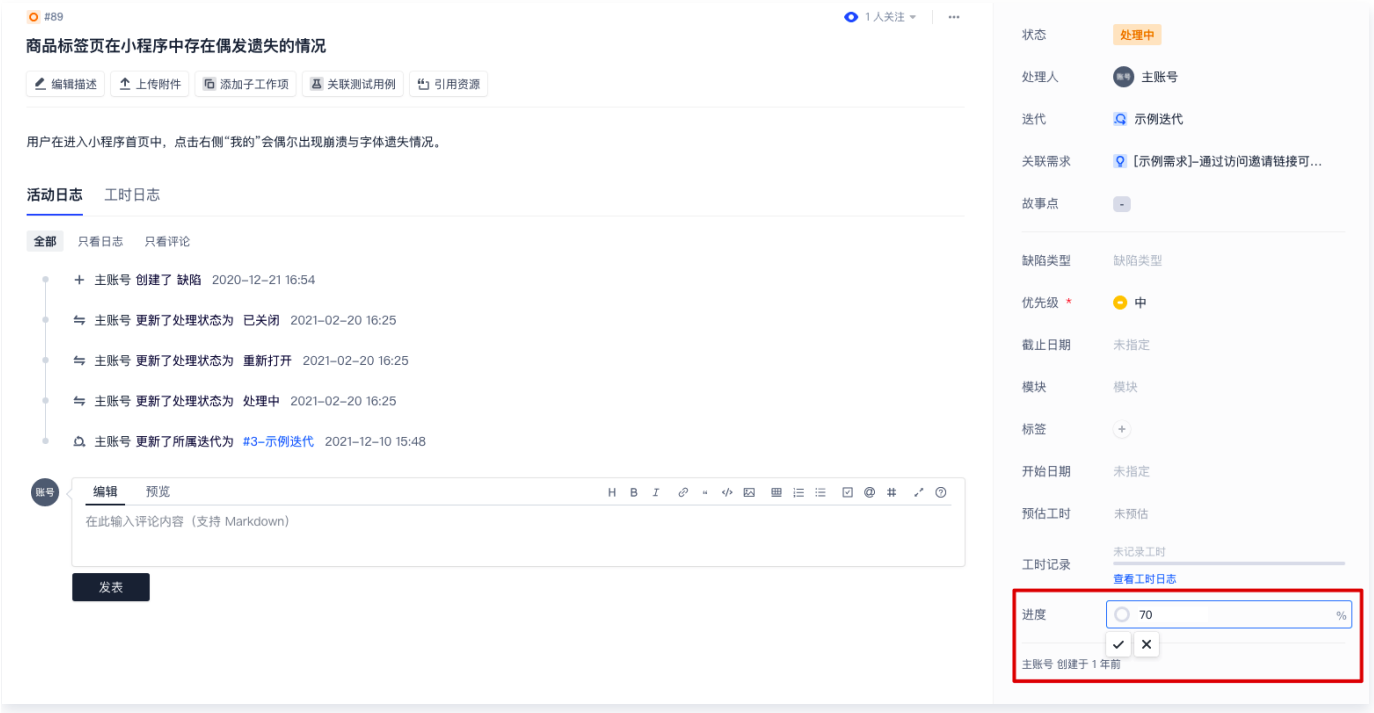
CODING 目前不支持跨团队转移项目，项目只能归属于同一个团队。您可以先导出旧有项目内的事项，再手动导入至新项目中。

## 项目被误删后怎么办?

项目删除属敏感操作，CODING 在删除环节内置了权限控制与二次确认机制。项目一旦被删除后将无法被找回，属不可逆操作，请谨慎删除。

## 已完成迭代内所有事项，但为什么进度不是 100%?

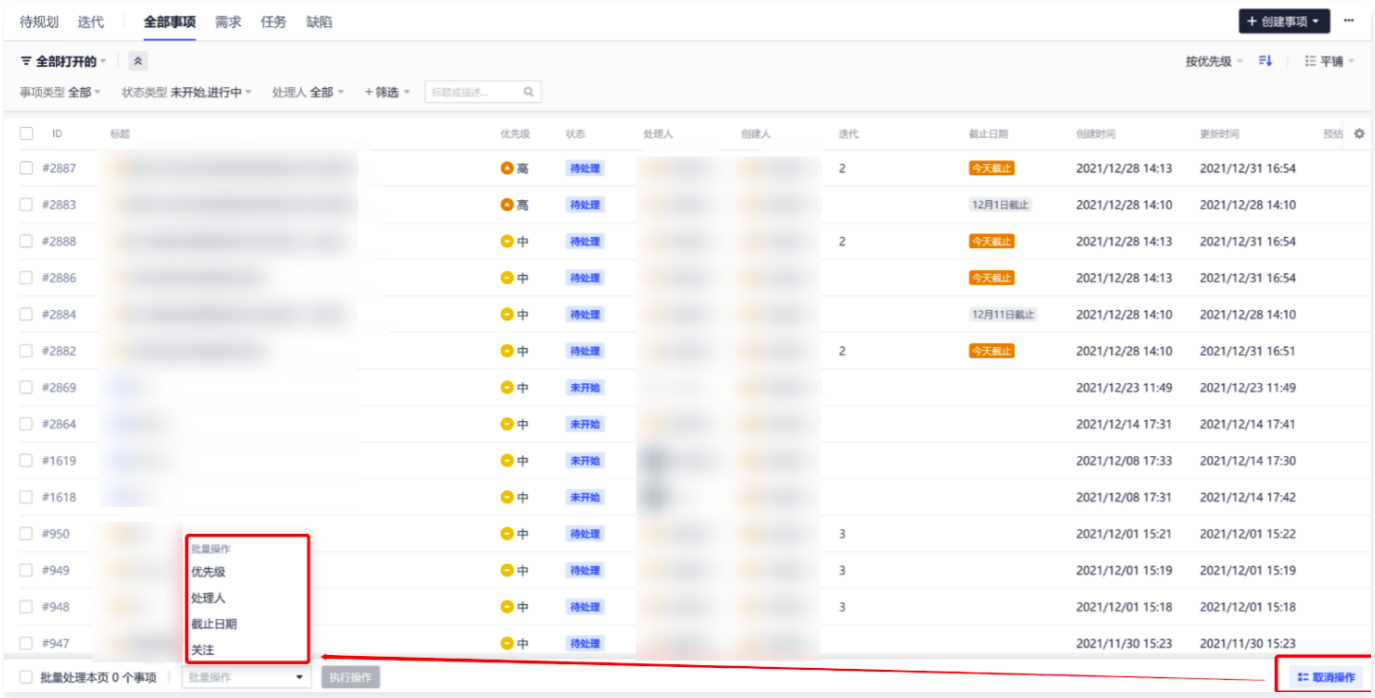
迭代中的进度条取决于事项中进度数值，而与事项的完成状态无关。



您可以参见 自定义工作流，设置事项状态改变时自动修改进度数值。例如将**已完成**事项的进度自动修改为 100%，当迭代内事项全部完成后迭代进度自动变更为 100%。


## 如何批量修改处理人?

单击事项列表页右下角的**批量操作**后按照提示进行操作，目前支持批量修改优先级、处理人、截止日期和关注人。




## 如何修改协作模式?

单击项目中左下角的**项目设置 > 协作模式 > 更改协作模式**进行切换。

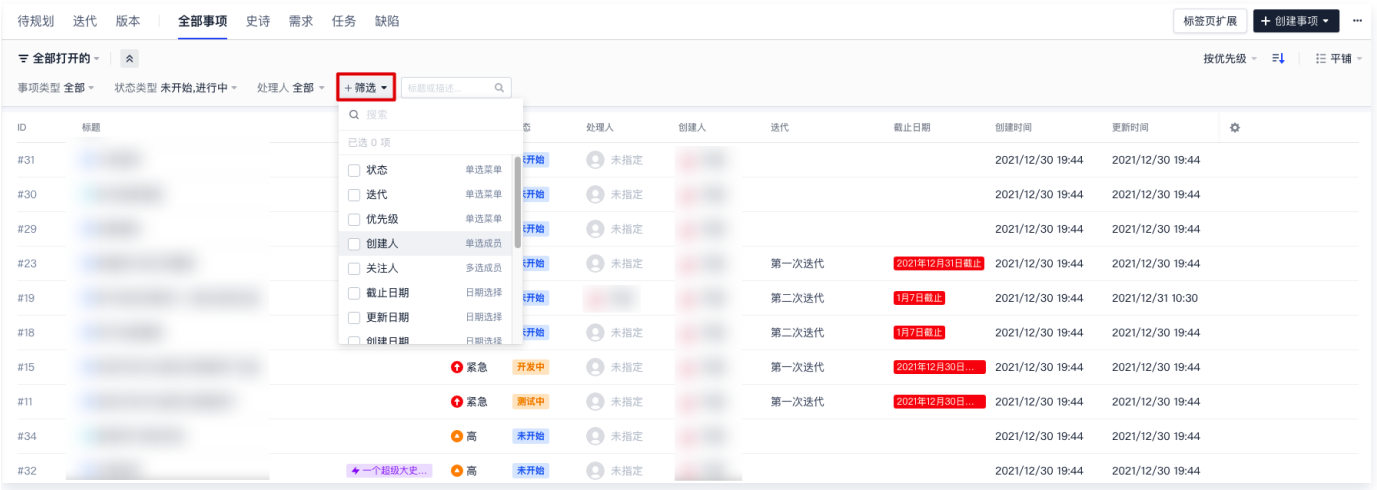
> **注意：**
> 切换模式时可能会修改部分事项类型。


## 编辑事项时无法选择迭代怎么办?

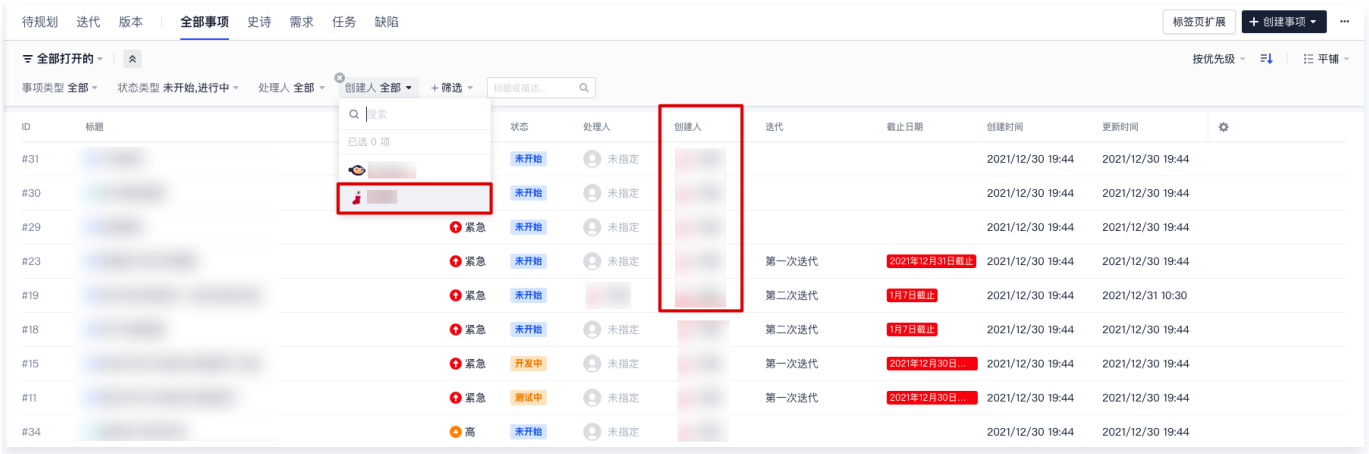
问题描述：在项目协同中，协作成员在提交事项（需求、任务、缺陷）时无法选择已规划的迭代，需要如何设置?
问题原因：该成员所在的项目权限方案未具备**编辑迭代**权限，因此无法将事项添加进已规划的迭代中。需联系管理员前往**团队设置中心 > 项目权限方案**为该成员分配具备**编辑迭代**权限的方案，详情请参见 配置项目权限方案 。


## 如何查看与“我”有关的全部事项?

您可以通过设置项目中的过滤器条件查看与**我**有关的所有事项，例如由我创建的事项、我所关注的事项等。
进入任意项目后，前往项目协同，单击**全部事项** tab 页，新增筛选条件。




勾选**创建人**、**关注人**等筛选条件，并选择**我**，即可查看所有与我有关的事项。




单击**另存为**即可储存当前筛选条件。


## 事项是否支持指派多个处理人？

事项（需求、任务与缺陷）不支持指派多个处理人，您可以将事项进行拆分，再逐一分配给相关处理人。

## 事项删除后是否支持恢复？

事项删除后无法恢复，请谨慎删除。

# 代码托管相关

最近更新时间：2024-09-14 14:45:01

## 提示验证失败怎么办？

在克隆代码仓库时需输入服务邮箱与服务密码。您可以前往**个人账户设置 > 个人设置**查看服务凭证。


## 克隆代码时出错怎么办？

1. 请确保安装并使用了最新版官方 Git 客户端。
2. 在终端中输入命令 `git remote -v` 查询所关联的 remote url（大小写敏感）是否为正确的，若 remote url 有误，请参见以下命令行修改远程仓库地址。

```
$ git remote set-url origin https://git.coding.net:username/right-name.git
```

## 推送代码时提示其他错误怎么办？

请参见 快速开始 确保您执行了正确的操作。如果仍然报错请 提交工单，并在工单中附上以下信息（可选）：

- Git 报错信息。
- 执行 `git --version` 的结果。
- 其他可能有用的信息，例如输入 `ping coding.net` 命令后的运行结果、您目前的 IP 地址及所使用的 DNS 等。

## 提示 Couldn't resolve host 怎么办？

这可能是由于您的 DNS 设置造成的，请更换您的 DNS 为 114.114.114.114 或 1.1.1.1 后重启网络。

## 提示 RPC failed 错误怎么办？

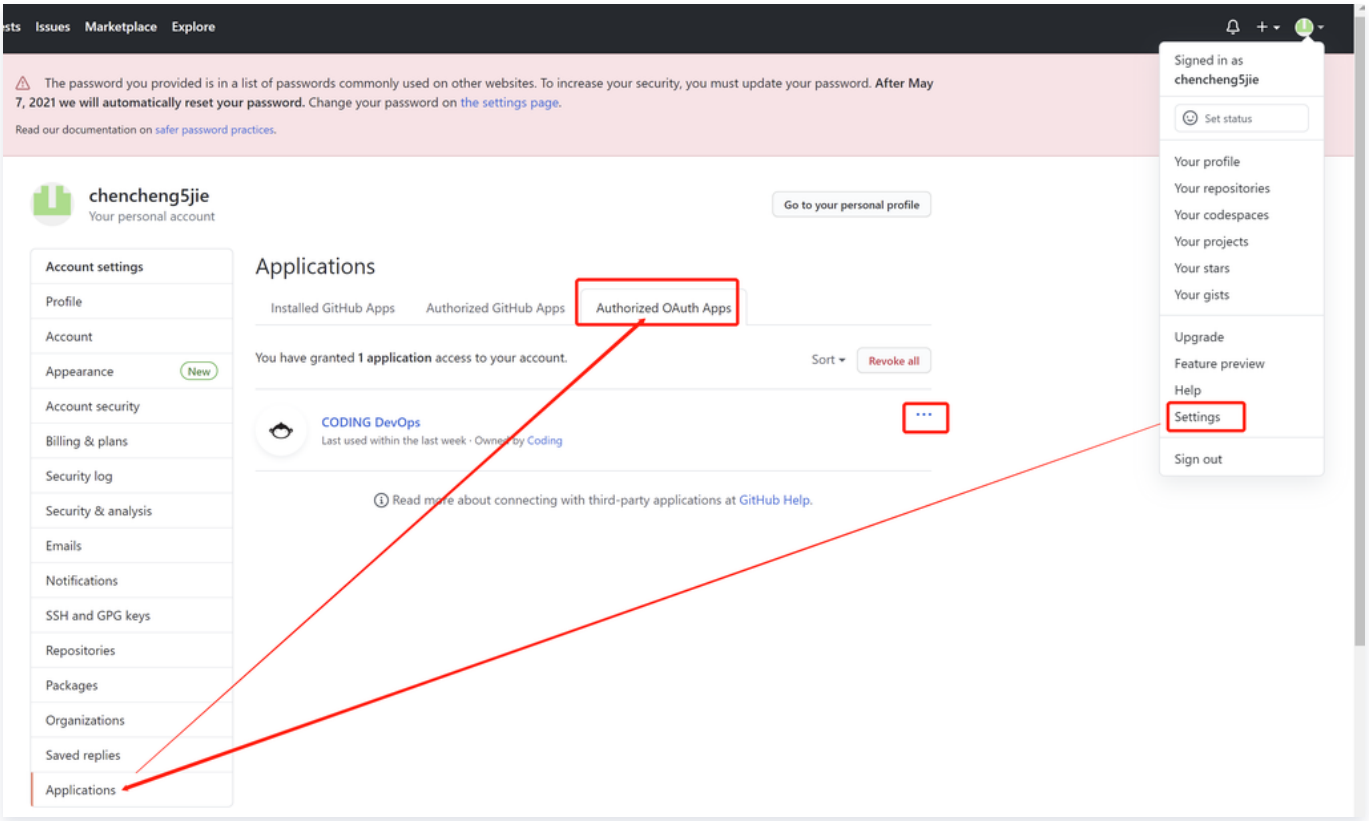
这是由于 HTTPS 推送方式的 `http.postBuffer` 限制推送文件大小。可以在终端中执行 `git config http.postBuffer 524288000` 命令以设置更大的限制值，或者切换为 SSH 方式进行推送。

若返回错误码：`HTTP 403 curl 22 The requested URL returned error: 403 Forbidden`，还可以检查是否具备仓库访问权限、账号密码是否正确、仓库存储是否已满。

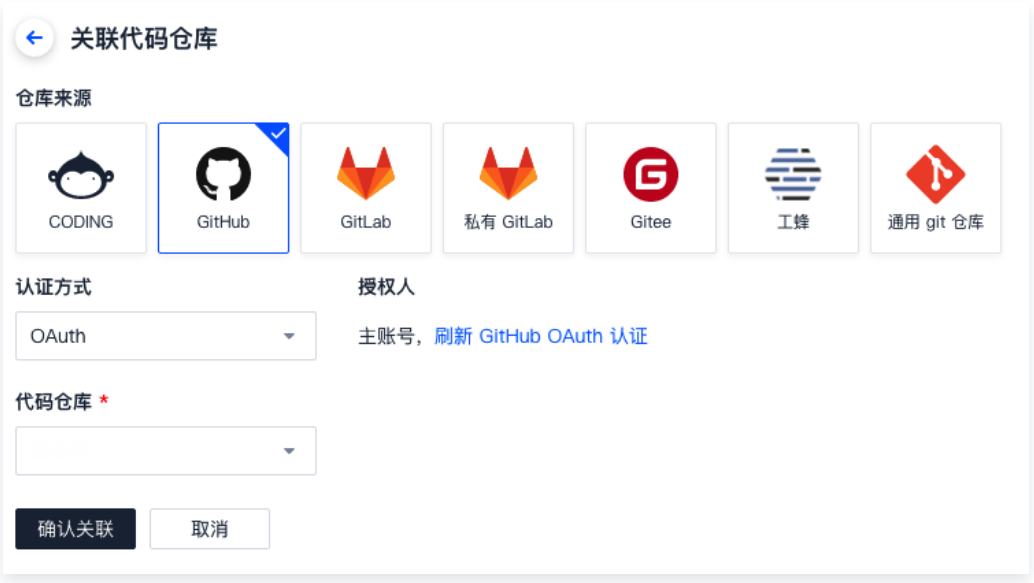
## 如何修改已关联的外部仓库?

以修改已关联的 GitHub 仓库为例：

1. 登录 GitHub，单击右上角头像处的 **Settings** > **Applications** > **Authorized OAuth Apps** 取消关联。




2. 登录新的 GitHub 账号。
3. 返回 CODING 的代码仓库，单击**关联仓库**后重新刷新 GitHub OAuth 认证。




## 认证失败超过 20 次提示冻结 3600s 怎么办?


使用错误的凭据连续认证失败累计 20 次将触发安全保护机制，暂时冻结该仓库的 https 访问，需要暂时停止所有自动或者手动认证的动作，等待冻结时间结束后重新进行登录，否则即使使用正确的账号密码也会认证失败并导致冻结时间重置。
若需紧急进行代码操作，您也可以改为 SSH 协议进行代码推拉。

## 公钥拉取失败提示 Permission denied（publickey）

可能是由于您没有目标仓库和分支的权限，导致无法更新数据。确认对目标分支已有读写权限后依然出现此错误码，请参见以下解决办法：

### 本地单公钥

删除本地的公钥与私钥后，根据 配置 SSH 公钥 提示重新生成新的公钥与私钥，并将新的公钥重新添加至项目中。

### 本地存在多公钥

在 `~/.ssh` 目录下参考下列样例新建 `config` 文件：

```
Host "demo1"
        HostName "code.csdn.net"
        User "git"
        IdentityFile "C:/Users/your-name/.ssh/id_rsaDemo1"

Host "demo2"
        HostName "code.csdn.net"
        User "git"
        IdentityFile "C:/Users/your-name/.ssh/id_rsaDemo2"
```

例如目标仓库的 Git 地址为：`git@e.coding.net:example/example.git`，那么可以使用 `git clone git@demo1:your-name/demo1.git` 命令进行拉取。针对不同的目标仓库，修改命令中的别名完成拉取。

### RSA 算法被禁用

此问题通常发生在 Git 客户端是 v2.33.1 及以上的版本或者 OpenSSH 是 8.8 及以上用户群体中。您可以添加 `~/.ssh/config` 文件并使用 `vim ~/.ssh/config` 命令重新修改 `ssh-rsa` 配置。

```
Host e.coding.net
HostkeyAlgorithms +ssh-rsa
PubkeyAcceptedAlgorithms +ssh-rsa
```

使用新算法生成新的公私钥：`ssh-keygen -t ed25519 -C "your email"`。

## 部署公钥与个人公钥有何区别?

部署公钥归项目所有，配置后默认拥有该项目的只读权限（可以调整为读写权限），能够添加至多个项目中；个人公钥归个人所有，配置后拥有账户下所有项目的读写权限，常用于认证 OPEN API 鉴权。同一份公钥不能既作为部署公钥，又配置为个人公钥。

## Git 账号密码相关

**问题描述：**每次提交代码后需要反复进行账号与密码验证、无法删除仓库缓存中的账号与密码。

**解决办法：**

**方法一：**切换为 SSH 协议。

使用 SSH 协议进行代码推拉不仅是更加安全的方式，也能够避免频繁输入账号密码。您可以参见 配置 SSH 公钥 生成 SSH 公钥后将其上传至代码仓库中作为部署公钥。


在本地终端中使用命令 `git remote repo-url`，再输入 `git remote -v` 以验证是否生效。

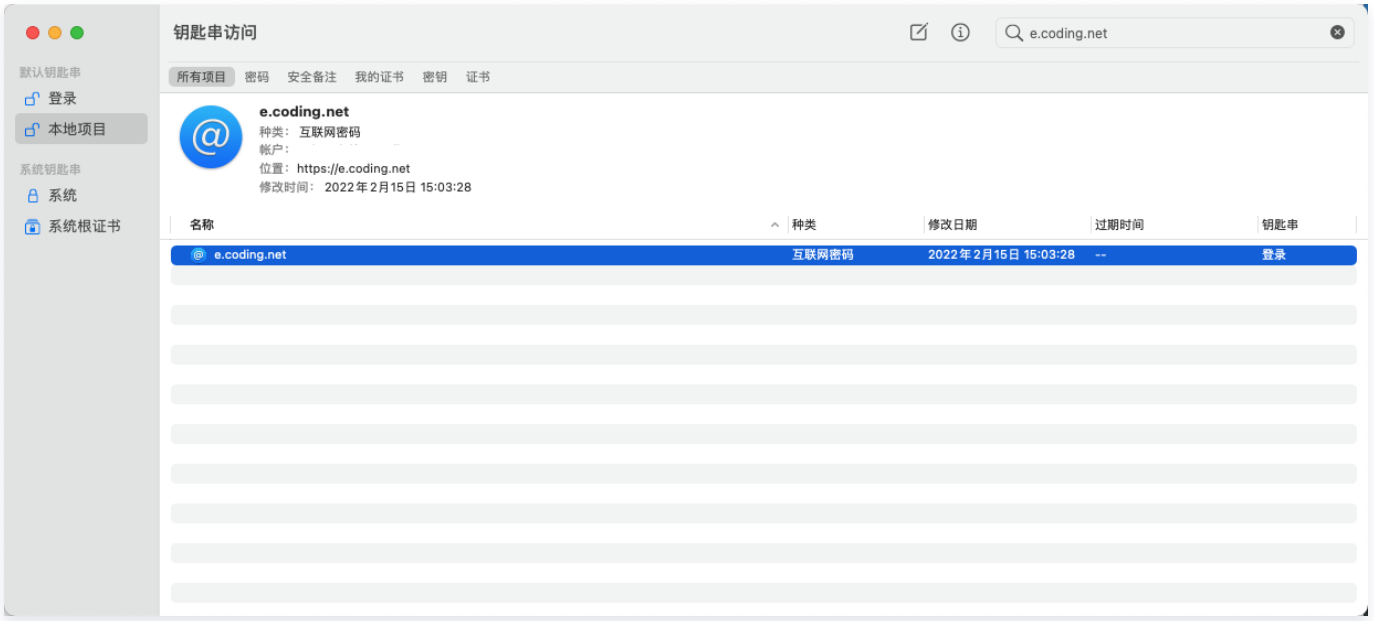
```
/Volumes/CODING-Help/coding-help-generator  issue/393 ●  git remote set-url origin git@e[redacted]net:devops/help-coding-website/coding-help-generator.git
/Volumes/CODING-Help/coding-help-generator  issue/393 ●  git remote -v                                                    ✔  10174  11:24:44
origin  git@e[redacted]net:devops/help-coding-website/coding-help-generator.git (fetch)
origin  git@e[redacted]net:devops/help-coding-website/coding-help-generator.git (push)
/Volumes/CODING-Help/coding-help-generator  issue/393 ●                                                                    ✔  10175  11:25:18
```

生效后即可免密进行代码推拉。若更换 SSH 密钥，请重新上传至代码仓库中。

**方法二：**删除缓存中的账号密码并重新录入正确的账号与密码。

### macOS

多数情况下 macOS 采用 osxkeychain（钥匙串）工具保存 Git 账号密码，前往**钥匙串访问**，搜索以 `e.coding.net` 开头的钥匙串并删除。




**删除账号与密码后，重新推送或拉取代码，此时会提醒您输入正确的账号与密码，无误后系统将会记住新的账号与密码。若仍未删除成功，在终端中运行此命令：**

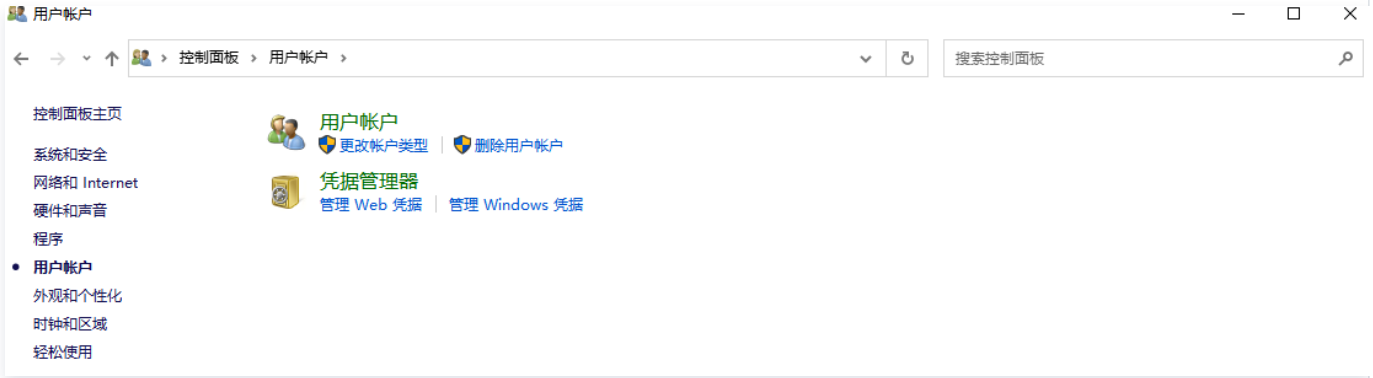
```
git credential-osxkeychain erase
host=e.codingcorp.net
protocol=https
```

**删除后重新推送或拉取代码，按照提示输入新的账号与密码。**

> **说明**
> Git 账号密码将同时保存在钥匙串与 `.git-credential` 文件中。若删除钥匙串后依然提示密码错误，有可能是系统读取了 `.git-credential` 文件中的账号与密码。

## Windows

1. **单击设置后，前往控制面板中的用户账户。**

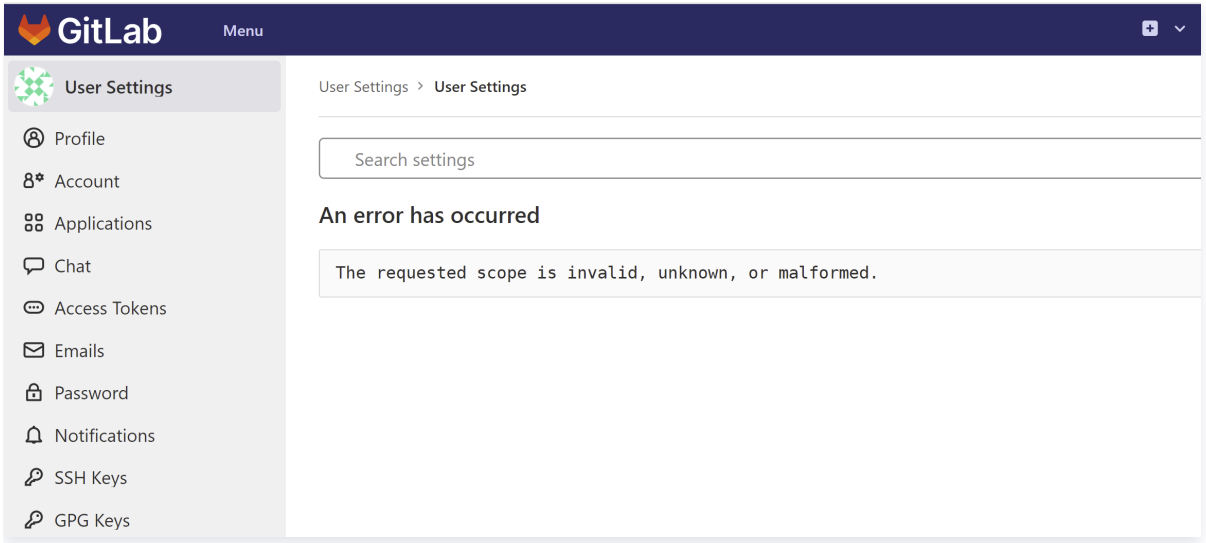



2. **在普通凭据中搜索以 `e.coding.net` 命名的普通凭据。**

3. 删除凭据后，重新推送或拉取代码，按照提示录入正确的账号与密码。

## 绑定 GitLab（私有部署）服务时提示"The request scope is invalid, unknown, or malformed"错误如何处理?

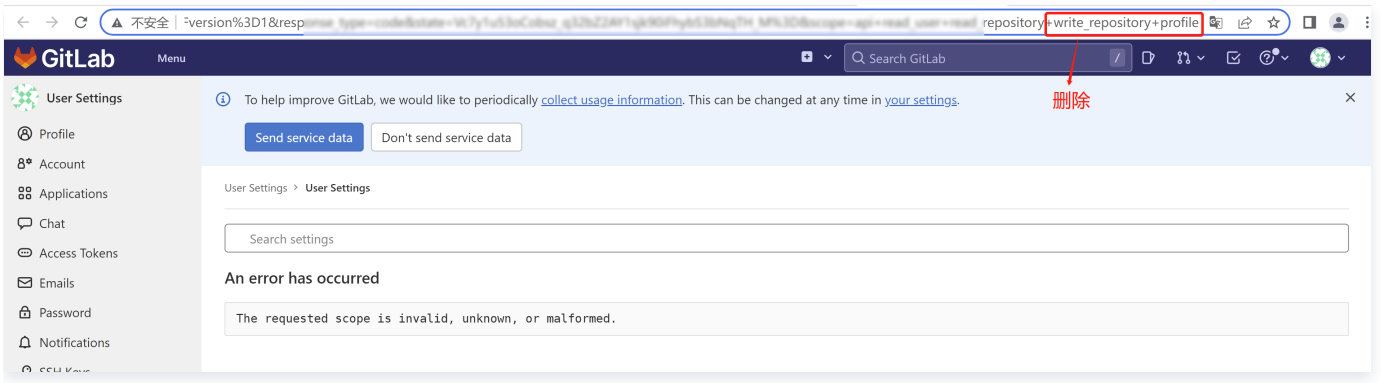
**问题描述：**如下图所示，在团队设置中绑定私有 GitLab 服务时出现 `The request scope is invalid, unknown, or malformed` 错误。



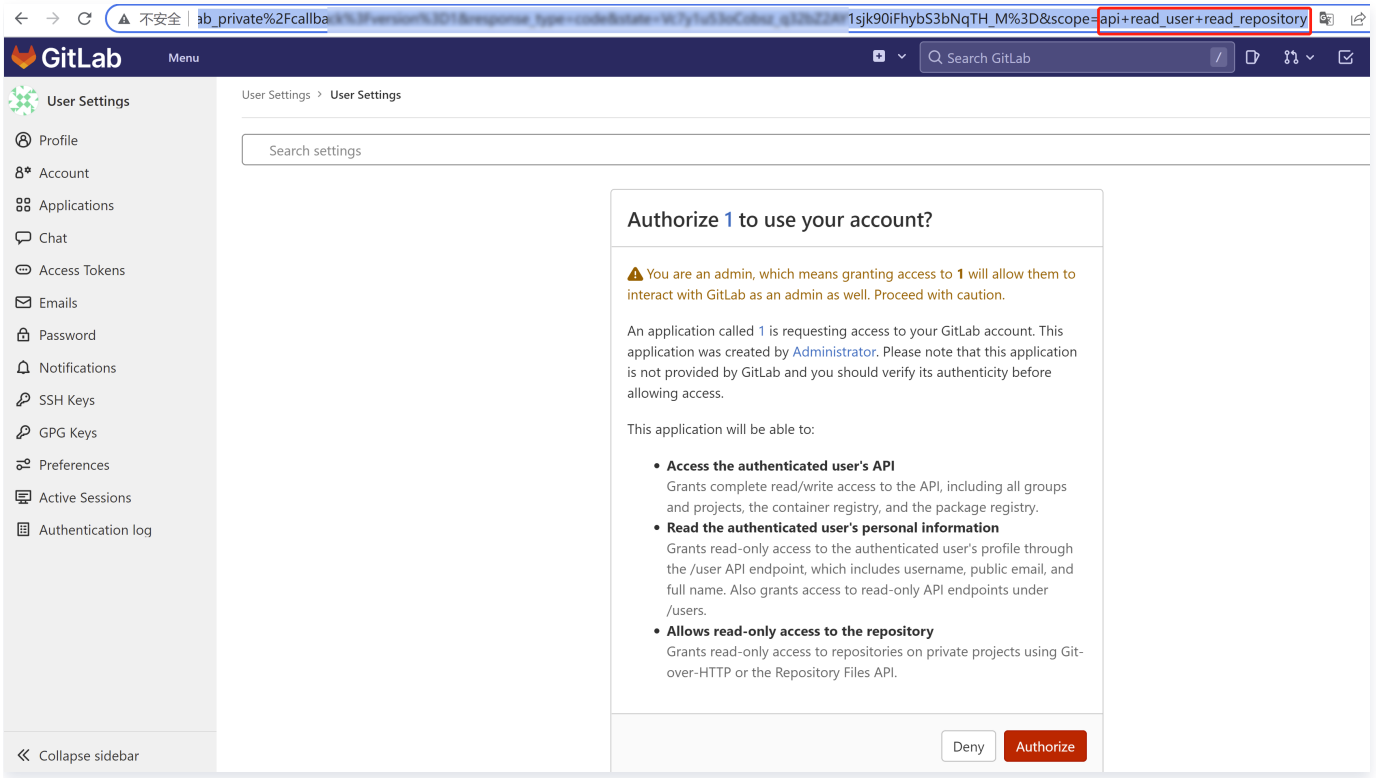

**解决方案：**出现此问题的原因是 GitLab 私有部署版本不同导致 GitLab Applications 中的 Scopes 不一致。绑定 GitLab 私有部署服务时，CODING 要求提供 Scopes 以下五种参数：

- api
- read_user
- read_repository
- write_repository
- profile

若私有 GitLab 服务缺失上述参数中的某一项，例如没有 `write_repository` 和 `profile` 参数，那么绑定 GitLab 服务时仅需勾选剩余的 3 个参数。然后跳转至绑定 GitLab 页，将 URL 中多余的 `+write_repository+profile` 部分删除。




如下图所示，`&scope` 部分仅保留 `&scope=api+read_user+read_repository` 后回车，出现下图所示的绑定页面即表示问题解决。




## 导入仓库后没有合并请求记录?

目前仅支持导入代码仓库中的代码资源、版本号与提交记录。合并请求记录存储于 Git 客户端数据库中，此部分信息无法被导入。

## 创建合并请求时提示不可自动合并怎么办?


出现此错误的原因是两个分支中的代码存在冲突。您可以在本地终端中，在源分支执行 `git merge [target branch]` 命令查看所有冲突文件。依次解决所有冲突后，重新提交一个 commit 至远端仓库合并请求将自动变更为**可合并**状态。

## 代码提交失败，并提示仓库使用量之和已达限额该怎么办?

对于标准版团队，若总代码仓库容量超过最大限额后将无法继续推送代码。您需要联系团队负责人或管理员升级团队版本以提升团队总代码容量，请参见 服务方案与计费 。

## 代码托管在 CODING 是否安全?

这个问题涉及两个层面，一是代码托管至云端是否安全，二是代码托管至 CODING 是否安全。

### 相较于把代码上云，将代码放在本地是否就安全呢?

代码放在本地，首先需要搭建一套代码管理系统配有专人维护，并且还需辅以建设安全机制，保障服务稳定和数据的安全性，以上两项都有一定的技术要求和较高的成本支出；专业的人负责专业的事。
代码托管在具备技术实力和安全保障的专业云服务上，在保障代码安全的基础时还能够使用更多功能，享受专业的服务，更加省时省力省钱。
国内企业 SaaS 服务经过多年发展，企业将人力信息、财务信息、客户信息、产品部署都放在云服务上，对于中小企业来说，选择一家有技术安全实力的云服务提供商比自己搭建一套系统更加安全高效。

### 代码托管到 CODING 是否安全呢?

- CODING 是目前国内市场上唯一基于企业级 Java 技术完全自主研发的一站式软件研发平台，相比其它基于开源项目改造的更加安全、可控性更高。同时具备更强的扩展性，不容易受到被公开的漏洞攻击。
- 网站通信采用 HTTPS 协议，代码推拉采用 HTTPS 或 SSH 协议，全程加密传输数据，避免数据传输过程中的泄漏。
- 可设置成员权限，可锁定异常成员，访问资源都须经用户 OAuth 认证，有效控制企业资源的访问权限。
- 提供二次验证、密码规则设置等功能，可以有效防止由团队成员账号遗失或被盗导致的团队信息泄漏风险。
- 提供详细的操作日志，可通过记录追溯团队成员的操作行为，可及时发现异常操作。
- 具有完善的运维安全机制，详情请参见 CODING 服务安全性说明 。
- 架构设计高可用。代码仓库使用分区存储策略存放于不同的存储服务器上，将其挂载到计算服务器工作空间上，相互之间不产生影响。若单点故障，负载会自动分配到其他服务器上。单个代码仓库出现问题后，仅需恢复此仓库的数据即可，其他仓库不受影响。
- 在客户环境满足的情况下，CODING 可以提供卓越的高可用能力，保证代码的安全性。
- 代码仓库支持硬盘 Raid、全程加密传输、定时备份、实时备份等多种手段保证代码的安全，降低代码管理风险。
- 为收费服务，收费服务意味着契约精神，CODING 有责任保障客户的代码及相关资产的安全。
- 自 2014 年运营至今，服务了 300 万研发团队与数万家企业用户，未出现信息泄漏的事件，在业内中拥有良好的口碑和信誉。

# 持续集成相关

最近更新时间：2024-11-01 11:11:02

## CODING 持续集成兼容的 Jenkins 版本?

目前，CODING 持续集成提供的公共构建节点使用 Jenkins 2.293 版本。
CODING 目前已经支持了凭据管理，我们建议用户使用更安全的凭据来管理密钥。如需了解如何使用凭据，请参见 凭据管理 。

## 单引号和双引号用法差异是什么?

使用 CODING 持续集成时经常需要在 Jenkinsfile 内拼接字符串或使用环境变量作为参数，Jenkinsfile 中的单引号和双引号在使用时，会有些许差异，以下演示常用的 echo 与 sh 两个命令的差异。

```
pipeline {
    agent any
    environment {
        MY_ENV = 'this is my env'
    }
    stages {
        stage('Test') {
            steps {
                script    {
                    def MY_ENV = 'define in script'

                    echo "${env.MY_ENV}"
                    // 输出内容为 this is my env

                    echo "\${env.MY_ENV}"
                    // 输出内容为 ${env.MY_ENV}

                    echo "${MY_ENV}"
                    // 输出内容为 define in script

                    echo '${MY_ENV}'
                    // 输出内容为 ${MY_ENV}

                    sh 'echo ${MY_ENV}'
                    // 输出内容为 this is my env

                    sh "echo ${MY_ENV}"
                    // 输出内容为 define in script

                    sh "echo ${env.MY_ENV}"
                    // 输出内容为 this is my env
                }
            }
        }
    }
}
```

- echo 在使用单引号时，并不会解析里面的 $ 符号，而是直接输出原文；在使用双引号时，会打印出环境变量里的 MY_ENV。
- sh 在使用单引号时，将原文当作我们平时在终端里 sh 的命令一样执行，所以可以打印出环境变量里的 MY_ENV。

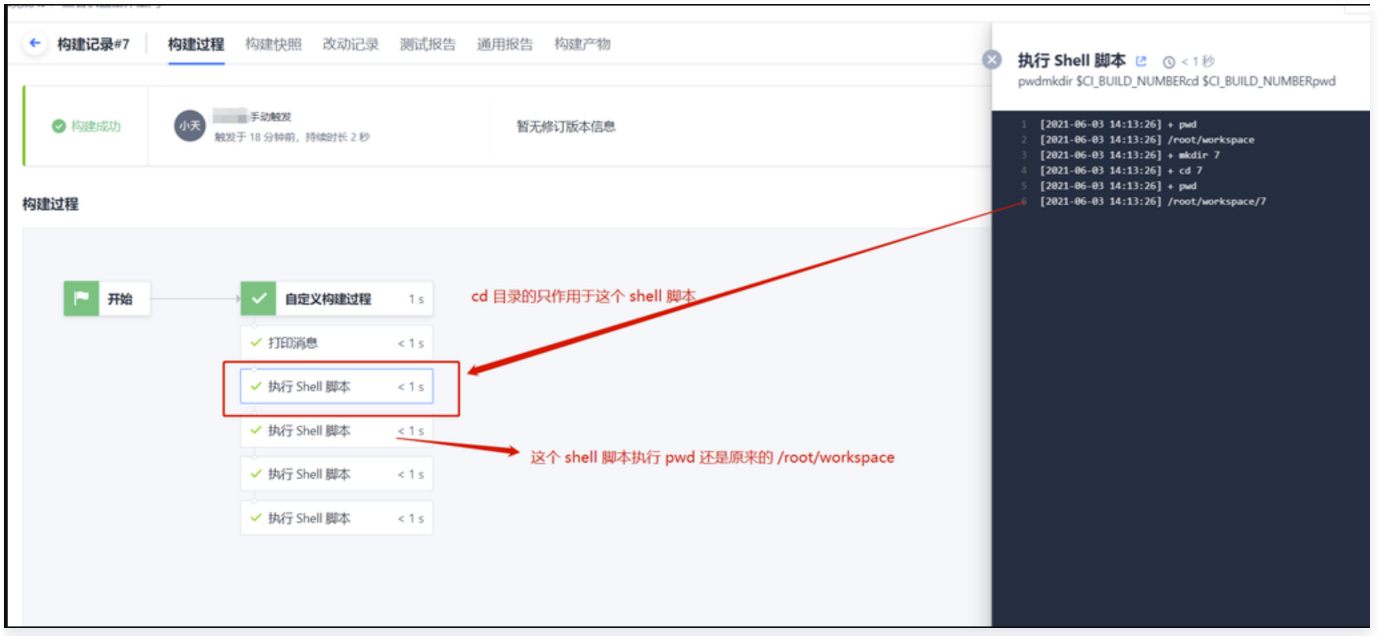
## 持续集成的配置来源区别是什么?

在创建构建计划时选择使用 **代码仓库中的 Jenkinsfile** 与 **静态配置的 Jenkinsfile** 有以下区别：

- 选择使用代码仓库中的 Jenkinsfile 后，该文件将存储至代码仓库中。修改 Jenkinsfile 意味着需在代码仓库中提交修改记录，若修改持续集成的触发条件，还可以自动触发集成任务。
- 使用静态配置的 Jenkinsfile 后，该文件将不会存储在代码仓库中，修改 Jenkinsfile 不会更新代码仓库内容，执行构建时将统一使用静态配置，保障构建流程的一致性。

## 如何查看工作空间目录?

在持续集成的部署流程中添加**执行 Shell 脚本**步骤，并在其中添加 pwd 命令。持续集成运行后将输出工作空间目录。




## 如何自定义环境变量?

详细说明请参见 自定义变量 。

## 远程 SSH 执行命令时环境变量不生效怎么办?

由于在使用构建机连接远程 SSH 时使用了**非交互非登录式**连接，因此无法引用远程机器的 `/etc/profile` 、 `~/.bashrc` 等文件配置中的环境变量。您可以参见以下示例，使用 export 命令再设置变量且用 `&&` 符号连续输入命令。

```
export PATH=/opt/jdk1.8.0_281/bin:$PATH && java -version
```

# 持续部署相关

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 。

## 持续部署支持哪些制品类型?

持续部署支持 Docker 镜像、Generic 文件、War 包。

## 持续部署支持哪些集群类型?

持续部署支持 CVM（Linux 操作系统）、TKE、SCF。

## 如何在配置持续部署过程的时候保护敏感信息?

对于 Token、SSH 私钥、Kubernetes 证书等保密信息，可以在 CODING DevOps 网页端的任一项目选择**项目设置 > 开发者选项 > 凭证管理**，进入**凭证管理**页面中设置，并可以在此设定哪些持续部署发布流程可用。

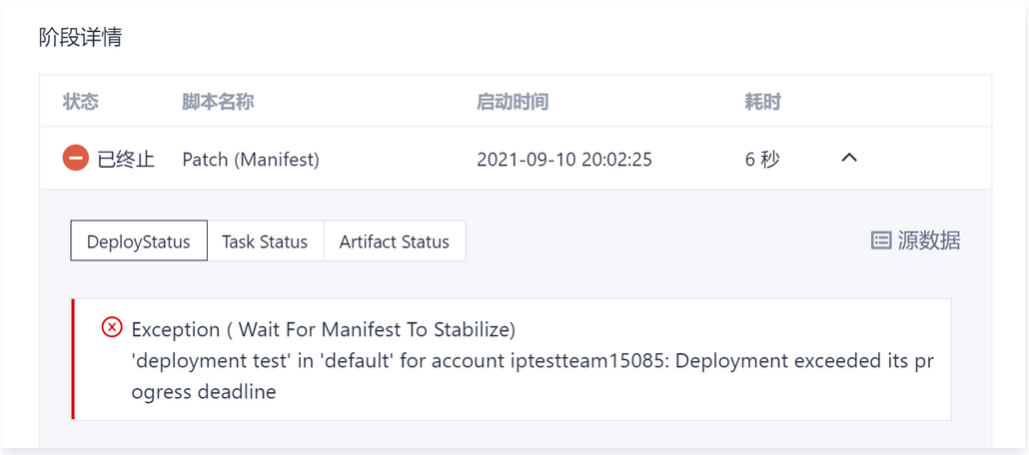
## 如何发布源代码?

很多常见的动态语言是没有编译和构建过程的，可以直接在应用的制品设置中配置 Git 代码仓库指明文件路径后发布。

## 常见错误码

## Deployment exceeded its progress deadline

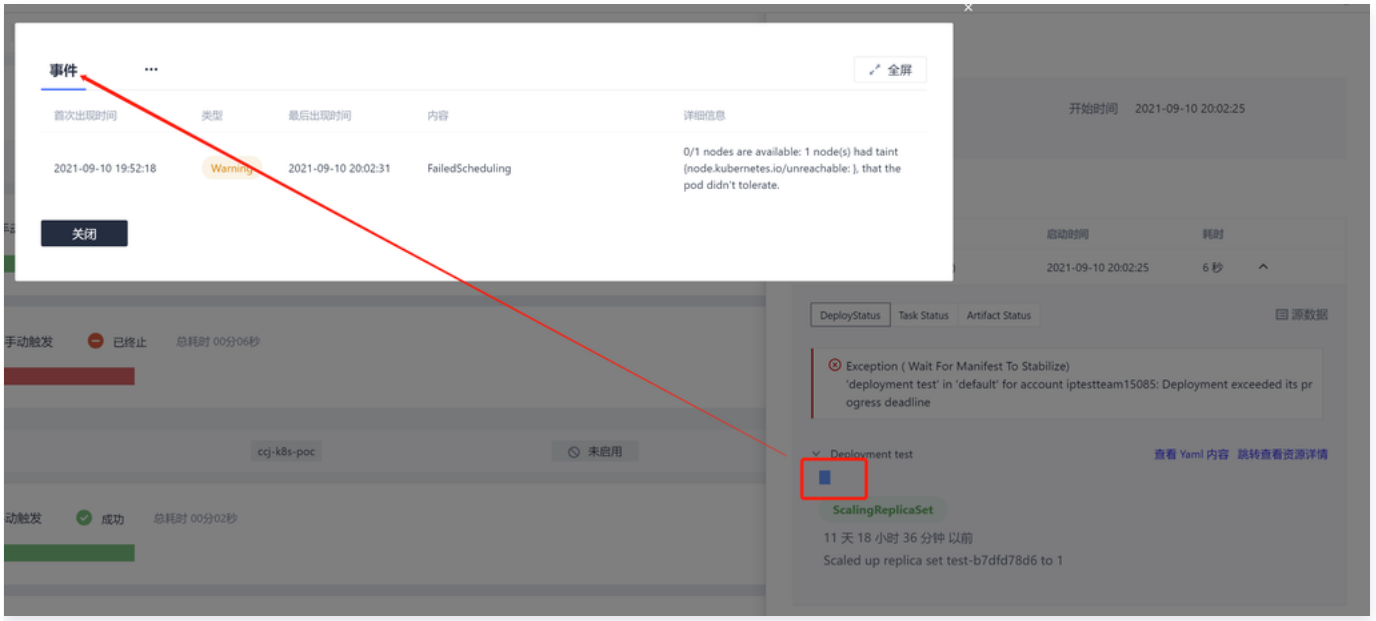
错误截图如下：




这是由于 deployment 的 pod 没有成功运行所导致的错误。可以通过以下方式查看 pod 事件详情，需要根据 pod 的事件排查问题。

**方法一：**

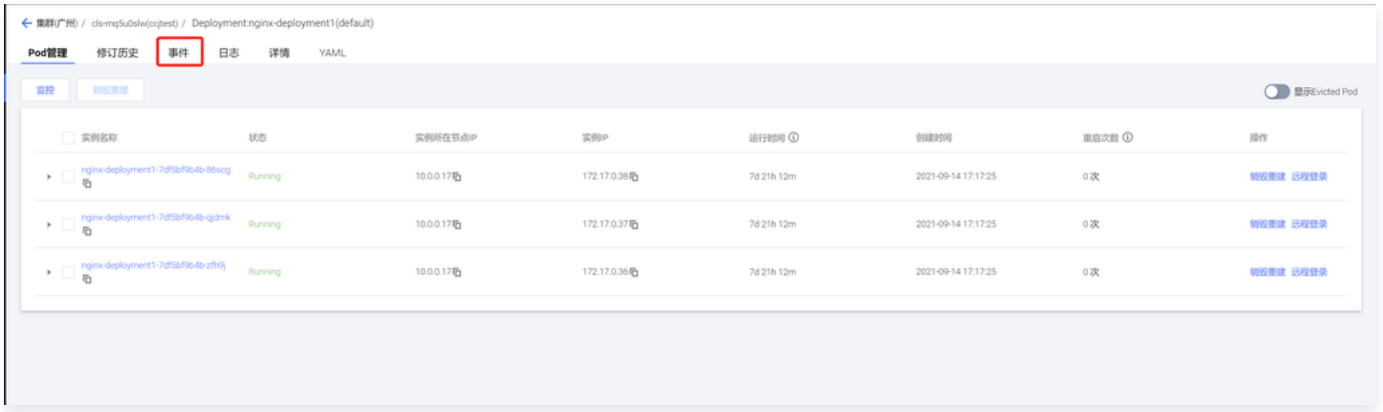
在产品页面单击**方块**查看事件。




**方法二：**

**前往您的集群查看 pod 事件。**

**例如在 TKE 界面查看 pod 的事件：**




**使用命令行查看 pod 的事件：**

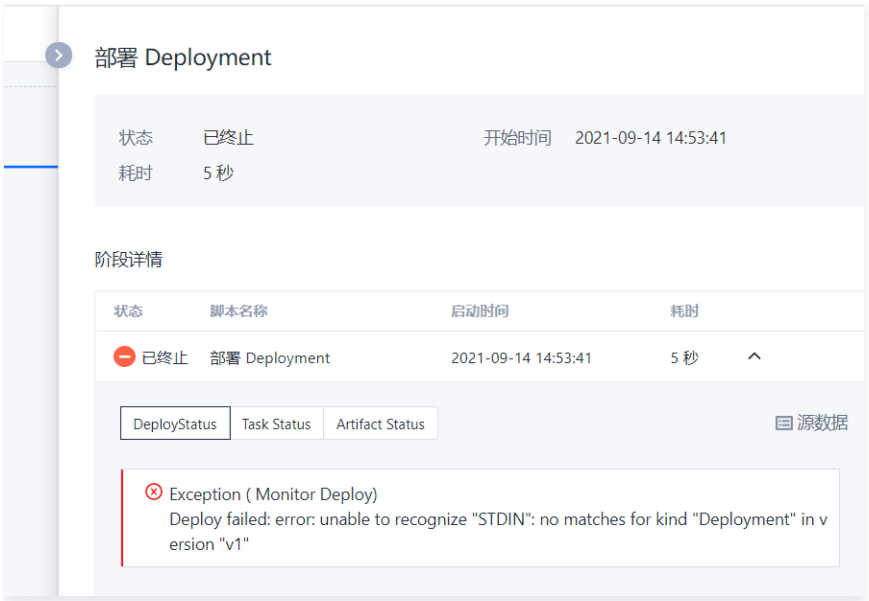
```
kubectl describe pod [pod name] -n [pod location]

kubectl describe pod nginxtest -n test
```

## Failed to pull image xxxxxx

**这是因为在 yaml 文件里配置了拉取 CODING docker 镜像，但没有配置 CODING docker 仓库的凭据，需要在 yaml 文件添加相关的 imagePullSecrets。**

**若要解决此问题，需要将 CODING 登录凭据添加到您的 kubernetes 集群的 secret，请参见 部署 Kubernetes 资源时如何拉取私有库镜像 。**

## Deploy failed: error: unable to recognize “STDIN”




**出现此报错的原因一般为 yaml 文件格式问题。建议在本地集群验证 yaml 文件，确保能够在本地集群正确执行之后，再粘贴至 CODING 上使用。**

## ConfigMap/Secret

**问题详情：**

**部署完成后，** `ConfigMap/Secret` **参数中新增了** `v00x` **版本号。**

**解决办法：**

**在** `configmap/secret` **的** `annotations` **参数中加入** `strategy.spinnaker.io/versioned: false` **命令行。**

# 制品管理相关

最近更新时间：2024-11-01 15:36:22

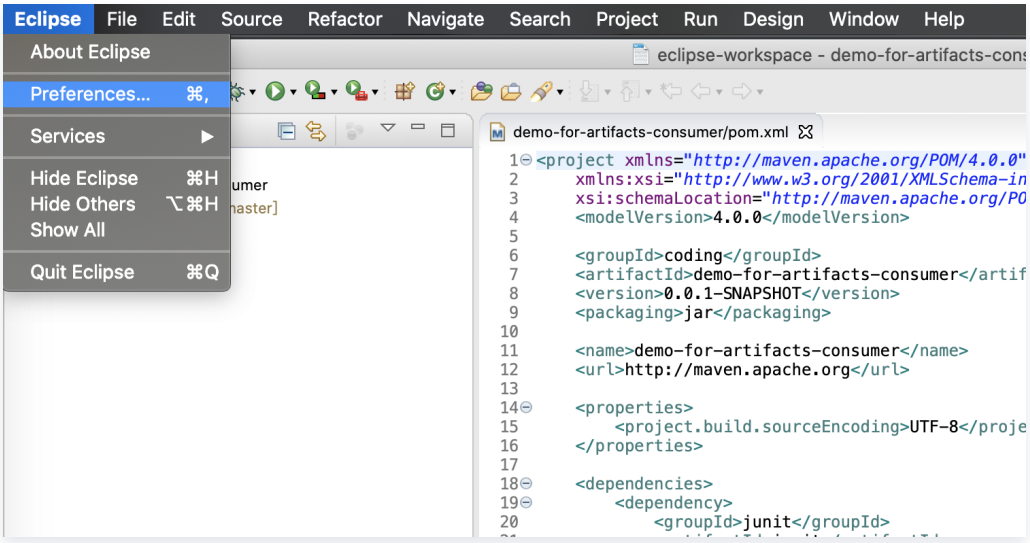
## Maven 的 settings.xml 配置文件在哪?

在生成 Maven 类型制品时，您需要配置您的 settings 文件，通常这个文件存放的位置有如下几个地方，您都可以按需使用，只不过配置生效的范围和优先级不同：

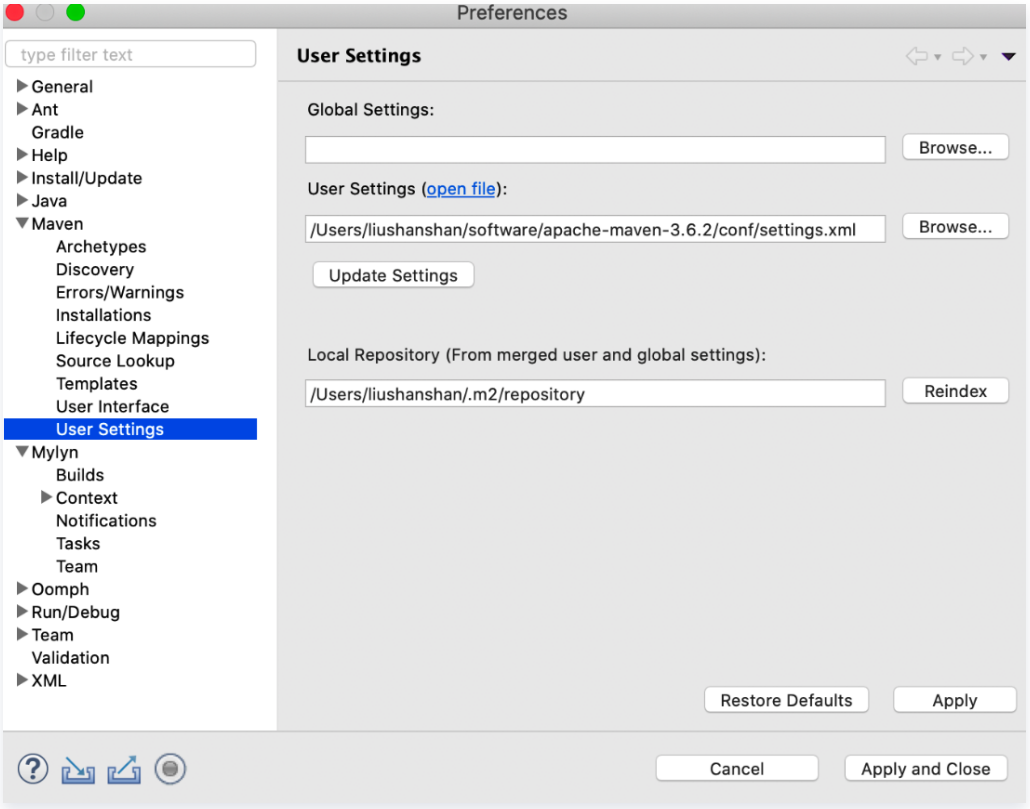
1. 全局配置：`${M2_HOME}/conf/settings.xml` 。
   如果您不记得 Maven 的安装目录 `${M2_HOME}` ，您可以在终端中执行 `echo ${M2_HOME}` 或者 `mvn -version` 就可以看到 Maven home 的路径。
2. 用户配置：`${user.home}/.m2/settings.xml`
   您可以通过 echo 环境变量的方式找到该文件目录，有时候这个目录下是没有 settings.xml 文件，您可以去全部配置里拷贝一份 settings.xml 再进行修改。
3. 指定路径下的 settings.xml。

```
mvn deploy --settings settings.xml
```

4. 在终端执行 mvn 相关命令时，settings.xml 配置生效的优先级：指定路径 > 用户配置 > 全局配置。
5. 除了在终端当中执行 mvn 命令，有时候您在 Eclipse 等 IDE 中也会用 Maven，该怎么修改 settings.xml 文件的配置?
   以 Eclipse 为例（其它类型 IDE 网上也有丰富的资料供参考）：
   5.1 单击 Preferences 进入偏好设置。

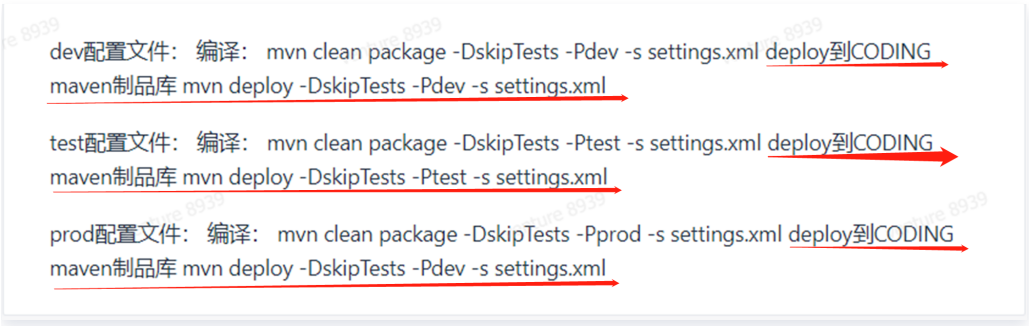



5.2 在 Maven > User Settings 当中您就可以看到您使用的配置文件路径，并且修改配置文件。




## 打包时指定了环境变量未生效?

打包 Maven 制品时指定了环境变量，例如指定了 `-Ptest` ，但上传至制品仓库时依然没有生效，依然为默认的 Dev 环境。若要解决此问题，请确认在使用 `mvn deploy` 命令推送到 CODING maven 制品库时是否有加 -P 参数指定环境。




## 如何将 npm @scope 指向 CODING 私有制品库?

1. 可以通过配置 `.npmrc` 来指定 @scope 的 registry。

   例如：有一个 npm 包，位置信息如下：

   - 企业：my-team
   - 项目：my-project
   - 制品仓库：my-npm-repo
   - 名：@my-scope/my-pkg

   可以通过配置 `.npmrc` ，让 package.json 中的 `@my-scope/my-pkg` 指向该链接地址：

```
@my-scope:registry=https://my-team-npm.pkg.coding.net/my-project/my-npm-repo/
```

2. 直接将 npm 包的 registry 指向 CODING 私有制品库。

   直接单击 npm 制品库指引页面中的**使用访问令牌生成配置**生成 `.npmrc` 。
   请妥善保管生成的配置：

```
registry=https://my-team-npm.pkg.coding.net/my-project/my-npm-repo/
always-auth=true


//my-team-npm.pkg.coding.net/my-project/my-npm-repo/:email=xxxxx
```

   由于 CODING 的 npm 制品库支持代理功能，可以直接将 npm registry 设置为 CODING 私有制品库，公共制品也可以被拉取到。
   关于如何在 CODING 持续集成中使用 npm 制品库，请参见 《持续集成——构建 npm 类型制品》 。

## 如何管理第三方依赖包?

通常在研发过程中，代码存在依赖库，编译时也会用到第三方的依赖包。此时可以通过 制品扫描 检测第三方依赖包的安全情况。另外还可以在企业内部建立可信软件源，保证第三方依赖包能够被统一纳管。

## 第三方制品仓库如何迁移至 CODING?

您可以在脚本中调用 Open API 接口实现制品仓库批量迁移。

# 测试管理相关

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 。

## 测试报告是否支持导出或下载?

测试报告支持以 pdf 文件格式进行下载。生成测试报告后，单击**下载**后等待报告下载完成。


## 导入测试用例时测试步骤不符怎么办?

**问题描述：** 使用 Excel 文件导入测试用例时，发现测试用例中的步骤数量不符。
**解决方案：** 出现此问题的原因是 Excel 表格内的测试步骤中额外填写了空白换行符，导致识别单个测试用例时包含预期外的步骤。您可以检查出现错误的测试用例，在 Excel 表格中删除其中多余的换行符。

## 自动化测试所触发的持续集成任务会消耗团队资源吗?

由系统触发的自动化持续集成任务并不会消耗团队中的构建分钟数。

# 效能洞察相关

最近更新时间：2024-09-04 14:25:11

## 研发团队为什么需要效能洞察?

在没有使用效能洞察前，研发团队通常面临着研发数据分散在各处而导致数据不完整或不准确，影响研发投入决策；并且团队内缺乏掌握数据分析能力的人才，有了度量意识但却无从下手。
效能洞察可以帮助研发团队改善业务流程和提高生产效率。效能洞察提供模板视图、模板图表、图表数据下钻等功能，可以帮助研发团队发现业务瓶颈，找到低效率环节；洞察业务流程和资源使用情况，避免资源浪费和不合理分配；追踪关键业务指标，及时发现问题和机会并进行相应的调整和优化。

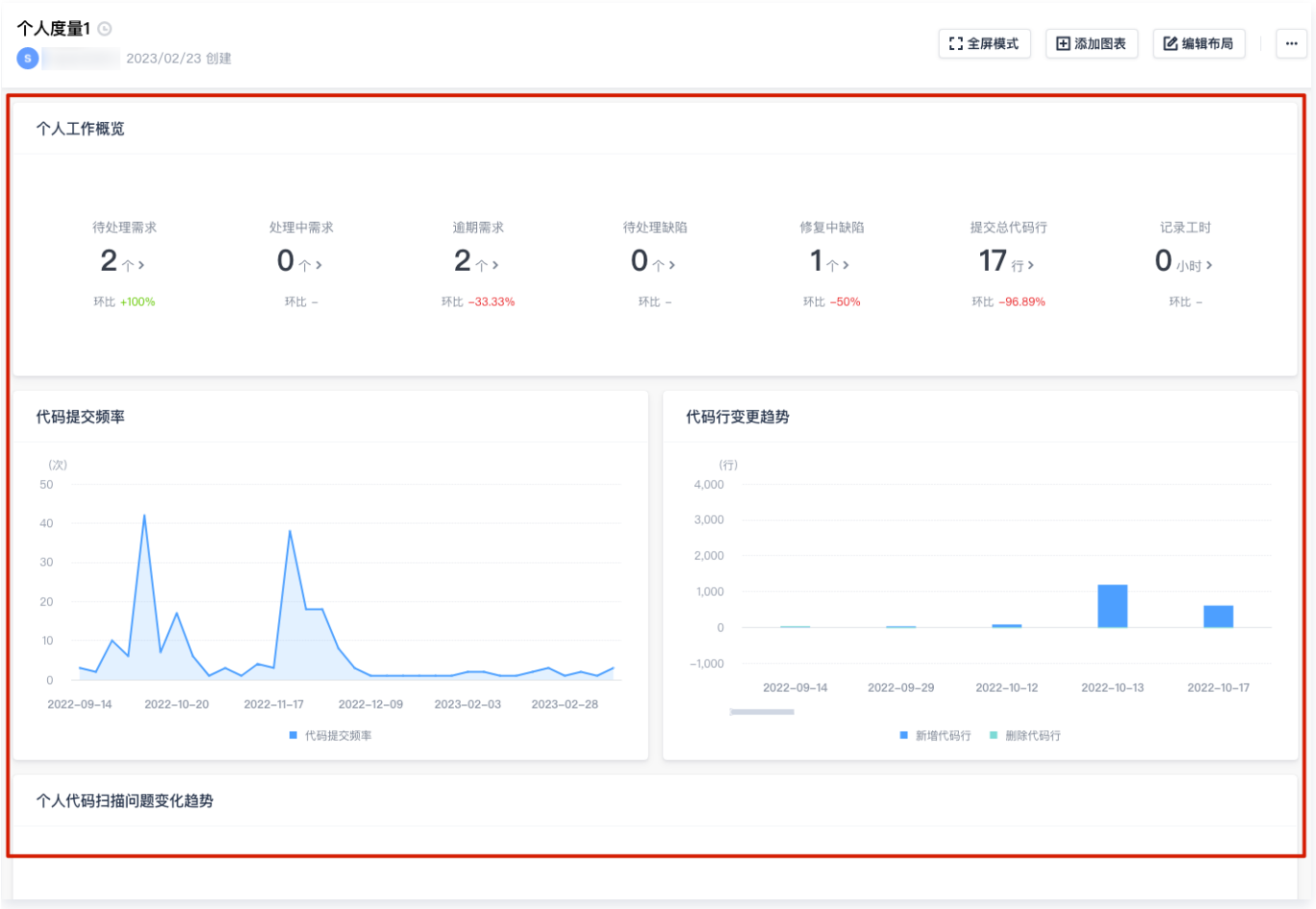
## 效能洞察的应用场景有哪些?

效能洞察有如下应用场景：

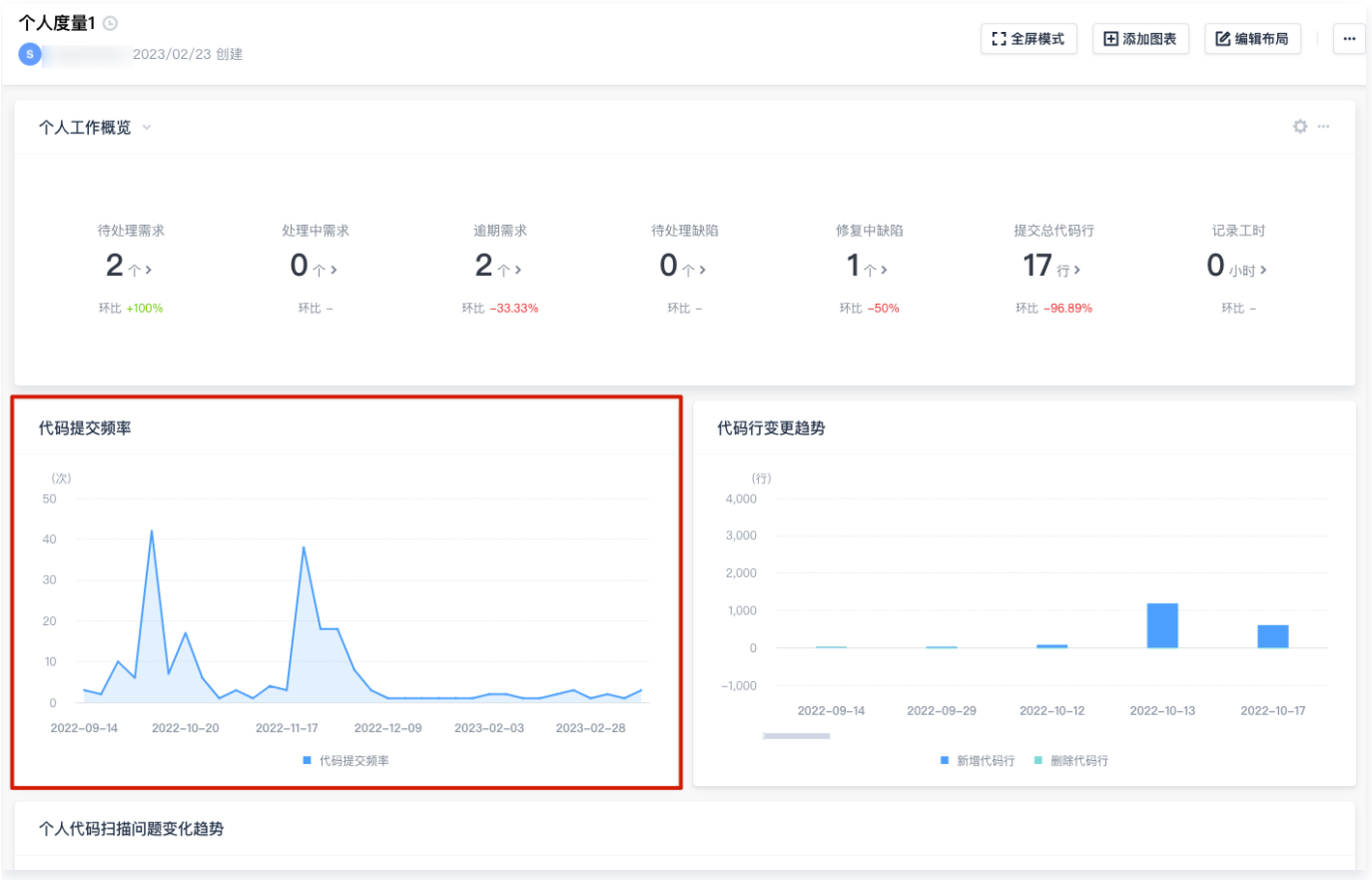
- **研发过程检视场景：**随时查看项目推进进度。
- **项目结束后复盘场景：**项目生命周期结束后的总结与复盘。
- **团队业绩目标制订场景：**在项目开始前科学制订团队业绩目标报表，使得后续开展的工作指标有对照与指引。
- **数字化运营分析场景：**效能洞察可以广泛适用于工作分析、项目复盘等需要统计分析的工作场景。

## 视图和图表有什么区别?

视图类似一个仪表盘，用于集中展示多项图表。

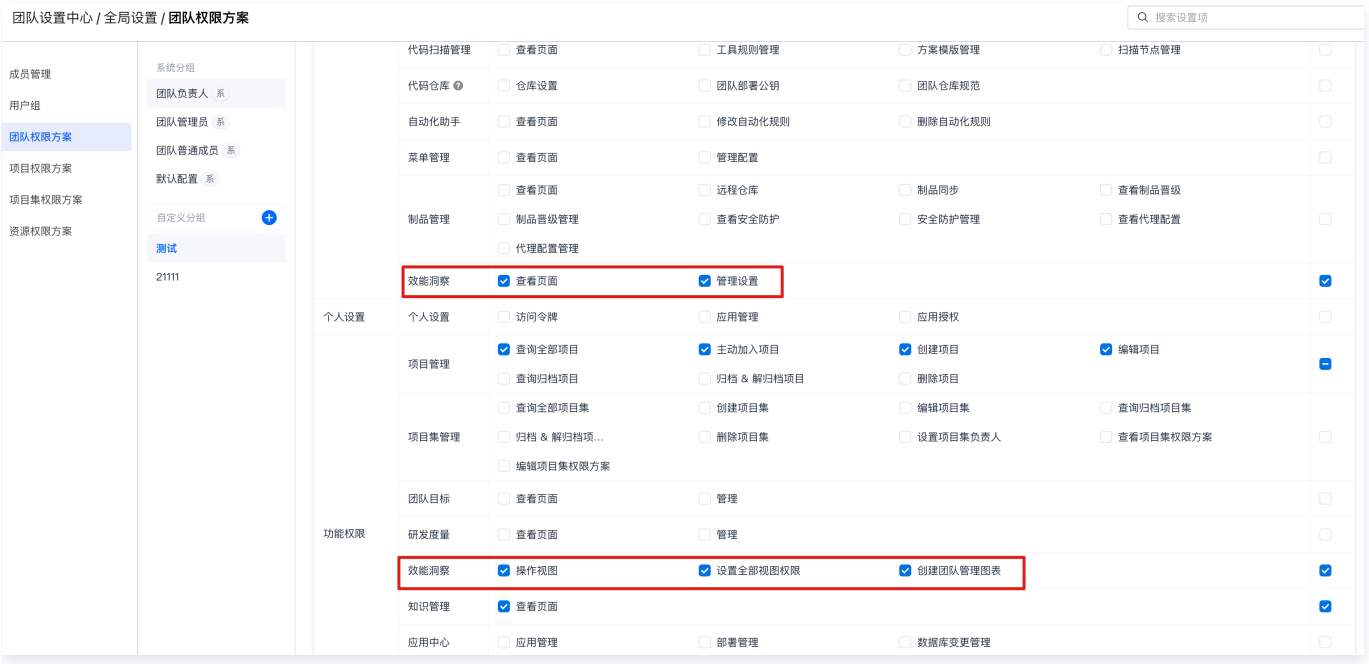



图表是一个单独的统计窗口，用于展示特定统计项目中的数据。




## 在首页中没有看到效能洞察怎么办？

这可能是当前团队成员不具备效能洞察的访问权限。请联系团队管理员，参考 权限配置 文档将团队成员添加至拥有效能洞察的“查看页面”权限的用户组中。




## 如何添加周期指标方案？

仅以下六种图表支持添加指标方案：

- 需求领域：需求开发周期、需求测试周期、需求交付周期。


- 缺陷领域：缺陷响应周期、缺陷修复周期、缺陷关闭周期。


在视图模板页右上角单击**添加**，添加图表后，使用上述类型图表，选择项目范围后单击**查看方案**。


**指定方案统计范围并配置参数后即可使用。**


修改方案
你可前往「团队设置中心–功能设置–周期指标方案设置」批量设置项目周期指标
* 方案名称
测试方案
方案描述
请输入方案描述，不超过 100 个字符
* 应用项目
* 周期指标设置
需求周期
事项类型： 任务
周期设置
开始阶段
结束阶段
需求交付周期
未开始
已完成
需求开发周期
未开始
处理中
需求测试周期
未开始
处理中
缺陷周期
事项类型： 缺陷
周期设置
开始阶段
结束阶段
缺陷关闭周期
待处理
已关闭
缺陷响应周期
处理中
待验证
保存
取消