CODING DevOps

Code Hosting

Contents

Code Hosting Quick Start

Last updated：2024-09-05 16:11:00

Initialization Phase

Local Installation of Git

Windows

1. Download the Git client from Git Official Website and follow the instructions to complete the installation. It is recommended to use the default options.
2. After the installation is complete, you can use the right mouse button to bring up the Git Bash terminal and start your Git journey.

Linux

In Linux, Git can be installed directly using the package management tool provided by the system to install precompiled Git binary packages.

- Install with yum in Fedora/CentOS: `yum install git-core`
- Install with apt-get in Ubuntu/Debian: `apt-get install git`

macOS

1. Run the following command to install the package management tool Homebrew.

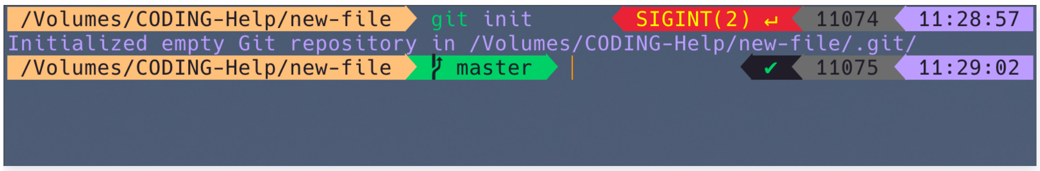
```
/bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"
```

> **Note:**
> If an error is reported, refer to this Open-source Repository and switch to a domestic source for downloading.

2. After Homebrew is installed, enter the command `brew install git` in the terminal to complete the Git installation.
3. After installing Git, you can run `git --version` to check the current Git version.

Local Environment Initialization

After installation, create a new folder and enter the folder in the terminal. Enter the command git init to complete the local environment initialization.



```
 /Volumes/CODING-Help/new-file  git init        SIGINT(2) ↵  11074  11:28:57
Initialized empty Git repository in /Volumes/CODING-Help/new-file/.git/
 /Volumes/CODING-Help/new-file   master |                 ✔  11075  11:29:02
```

Setting User Information

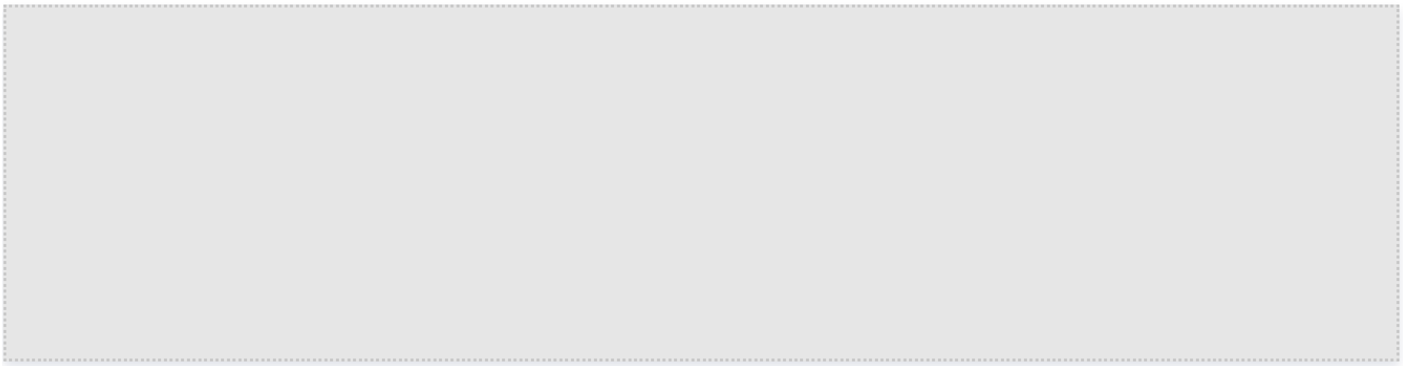
After installing Git, you should set the committer's name and email address promptly. Each subsequent commit will use this information as a record. The commands to set user information are as follows:

For example, your CODING account nickname is "大黑", and the user information configured in Git is: `user.name 大白`, `user.email dabai@coding.net`. After you push the code to the CODING repository, the activity is displayed as follows:

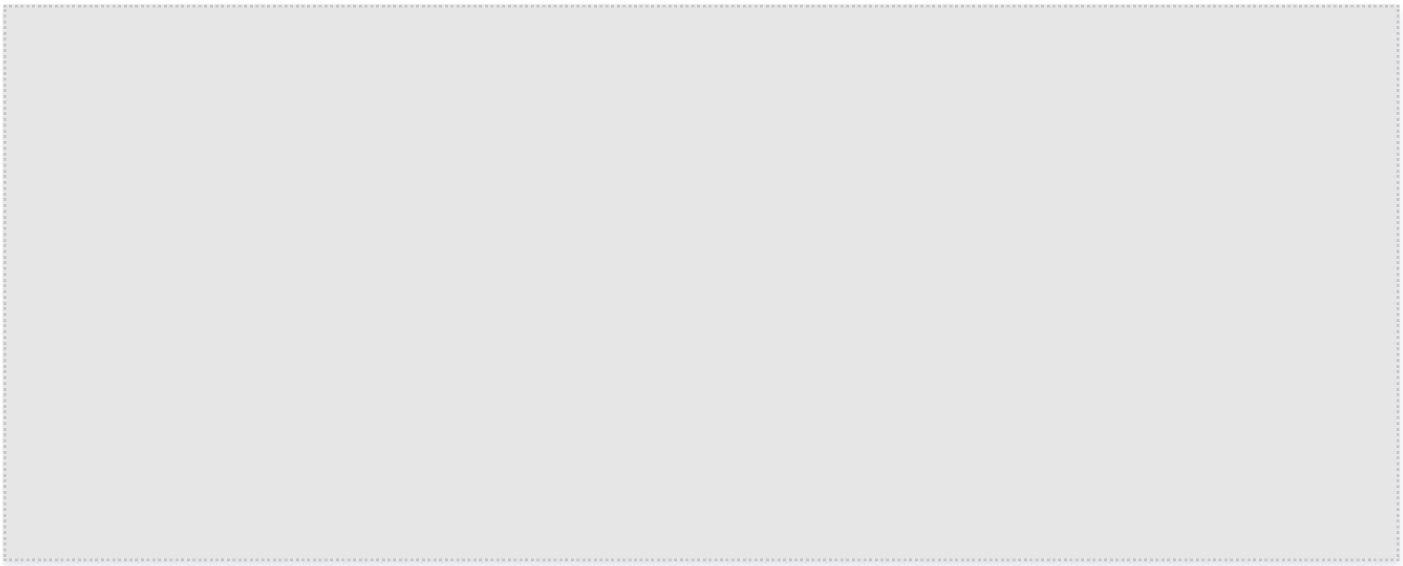
The user information configured in Git is customizable. It is recommended to directly fill in your CODING username and registered email for better collaboration.

Creating a CODING Code Repository

Enter any project and click on **Code Repository** in the left navigation bar to enter the code repository management page, and then create or enter any repository.



If the code repository entry is not shown, the project administrator needs to enter the project, click on **Project Settings** at the bottom left corner, and then open the code repository feature in **Project and Members > Menu Management**.

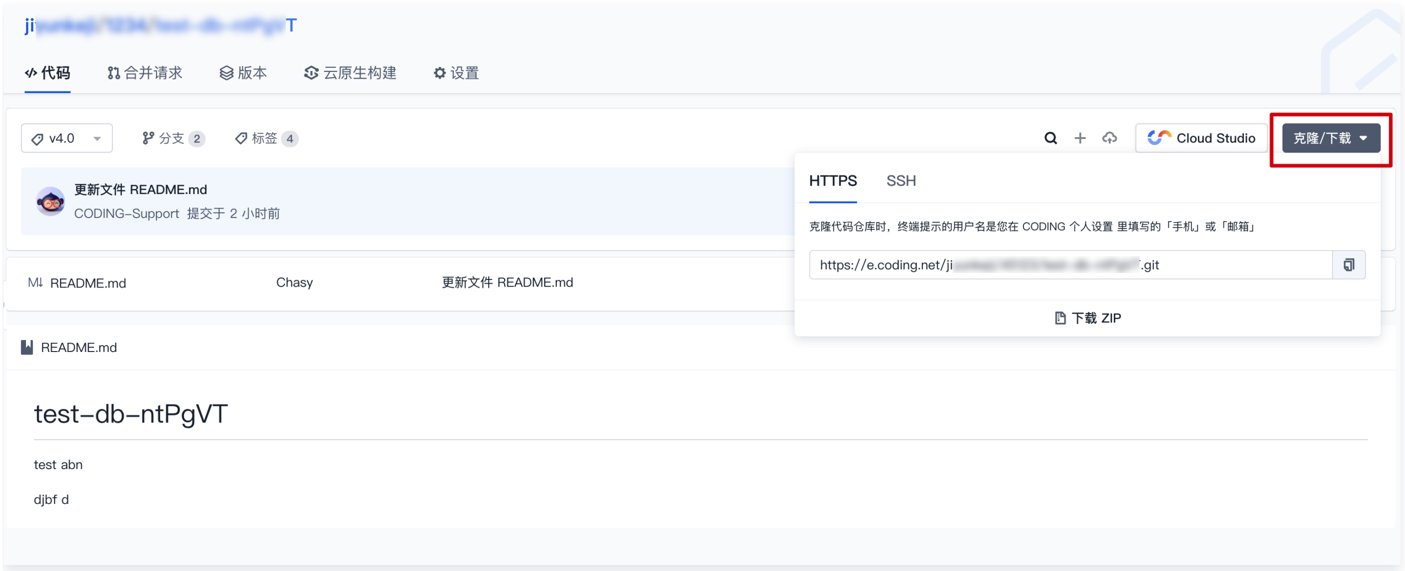


Push and Pull Code

This operation will demonstrate how to pull code from a remote repository / upload local code to a remote repository, aiding your code cloud journey.

Pulling Remote Repository Code

After initializing the code repository locally, bring up the terminal in the folder and input the command `git clone + repository address` to pull the code.



ji[redacted]T

代码 合并请求 版本 云原生构建 设置

v4.0 分支 2 标签 4 Cloud Studio 克隆/下载

更新文件 README.md
CODING-Support 提交于 2 小时前

README.md Chasy 更新文件 README.md

HTTPS SSH

克隆代码仓库时，终端提示的用户名是您在 CODING 个人设置 里填写的「手机」或「邮箱」

https://e.coding.net/ji[redacted].git

下载 ZIP

README.md

test-db-ntPgVT

test abn

djbf d

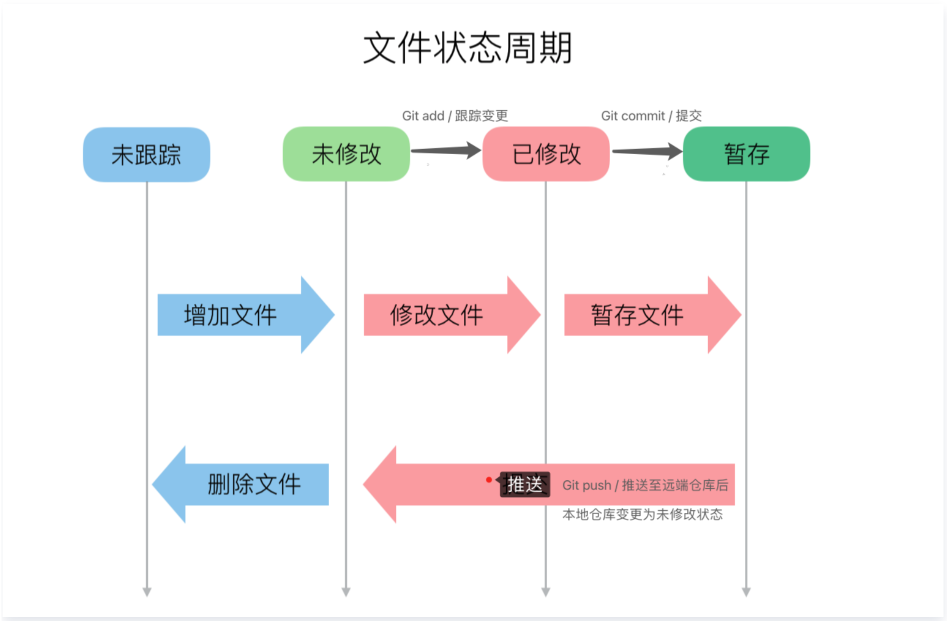
After the first pull, you will be prompted to fill in the credentials. Here, you can enter the email and password used at the time of registration with CODING. You can also go to the **Personal Account Settings** at the bottom left corner to modify the credential information.
After the command operation is successful, you can modify the code in the local code repository.

Editing Files

In the folder, create `readme.txt` and `learn-git.txt` files, write the sentence `I'm learning git.` (you can fill in the content as needed) in one of the files and save it.

Flow Diagram

Before officially submitting the code, you can refer to this flow diagram to understand the status cycle Git goes through when tracking files.

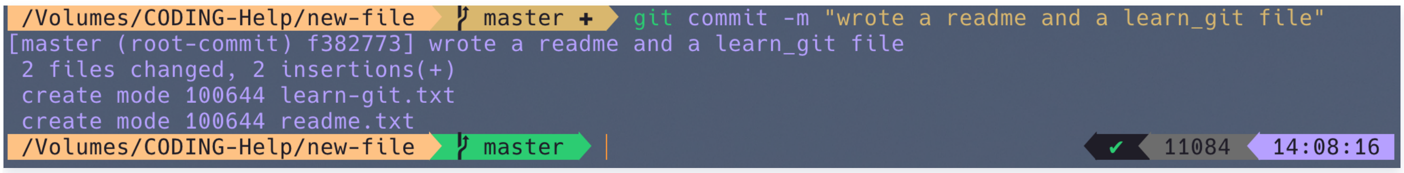


Track Files (git add)

After creating or modifying files, call the `git add` command to add the changed files to the local Git repository's staging area (Index Stage).
Command to track specific files:
`git add readme.txt`
Command to add multiple files:
`git add readme.txt learn_git.txt`
If you want to **track all files** at once, you can directly enter `git add .` in the terminal

Commit Files (git commit)

Once the files are staged, run the git commit command to officially commit the files to the local repository. This command will commit all files in the staging area at once:
`git commit -m "wrote a readme and a learn_git file"`

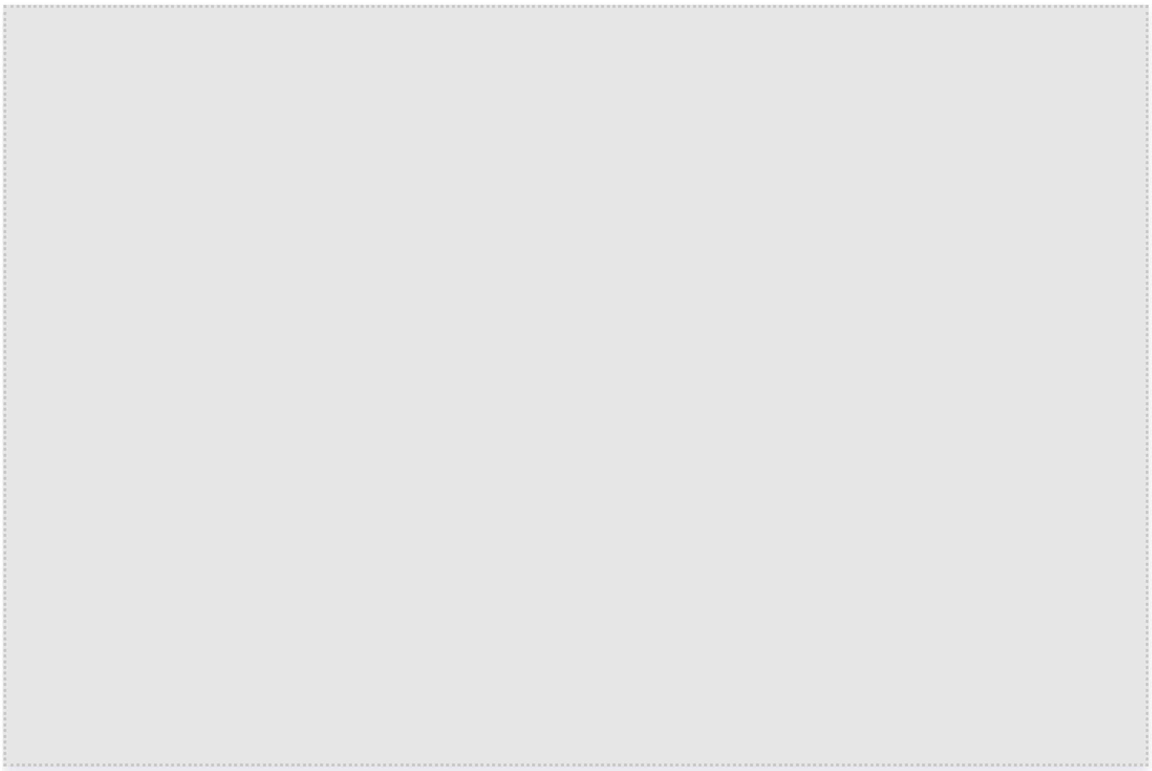


```
 /Volumes/CODING-Help/new-file  master ✚  git commit -m "wrote a readme and a learn_git file"
[master (root-commit) f382773] wrote a readme and a learn_git file
 2 files changed, 2 insertions(+)
 create mode 100644 learn-git.txt
 create mode 100644 readme.txt
 /Volumes/CODING-Help/new-file  master                                        ✔  11084  14:08:16
```

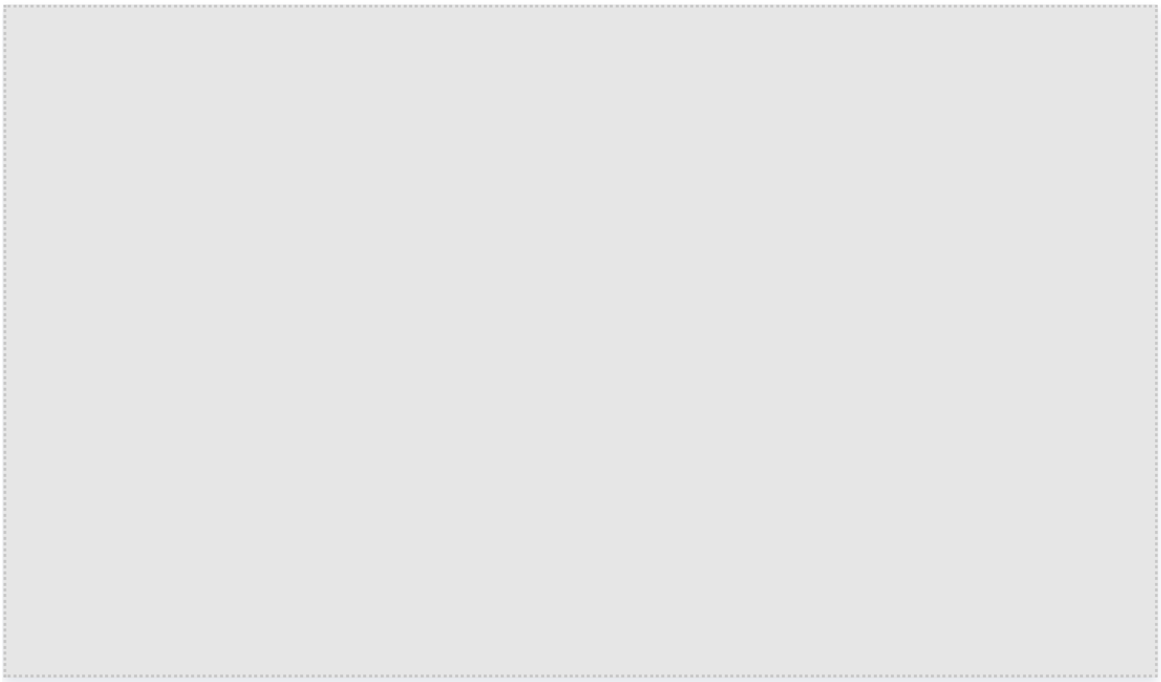
`The content inside the quotes after >-m` is your commit message. The following lines are the terminal's return results. By attaching a change description to each commit, you can clearly control what modifications were committed.
In addition to the `git commit` command, you can also use standardized plugins to standardize commit information in the repository, making it easier to backtrack later.

Automatic Issue Association

When committing code, you can also include the `# Issue ID` in the commit message to directly associate with issues in the project, allowing you to view related code commit records within the issue.
For example: If the issue (type: requirement) ID is 630, adding `#630` in the commit message will automatically associate it with the requirement; adding `project-1#630` will associate it with issue 630 in `project-1`.



After associating issues, hover the mouse over the commit record's ID to preview the issue.



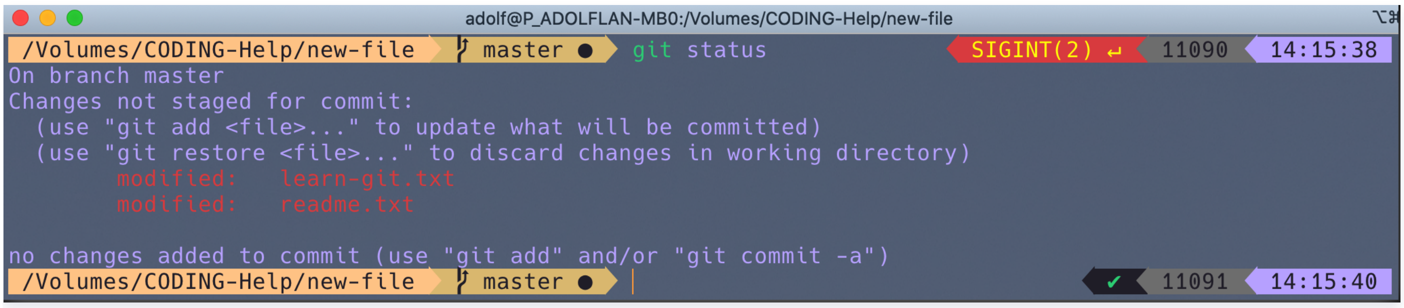
Check File Status (git status)

Use the `git status` command to check the file status. This command allows you to confirm whether Git is accurately tracking the modified files and their status.
When no files in the current repository are being tracked, the result is as follows:

```
$ git status
```

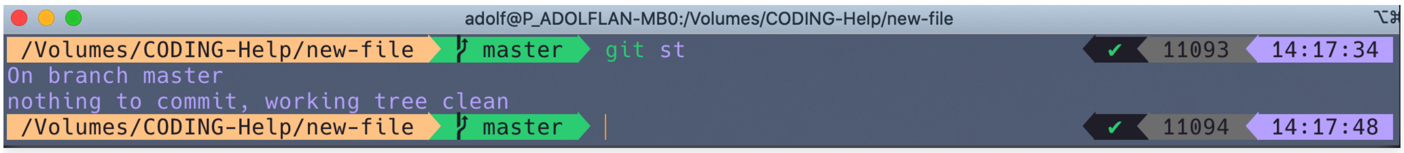
```
On branch master
Your branch is up-to-date with 'origin/master'
nothing to commit, working directory clean
```

When files have changes but are not being tracked, the result is as follows:



```
adolf@P_ADOLFLAN-MB0:/Volumes/CODING-Help/new-file
/Volumes/CODING-Help/new-file  master ● git status          SIGINT(2) ↵  11090  14:15:38
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   learn-git.txt
        modified:   readme.txt

no changes added to commit (use "git add" and/or "git commit -a")
/Volumes/CODING-Help/new-file  master ●                          ✔  11091  14:15:40
```

At this point, running the `git add` command will track the files. Once tracked, the text color changes from red to green.
When files are tracked and have been committed to the repository, the result is as follows:



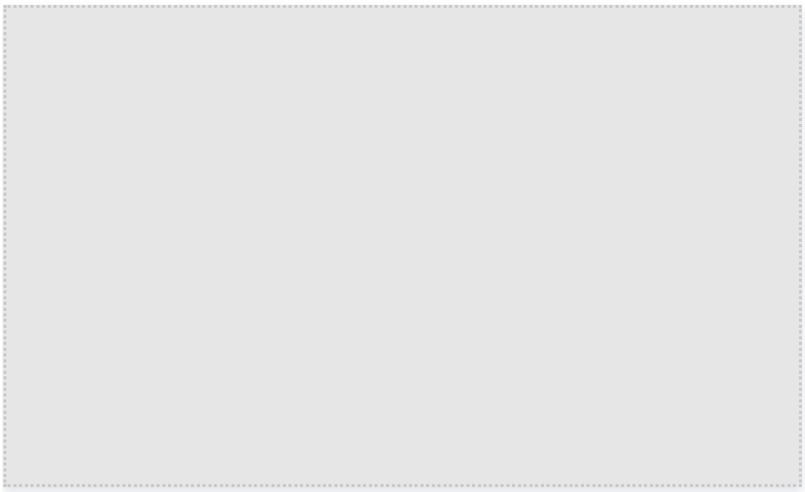
```
adolf@P_ADOLFLAN-MB0:/Volumes/CODING-Help/new-file
/Volumes/CODING-Help/new-file  master  git st                    ✔  11093  14:17:34
On branch master
nothing to commit, working tree clean
/Volumes/CODING-Help/new-file  master                             ✔  11094  14:17:48
```

Push Files to Remote Repository (git push)

Run the following command in the terminal:
`git push`
To automatically create a merge request and associate it with the project upon submission, you can use the following command:
`git push origin local-branch:mr/target-branch/local-branch`
`git push` is the push command, which essentially pushes the local branch to the remote repository, creating a backup on the remote. Go to the CODING repository to view the pushed files. If multiple people are collaborating on the remote repository, once others have committed their code, you only need to run the `git pull` command locally to keep your local code in sync with the remote.

View or Edit Remote Repository

After the files are pushed to the CODING repository, you can edit the code, save, and submit changes on the webpage.
For example, with the `README.md` file, after editing and committing, you can provide a brief description of the changes. If not filled in, the default commit message is "Update file xxx".



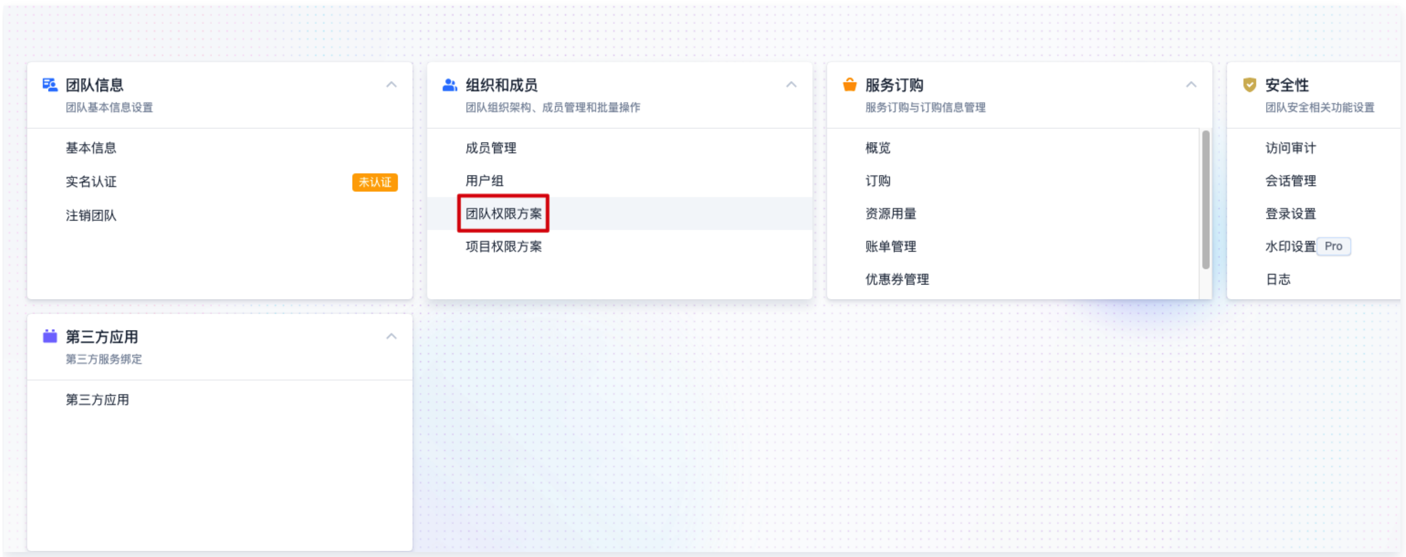
Configuring permission

Last updated：2024-09-05 16:11:19

Based on the usage scenarios and feature positioning of code repositories, the types of permissions include "Team Permission Scheme", "Project Permission Group", and "Individual Repository Permission".

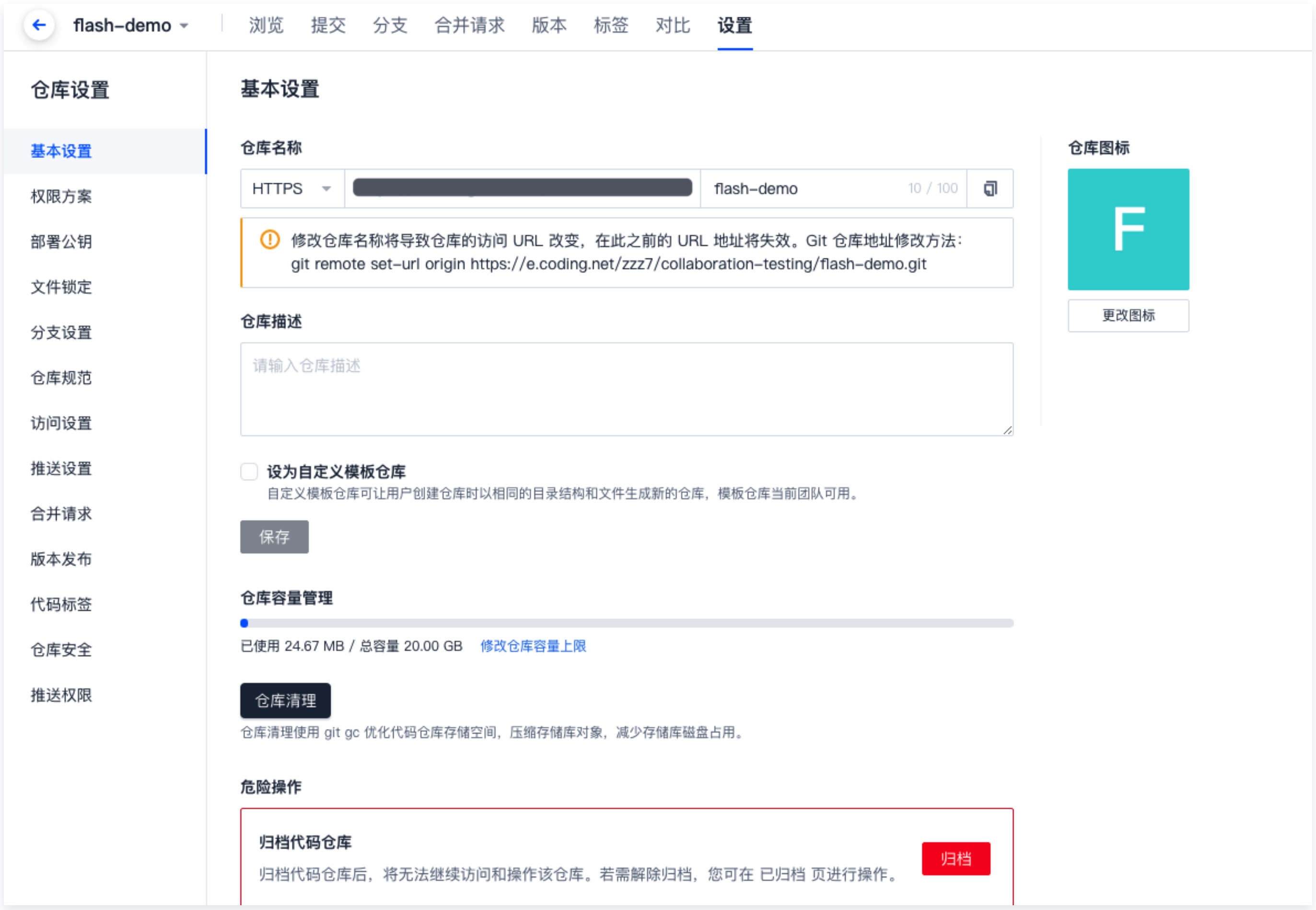
Team Permission Scheme

Repository settings, team deployment of public keys, and team repository standards fall under team-level permission management. This is adjusted by the team owner/administrator in the Team Settings Center under "Team Permission Scheme".



- Repository Settings

  Members in the group with this permission point can access the **Settings** page within all repositories of the project.



flash-demo 浏览 提交 分支 合并请求 版本 标签 对比 设置

仓库设置

基本设置
权限方案
部署公钥
文件锁定
分支设置
仓库规范
访问设置
推送设置
合并请求
版本发布
代码标签
仓库安全
推送权限

基本设置

仓库名称

HTTPS [redacted] flash-demo 10 / 100

修改仓库名称将导致仓库的访问 URL 改变，在此之前的 URL 地址将失效。Git 仓库地址修改方法：
git remote set-url origin https://e.coding.net/zzz7/collaboration-testing/flash-demo.git

仓库描述

请输入仓库描述

设为自定义模板仓库
自定义模板仓库可让用户创建仓库时以相同的目录结构和文件生成新的仓库，模板仓库当前团队可用。

保存

仓库容量管理

已使用 24.67 MB / 总容量 20.00 GB 修改仓库容量上限

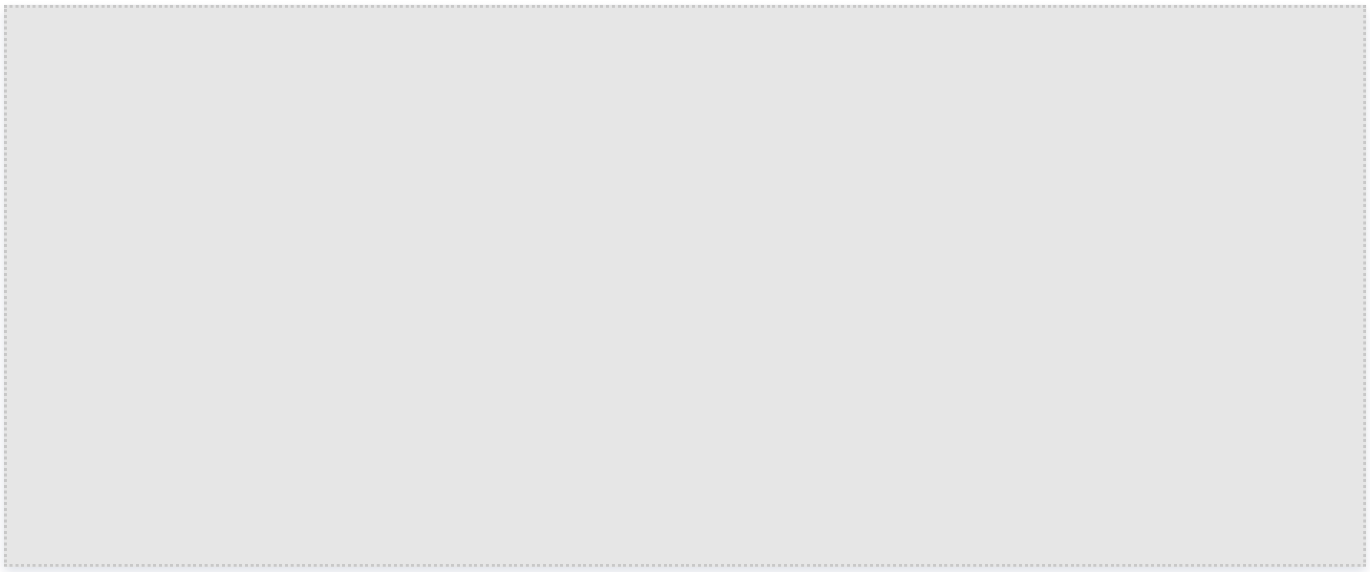
仓库清理
仓库清理使用 git gc 优化代码仓库存储空间，压缩存储库对象，减少存储库磁盘占用。

危险操作

归档代码仓库
归档代码仓库后，将无法继续访问和操作该仓库。若需解除归档，您可在 已归档 页进行操作。 归档

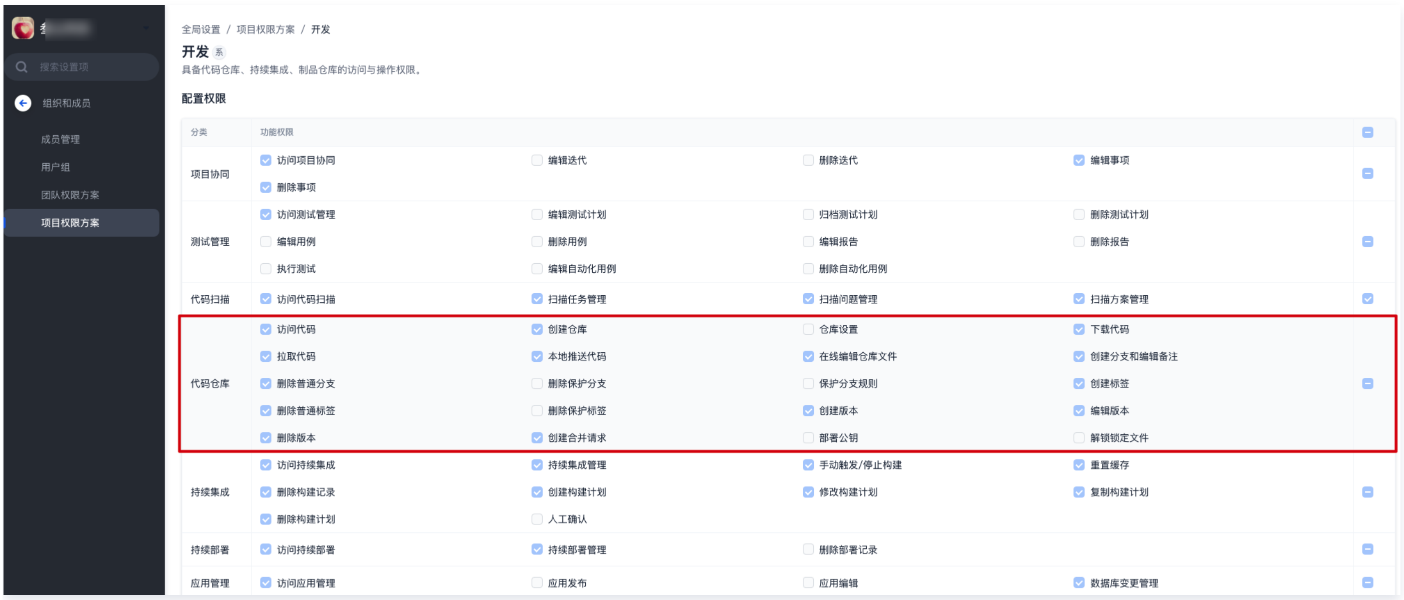
仓库图标
F
更改图标

- Team Deployment Public Key and Repository Standards
  Members in the group with this permission point can access the **feature settings** for **Team Deployment Public Key** and **Repository Standards**.



Project Permission Scheme

The project permission group is used to control various features for project members collaborating in the code repository, including protection branches, Tag and version creation, and other permission points.



组织和成员
成员管理
用户组
团队权限方案
项目权限方案

全局设置 / 项目权限方案 / 开发

开发 系

具备代码仓库、持续集成、制品仓库的访问与操作权限。

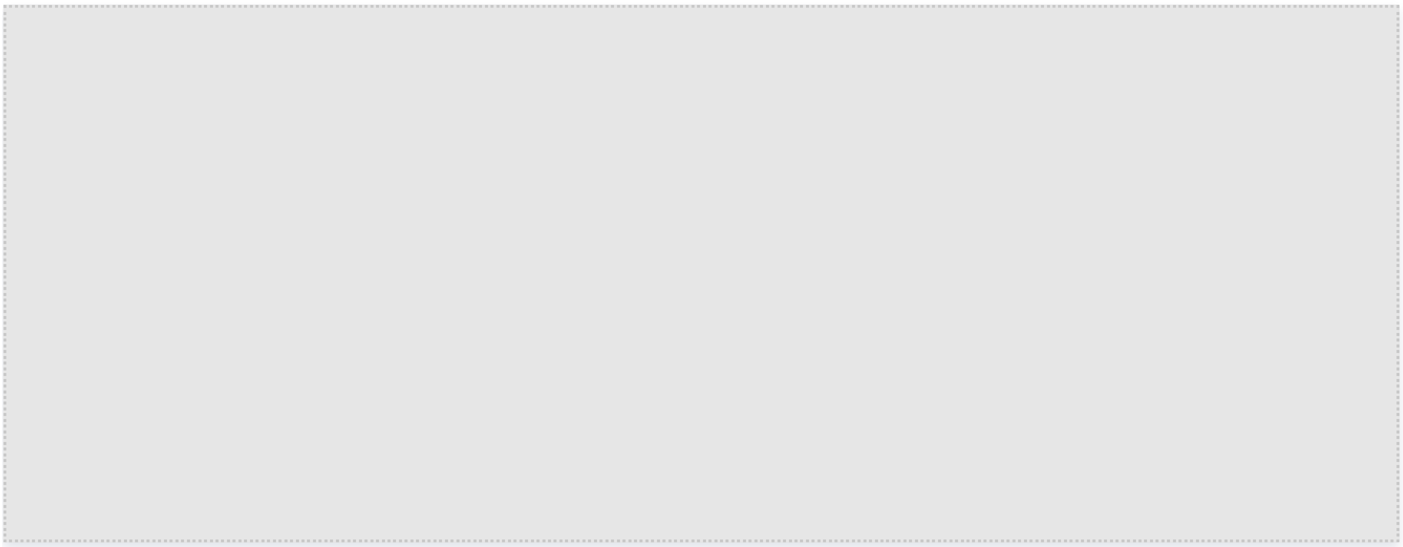
配置权限

分类	功能权限				
项目协同	✓ 访问项目协同	☐ 编辑迭代	☐ 删除迭代	✓ 编辑事项	−
	✓ 删除事项				
测试管理	✓ 访问测试管理	☐ 编辑测试计划	☐ 归档测试计划	☐ 删除测试计划	−
	☐ 编辑用例	☐ 删除用例	☐ 编辑报告	☐ 删除报告	
	☐ 执行测试	☐ 编辑自动化用例	☐ 删除自动化用例		
代码扫描	✓ 访问代码扫描	✓ 扫描任务管理	✓ 扫描问题管理	✓ 扫描方案管理	✓
代码仓库	✓ 访问代码	✓ 创建仓库	☐ 仓库设置	✓ 下载代码	−
	✓ 拉取代码	✓ 本地推送代码	✓ 在线编辑仓库文件	✓ 创建分支和编辑备注	
	✓ 删除普通分支	☐ 删除保护分支	☐ 保护分支规则	✓ 创建标签	
	✓ 删除普通标签	☐ 删除保护标签	✓ 创建版本	✓ 编辑版本	
	✓ 删除版本	✓ 创建合并请求	☐ 部署公钥	☐ 解锁锁定文件	
持续集成	✓ 访问持续集成	✓ 持续集成管理	✓ 手动触发/停止构建	✓ 重置缓存	−
	✓ 删除构建记录	✓ 创建构建计划	✓ 修改构建计划	✓ 复制构建计划	
	✓ 删除构建计划	☐ 人工确认			
持续部署	✓ 访问持续部署	✓ 持续部署管理	☐ 删除部署记录		−
应用管理	✓ 访问应用管理	☐ 应用发布	☐ 应用编辑	✓ 数据库变更管理	−

After the team leader/administrator completes the project permission group configuration, they can go to the **Global Settings > Project Permission Scheme > Project Member Management** page to assign appropriate permission schemes to project members.

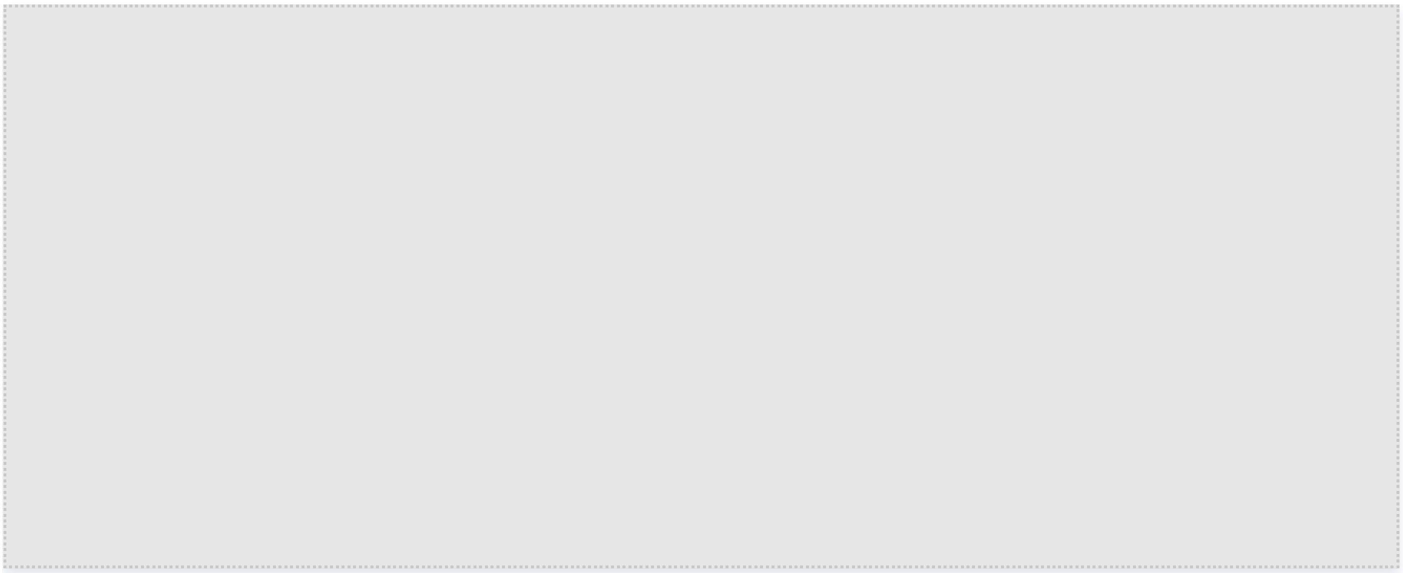
Repository Permission Scheme

In addition to using the project permission scheme, it also supports configuring repository permission schemes within a single code repository to meet the requirements for development process security between different code repositories within the same project. For detailed configuration instructions, please refer to Creating a Resource Permission Scheme Group.



Specifying Permission Scheme Type

After the creation of the code repository permission group, the repository administrator goes to the repository settings to adjust the "Permission Scheme".



Note:
A repository administrator is a user whose project permission scheme includes the "Repository Settings" permission point.

To distinguish the applied permission schemes, the repository administrator must select the appropriate permission scheme based on the scenario before adding members.

- **Using Project Permissions & Repository Scheme Simultaneously**
  Balance the project permission scheme and the repository permission scheme for repository members. For example, if Member A's project permission scheme includes the "Delete Normal Branch" permission but the code repository permissions do not, Member A will have the "Delete Normal Branch" permission.
- **Using Repository Permission Scheme Only**
  Discard the project permission scheme, and follow the repository permission scheme for the repository members. For example, if Member A’ s project permission scheme includes the "Delete Normal Branch" permission but the code repository permissions do not, Member A will not have the "Delete Normal Branch" permission.

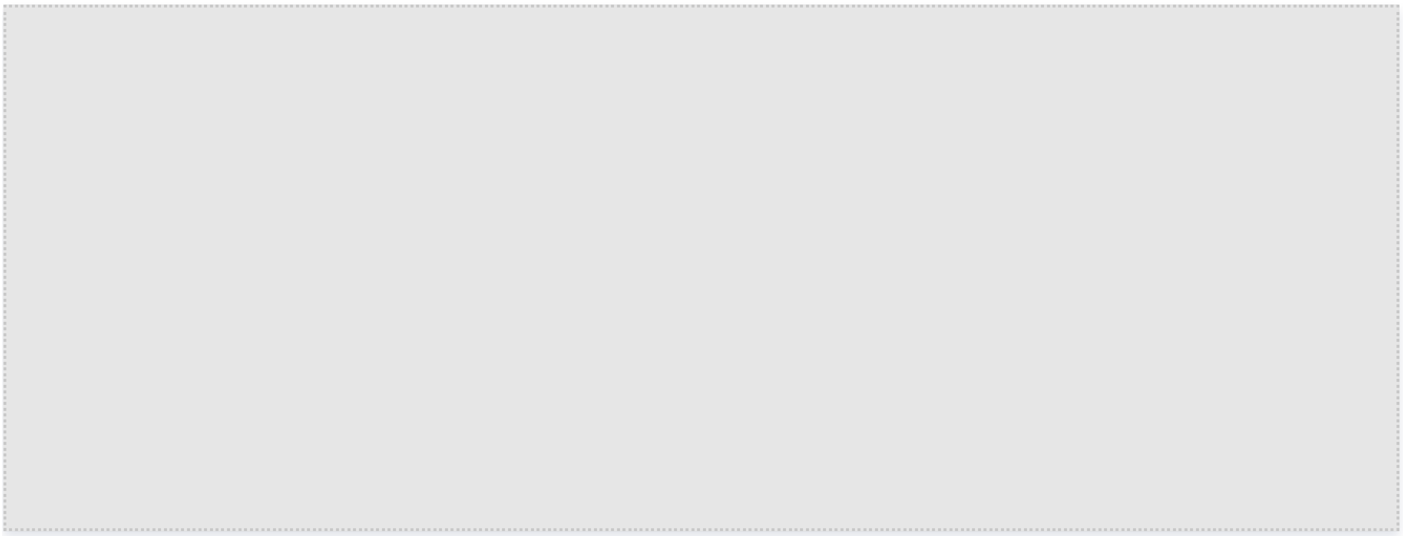
  Note:

> If you choose "Using Repository Permission Scheme Only", only the team leader/administrator or other members with code permissions in the team are allowed to access the repository page for operations.

- **Using Project Permission Scheme Only**
  Discard the repository permission scheme and follow the project permission scheme for the repository members. For example, if Member A’ s project permission scheme includes the "Delete Normal Branch" permission but the code repository permissions do not, Member A will have the "Delete Normal Branch" permission.

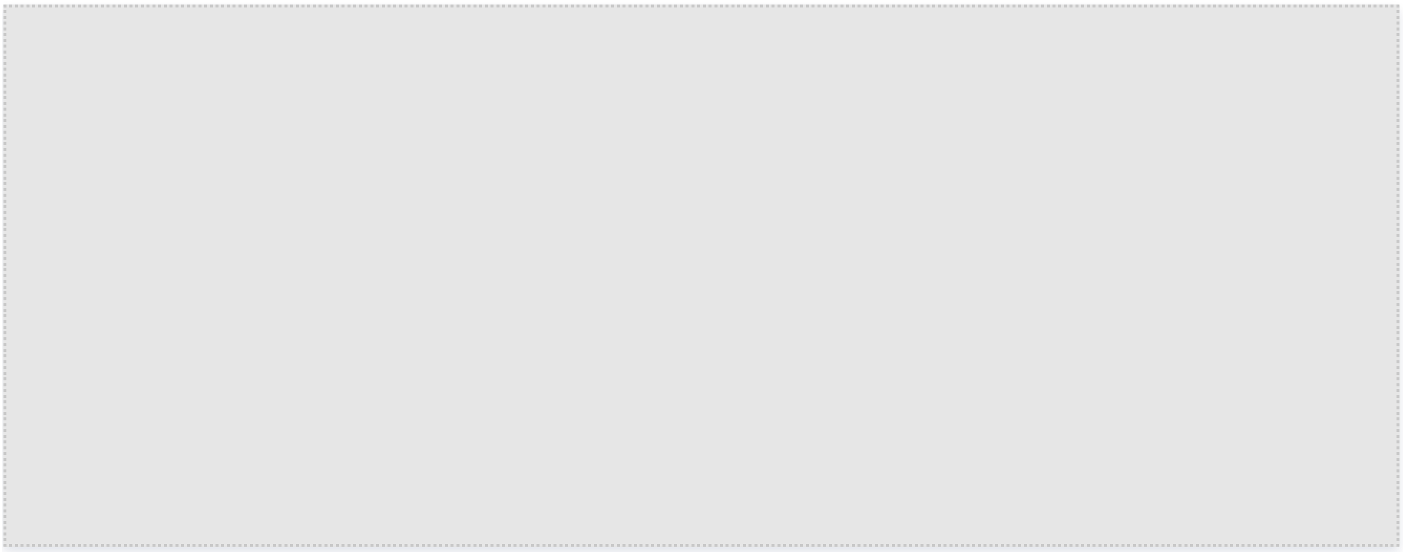
Assign Code Repository Permissions

After adding members, assign them to a code repository permission group.



View Code Repository Permission Group

Switch to the **Repository Permission Group** tab to view the current permission points included in the permission group.



If the assigned permission group lacks the necessary permission points, please contact the team administrator for Resource Permission Scheme Configuration to add the relevant permission points.

Git repository
Creating a Repository

Last updated：2024-09-05 16:11:39

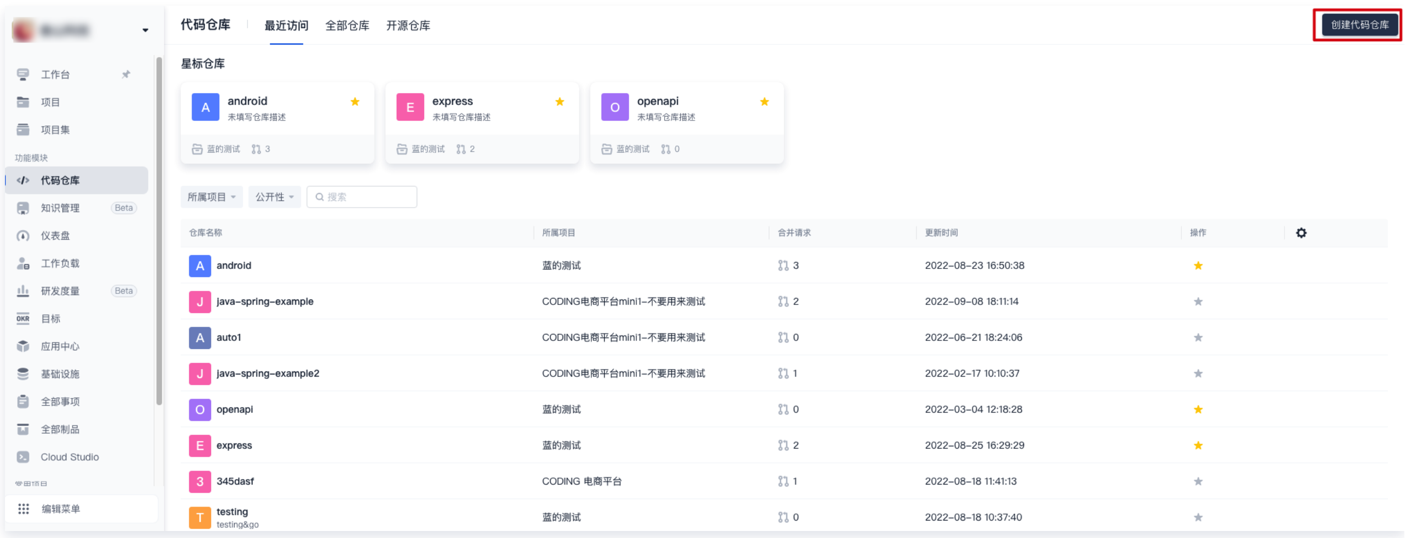
This document describes how to create a Git code repository.

Open Project

1. Log in to the CODING console and click the team domain name to enter the CODING page.
2. Click **Projects** on the left side of the team homepage to enter the project list page, then select the target project.
3. In the menu on the left, select **Code Repositories** to enter the code repository homepage.

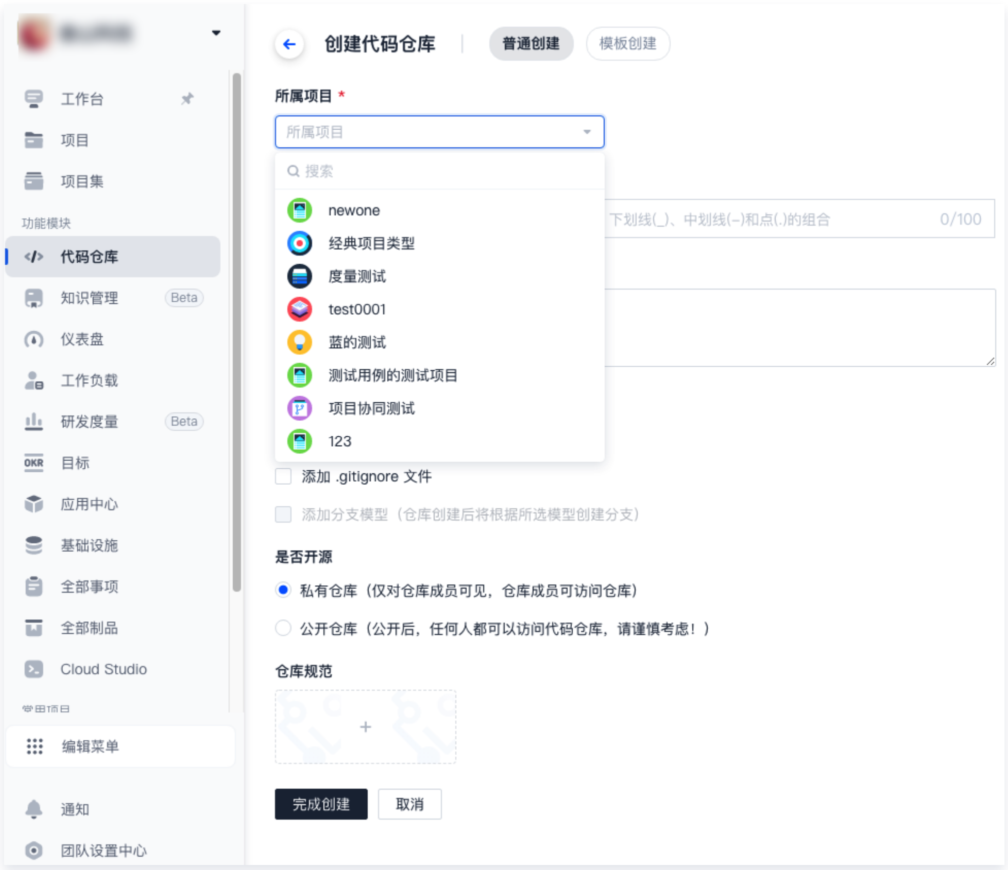
Manual Repository Creation

1. Log in to the team, then click **Code Repositories** on the left side of the homepage and click **Create Code Repository**.



仓库名称	所属项目	合并请求	更新时间	操作
android	蓝的测试	3	2022-08-23 16:50:38	★
java-spring-example	CODING电商平台mini1-不要用来测试	2	2022-09-08 18:11:14	☆
auto1	CODING电商平台mini1-不要用来测试	0	2022-06-21 18:24:06	☆
java-spring-example2	CODING电商平台mini1-不要用来测试	1	2022-02-17 10:10:37	☆
openapi	蓝的测试	0	2022-03-04 12:18:28	★
express	蓝的测试	2	2022-08-25 16:29:29	★
345dasf	CODING 电商平台	1	2022-08-18 11:41:13	☆
testing testing&go	蓝的测试	0	2022-08-18 10:37:40	☆

2. Code repositories cannot exist independently and must belong to a project. One project can correspond to multiple code repositories.



创建代码仓库 普通创建 模板创建

所属项目 *

所属项目

搜索

newone
经典项目类型
度量测试
test0001
蓝的测试
测试用例的测试项目
项目协同测试
123

下划线(_)、中划线(-)和点(.)的组合 0/100

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

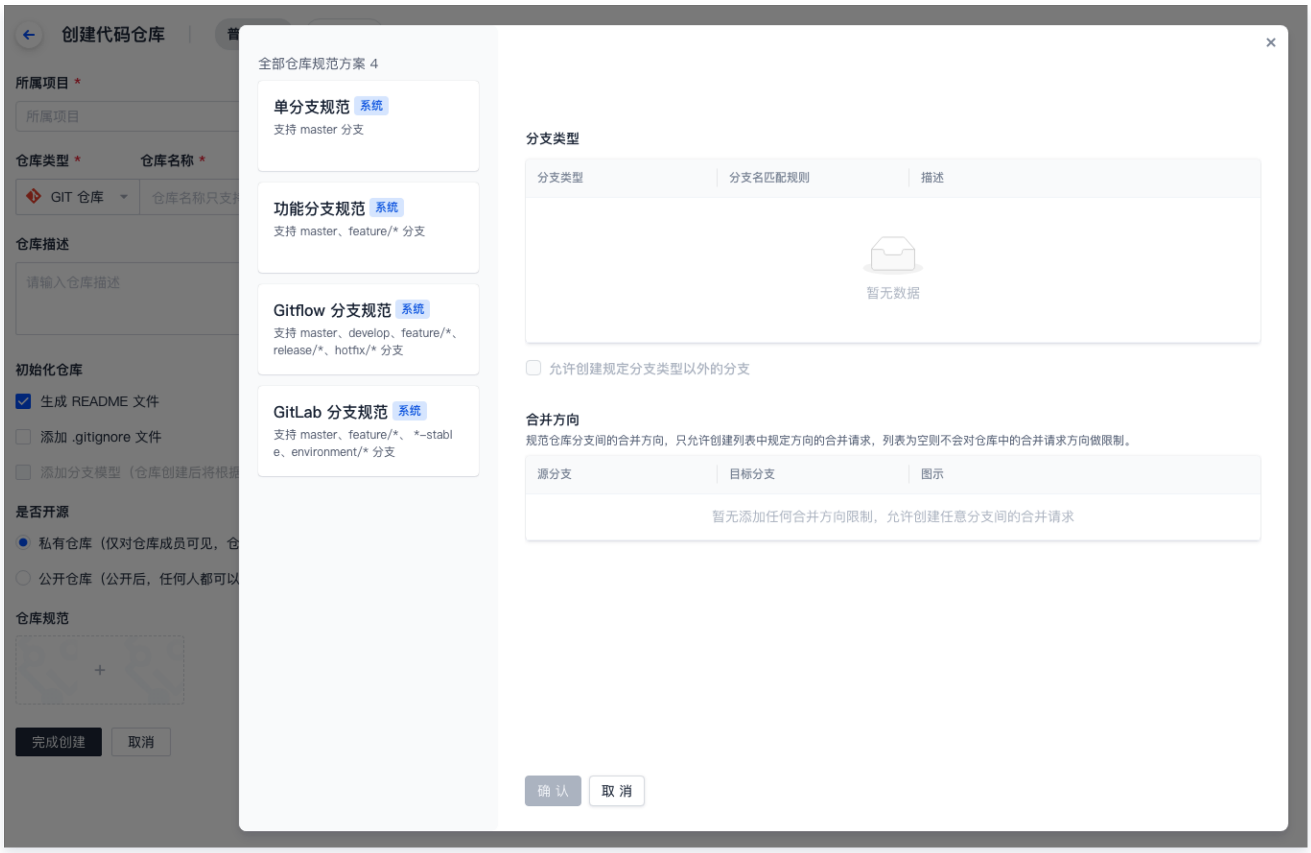
● 私有仓库（仅对仓库成员可见，仓库成员可访问仓库）

○ 公开仓库（公开后，任何人都可以访问代码仓库，请谨慎考虑！）

仓库规范

完成创建 取消

3. Select Git as the repository type and enter a valid repository name.
4. When creating a code repository, you can also choose whether to enable repository standards to quickly establish the team development workflow.


创建代码仓库

所属项目 *

仓库类型 * 仓库名称 *

GIT 仓库

仓库描述

请输入仓库描述

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

是否开源

● 私有仓库（仅对仓库成员可见，仓

○ 公开仓库（公开后，任何人都可以

仓库规范

完成创建 取消

全部仓库规范方案 4

单分支规范 系统
支持 master 分支

功能分支规范 系统
支持 master、feature/* 分支

Gitflow 分支规范 系统
支持 master、develop、feature/*、release/*、hotfix/* 分支

GitLab 分支规范 系统
支持 master、feature/*、*-stable、environment/* 分支

分支类型

分支类型	分支名匹配规则	描述
暂无数据		

允许创建规定分支类型以外的分支

合并方向

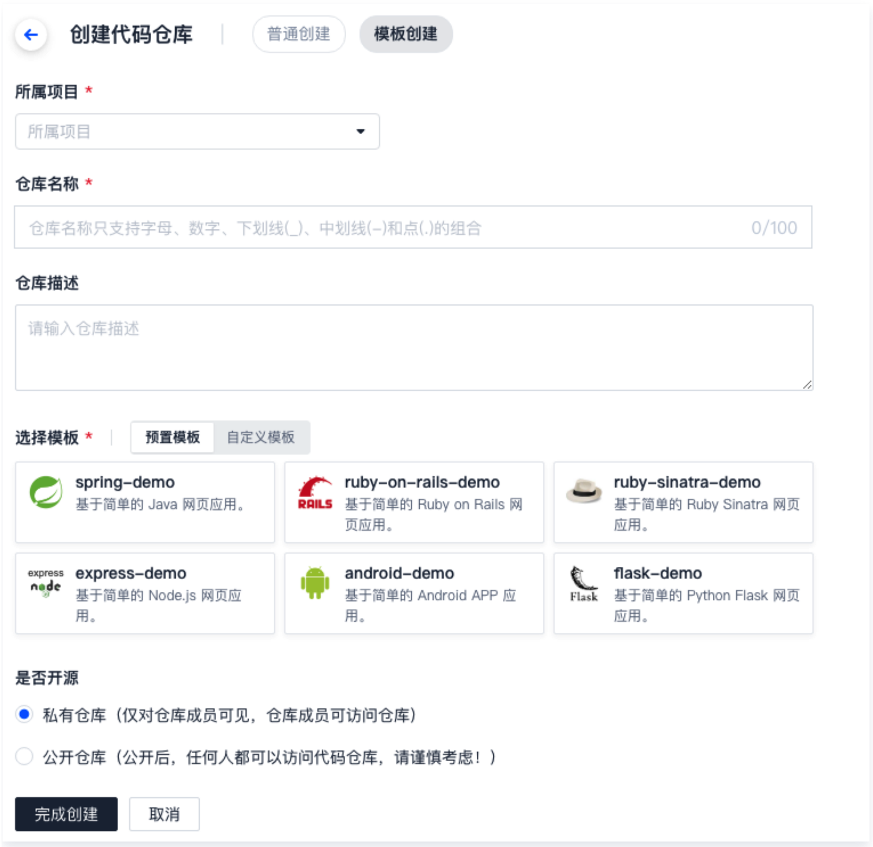
规范仓库分支间的合并方向，只允许创建列表中规定方向的合并请求，列表为空则不会对仓库中的合并请求方向做限制。

源分支	目标分支	图示
暂无添加任何合并方向限制，允许创建任意分支间的合并请求		

确 认 取 消

Template Creation

CODING provides several preset code repository modules with development frameworks to help you quickly experience how code repositories work with continuous integration or artifacts.



创建代码仓库 | 普通创建 | 模板创建

所属项目 *

所属项目

仓库名称 *

仓库名称只支持字母、数字、下划线(_)、中划线(-)和点(.)的组合 0/100

仓库描述

请输入仓库描述

选择模板 * | 预置模板 | 自定义模板

spring-demo 基于简单的 Java 网页应用。	ruby-on-rails-demo 基于简单的 Ruby on Rails 网页应用。	ruby-sinatra-demo 基于简单的 Ruby Sinatra 网页应用。
express-demo 基于简单的 Node.js 网页应用。	android-demo 基于简单的 Android APP 应用。	flask-demo 基于简单的 Python Flask 网页应用。

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（公开后，任何人都可以访问代码仓库，请谨慎考虑！）

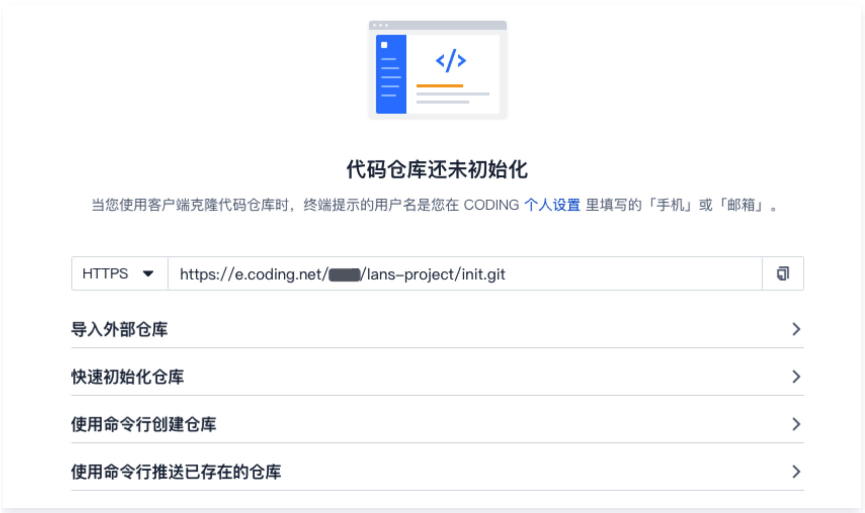
完成创建 取消

Import an External Repository

You can quickly migrate open-source Git repositories or repositories from other platforms to the CODING DevOps platform. For details, see Import or Associate External Repository.

Git Repository Initialization

After creating a repository, you can initialize it in four ways.



代码仓库还未初始化

当您使用客户端克隆代码仓库时，终端提示的用户名是您在 CODING 个人设置 里填写的「手机」或「邮箱」。

HTTPS https://e.coding.net/[redacted]/lans-project/init.git

导入外部仓库 >

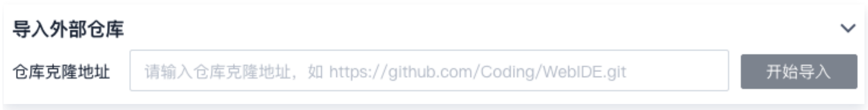
快速初始化仓库 >

使用命令行创建仓库 >

使用命令行推送已存在的仓库 >

Import an External Repository

Enter the Git repository address for import to complete the initialization.

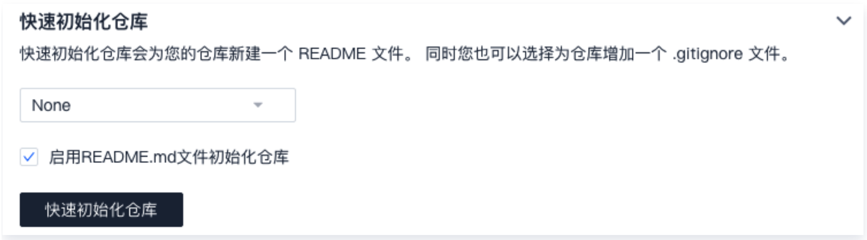


导入外部仓库

仓库克隆地址 请输入仓库克隆地址，如 https://github.com/Coding/WebIDE.git 开始导入

Quick Repository Initialization

This method completes the initialization by generating a README.md file.



快速初始化仓库

快速初始化仓库会为您的仓库新建一个 README 文件。 同时您也可以选择为仓库增加一个 .gitignore 文件。

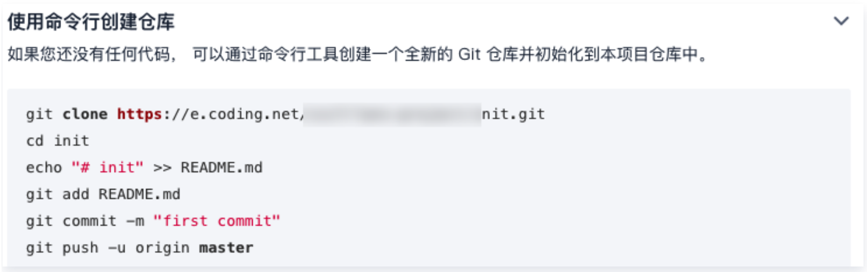
None

[✓] 启用README.md文件初始化仓库

快速初始化仓库

Creating a Repository Using Command Line

This method essentially generates a README.md file in the local repository and then uploads it to the remote repository for initialization.



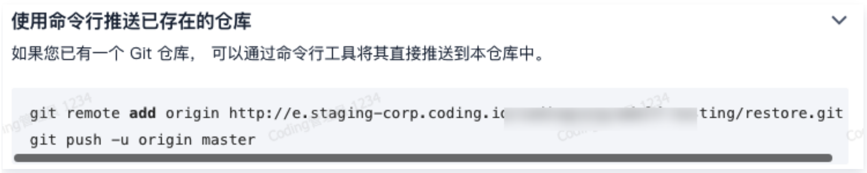
使用命令行创建仓库

如果您还没有任何代码， 可以通过命令行工具创建一个全新的 Git 仓库并初始化到本项目仓库中。

```
git clone https://e.coding.net/[redacted]nit.git
cd init
echo "# init" >> README.md
git add README.md
git commit -m "first commit"
git push -u origin master
```

Pushing Local Repository Using Command Line

You can also directly upload a locally initialized repository to the remote repository.



使用命令行推送已存在的仓库

如果您已有一个 Git 仓库， 可以通过命令行工具将其直接推送到本仓库中。

```
git remote add origin http://e.staging-corp.coding.i[redacted]ting/restore.git
git push -u origin master
```

Import or Associate External Repository

Last updated：2024-09-05 16:12:32

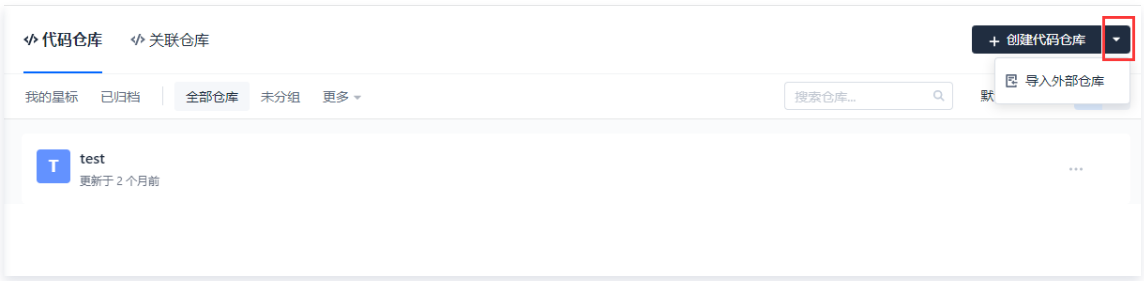
Open Project

1. Log in to CODING Console, click the team domain name to enter the CODING page.
2. Click **Item** on the left side of the team homepage to enter the project list page and select the target project.
3. In the menu on the left, select **Code Repository** to enter the code repository homepage.

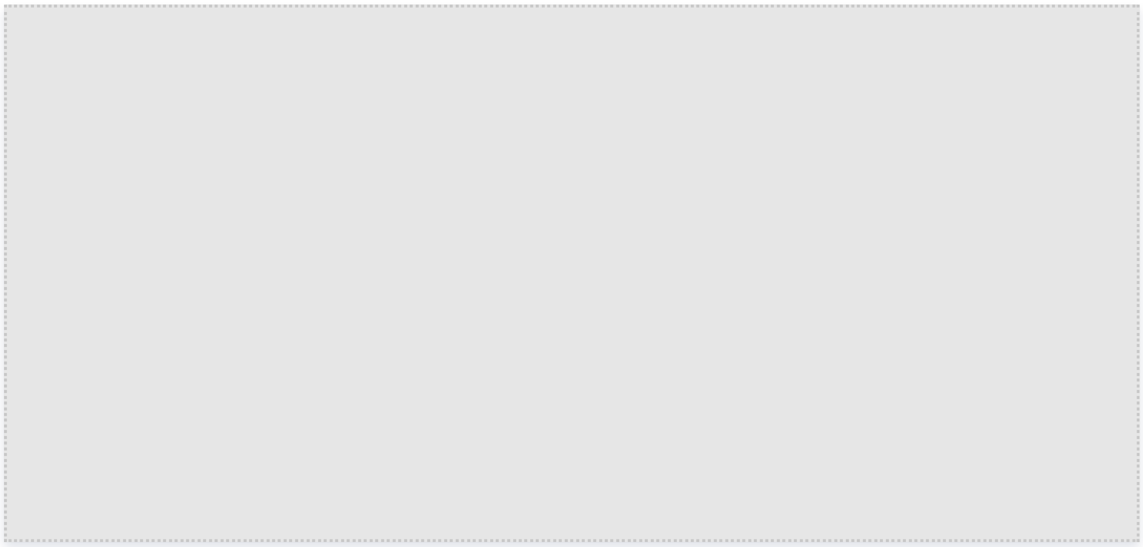
Import Repository

CODING provides not only a one-click import feature for external repositories but also supports scheduling syncs for external open-source repositories.

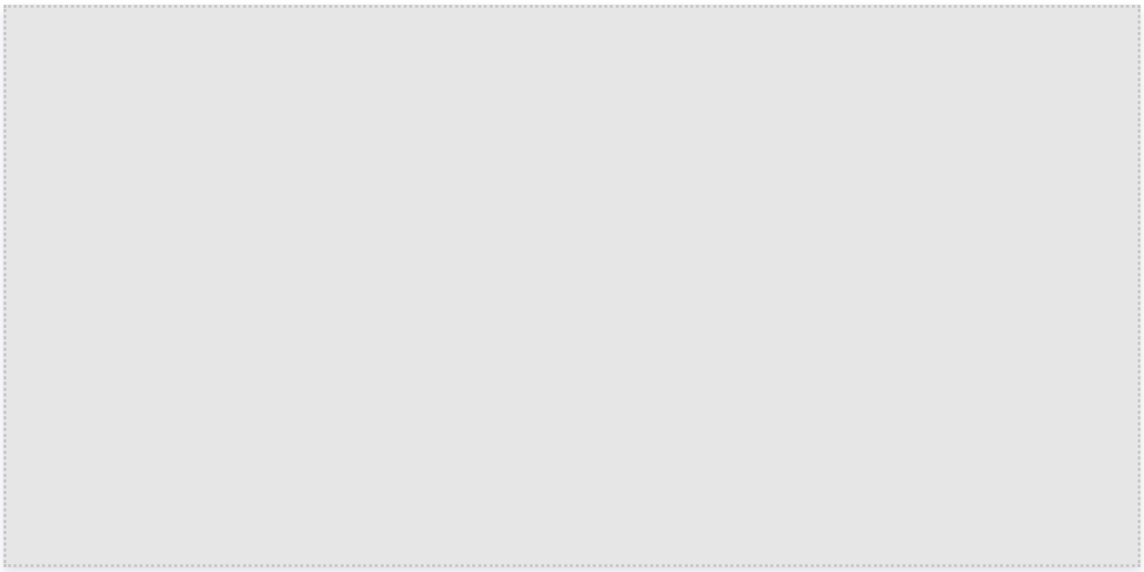
1. Go to any project, click **Code Repository** on the left sidebar, and click **Import External Repository** in the top right corner of the page.



2. Follow the prompts to select the type of repository source.

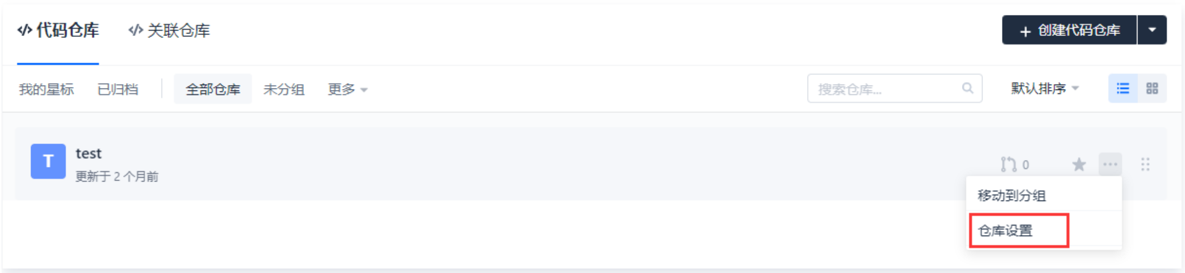


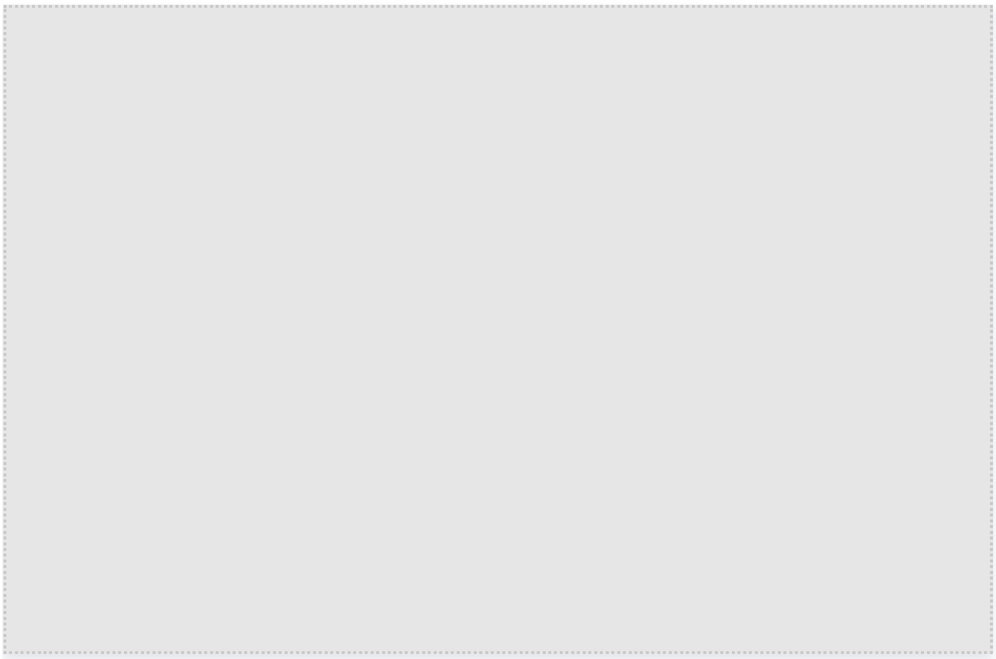
3. You can choose to import private or open-source repositories via URL. If the source repository is private, you will need to enter the username and password as prompted by the page.



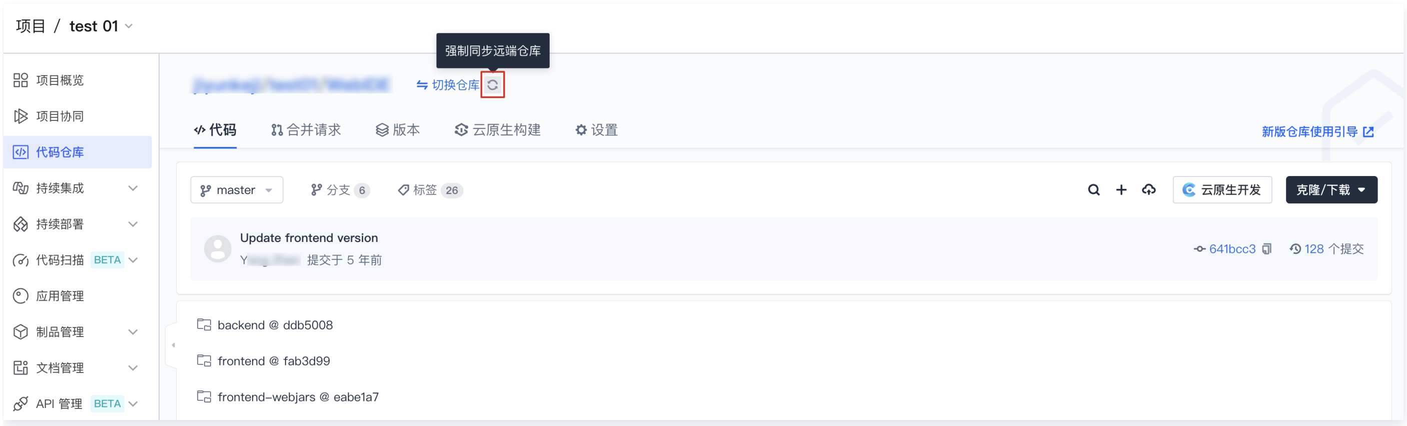
Synchronize Repository

The sync feature is only available for open-source repositories. This means that it will keep consistent with the source repository and overwrite any changes made in the CODING repository. Click **Repository Settings** to change the sync frequency or disable the auto-sync feature.





On the homepage of the imported repository, you can also manually force sync the code repository by clicking the import button.

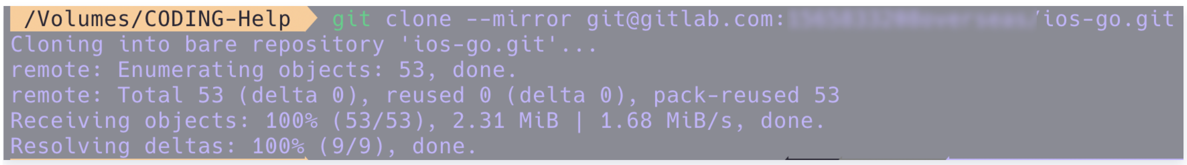


Import Mirror Repository

For security reasons, you can also choose to import an external private (non-open source) Git mirror repository. Refer to the following steps.

Step 1: Pull the private repository to your local machine

```
git clone --mirror [private repository URL]
```

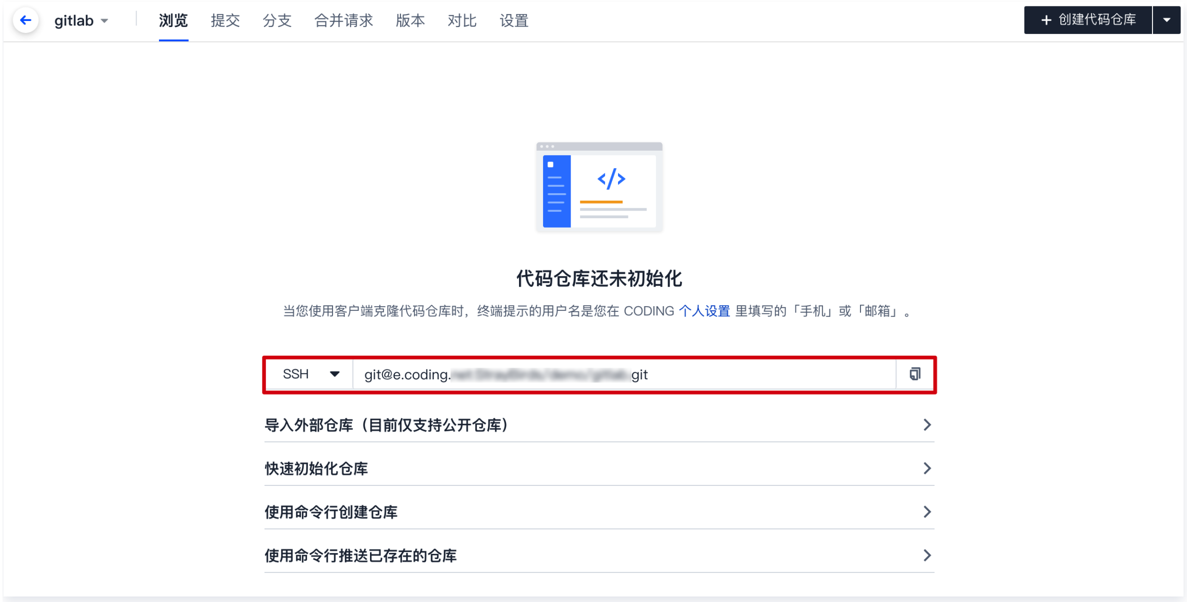


```
/Volumes/CODING-Help  git clone --mirror git@gitlab.com:[redacted]/ios-go.git
Cloning into bare repository 'ios-go.git'...
remote: Enumerating objects: 53, done.
remote: Total 53 (delta 0), reused 0 (delta 0), pack-reused 53
Receiving objects: 100% (53/53), 2.31 MiB | 1.68 MiB/s, done.
Resolving deltas: 100% (9/9), done.
```

Note:
The local mirror repository folder typically has a `.git` suffix.

Step 2: Obtain the target CODING code repository address

Enter the CODING project on the web to get the address of the target repository.


gitlab
浏览 提交 分支 合并请求 版本 对比 设置
创建代码仓库
导入外部仓库（目前仅支持公开仓库）
快速初始化仓库
使用命令行创建仓库
使用命令行推送已存在的仓库

代码仓库还未初始化

当您使用客户端克隆代码仓库时，终端提示的用户名是您在 CODING 个人设置 里填写的「手机」或「邮箱」。

SSH git@e.coding.[redacted].git

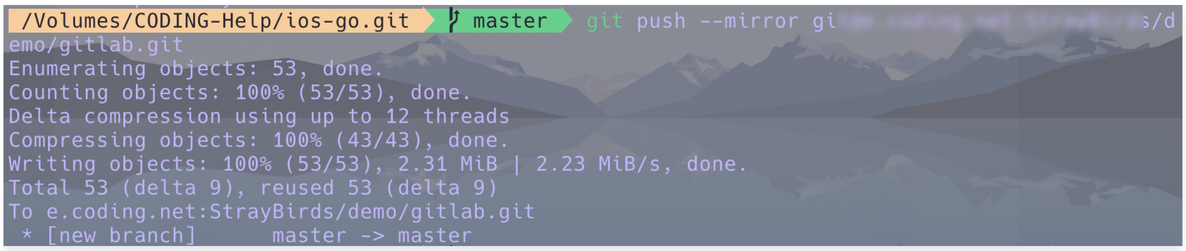
Step 3: Push to the target CODING repository

Enter the terminal command to access the private repository.

```
cd private repository
```

Use the push command to push the private repository to the target CODING repository.

```
git push --mirror target repository address
```

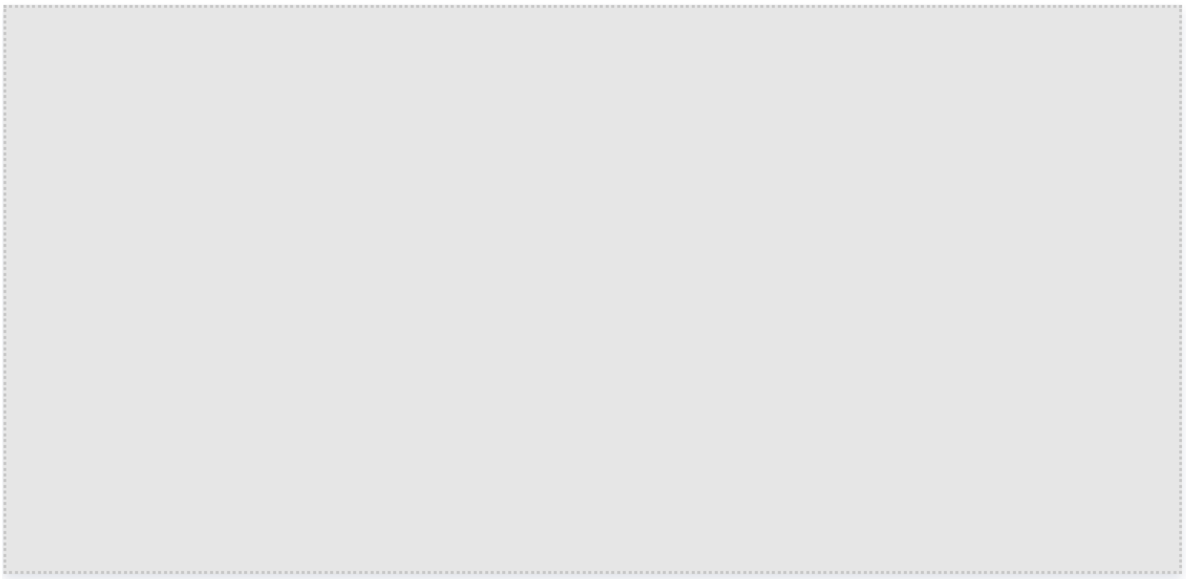


```
/Volumes/CODING-Help/ios-go.git  master  git push --mirror gi[redacted]s/d
emo/gitlab.git
Enumerating objects: 53, done.
Counting objects: 100% (53/53), done.
Delta compression using up to 12 threads
Compressing objects: 100% (43/43), done.
Writing objects: 100% (53/53), 2.31 MiB | 2.23 MiB/s, done.
Total 53 (delta 9), reused 53 (delta 9)
To e.coding.net:StrayBirds/demo/gitlab.git
 * [new branch]      master -> master
```

If an `refs/pull` error occurs, you can use the following command to avoid the error:

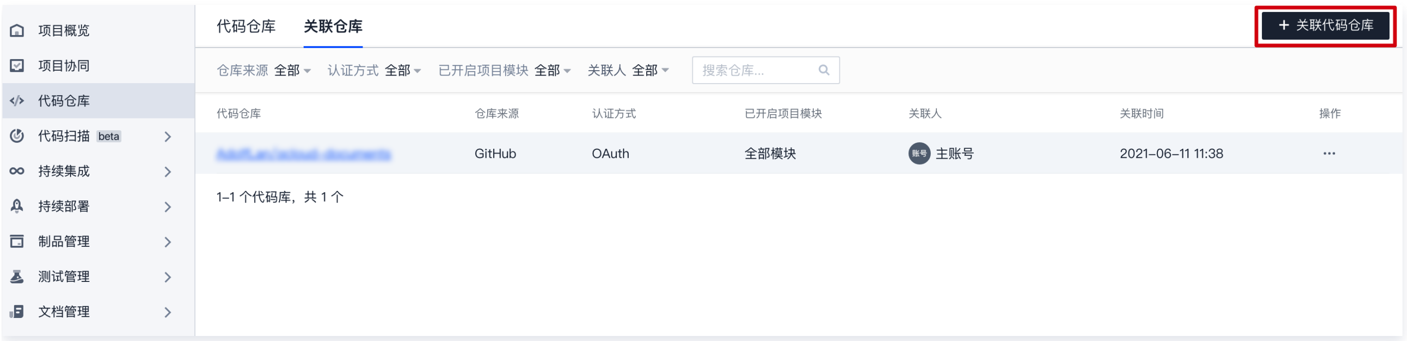
```
git push URL "+refs/heads/*:refs/heads/*" "+refs/tags/*:refs/tags/*"
```

After the push is successful, you can see that the private repository has been uploaded to the target repository.



Associate a Code Repository

The Link Repository feature essentially "temporary stores" the credentials for accessing external repositories on CODING. When you use continuous integration or continuous deployment, it allows direct calls to third-party repositories as the source code, thereby eliminating the cumbersome process of frequent migration.

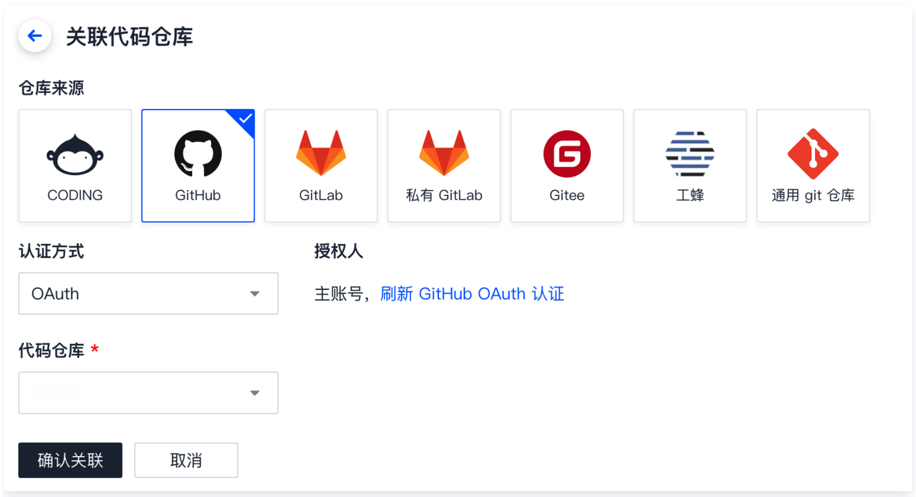


仓库来源 全部 认证方式 全部 已开启项目模块 全部 关联人 全部 搜索仓库...

代码仓库	仓库来源	认证方式	已开启项目模块	关联人	关联时间	操作
[redacted]	GitHub	OAuth	全部模块	主账号	2021-06-11 11:38	···

1-1 个代码库，共 1 个

The supported types of linked repositories include GitHub, GitLab, private GitLab, Gitee, Gongfeng, Universal Git repositories, and other projects' CODING repositories. The first five types of repositories support OAuth authentication method, while Universal Git repositories support account password authentication. The code of the linked repositories will not be stored in the CODING code repository.


关联代码仓库

仓库来源

认证方式: OAuth

授权人: 主账号，刷新 GitHub OAuth 认证

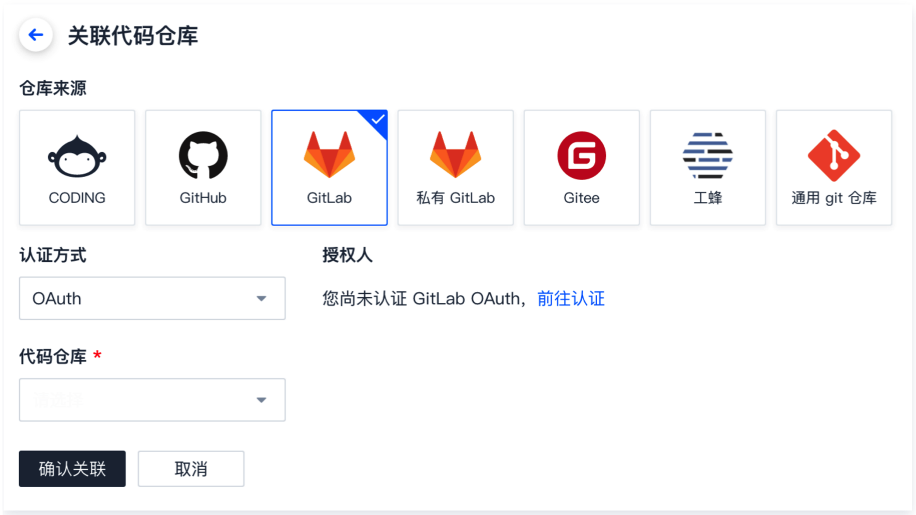
代码仓库 *

Associate a private GitLab repository

To associate a private GitLab repository, you must create an application in GitLab and then the team admin must bind the private GitLab service. For details, see Bind Private GitLab.

Associate a GitLab SaaS repository

To associate a GitLab SaaS repository, select **Associate Code Repository** page and choose **GitLab** as the code source, then click **Verify Now** to go to the GitLab Authorization page and click **Authorize** to complete authorization. After successful authorization, select the code repository to complete the operation.


关联代码仓库

仓库来源

认证方式: OAuth

授权人: 您尚未认证 GitLab OAuth，前往认证

代码仓库 *

Associate a GitHub repository

On the **Associate Code Repository** page, select **GitHub** as the code source, then click **Verify Now** to use OAuth authorization and go to GitHub for authorization. If authorization fails, this may be because you did not enter your username in GitHub. In this case, go to **Settings** > **Profile** > **Name** and enter your username.

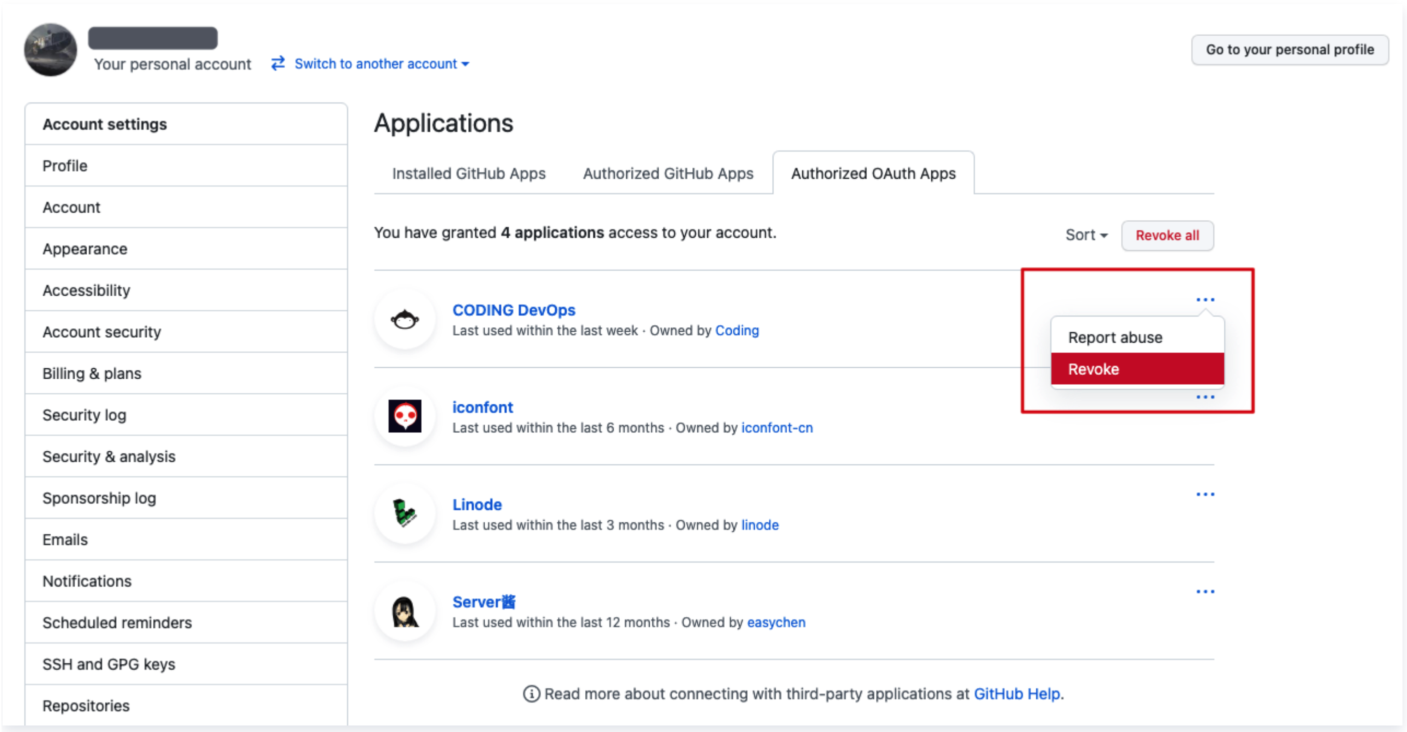
Associate repositories from other projects

To associate code repositories under your name from other CODING projects, go to the **Associate Code Repository** page, select CODING as the repository source, check the target repository to associate it. After association, it can be used as a code source for continuous integration or continuous deployment.

If you want to import the repository directly into the project for modification, go to the target repository' s **Access Settings** and set it to open source status, then refer to Import Open Source Repository to import the target code.

Modify Associated Repositories

This chapter takes GitHub as an example:
After logging in to GitHub, click on the avatar in the upper right corner **Settings** > **Applications** > **Authorized OAuth Apps** to disassociate.


[redacted]
Your personal account
Switch to another account
Go to your personal profile

Account settings
- Profile
- Account
- Appearance
- Accessibility
- Account security
- Billing & plans
- Security log
- Security & analysis
- Sponsorship log
- Emails
- Notifications
- Scheduled reminders
- SSH and GPG keys
- Repositories

Applications

Installed GitHub Apps | Authorized GitHub Apps | Authorized OAuth Apps

You have granted **4 applications** access to your account.

Sort | Revoke all

CODING DevOps
Last used within the last week · Owned by Coding

Report abuse
Revoke

iconfont
Last used within the last 6 months · Owned by iconfont-cn

Linode
Last used within the last 3 months · Owned by linode

Server酱
Last used within the last 12 months · Owned by easychen

Read more about connecting with third-party applications at GitHub Help.

Switch to another GitHub account and re-authorize using OAuth to complete Associate GitHub Repository.

Manage Repository

Last updated：2024-09-05 16:12:49

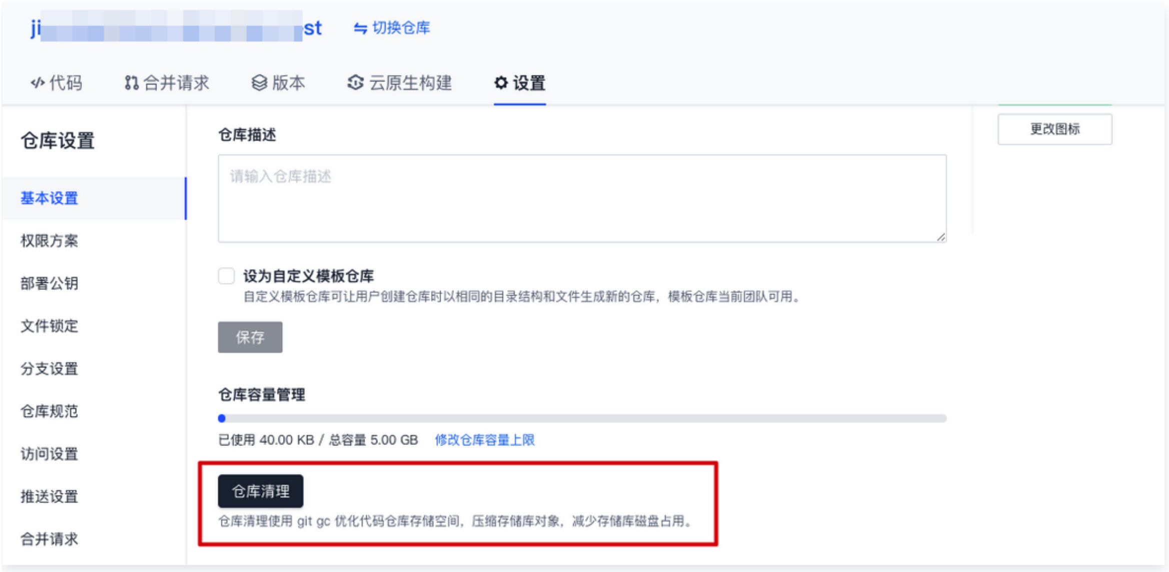
This document provides detailed instructions on how to perform some operations in the repository.

Open Project

1. Log in to CODING Console, click the team domain name to enter the CODING page.
2. Click **Item** on the left side of the team homepage to enter the project list page and select the target project.
3. In the menu on the left, select **Code Repository** to enter the code repository homepage.

Clean Repository

Remote repositories generate cached files during use. You can click **Repository Cleanup** to reduce the space occupied by cached files in the repository.



ji[redacted]st ⇋ 切换仓库

代码 合并请求 版本 云原生构建 设置

仓库设置

基本设置
权限方案
部署公钥
文件锁定
分支设置
仓库规范
访问设置
推送设置
合并请求

仓库描述

请输入仓库描述

更改图标

设为自定义模板仓库
自定义模板仓库可让用户创建仓库时以相同的目录结构和文件生成新的仓库，模板仓库当前团队可用。

保存

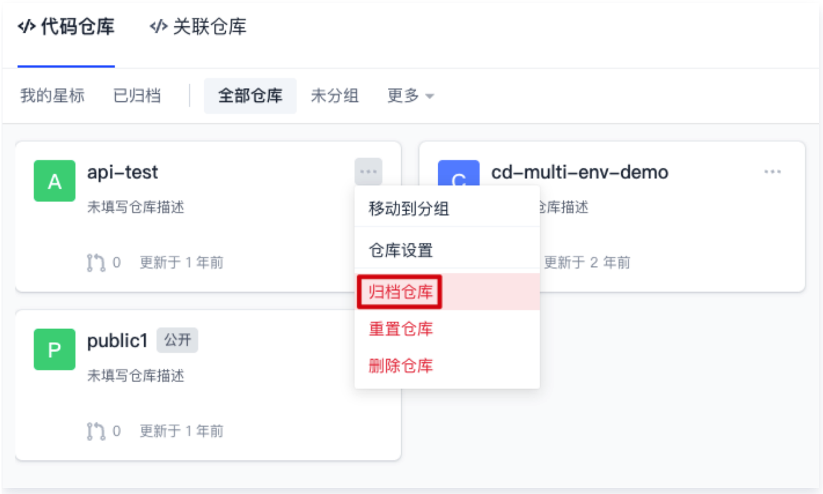
仓库容量管理

已使用 40.00 KB / 总容量 5.00 GB 修改仓库容量上限

仓库清理
仓库清理使用 git gc 优化代码仓库存储空间，压缩存储库对象，减少存储库磁盘占用。

Archive Repository

To archive a repository, click ··· in the code repository list and follow the prompts to confirm.



代码仓库 关联仓库

我的星标 已归档 全部仓库 未分组 更多

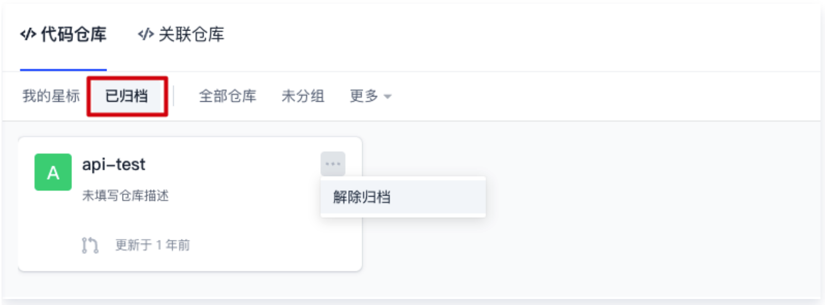
A api-test
未填写仓库描述
0 更新于 1 年前

C cd-multi-env-demo
仓库描述
更新于 2 年前

P public1 公开
未填写仓库描述
0 更新于 1 年前

移动到分组
仓库设置
归档仓库
重置仓库
删除仓库

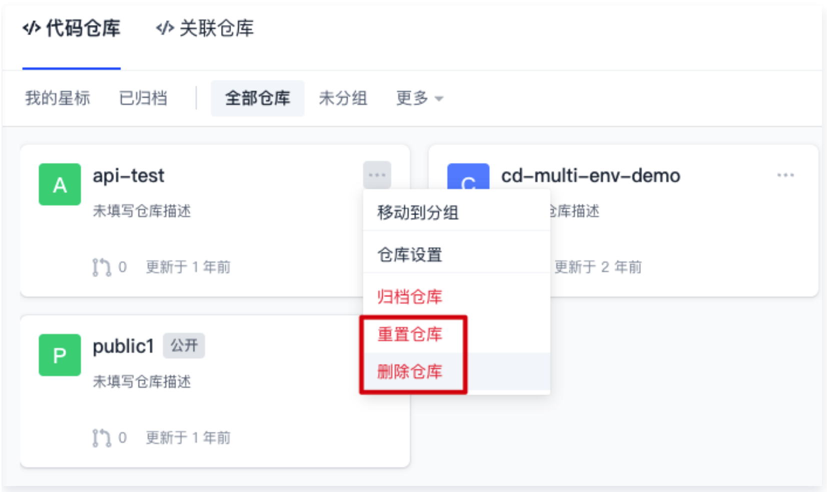
After archiving the repository, Git or web access requests will be automatically blocked. Users will not be able to access or operate the repository. Archived repositories can only be viewed under the **Archived** classification. To restore normal access to the repository, you need to unarchive it.



Delete and Reset

To reset or delete a repository, click ··· in the code repository list and follow the prompts to confirm.

Deleted repositories are moved to the recycle bin and retained for 30 days. Project admins can restore them from the recycle bin before the time limit. After the limit, all code, including code branches, merge requests, code versions, within the repository will be permanently deleted and cannot be recovered.

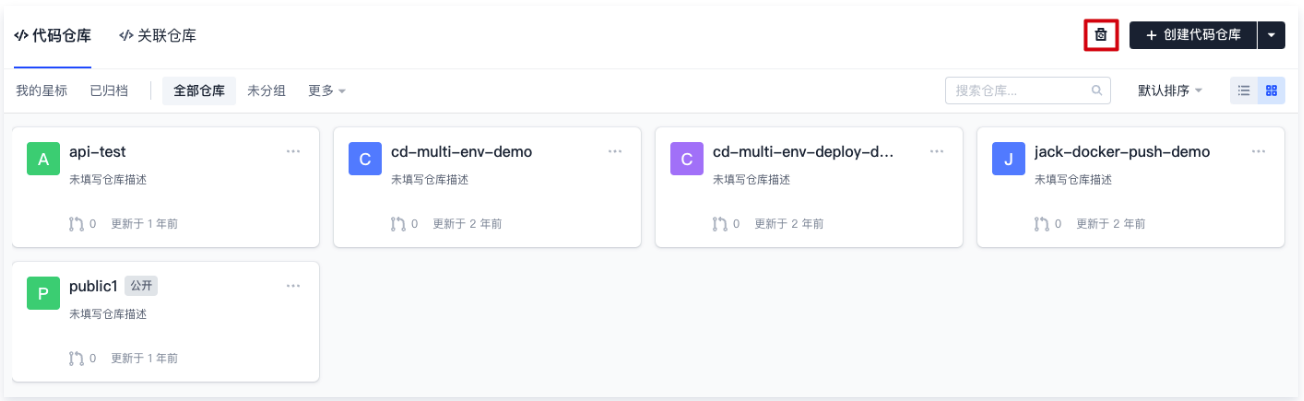


Click **Reset Repository** to reset all the code in the repository, including code branches, merge requests, and code versions. This action cannot be undone, and the repository will be reset to an empty state.

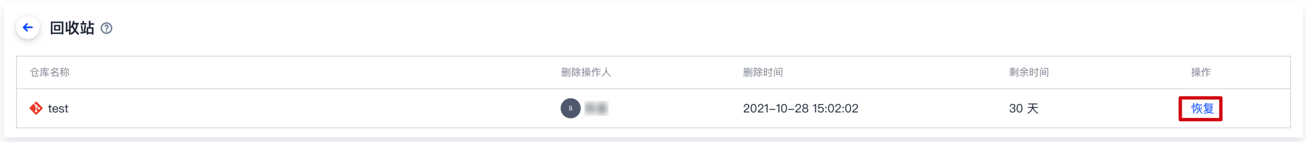
Restore Deleted Repository

Project admins can restore deleted repositories from the recycle bin within 30 days.

1. On the **Code Repository** page, click the Recycle Bin icon in the top-right corner.



2. In the recycle bin, select the repository to restore and click **Recover** to restore the repository.


回收站

仓库名称	删除操作人	删除时间	剩余时间	操作
test	[redacted]	2021-10-28 15:02:02	30 天	恢复

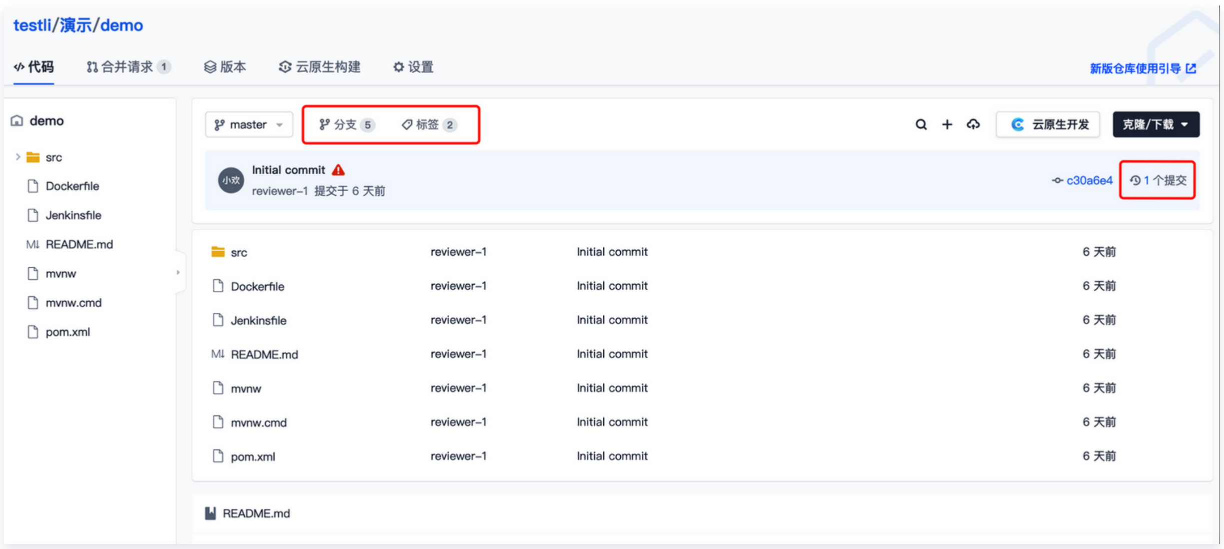
Code Comparison

Last updated: 2024-09-05 16:13:04

This article provides a detailed introduction on how to use the code comparison feature.

Operation step

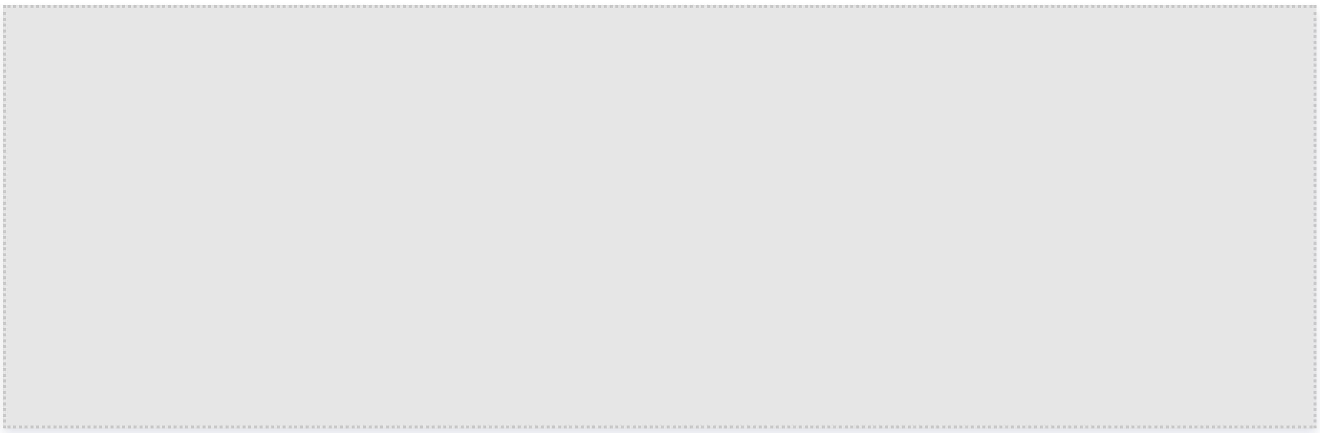
1. Log in to CODING Console, click the team domain name to enter the CODING page.
2. Click **Item** on the left side of the team homepage to enter the project list page and select the target project.
3. In the menu on the left, select **Code Repository** to enter the code repository homepage.
4. Enter the specified code repository, then click **Branch**, **Tag**, or Revision Version (hash value) to go to the corresponding page.



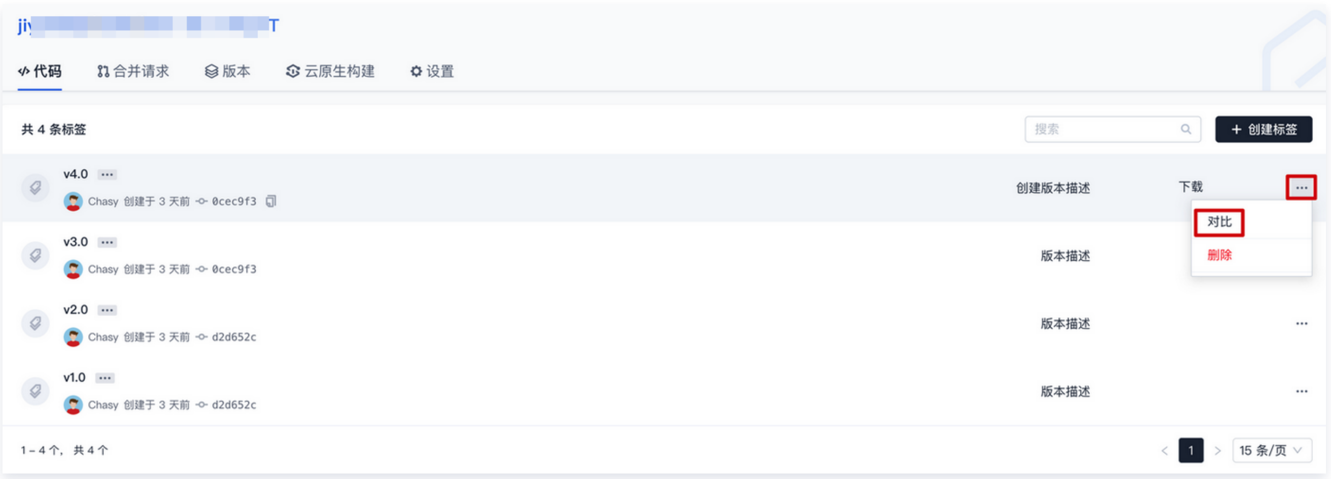
Initial commit
reviewer-1 提交于 6 天前

src	reviewer-1	Initial commit	6 天前
Dockerfile	reviewer-1	Initial commit	6 天前
Jenkinsfile	reviewer-1	Initial commit	6 天前
README.md	reviewer-1	Initial commit	6 天前
mvnw	reviewer-1	Initial commit	6 天前
mvnw.cmd	reviewer-1	Initial commit	6 天前
pom.xml	reviewer-1	Initial commit	6 天前

5. On the corresponding page, click ··· and select **Compare**.
   - For the branch list, select a specified branch, click ··· and select **Compare**.



   - For the tag list, select a specified tag, click ··· and select **Compare**.



共 4 条标签

v4.0 ··· Chasy 创建于 3 天前 0cec9f3	创建版本描述	下载	···
v3.0 ··· Chasy 创建于 3 天前 0cec9f3	版本描述		
v2.0 ··· Chasy 创建于 3 天前 d2d652c	版本描述		···
v1.0 ··· Chasy 创建于 3 天前 d2d652c	版本描述		···

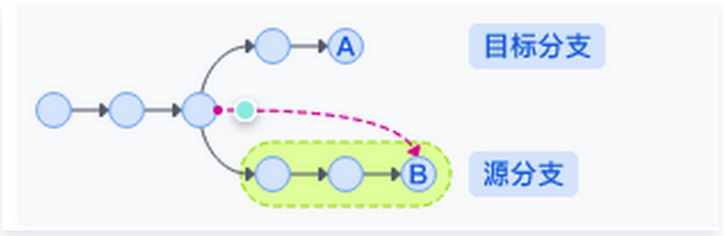
1 - 4 个， 共 4 个 &lt; 1 &gt; 15 条/页

- ○ If you have entered the specified revision version (hash value), click ··· in the upper-right corner and select **Compare**

6. In the code comparison page, select the corresponding branch, tag, or revision version for comparison.

Feature Overview

- To ensure web page responsiveness, you cannot use the **Expand All** feature if there are differences in more than 20 files.
- The code comparison feature essentially provides an online graphical representation of the `git diff` command. If the **Difference from Common Ancestor** option is selected, it will show the differences between the source branch and the common ancestor.



- If the **Difference from Common Ancestor** option is not selected, it will directly show the comparison between the source branch and the target branch.

SVN Repository
Create an SVN repository

Last updated：2024-09-05 16:13:23

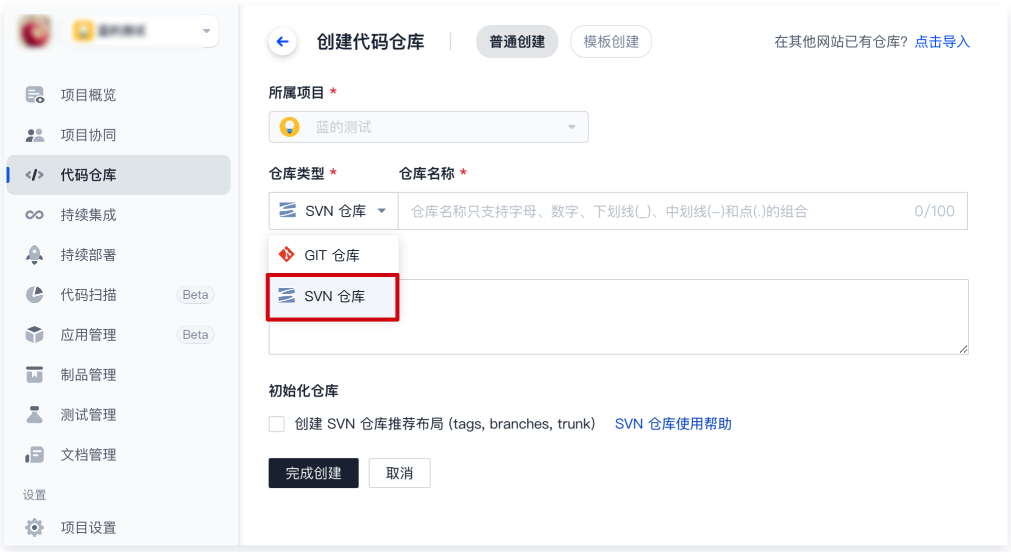
This document describes how to create an SVN repository.

Open Project

1. Log in to CODING Console, click the team domain name to enter the CODING page.
2. Click **Item** on the left side of the team homepage to enter the project list page and select the target project.
3. In the menu on the left, select **Code Repository** to enter the code repository homepage.

Operation step

1. After entering a project, click on the left navigation bar **Code Repository** to access the Code Repository Management page.
2. In the upper-right corner of the page, click **Create Code Repository**, and then select **SVN Repository** as the repository type.



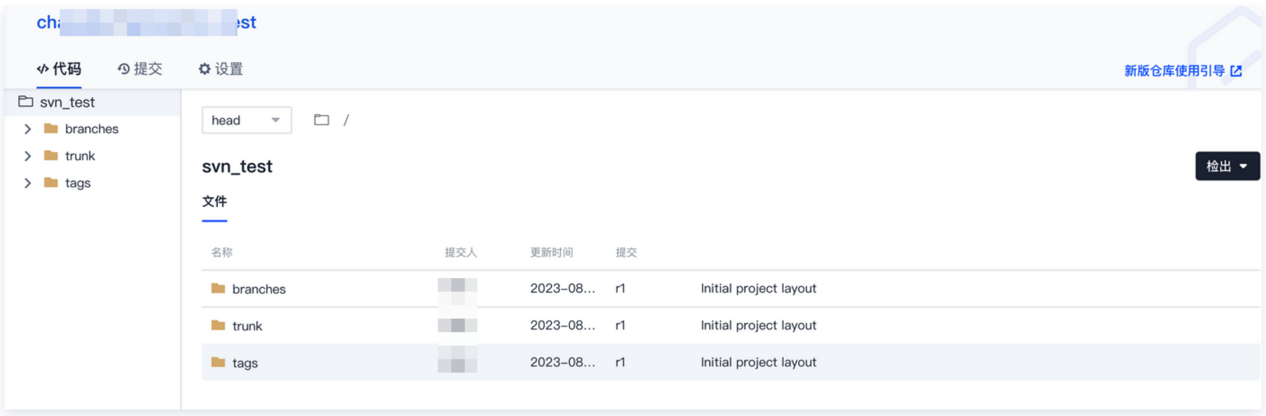
所属项目 * 蓝的测试

仓库类型 * SVN 仓库 仓库名称 * 仓库名称只支持字母、数字、下划线(_)、中划线(-)和点(.)的组合 0/100

初始化仓库

☐ 创建 SVN 仓库推荐布局 (tags, branches, trunk) SVN 仓库使用帮助

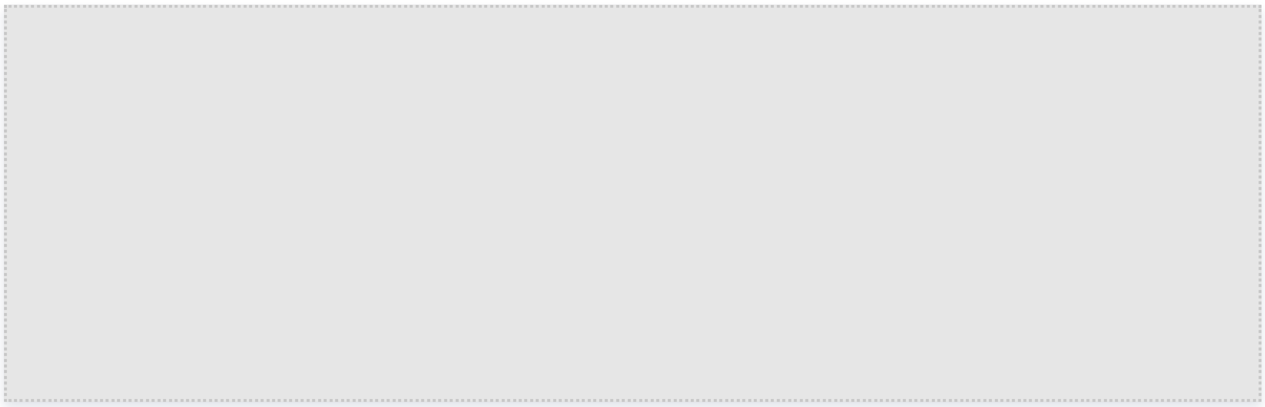
3. If you choose **Create Recommended SVN Repository Layout**, the system will automatically create the tags, branches, and trunk directories. This is the recommended directory layout for most SVN repositories.
4. After the repository initialization is complete, you can view the content of the SVN repository in the **Code** interface.



svn_test

名称	提交人	更新时间	提交	
branches	[redacted]	2023-08...	r1	Initial project layout
trunk	[redacted]	2023-08...	r1	Initial project layout
tags	[redacted]	2023-08...	r1	Initial project layout

5. Click **Detection** in the upper-right corner to see the SVN address of the repository.



Note:
Currently, you must create SVN repositories in a project. You cannot create an SVN repository in a Git repository.

Access SVN Repository

Last updated：2024-09-05 16:13:39

The SVN repository service currently supports most mainstream SVN clients. We recommend you use the latest stable version of the client.

Open Project

1. Log in to CODING Console, click the team domain name to enter the CODING page.
2. Click **Item** on the left side of the team homepage to enter the project list page and select the target project.
3. In the menu on the left, select **Code Repository** to enter the code repository homepage.

Mac Environment

In a Mac environment, you can use Homebrew to install the SVN client.

1. Run the following command to install Homebrew:

```
/usr/bin/ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)"
```

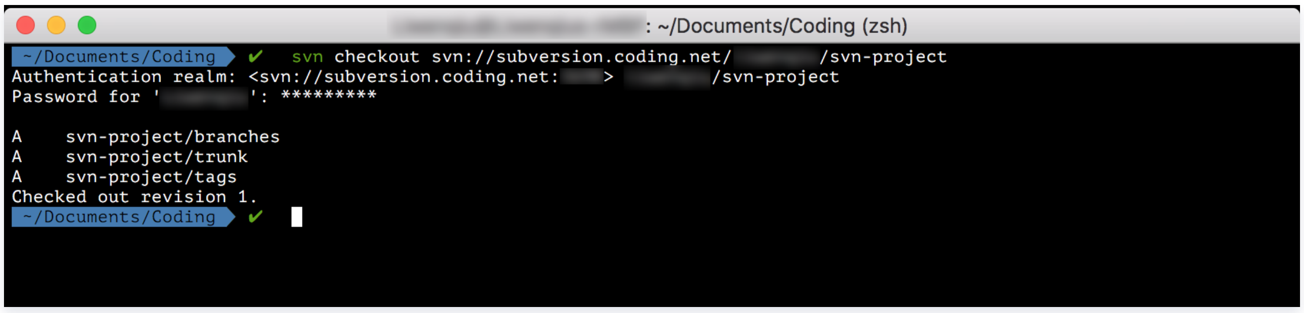
2. After you install Homebrew, input the following command in your terminal to install SVN:

```
brew install subversion
```

3. Run the command `svn --version` to verify that SVN has been correctly installed:

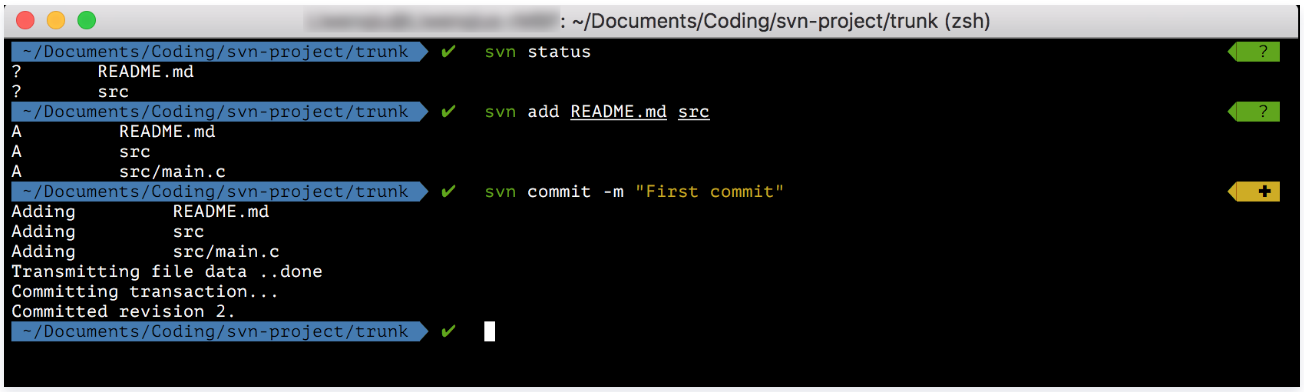
```
svn, version 1.9.7 (r1800392)
compiled Feb 28 2018, 15:54:50 on x86_64-apple-darwin17.3.0
Copyright (C) 2017 The Apache Software Foundation.
This software consists of contributions made by many people;
see the NOTICE file for more information.
Subversion is open source software, see http://subversion.apache.org/
The following repository access (RA) modules are available:

* ra_svn : Module for accessing a repository using the svn network protocol.
- with Cyrus SASL authentication
- handles 'svn' scheme
* ra_local : Module for accessing a repository on local disk.
- handles 'file' scheme
* ra_serf : Module for accessing a repository via WebDAV protocol using serf.
- using serf 1.3.9 (compiled with 1.3.9)
- handles 'http' scheme
- handles 'https' scheme
The following authentication credential caches are available:
* Plaintext cache in /Users/Liwenqiu/.subversion
* Mac OS X Keychain
```

4. Run the command `svn checkout svn://subversion.e.coding.net/example/example-project` (replacing the URL with your SVN repository URL) to check out the SVN repository:



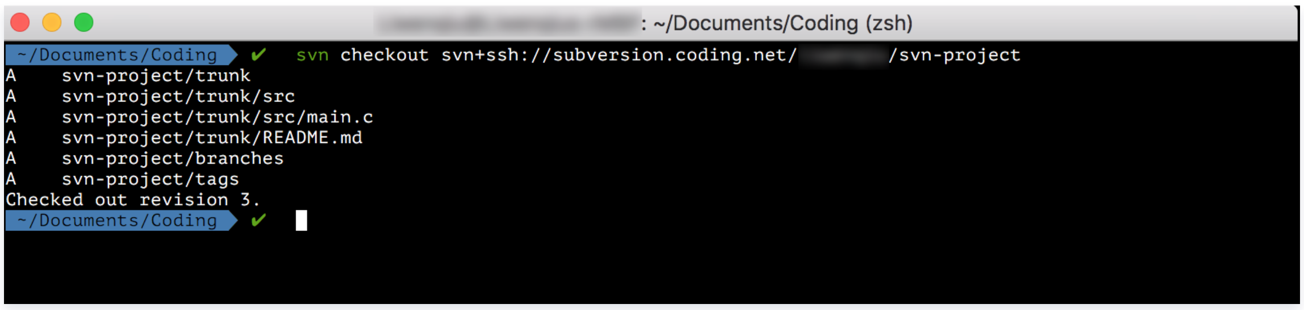
```
[redacted]: ~/Documents/Coding (zsh)
~/Documents/Coding ✔  svn checkout svn://subversion.coding.net/[redacted]/svn-project
Authentication realm: <svn://subversion.coding.net:[redacted]> [redacted]/svn-project
Password for '[redacted]': *********

A    svn-project/branches
A    svn-project/trunk
A    svn-project/tags
Checked out revision 1.
~/Documents/Coding ✔
```

5. Then, you can use the add and commit commands to add content to the repository:



```
[redacted]: ~/Documents/Coding/svn-project/trunk (zsh)
~/Documents/Coding/svn-project/trunk ✔  svn status                                ?
?       README.md
?       src
~/Documents/Coding/svn-project/trunk ✔  svn add README.md src                     ?
A         README.md
A         src
A         src/main.c
~/Documents/Coding/svn-project/trunk ✔  svn commit -m "First commit"              ✚
Adding          README.md
Adding          src
Adding          src/main.c
Transmitting file data ..done
Committing transaction...
Committed revision 2.
~/Documents/Coding/svn-project/trunk ✔
```

6. In addition to using the SVN protocol, you can use the `svn+ssh` protocol to access the repository, as shown below:

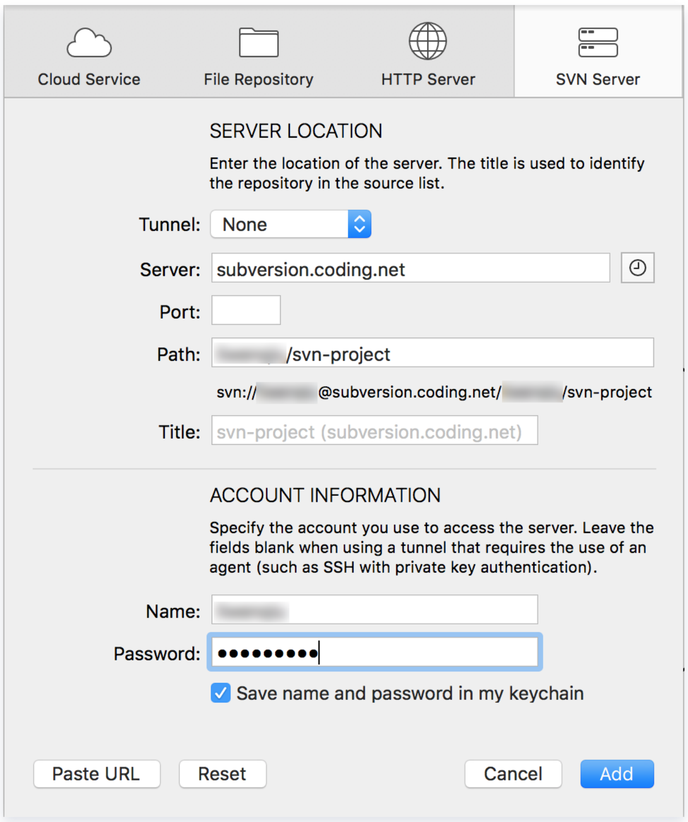


```
[redacted]: ~/Documents/Coding (zsh)
~/Documents/Coding ✔  svn checkout svn+ssh://subversion.coding.net/[redacted]/svn-project
A    svn-project/trunk
A    svn-project/trunk/src
A    svn-project/trunk/src/main.c
A    svn-project/trunk/README.md
A    svn-project/branches
A    svn-project/tags
Checked out revision 3.
~/Documents/Coding ✔
```

Cornerstone Tool

You can use SVN repositories through Cornerstone.

1. Open Cornerstone and click **Add Repository** to add an SVN repository reference (replacing the URL with your SVN repository URL).



SERVER LOCATION

Enter the location of the server. The title is used to identify the repository in the source list.

Tunnel:	None
Server:	subversion.coding.net
Port:	
Path:	[redacted]/svn-project
	svn://[redacted]@subversion.coding.net/[redacted]/svn-project
Title:	svn-project (subversion.coding.net)

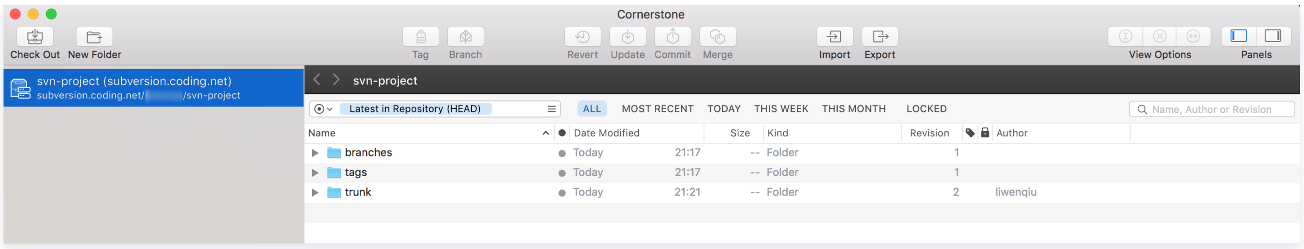
ACCOUNT INFORMATION

Specify the account you use to access the server. Leave the fields blank when using a tunnel that requires the use of an agent (such as SSH with private key authentication).

Name:	[redacted]
Password:	•••••••••

✓ Save name and password in my keychain

Then, you can view the repository content as follows:

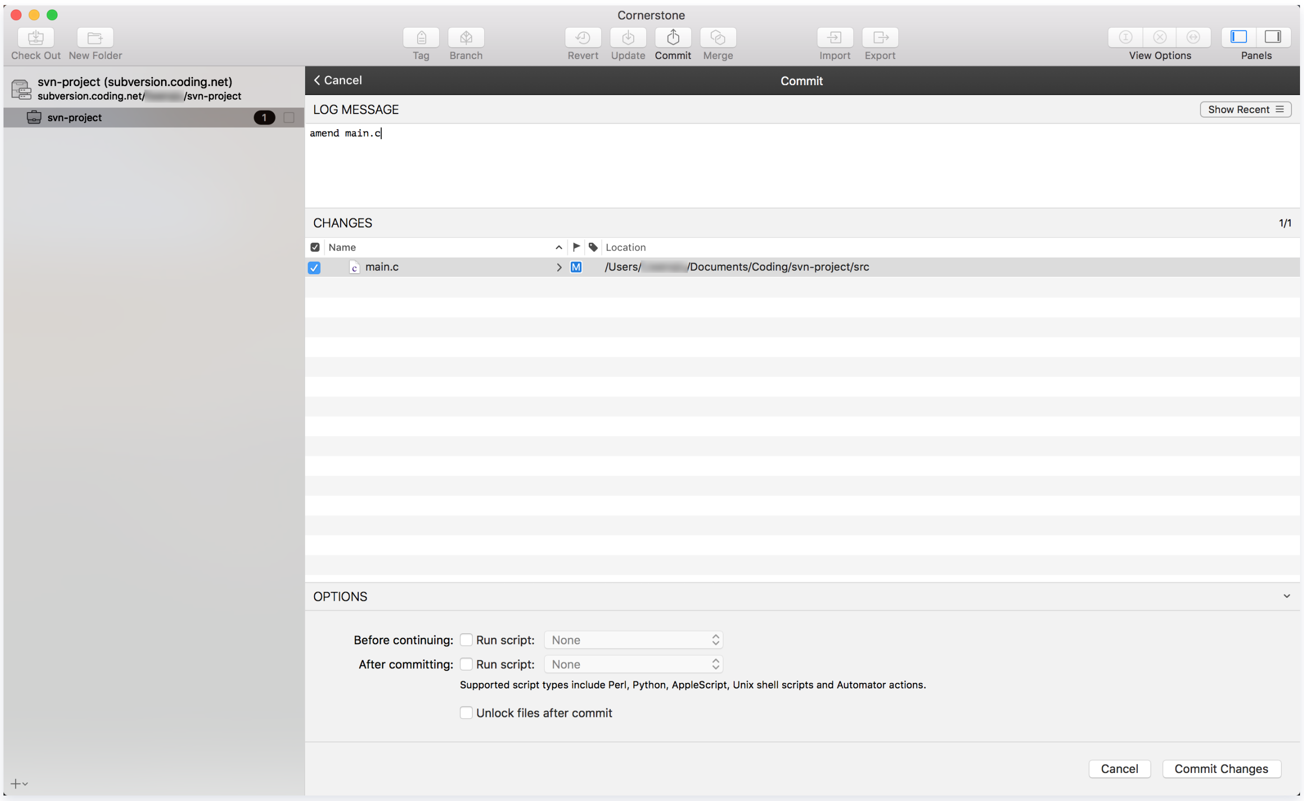


svn-project (subversion.coding.net)
subversion.coding.net/[redacted]/svn-project

svn-project

Name	Date Modified		Size	Kind	Revision	Author
branches	Today	21:17	--	Folder	1	
tags	Today	21:17	--	Folder	1	
trunk	Today	21:21	--	Folder	2	liwenqiu

2. Check out the repository, edit the files, and use commit to commit the changes, as shown below:



```
amend main.c
```

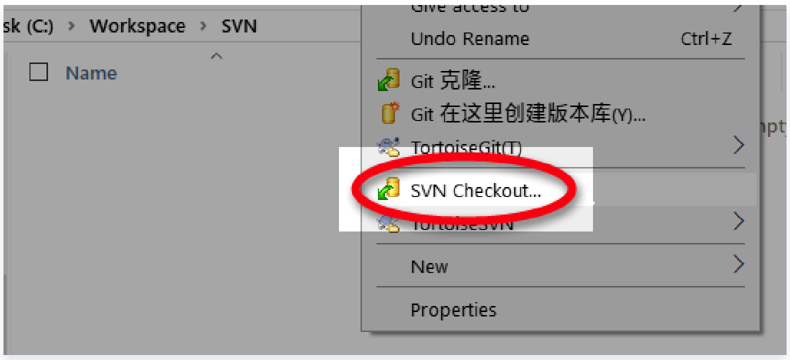
main.c — M — /Users/[redacted]/Documents/Coding/svn-project/src

Before continuing: Run script: None
After committing: Run script: None
Supported script types include Perl, Python, AppleScript, Unix shell scripts and Automator actions.
Unlock files after commit

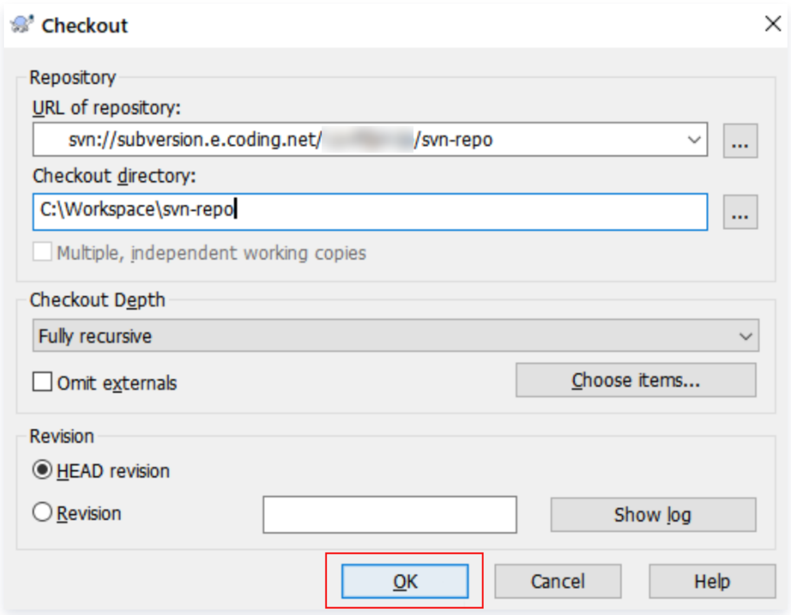
Windows Environment

In Windows, we recommend you use TortoiseSVN.

1. After download and installation are complete, right-click in any file directory.



Select `checkout` to check out the SVN repository `checkout` to your local environment (replacing the URL with your SVN repository URL).



Checkout

Repository

URL of repository: svn://subversion.e.coding.net/[redacted]/svn-repo

Checkout directory: C:\Workspace\svn-repo

Multiple, independent working copies

Checkout Depth

Fully recursive

Omit externals　Choose items...

Revision

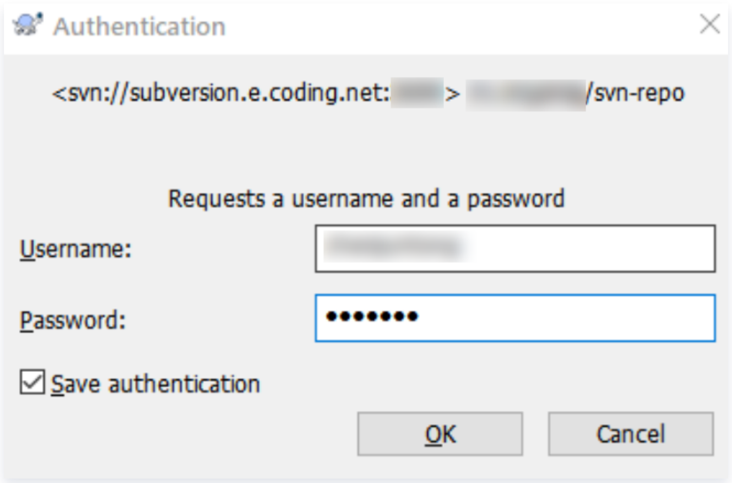
HEAD revision

Revision　Show log

OK　Cancel　Help

2. The first time you `checkout` , you need to enter a username and password. Check "Save authentication" to save the authentication information, so you won't need to enter the password every time.

   The username you enter should be the email linked to your CODING account.



Authentication

<svn://subversion.e.coding.net:[redacted]> [redacted]/svn-repo

Requests a username and a password

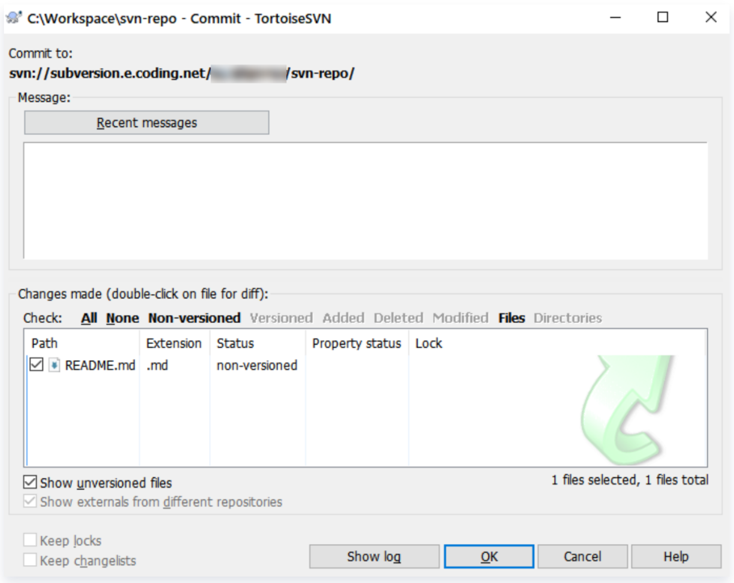
Username: [redacted]

Password: •••••••

☑ Save authentication

OK　Cancel

3. Open the checked-out folder and create a `README.md` file.



C:\Workspace\svn-repo - Commit - TortoiseSVN

Commit to:

svn://subversion.e.coding.net/[redacted]/svn-repo/

Message:

Recent messages

Changes made (double-click on file for diff):

Check: All None Non-versioned Versioned Added Deleted Modified Files Directories

Path	Extension	Status	Property status	Lock
☑ README.md	.md	non-versioned		

☑ Show unversioned files　1 files selected, 1 files total

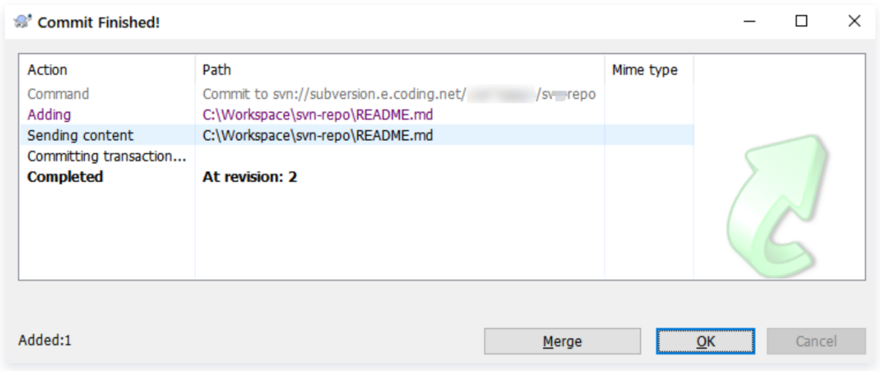
Show externals from different repositories

Keep locks

Keep changelists

Show log　OK　Cancel　Help

Right-click on a blank space and select `SVN commit...` to save the new file to the version repository:


Commit Finished!

Action	Path	Mime type
Command	Commit to svn://subversion.e.coding.net/[redacted]/sv[redacted]repo	
Adding	C:\Workspace\svn-repo\README.md	
Sending content	C:\Workspace\svn-repo\README.md	
Committing transaction...		
Completed	**At revision: 2**	

Added:1

Linux Environment

On a Linux system, you can use the system's package management tool to install SVN.

Install with yum in Fedora

```
$ sudo yum install subversion
```

Install with apt-get in Ubuntu or Debian

```
$ sudo apt-get install subversion
```

After successful installation, you can use `svn checkout / commit` to access the SVN repository.

Note:
This method is similar to the command line used in Mac systems.

Negotiation authentication mechanism error using SVN command line on Ubuntu

The following errors might occur when using the SVN command line client on Ubuntu:

```
svn: E210007: Cannot negotiate authentication mechanism
```

This is because the SVN authentication process uses the SASL library, so you need to run the following command to install the dependency library required for SASL authentication:

```
$ sudo apt-get install cyrus-sasl2-dbg
```

Manage SVN Directory Permissions

Last updated：2024-09-05 16:13:55

This document describes how to manage SVN repository permissions.

Open Project

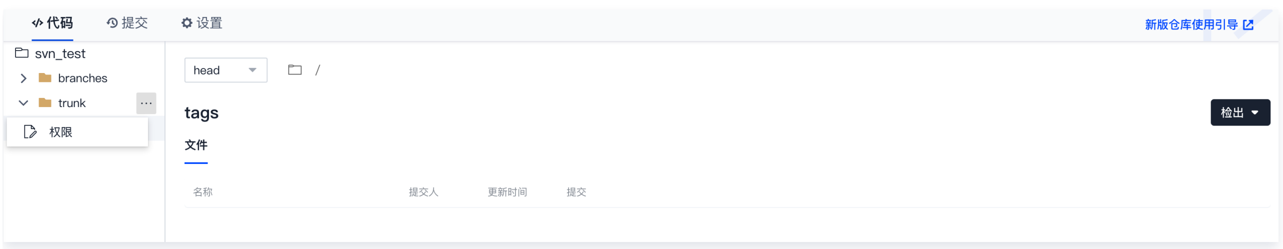
1. Log in to CODING Console, click the team domain name to enter the CODING page.
2. Click **Item** on the left side of the team homepage to enter the project list page and select the target project.
3. In the menu on the left, select **Code Repository** to enter the code repository homepage.

The SVN repository now supports permission control, allowing administrators to set directory permissions for individual users. Administrators can set the following three permissions for repositories and subdirectories individually:

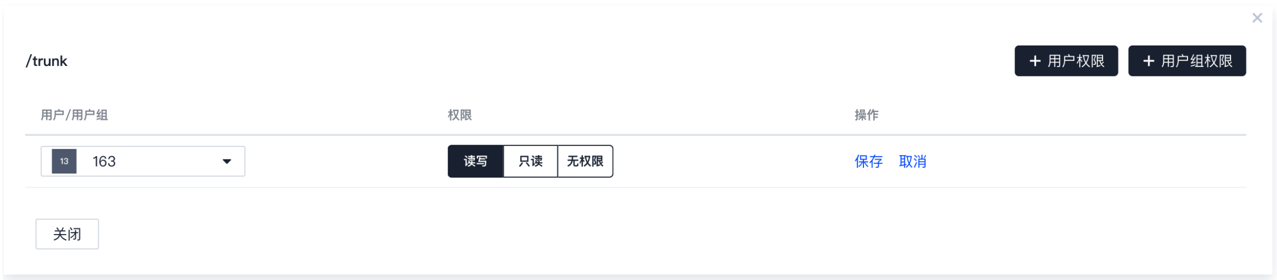
- **Read-only**: Users can check out and view the specified directory, but cannot write to it.
- **Read/Write**: Users can check out, view, and write to the specified directory.
- **No permission**: Users cannot check out, view, or write to the specified directory.

Setting permissions

Because each user has read/write permission for a repository by default, to set permission control for a directory in an SVN repository, click the directory's ··· and select **Permissions**.



On the permission settings page that appears, you can add individual users/user groups and corresponding permissions for this directory.



After you configure permission control for a directory, the directory is shown in a different color. Directories for which you have not configured permission control are shown in black, and all users have **Read/Write** permission for this directory.

- **Read/Write**: black (default)
- **Read-only**: yellow
- **No permission**: gray

Permission Override Description

In some scenarios, different permission settings may be configured for a parent directory and its subdirectories, and these permissions might conflict. For example, the parent directory might have read-only permission while the subdirectory has read/write permission.

The precedence rules for parent directory and subdirectory permissions in an SVN repository are as follows:

- If permissions are set for the parent directory but not its subdirectory, the subdirectory inherits the permission settings of the parent directory.

- If permissions are set for both the parent directory and subdirectory, the subdirectory permissions take precedence. For example:
  - If the parent directory has **Read/Write** permission and the subdirectory has **Read-only** permission, the subdirectory will actually have **Read-only** permission.
  - If the parent directory has **Read-only** permission and the subdirectory has **Read/Write** permission, the subdirectory will actually have **Read/Write** permission.
  - If the parent directory has **Read/Write** or **Read-only** permission, but the subdirectory has **No permission**, the subdirectory will actually have **No permission**.

SSH Protocol
Configure SSH Public Keys

Last updated：2024-09-05 16:14:11

This document describes how to configure SSH public keys.

Open Project

1. Log in to CODING Console, click the team domain name to enter the CODING page.
2. Click **Item** on the left side of the team homepage to enter the project list page and select the target project.
3. In the menu on the left, select **Code Repository** to enter the code repository homepage.

Description of the Feature

SSH stands for Secure Shell, an encryption protocol for network communication. It provides a secure data transmission environment in a public network and is generally used to log in to remote hosts and push/pull code.
If an SSH public key is added to a code repository, it is called a **Deploy Public Key**. By default, it has read-only permission to the project with an option to add read/write permission. If it is added to the Team Settings Center, it is called a **Team Deployment Public Key** and has read-only permission. If added to a personal account, it is called an **Account SSH Public Key** and has read/write permission to all code repositories under the account. The same SSH public key cannot be used as both a Deploy Public Key and an Account SSH Public Key.

Note:
If the SSH public key is not used as an Account SSH Public Key but you still encounter errors when adding it as a Deploy Public Key, it is possible that someone else has used this public key as an Account SSH Public Key. If you belong to multiple teams, the key might have been added to another team member's account settings.

Generate Public Key

Here, we use the `ssh-keygen` tool to generate an SSH public key. Run the following command:

```
ssh-keygen -m PEM -t ed25519 -C "your.email@example.com"  // Create a new SSH private and public key pair,
using your email as the Tag
Enter file in which to save the key (/Users/you/.ssh/id_rsa): [Press enter]  // We recommend using the
default address
Enter passphrase (empty for no passphrase):  // Press Enter directly here; if you set a password, you will
be prompted to enter it each time you push code using SSH
```

Note:
- If you need to use multiple SSH key pairs, enter a new file name at the `Enter file in which to save the key` step to avoid overwriting existing keys. For more information about SSH, visit Baidu Baike.
- If your system does not support the Ed25519 algorithm, use the command `ssh-keygen -m PEM -t rsa -b 4096 -C "your.email@example.com"`.

After the operation succeeds, you will see the following information:

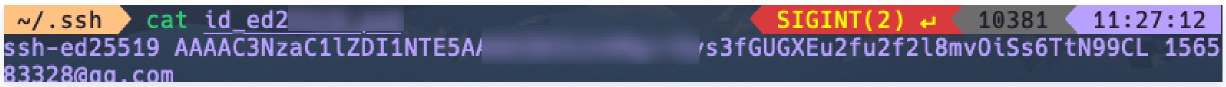
```
Your identification has been saved in /Users/you/.ssh/id_ed25519.
# Your public key has been saved in /Users/you/.ssh/id_ed25519.pub.
# The key fingerprint is:
# 01:0f:f4:3b:ca:85:d6:17:a1:7d:f0:68:9d:f0:a2:db your.email@example.com
```

Add Public Key

You can add the public key to a single repository or a personal account as needed. The same SSH public key cannot be added more than once.

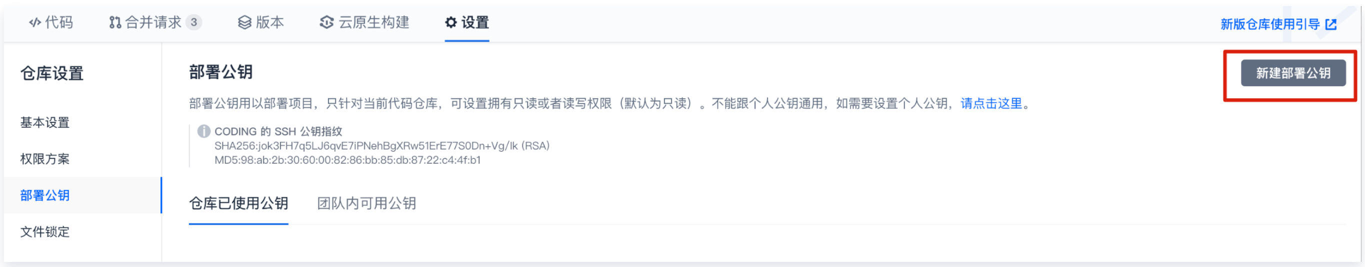
Add to Deploy Public Key

1. Navigate to the address where the key pair was generated (usually the `~/.ssh/` folder) and find the public key file with the pub extension. Use the cat command to output all content and copy it.



```
~/.ssh  cat id_ed2[redacted]                              SIGINT(2) ↵  10381  11:27:12
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AA[redacted]/s3fGUGXEu2fu2f2l8mvOiSs6TtN99CL 1565
83328@qq.com
```

2. Go to the code repository's **Settings** > **Deploy Public Key** page, click Add Deploy Key, paste the copied public key content, and the email at the end will automatically fill in as the public key name.



部署公钥

部署公钥用以部署项目，只针对当前代码仓库，可设置拥有只读或者读写权限（默认为只读）。不能跟个人公钥通用，如需要设置个人公钥，请点击这里。

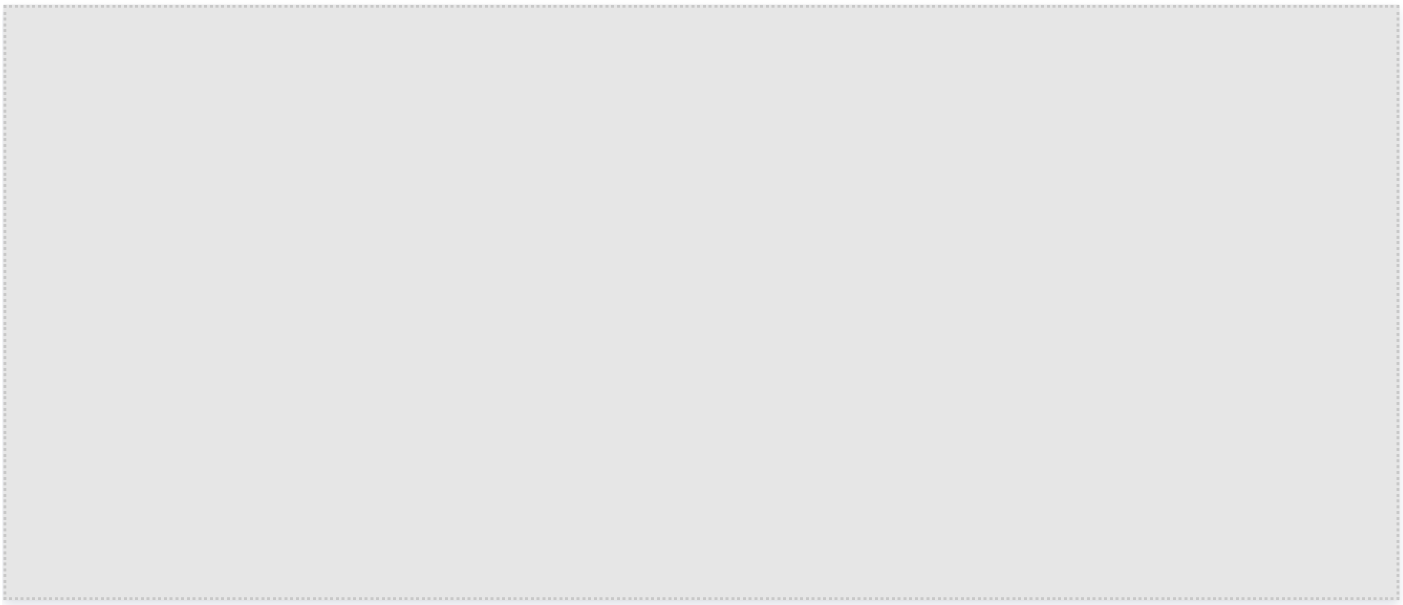
CODING 的 SSH 公钥指纹
SHA256:jok3FH7q5LJ6qvE7iPNehBgXRw51ErE77S0Dn+Vg/Ik (RSA)
MD5:98:ab:2b:30:60:00:82:86:bb:85:db:87:22:c4:4f:b1

> **Note:**
> Public deploy keys have read-only permission to the project by default. Select Grant Push Permission in the public deploy key settings to obtain push permission.

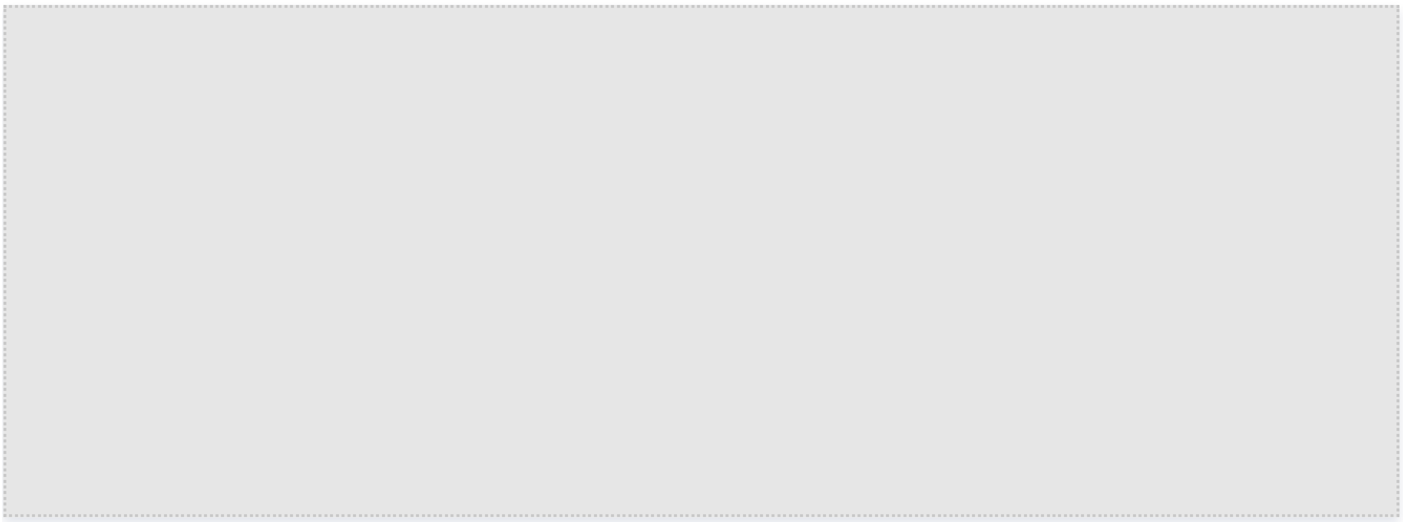
3. Afterward, run the authentication command of the public key locally for the first connection: `ssh -T git@e.coding.net` .

Add to Team Deployment Public Key

Team Deployment Public Keys have pull-only permissions and are mainly used to conveniently pull code repositories from different projects. After a team administrator inputs an SSH public key as a Team Deployment Public Key, they can share the corresponding private key with team members. This allows code repository access without needing to open source the repositories or worry about personal private key leaks.
In the context of continuous delivery, providing the private key allows the corresponding build machine or host group to check out code. This eliminates the need for complex secondary associations and verification, making the build process more convenient.
Team Owners/Administrators can click the settings button at the bottom left corner of the home page navigation to go to **Team Settings Center** > **Feature Settings** > **Code Repository** > **Team Deployment Public Keys** to input the SSH Public Key.



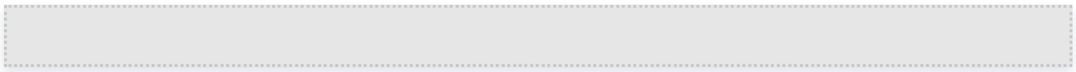
After input, add it to the respective target repository.



Once added, you can use the private key to pull the code repository through the SSH protocol.

Add to Personal Account SSH Public Key

1. Go to the address of the key pair generated above (the default address is generally `~/.ssh/`), find the public key file with the pub extension, use the cat command to output all content, and copy it.

2. Log in to CODING, click the profile photo in the upper-right corner to go to **Personal Account Settings** > **SSH Public Keys**, and then click **Create Public Key**.

3. Follow the instructions to paste the public key content you copied and enter a name for the public key.
4. Then, the first time you attempt to connect from your local device, run the public key authentication command:
   `ssh -T git@e.coding.net`

Key Fingerprint Authentication

Last updated：2024-09-05 16:14:25

This document describes how to use key fingerprints for code repository authentication. This ensures that the connected remote repository is a genuine CODING repository.

Open Project

1. Log in to CODING Console, click the team domain name to enter the CODING page.
2. Click **Item** on the left side of the team homepage to enter the project list page and select the target project.
3. In the menu on the left, select **Code Repository** to enter the code repository homepage.
4. If the Code Repository is not shown on the left, the project admin needs to go to **Project Settings** > **Projects and Members** > **Feature Toggle** to enable it.

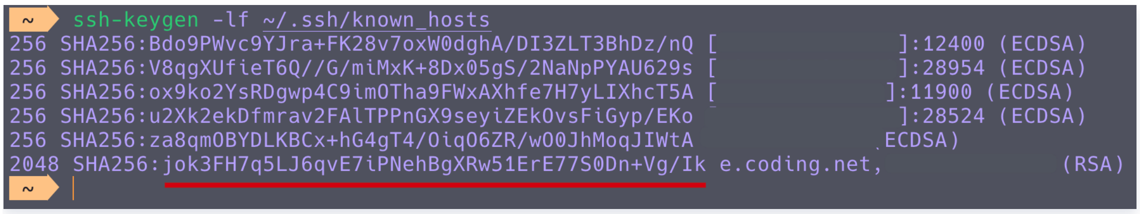
Description of the Feature

Code security is always essential. To ensure that the remote repository you are connecting to is an authentic CODING code repository, SSH key fingerprints are now provided for authentication. You simply need to run the command locally and verify the returned result to determine the authenticity of the remote repository.

Verify SHA256 Fingerprint

View the `.ssh/know_hosts` file locally for the SHA256 fingerprint of `e.coding.net`. If the return value is `jok3FH7q5LJ6qvE7iPNehBgXRw51ErE77S0Dn+Vg/Ik`, it confirms that you are connected to the correct CODING server. You can view the result of the command on the terminal.

```
ssh-keygen -lf ~/.ssh/known_hosts
```



```
~ ssh-keygen -lf ~/.ssh/known_hosts
256 SHA256:Bdo9PWvc9YJra+FK28v7oxW0dghA/DI3ZLT3BhDz/nQ [[redacted]              ]:12400 (ECDSA)
256 SHA256:V8qgXUfieT6Q//G/miMxK+8Dx05gS/2NaNpPYAU629s [              ]:28954 (ECDSA)
256 SHA256:ox9ko2YsRDgwp4C9imOTha9FWxAXhfe7H7yLIXhcT5A [             ]:11900 (ECDSA)
256 SHA256:u2Xk2ekDfmrav2FAlTPPnGX9seyiZEkOvsFiGyp/EKo               ]:28524 (ECDSA)
256 SHA256:za8qmOBYDLKBCx+hG4gT4/0iqO6ZR/wO0JhMoqJIWtA              ECDSA)
2048 SHA256:jok3FH7q5LJ6qvE7iPNehBgXRw51ErE77S0Dn+Vg/Ik e.coding.net,              (RSA)
~ 
```

Verify MD5 Fingerprint

View the `.ssh/know_hosts` file locally for the MD5 fingerprint of `e.coding.net`. If the return value is `98:ab:2b:30:60:00:82:86:bb:85:db:87:22:c4:4f:b1`, it confirms that you are connected to the correct CODING server. You can view the result of the command on the terminal.

```
ssh-keygen -E md5 -lf ~/.ssh/known_hosts
```

Push/Pull Code Via SSH Protocol

Last updated：2024-09-05 16:14:40

This document describes how to use SSH protocol to push/pull code.

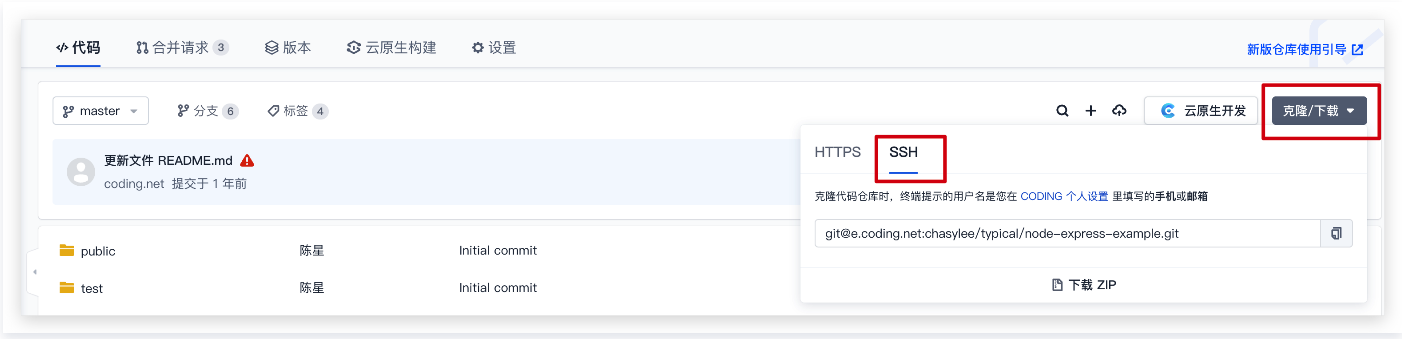
Open Project

1. Log in to CODING Console, click the team domain name to enter the CODING page.
2. Click **Item** on the left side of the team homepage to enter the project list page and select the target project.
3. In the menu on the left, select **Code Repository** to enter the code repository homepage.
4. If the Code Repository is not shown on the left, the project admin needs to go to **Project Settings** > **Projects and Members** > **Feature Toggle** to enable it.

Operation step

CODING supports pushing/pulling code via SSH protocol.

1. Refer to Configure SSH Public Keys to generate a public key and add it to a code repository or personal account.
2. Copy the SSH URL on the code repository's Browse page.



更新文件 README.md
coding.net 提交于 1 年前

public	陈星	Initial commit
test	陈星	Initial commit

克隆代码仓库时，终端提示的用户名是您在 CODING 个人设置 里填写的手机或邮箱

git@e.coding.net:chasylee/typical/node-express-example.git

3. Run `git clone + repository URL` command on your local machine to pull code.

```

```

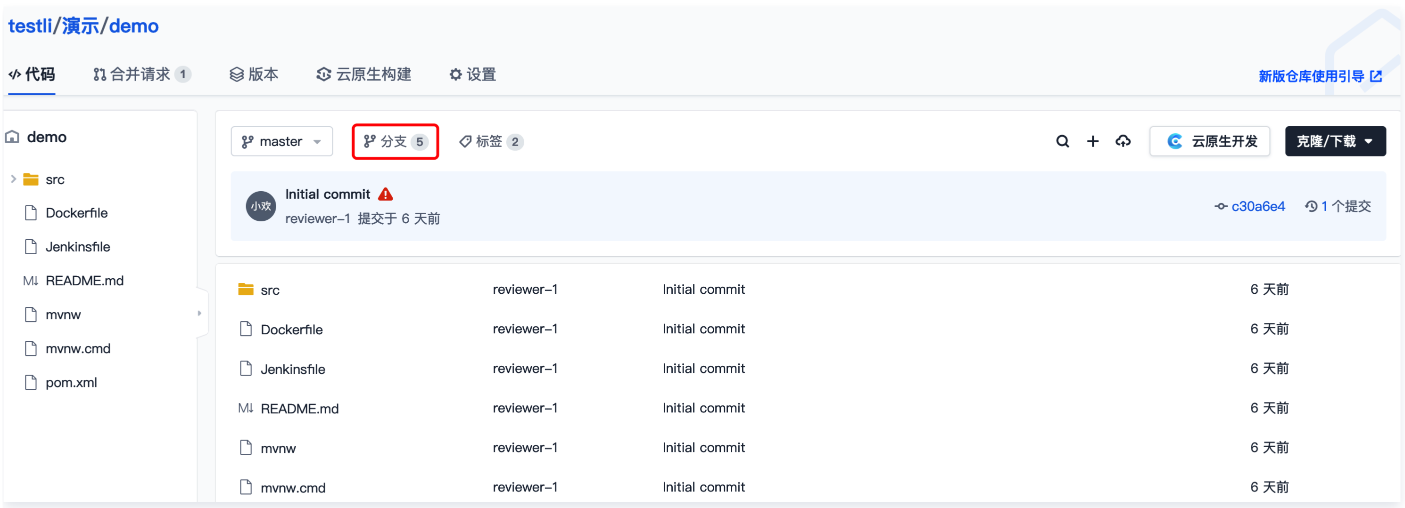
Branch management Create Branch

Last updated：2024-09-05 16:14:54

Branches are a common function in Git. The common development approach is to retain one trunk (or master branch), while each development or bug fix task is carried out on its own branch. When finished, the branches are merged back into the trunk. Because development directly on the trunk is very risky, the branch can be viewed as a safety valve, which isolates different development tasks. It can ensure that the stability of the master version is not compromised, while allowing different people to focus on different development tasks.
Below, we will refer to branches in CODING code repositories as remote branches and branches in local Git repositories as local branches. You can run relevant commands on your local terminal to quickly create branches. For details, see the Git Command Quick Reference.

View Branches

Go to the code repository details page and click **Branches** to see all branches in the remote repository. Here, you can create branches, enable protected branches, etc. The branch list page shows the number of commits ahead or behind the default branch.



Initial commit
reviewer-1 提交于 6 天前
c30a6e4
1 个提交

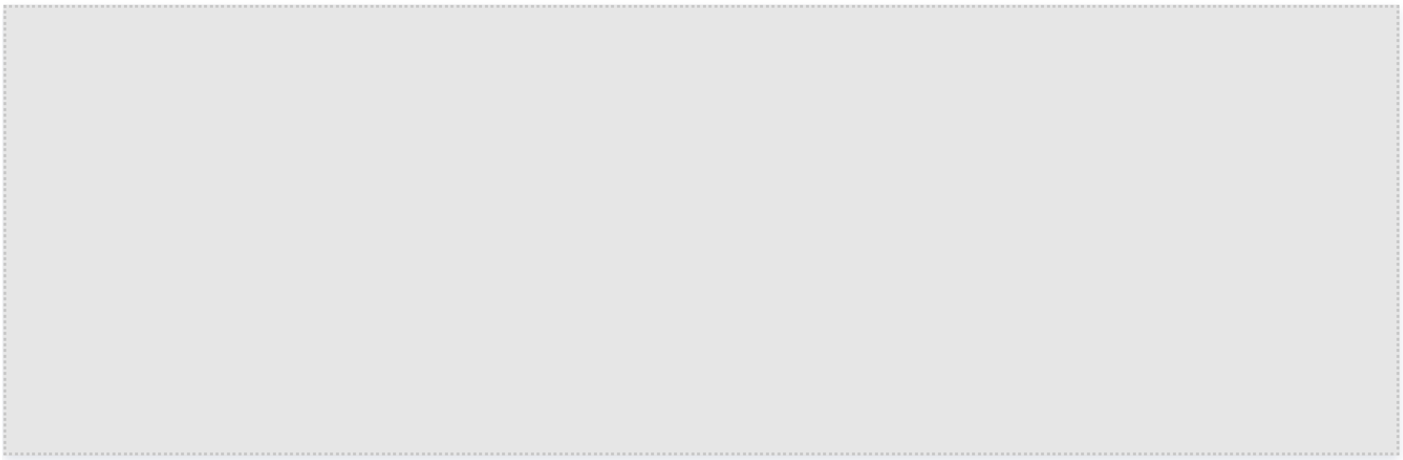
src	reviewer-1	Initial commit	6 天前
Dockerfile	reviewer-1	Initial commit	6 天前
Jenkinsfile	reviewer-1	Initial commit	6 天前
README.md	reviewer-1	Initial commit	6 天前
mvnw	reviewer-1	Initial commit	6 天前
mvnw.cmd	reviewer-1	Initial commit	6 天前

Create Branch

Click the upper-right corner **Create Branch**, and follow the instructions in the pop-up window to enter the relevant configuration information. The master branch is used as the default source. The new branch content is identical to the original branch, essentially deriving a new branch from the original.

Add Remarks

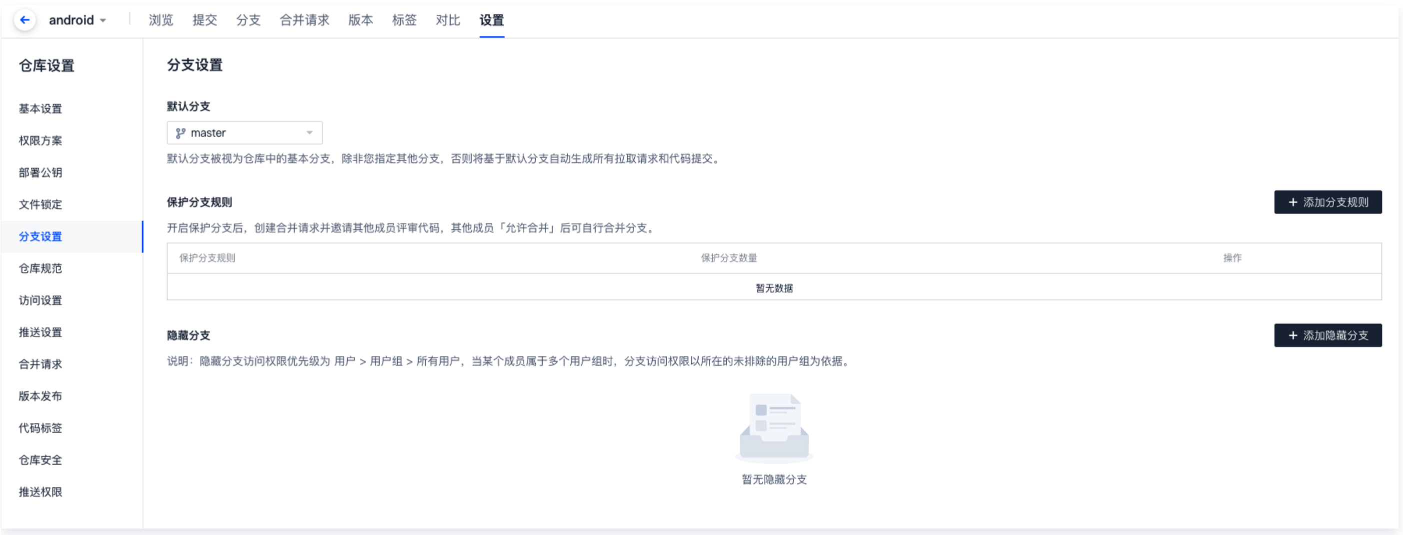
If the branch name can't fully describe the branch purpose, you can add a simple description to explain the branch's purpose.



Set Default Branch

Last updated：2024-09-05 16:15:10

The default branch is considered the base branch in the repository. Unless you specify another branch, all pull requests and code commits will be automatically generated based on the default branch. The default branch can be modified on the **Settings > Branch Settings** page of the code repository.



分支设置

默认分支

master

默认分支被视为仓库中的基本分支，除非您指定其他分支，否则将基于默认分支自动生成所有拉取请求和代码提交。

保护分支规则

开启保护分支后，创建合并请求并邀请其他成员评审代码，其他成员「允许合并」后可自行合并分支。

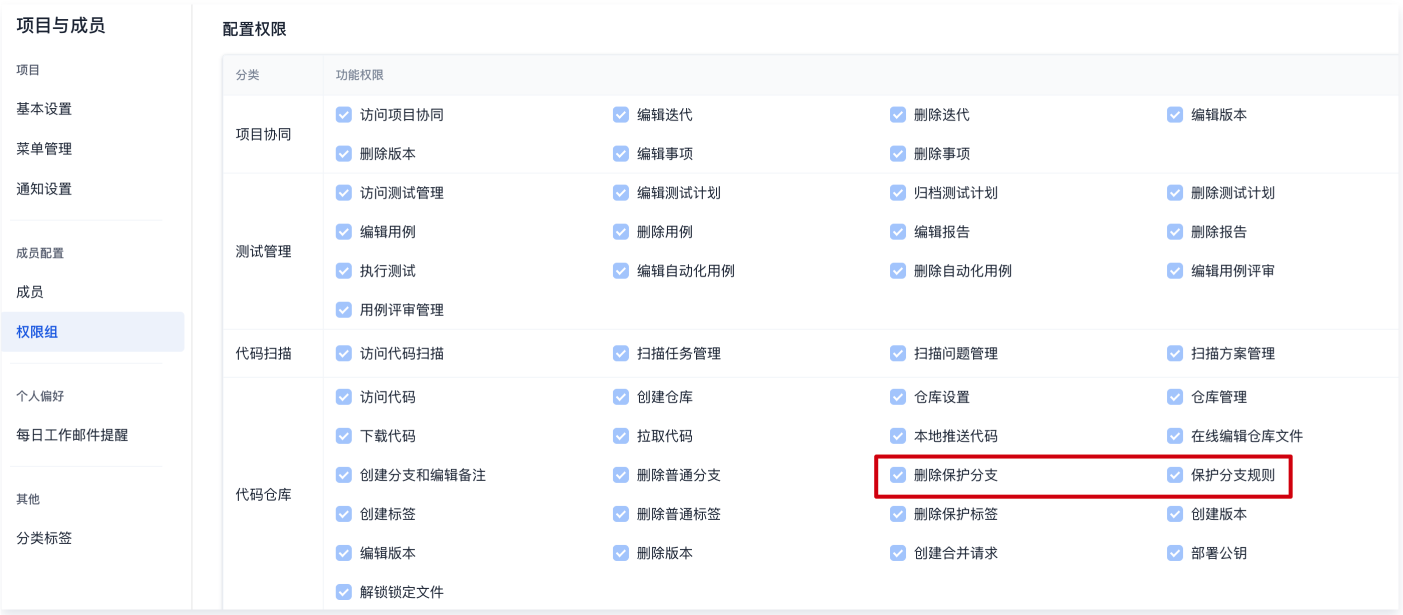
保护分支规则	保护分支数量	操作
	暂无数据	

隐藏分支

说明：隐藏分支访问权限优先级为 用户 > 用户组 > 所有用户，当某个成员属于多个用户组时，分支访问权限以所在的未排除的用户组为依据。

暂无隐藏分支

When the user's permission group has the **Protect Branch** permission, they will have the authority to adjust the default branch.



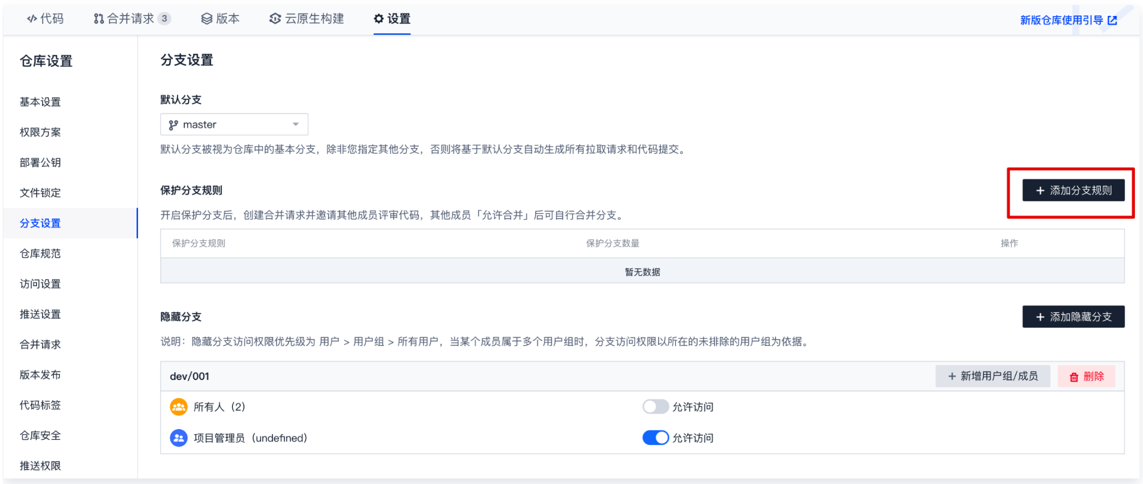
配置权限

分类	功能权限			
项目协同	✓ 访问项目协同	✓ 编辑迭代	✓ 删除迭代	✓ 编辑版本
	✓ 删除版本	✓ 编辑事项	✓ 删除事项	
测试管理	✓ 访问测试管理	✓ 编辑测试计划	✓ 归档测试计划	✓ 删除测试计划
	✓ 编辑用例	✓ 删除用例	✓ 编辑报告	✓ 删除报告
	✓ 执行测试	✓ 编辑自动化用例	✓ 删除自动化用例	✓ 编辑用例评审
	✓ 用例评审管理			
代码扫描	✓ 访问代码扫描	✓ 扫描任务管理	✓ 扫描问题管理	✓ 扫描方案管理
代码仓库	✓ 访问代码	✓ 创建仓库	✓ 仓库设置	✓ 仓库管理
	✓ 下载代码	✓ 拉取代码	✓ 本地推送代码	✓ 在线编辑仓库文件
	✓ 创建分支和编辑备注	✓ 删除普通分支	✓ 删除保护分支	✓ 保护分支规则
	✓ 创建标签	✓ 删除普通标签	✓ 删除保护标签	✓ 创建版本
	✓ 编辑版本	✓ 删除版本	✓ 创建合并请求	✓ 部署公钥
	✓ 解锁锁定文件			

Set Protected Branch

Last updated：2024-09-05 16:15:28

Branch protection is a special function CODING uses to limit Git code permissions. Selected branches can be protected from unreported and unauthorized changes.
When you enable branch protection, protected branches are marked with a green shield in the branch list. To modify a protected branch, members must create a new branch and modify it. Then, they submit a merge request to invite other members to review the code. If the review result is Allow Merge, the merge operation can be performed.



分支设置

默认分支

master

默认分支被视为仓库中的基本分支，除非您指定其他分支，否则将基于默认分支自动生成所有拉取请求和代码提交。

保护分支规则

开启保护分支后，创建合并请求并邀请其他成员评审代码，其他成员「允许合并」后可自行合并分支。

保护分支规则	保护分支数量	操作
暂无数据		

隐藏分支

说明：隐藏分支访问权限优先级为 用户 > 用户组 > 所有用户，当某个成员属于多个用户组时，分支访问权限以所在的未排除的用户组为依据。

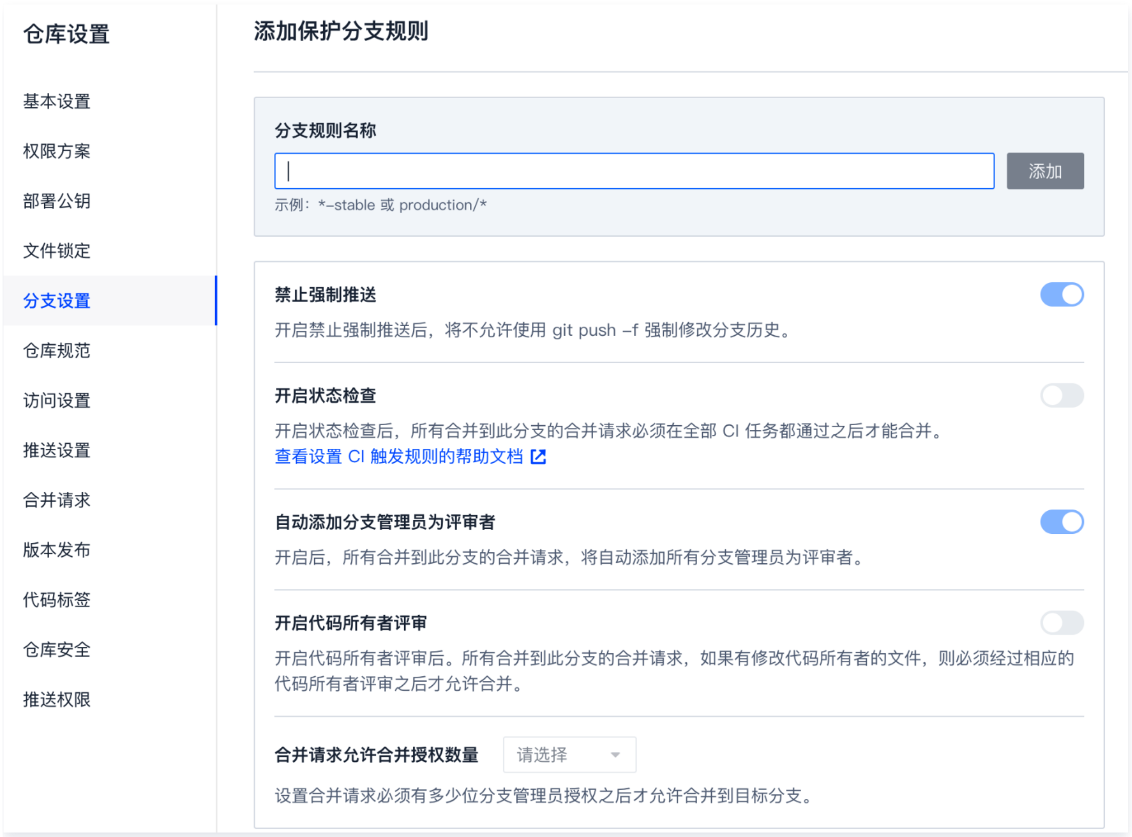
dev/001

所有人（2） 允许访问

项目管理员（undefined） 允许访问

Set Protected Branch Rules

In a code repository, in the **Settings > Branch Settings**, enter the branch name you need to protect in the branch rules name. Branches that match the naming rules will be considered protected branches.



添加保护分支规则

分支规则名称

示例：*-stable 或 production/*

禁止强制推送

开启禁止强制推送后，将不允许使用 git push -f 强制修改分支历史。

开启状态检查

开启状态检查后，所有合并到此分支的合并请求必须在全部 CI 任务都通过之后才能合并。
查看设置 CI 触发规则的帮助文档

自动添加分支管理员为评审者

开启后，所有合并到此分支的合并请求，将自动添加所有分支管理员为评审者。

开启代码所有者评审

开启代码所有者评审后。所有合并到此分支的合并请求，如果有修改代码所有者的文件，则必须经过相应的代码所有者评审之后才允许合并。

合并请求允许合并授权数量

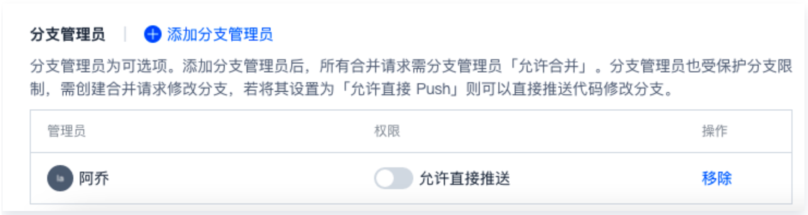
设置合并请求必须有多少位分支管理员授权之后才允许合并到目标分支。

- **Disallow Force Push:** Enabled by default. Even users with permission to perform git push cannot use `git push -f` to force modify the branch commit history. We strongly recommend you enable this option when multiple people collaborate on a branch. This ensures that users must use new commits to change the branch content, rather than modifying prior commits.

- **Enable Status Check**: By setting the specification check conditions or setting a code scanning scheme in the CI, merges are allowed only after the CI is successful. Check Trigger Rules for more information.
- **Automatically Add Branch Admin as Reviewer**: When this feature is enabled, all merge requests to this protected branch automatically add all admins of the current branch as reviewers.
- **Enable Review by Code Owner**: When this feature is enabled, for merge requests involving protected branches, any modification must be reviewed by the relevant code owner before the merge is allowed. Typically, different code paths determine specific features in the application, and maintainers naturally do not want their code to be altered inadvertently by other members.
- **Number of Reviewers Authorizing Merge**: This sets the number of branch admins that must authorize a merge request before it can be merged into the target branch. If no branch admins are set for a protected branch, authorization from one ordinary member is required before the merge is allowed.

Specify Branch Admins

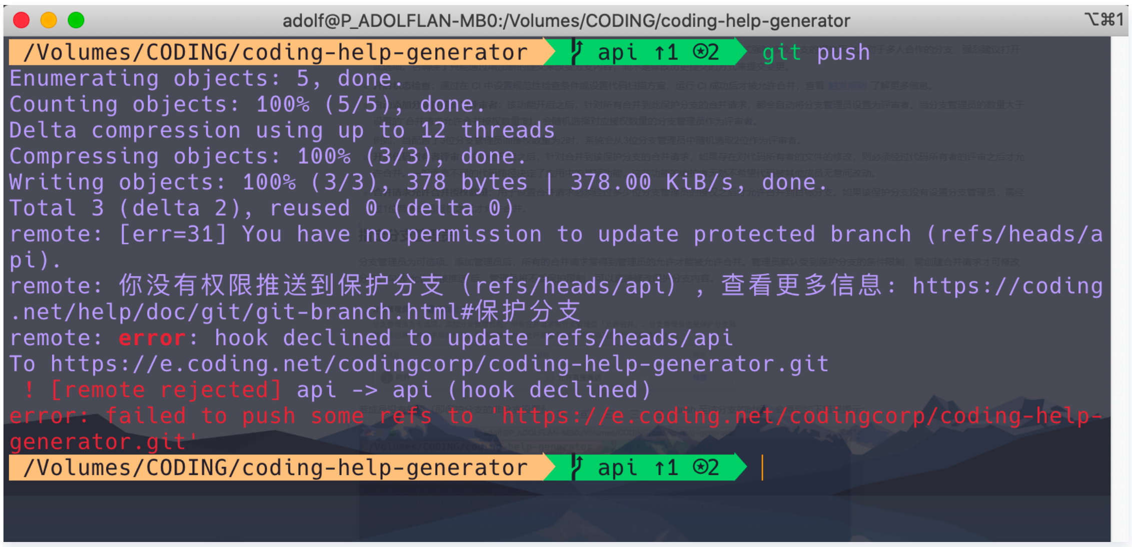
Branch administrators are optional. After adding an administrator, all merge requests need the administrator's permission to be merged. Administrators by default are subject to the protected branch's conditions and need to create a merge request to modify the branch. By selecting "Allow direct push", administrators will not be restricted by protection and can directly modify protected branch content.


分支管理员 | 添加分支管理员

分支管理员为可选项。添加分支管理员后，所有合并请求需分支管理员「允许合并」。分支管理员也受保护分支限制，需创建合并请求修改分支，若将其设置为「允许直接 Push」则可以直接推送代码修改分支。

管理员	权限	操作
阿乔	允许直接推送	移除

If members do not have permissions (i.e., they are not designated as branch admins for the protected branch) to push to the branch, they will receive the following error message when attempting to push to the branch:

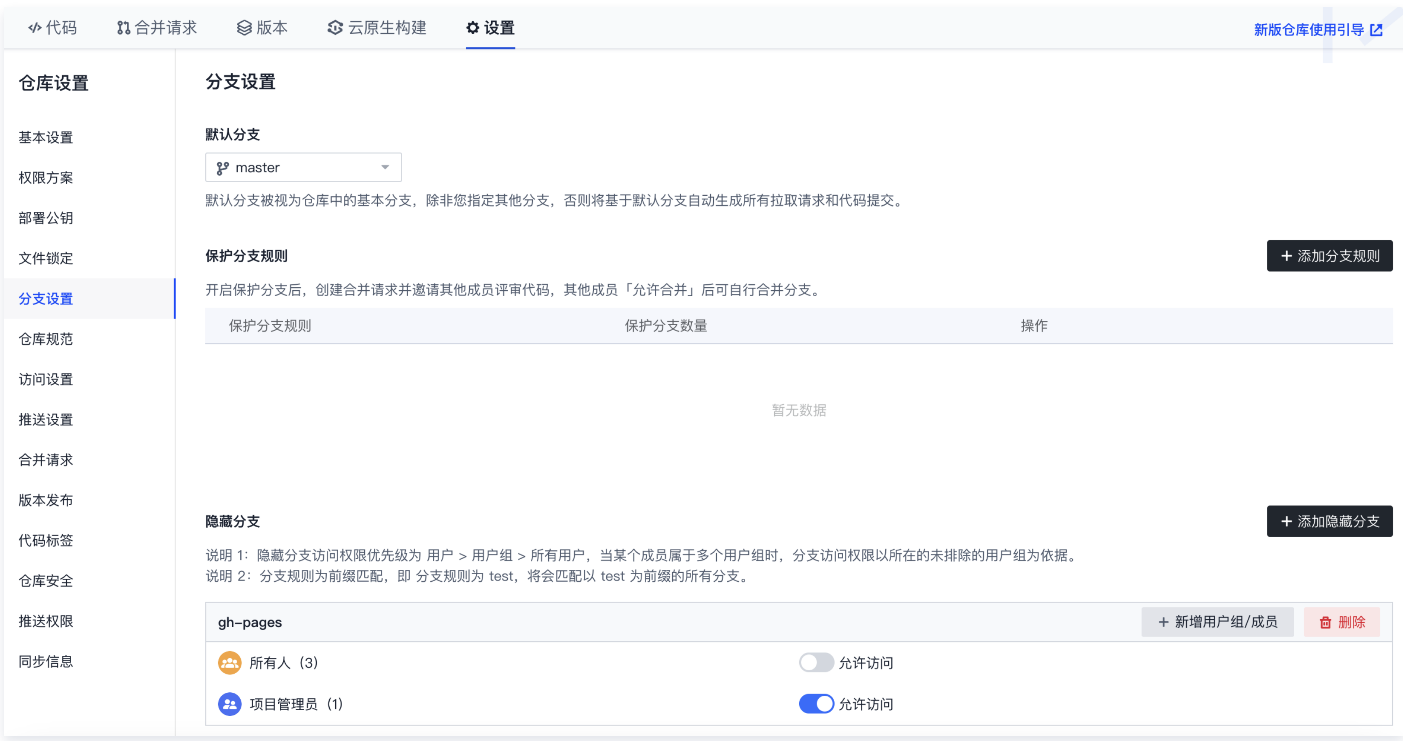

```
adolf@P_ADOLFLAN-MB0:/Volumes/CODING/coding-help-generator
 /Volumes/CODING/coding-help-generator  api ↑1 ⊛2  git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 12 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 378 bytes | 378.00 KiB/s, done.
Total 3 (delta 2), reused 0 (delta 0)
remote: [err=31] You have no permission to update protected branch (refs/heads/a
pi).
remote: 你没有权限推送到保护分支 (refs/heads/api) , 查看更多信息: https://coding
.net/help/doc/git/git-branch.html#保护分支
remote: error: hook declined to update refs/heads/api
To https://e.coding.net/codingcorp/coding-help-generator.git
 ! [remote rejected] api -> api (hook declined)
error: failed to push some refs to 'https://e.coding.net/codingcorp/coding-help-
generator.git'
 /Volumes/CODING/coding-help-generator  api ↑1 ⊛2 
```

Set Hidden Branch

Last updated：2024-09-05 16:15:45

To control access permissions to a single branch, you can make any branch other than the default branch a hidden branch. Only authorized users and user groups can access hidden branches, ensuring the security of the code.

1. On the details page of a code repository, click **Settings > Branch Settings** to go to the branch settings page.


代码 合并请求 版本 云原生构建 设置 新版仓库使用引导

仓库设置: 基本设置 权限方案 部署公钥 文件锁定 分支设置 仓库规范 访问设置 推送设置 合并请求 版本发布 代码标签 仓库安全 推送权限 同步信息

分支设置

默认分支

master

默认分支被视为仓库中的基本分支，除非您指定其他分支，否则将基于默认分支自动生成所有拉取请求和代码提交。

保护分支规则 + 添加分支规则

开启保护分支后，创建合并请求并邀请其他成员评审代码，其他成员「允许合并」后可自行合并分支。

保护分支规则	保护分支数量	操作
暂无数据		

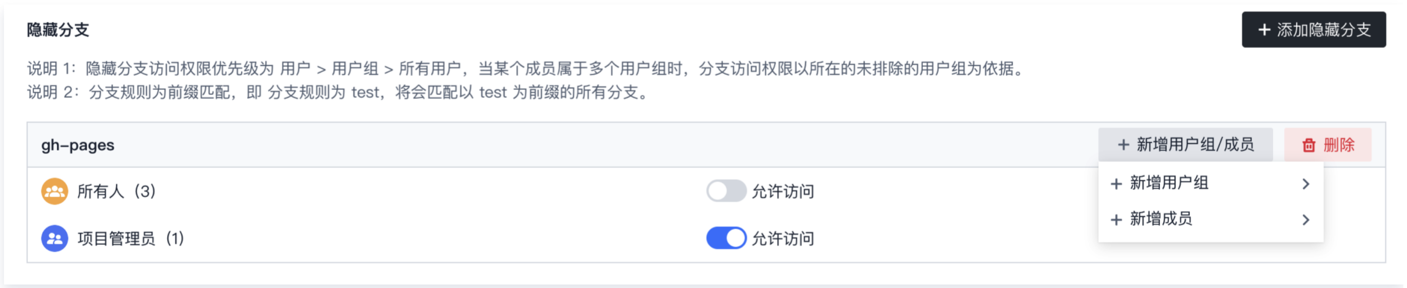
隐藏分支 + 添加隐藏分支

说明 1：隐藏分支访问权限优先级为 用户 > 用户组 > 所有用户，当某个成员属于多个用户组时，分支访问权限以所在的未排除的用户组为依据。
说明 2：分支规则为前缀匹配，即 分支规则为 test，将会匹配以 test 为前缀的所有分支。

gh-pages + 新增用户组/成员 删除

所有人（3）	允许访问
项目管理员（1）	允许访问

2. Click **Add Hidden Branch**, select or enter the branch, and click **Save**. When a branch is hidden, it is shown in the hidden branch list.
3. Use **Add User Group** or **Add Member** to specify the users that are allowed or denied access to the branch.


隐藏分支 + 添加隐藏分支

说明 1：隐藏分支访问权限优先级为 用户 > 用户组 > 所有用户，当某个成员属于多个用户组时，分支访问权限以所在的未排除的用户组为依据。
说明 2：分支规则为前缀匹配，即 分支规则为 test，将会匹配以 test 为前缀的所有分支。

gh-pages + 新增用户组/成员 删除

所有人（3）	允许访问
项目管理员（1）	允许访问

The priority of hidden branch access permissions is as follows: user > user group > all users. If a member belongs to multiple user groups, the member can access a hidden branch if any of their groups has access permission.

For example, if User A belongs to User Group 1 (allowed access to the `dev/001` branch) and User Group 2 (denied access to the `dev/001` branch), User A can access the `dev/001` branch.

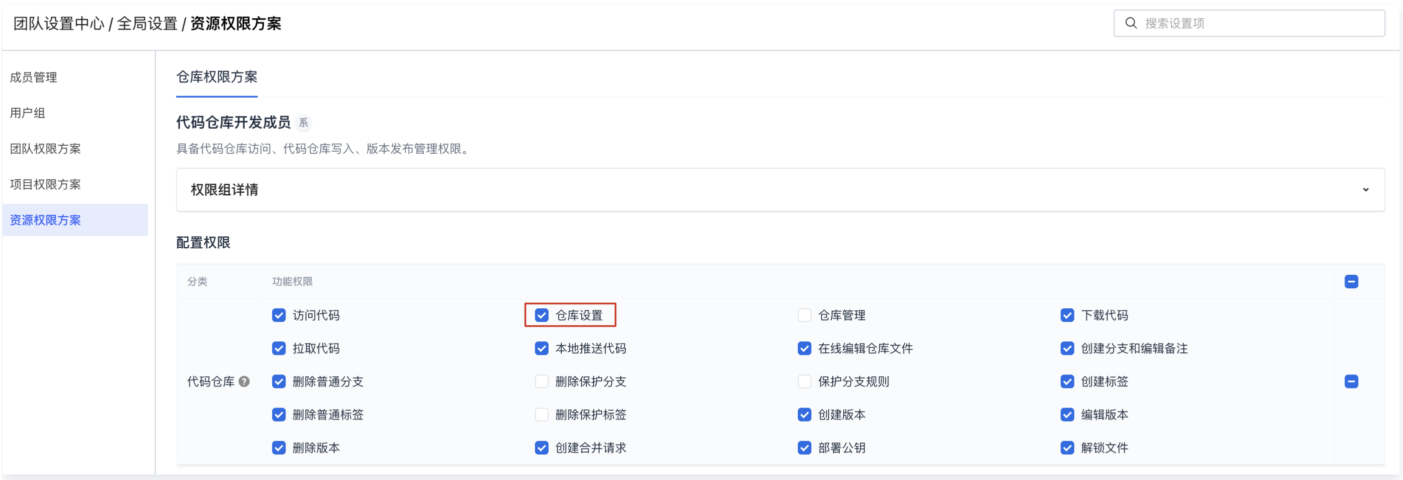
Set Read-only Branch

Last updated：2024-09-05 16:16:03

After setting a read-only branch, no team member can write to or submit merge requests to this branch. Only pulling is allowed.

Configuring permission

In the **Team Settings Center** > **Organization and Members** > **Resource Permission Scheme** tab of the **Repository Permission Scheme** page, click the name of the permission group in the permission list to set repository permissions. When the repository permissions include **Repository Settings**, the administrator can set or cancel read-only branches.



团队设置中心 / 全局设置 / **资源权限方案**

搜索设置项

成员管理
用户组
团队权限方案
项目权限方案
资源权限方案

仓库权限方案

代码仓库开发成员 系

具备代码仓库访问、代码仓库写入、版本发布管理权限。

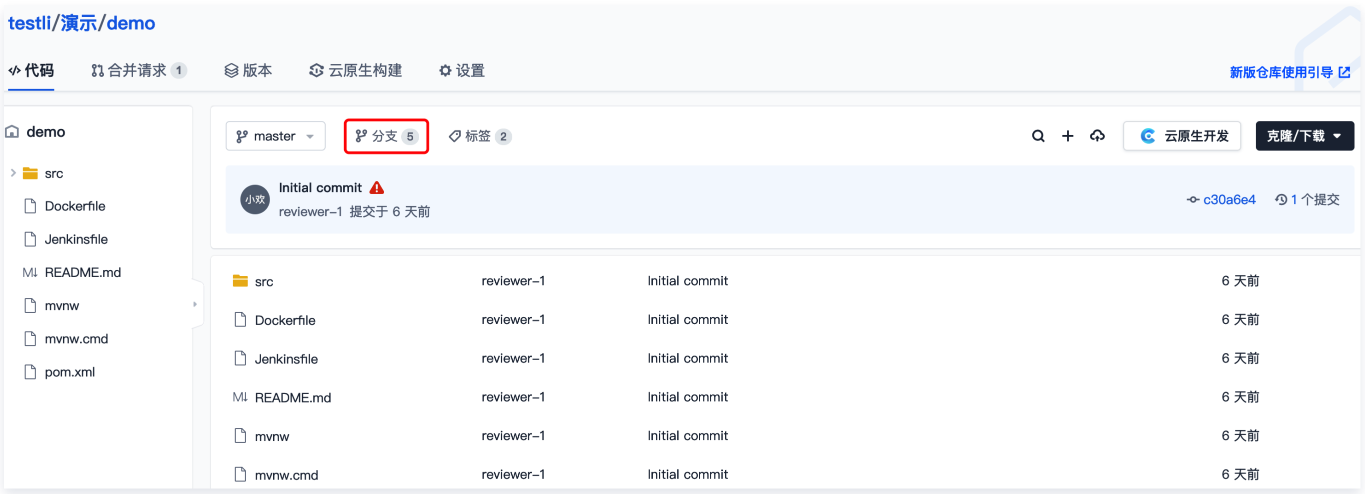
权限组详情

配置权限

分类	功能权限			
代码仓库	✓ 访问代码	✓ 仓库设置	☐ 仓库管理	✓ 下载代码
	✓ 拉取代码	✓ 本地推送代码	✓ 在线编辑仓库文件	✓ 创建分支和编辑备注
	✓ 删除普通分支	☐ 删除保护分支	☐ 保护分支规则	✓ 创建标签
	✓ 删除普通标签	☐ 删除保护标签	✓ 创建版本	✓ 编辑版本
	✓ 删除版本	✓ 创建合并请求	✓ 部署公钥	✓ 解锁文件

Set Read-only Branch

1. After entering any Code Repository, click **Branches** to enter the branch list.

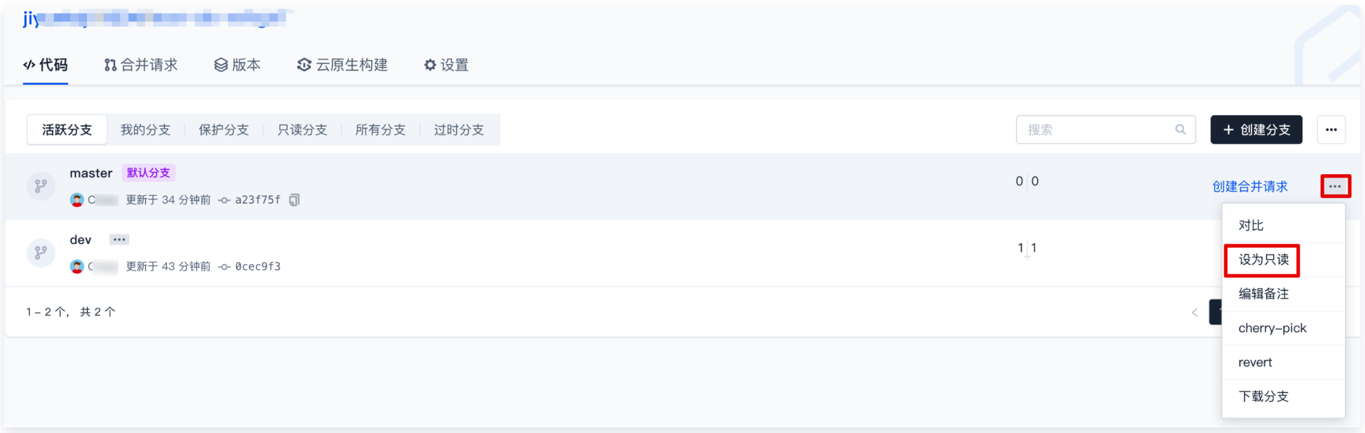


testli/演示/demo

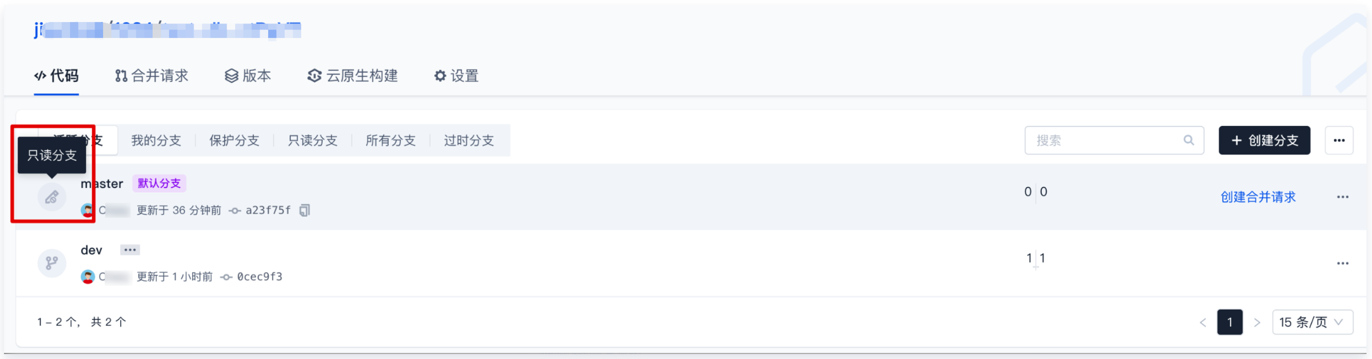
Initial commit
小欢 reviewer-1 提交于 6 天前
c30a6e4 1 个提交

src	reviewer-1	Initial commit	6 天前
Dockerfile	reviewer-1	Initial commit	6 天前
Jenkinsfile	reviewer-1	Initial commit	6 天前
README.md	reviewer-1	Initial commit	6 天前
mvnw	reviewer-1	Initial commit	6 天前
mvnw.cmd	reviewer-1	Initial commit	6 天前

2. Select the specified branch, click ··· and choose **Set to Read-only**.



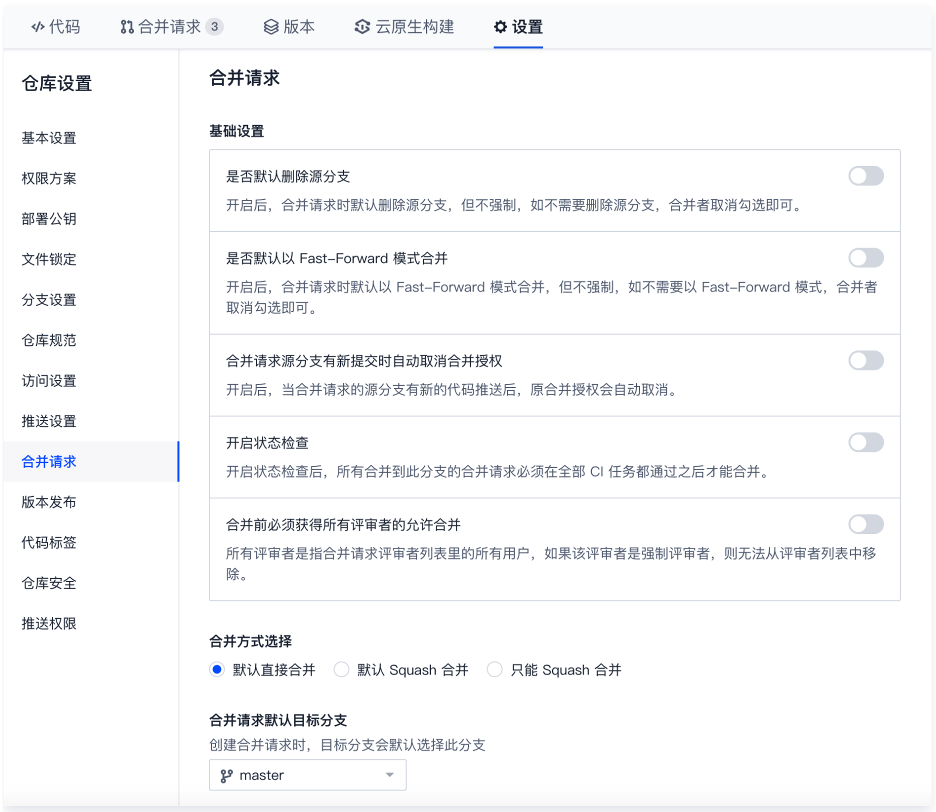
After enabling read-only mode, the branch will display a read-only indicator. To cancel read-only, click and choose **Cancel Read-only.**



Merge Request and Code Review
Merge Request Settings

Last updated：2024-09-05 16:16:24

The full name of a merge request is a Merge Request, abbreviated as MR below.
Project administrators can configure the basic settings for merge requests, adjust the default merge method, change the MR default target branch, and set the merge commit message template (Accept Merge Request) on the **Settings > Merge Requests** page of the code repository.



合并请求

基础设置

是否默认删除源分支
开启后，合并请求时默认删除源分支，但不强制，如不需要删除源分支，合并者取消勾选即可。

是否默认以 Fast-Forward 模式合并
开启后，合并请求时默认以 Fast-Forward 模式合并，但不强制，如不需要以 Fast-Forward 模式，合并者取消勾选即可。

合并请求源分支有新提交时自动取消合并授权
开启后，当合并请求的源分支有新的代码推送后，原合并授权会自动取消。

开启状态检查
开启状态检查后，所有合并到此分支的合并请求必须在全部 CI 任务都通过之后才能合并。

合并前必须获得所有评审者的允许合并
所有评审者是指合并请求评审者列表里的所有用户，如果该评审者是强制评审者，则无法从评审者列表中移除。

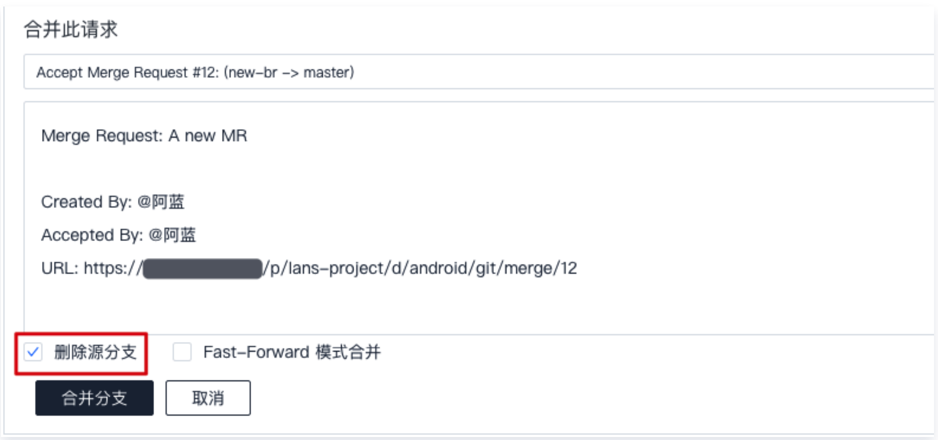
合并方式选择
[✓ 默认直接合并] 默认 Squash 合并 只能 Squash 合并

合并请求默认目标分支
创建合并请求时，目标分支会默认选择此分支
master

Basic settings

Delete Source Branch by Default

Deleting the source branch helps keep the repository clean. Once this switch is enabled, the MR request will auto-select the option shown in the image. If you do not want to delete the source branch, the merger can uncheck this button.



合并此请求

Accept Merge Request #12: (new-br -> master)

Merge Request: A new MR

Created By: @阿蓝
Accepted By: @阿蓝
URL: https://[redacted]/p/lans-project/d/android/git/merge/12

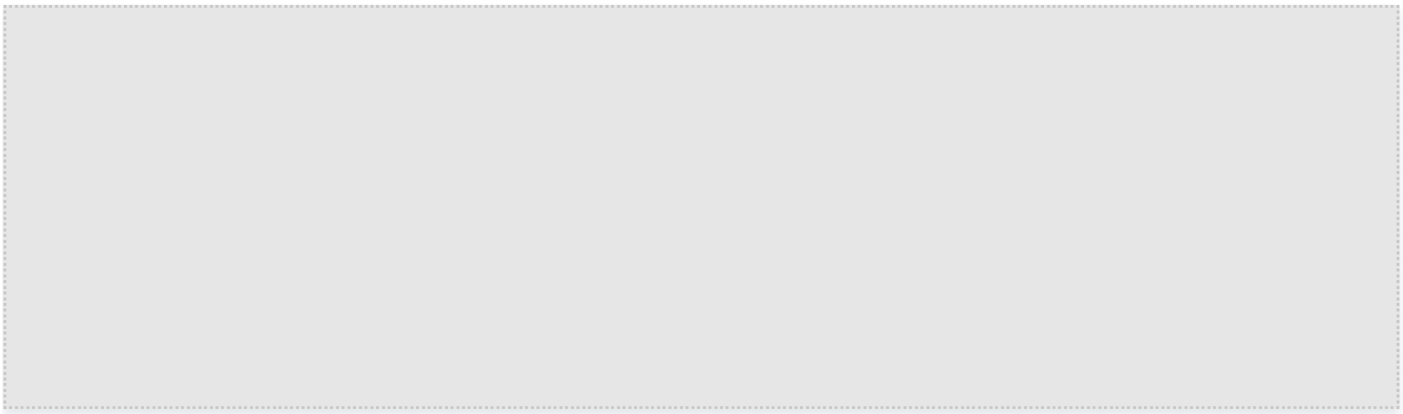
[✓ 删除源分支] Fast-Forward 模式合并

Fast-Forward Merge by Default

This feature is mainly to maintain the cleanliness of the trunk branch. This merge mode will not leave any historical information of the merges in the target branch. The commit messages of the source branch will directly merge into the target branch without creating a commit record of the merge, similar to adding the −−ff parameter when using the git merge command.

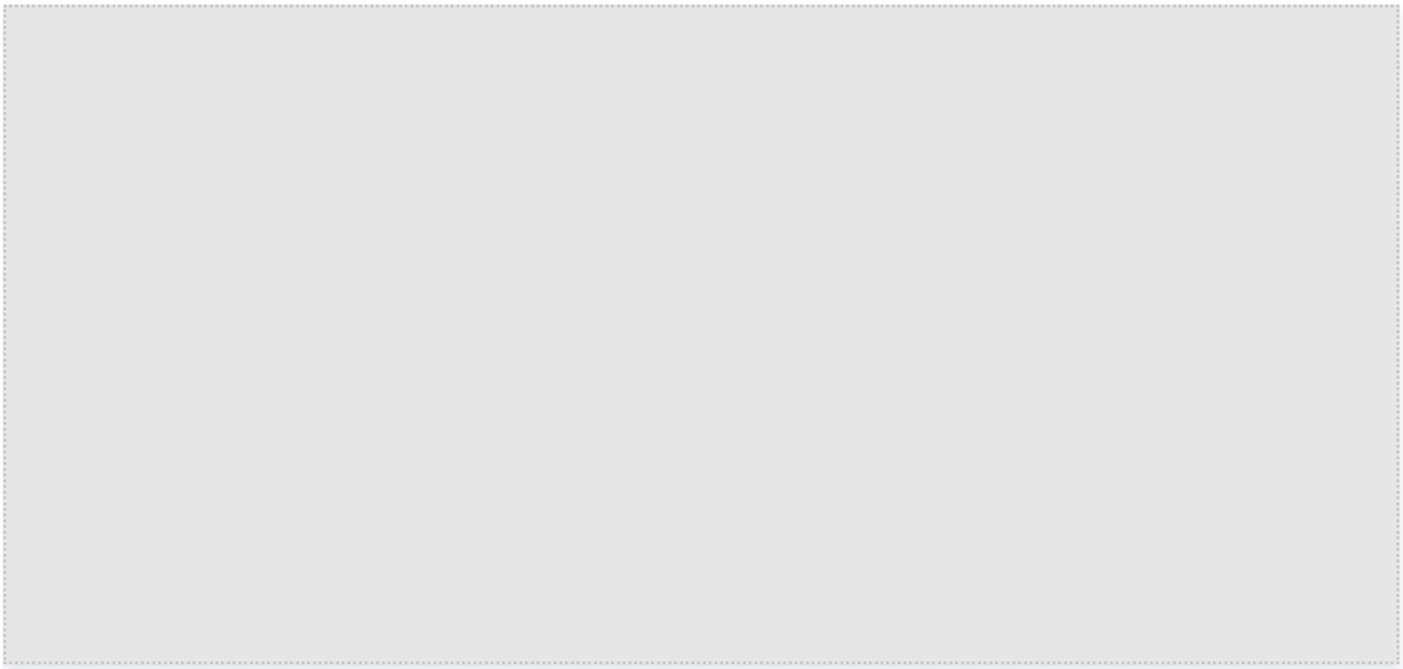
Status Check

Once this feature is enabled, all MRs will automatically trigger a Continuous Integration task on the source branch. Only after passing can they be allowed to merge into the target branch.



All reviewers must allow the merge before merging

This feature is mainly to ensure that all MRs can only be merged with the unanimous consent of all reviewers, regardless of whether the target branch is protected or not.



Merge Mode

Three merge modes are provided when the source branch has multiple commits:

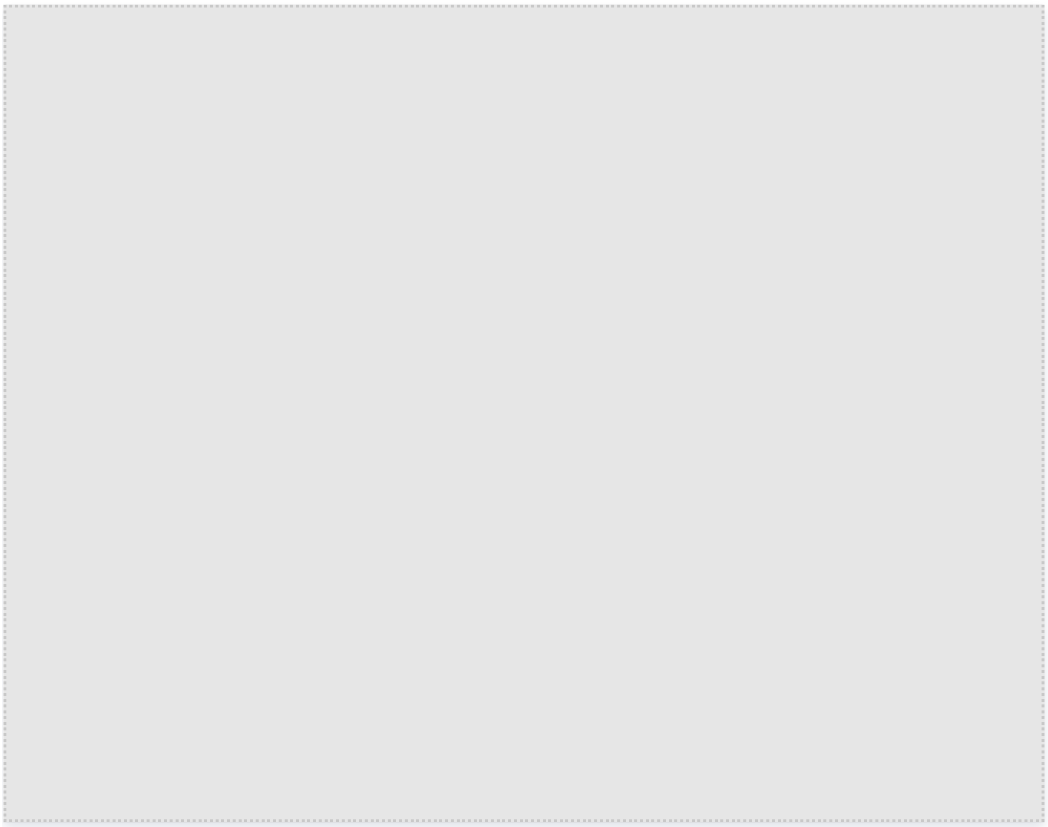
- Default direct merge: A merge commit will be created.
- Default Squash merge: Multiple commits from the source branch will be squashed into one commit, and users can opt out of this behavior.
- Squash merge only: Forces the squashing of multiple commits from the source branch into one commit, and users cannot opt out.

Default Target Branch

After the default branch is specified, it will be automatically set as the target branch when creating merge requests. It is recommended to use the trunk branch as the default target branch for merge requests.

Merge Commit Message Template

All merge requests will leave a commit message in the target branch. For example, the default message in Git is 'Accept Merge Request #xxx.' You can modify this message template here to record the merge results more clearly.



Submit Message Variable

Variable name	Variable Description	Variable Example
${source_branch }	Source Branch	feature/demo
${target_branch}	Target Branch	main
${title}	Merge Request Title	This is a merge request title
${id}	Merge Request Reference Issue Number	#1
${url}	Merge Request Access Link	https://team-name.coding.net/p/project-name/d/depot-name/git/merge/21
${reviewer}	Merge Request Reviewer	@DevelopmentTeamLead
${approver}	Merge Request Allowed to Merge Personnel	@DevelopmentTeamLead
${acceptor}	Merge into Branch Personnel	@Development Team Member

${creator}	Merge Request Creator	@Development Team Member

Initiate Merge Request

Last updated：2024-09-05 16:16:59

If a multi-branch development workflow is used, we recommend you set the master branch as a protected branch. Developers can create temporary develop branches and initiate merge requests for them. After continuous integration (CI) and code reviews, developers can merge the develop branch into the master branch.

Initiate Merge Request

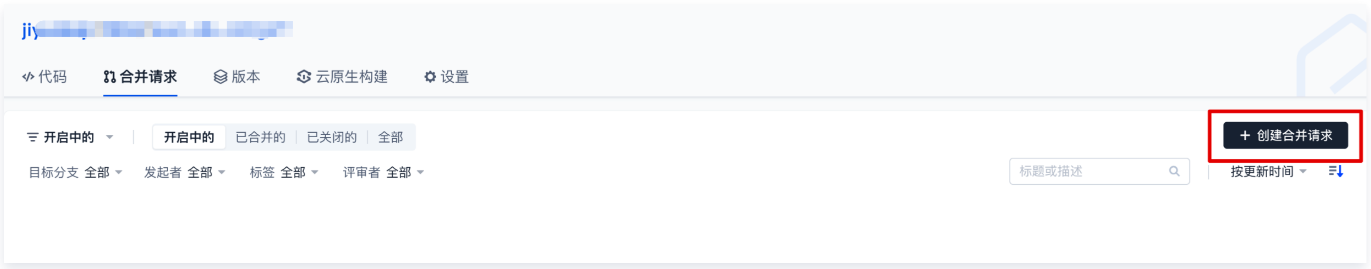
You can manually create merge requests on the command line or on the web.

Create via Command Line

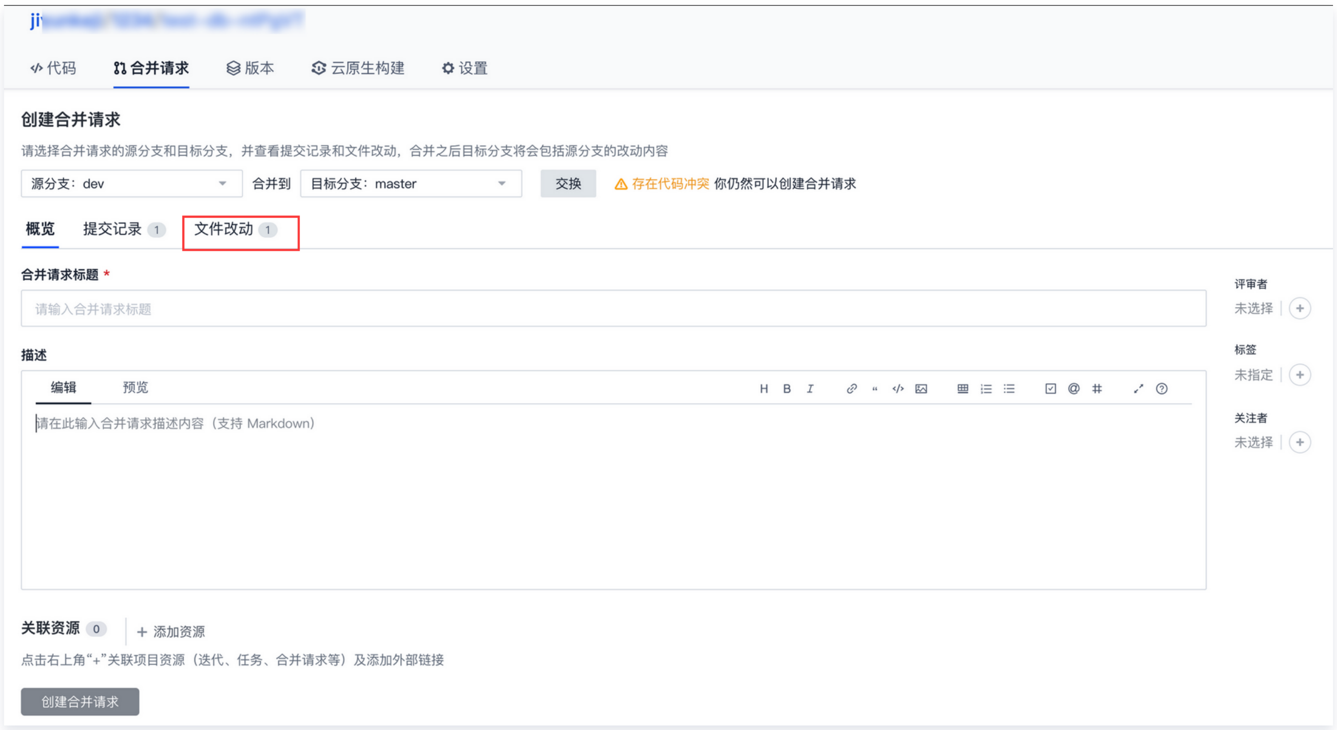
```
git push origin local-branch:mr/target-branch/local-branch
```

Manual Creation

1. On the details page of a code repository, go to the **Merge Requests** tab and then click **Create Merge Request**.



2. Specify a source branch and target branch for the merge request. If no conflicts are found after the source branch is compared with the target branch, the system will indicate that the source branch can be merged into the target branch. You can view the differences between the files on the **File Changes** tab.



创建合并请求

请选择合并请求的源分支和目标分支，并查看提交记录和文件改动，合并之后目标分支将会包括源分支的改动内容

源分支：dev 合并到 目标分支：master 交换 存在代码冲突 你仍然可以创建合并请求

合并请求标题 *

请输入合并请求标题

描述

请在此输入合并请求描述内容（支持 Markdown）

评审者 未选择

标签 未指定

关注者 未选择

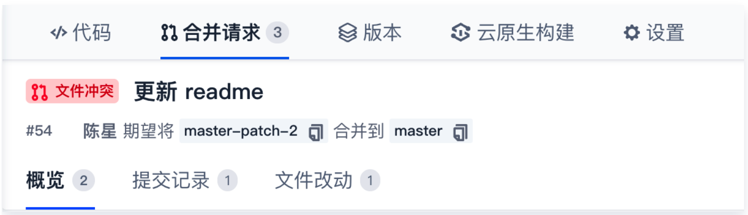
关联资源 0 + 添加资源

点击右上角“+”关联项目资源（迭代、任务、合并请求等）及添加外部链接

3. After creating a merge request, you can still view all the commit history and all individual commits included in a single push in the **Commit History** tab.

Resolve Merge Conflicts

The reason for merge conflicts is that the source branch and the target branch have made inconsistent changes to the same code. When a merge request is initiated, the system will automatically compare the differences between the source and target branches. If a conflict is found, the system will indicate that they cannot be automatically merged. You can resolve the conflict either online or locally before continuing to initiate the merge request.


代码 合并请求 3 版本 云原生构建 设置

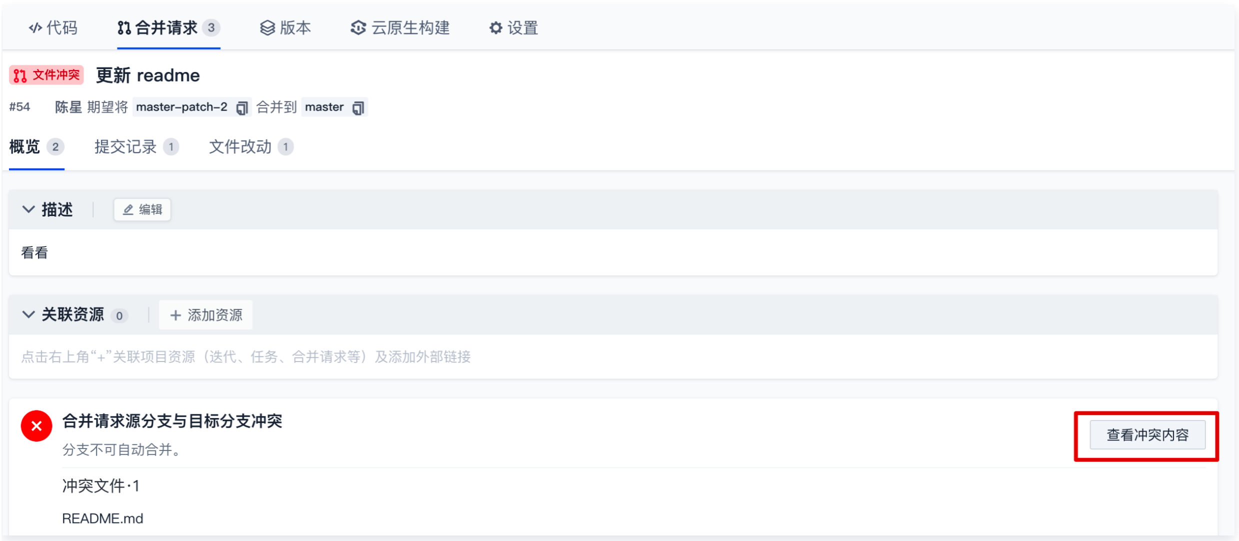
文件冲突 更新 readme

#54 陈星 期望将 master-patch-2 合并到 master

概览 2 提交记录 1 文件改动 1

Online Conflict Resolution

When encountering code conflicts, it is often necessary to repeatedly pull and resolve them in the local terminal before pushing them to the remote repository. The online conflict resolution feature simplifies these tedious steps to simple mouse operations. Click **View Conflict Content** on the webpage to preview the conflicting content online.


代码 合并请求 3 版本 云原生构建 设置

文件冲突 更新 readme

#54 陈星 期望将 master-patch-2 合并到 master

概览 2 提交记录 1 文件改动 1

描述 | 编辑

看看

关联资源 0 | + 添加资源

点击右上角"+"关联项目资源（迭代、任务、合并请求等）及添加外部链接

合并请求源分支与目标分支冲突

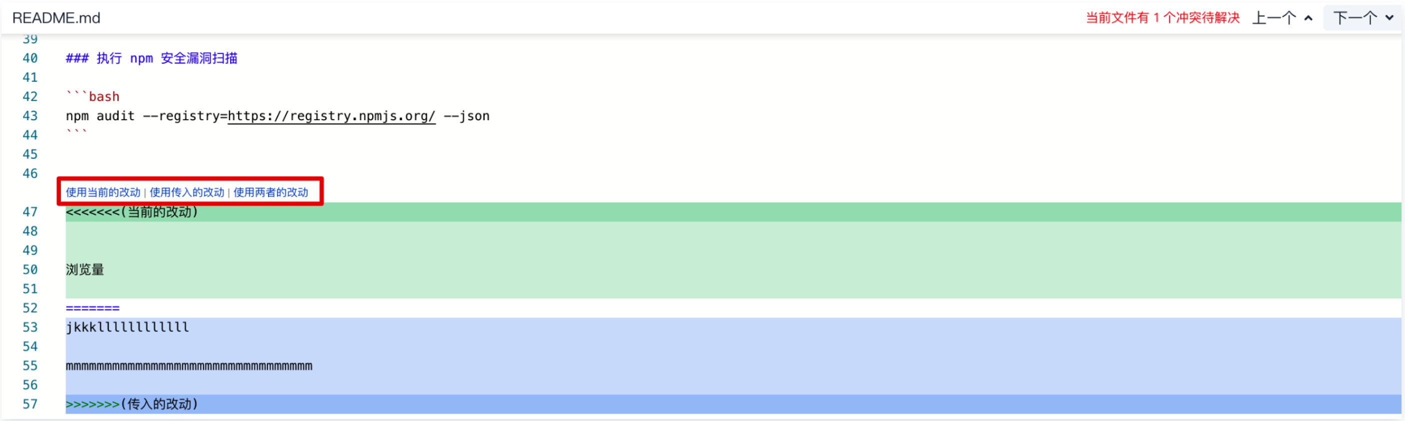
分支不可自动合并。

查看冲突内容

冲突文件·1

README.md

The conflicting content will be highlighted. You can select the content you want to keep. Click the toggle button in the upper-right corner to quickly switch to other conflicting lines. After resolving all conflicts and submitting the changes, a new commit record will be generated in the code repository.


README.md
当前文件有 1 个冲突待解决
上一个
下一个
使用当前的改动 | 使用传入的改动 | 使用两者的改动

~~~
### 执行 npm 安全漏洞扫描

```bash
npm audit --registry=https://registry.npmjs.org/ --json
```


<<<<<<<(当前的改动)


浏览量

=======
jkkkllllllllllll

mmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmm

>>>>>>>(传入的改动)
~~~

Local Conflict Resolution

For example, if a conflict is found when merging branch-01 into master, you can switch to the master branch locally and run the command:

```
git merge branch-01
```

Find the file with conflicts. The conflicts will be highlighted in the file. You will be prompted to select which content to keep. Select the content you want to keep and save the file before committing it again, and then switch to the branch-01 branch and enter the command:

```
git merge master
```

Then, push the modified code to the remote repository.

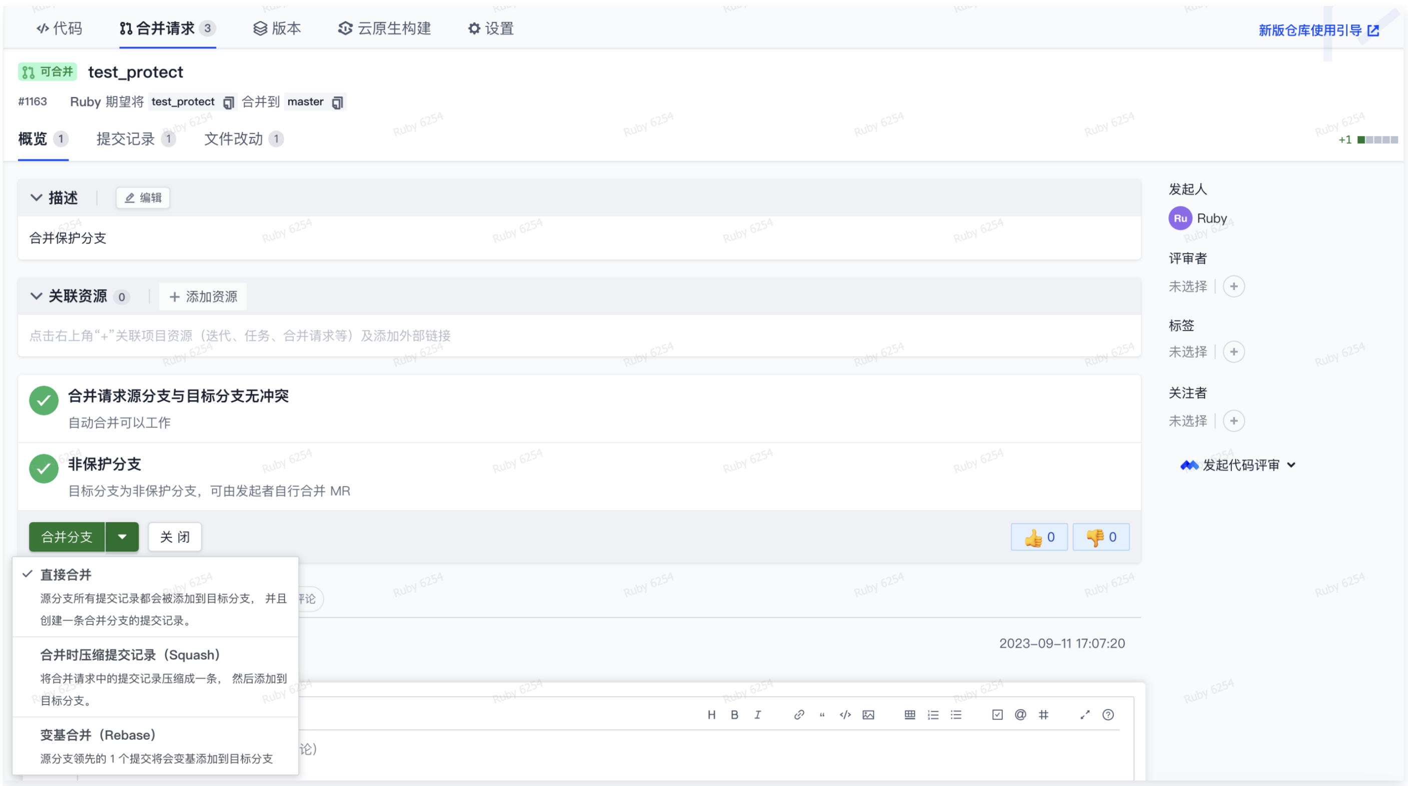
Confirm Merge Request

- **The target branch of the merge is a protected branch**
  If the initiator of the merge request is a branch admin, they can perform the merge on their own. If the initiator is a regular member, the merge can be completed only after it has passed a review by a branch admin.
- **The target branch of the merge is not a protected branch**
  The initiator can initiate and complete a branch merge without review or authorization.

Choose Merge Type

On the **Merge Request** page, click **Merge Branch** and select the merge type from the dropdown list. The system will process the merge according to the selected type.

- **Direct Merge**: All commits from the source branch will be added to the target branch and a merge commit will be created.
- **Squash Merge**: Compress all commits in the merge request into a single commit before adding it to the target branch.
- **Rebase Merge**: Rebase the leading commits of the source branch onto the target branch.



代码 合并请求 3 版本 云原生构建 设置 新版仓库使用引导

可合并 test_protect

#1163 Ruby 期望将 test_protect 合并到 master

概览 1 提交记录 1 文件改动 1

描述 编辑

合并保护分支

关联资源 0 + 添加资源

点击右上角"+"关联项目资源（迭代、任务、合并请求等）及添加外部链接

合并请求源分支与目标分支无冲突
自动合并可以工作

非保护分支
目标分支为非保护分支，可由发起者自行合并 MR

合并分支 关 闭 👍 0 👎 0

✓ 直接合并
源分支所有提交记录都会被添加到目标分支，并且创建一条合并分支的提交记录。

合并时压缩提交记录（Squash）
将合并请求中的提交记录压缩成一条，然后添加到目标分支。

变基合并（Rebase）
源分支领先的 1 个提交将会变基添加到目标分支

2023-09-11 17:07:20

发起人 Ruby
评审者 未选择
标签 未选择
关注者 未选择
发起代码评审

Delete source branch

When initiating a merge request, select Delete Source Branch to delete the source branch after the merge request is initiated.

Fast-Forward Merge

During a fast-forward merge, no historical information about the merge between the source and target branches is left in the target branch. The commits from the source branch will be directly combined with the target branch without creating a merge commit.

If Fast-Forward Merge is selected, the remote repository will determine if the fast-forward rules are met during the merge. If the rules are met, this merge will not create a new merge commit record; if this mode is not selected, previous development records will be kept and a new merge record will be created during the merge.
Selecting this option is equivalent to adding the -ff parameter when using git merge.

Note:
In Merge Request Settings, you can set default options for deleting the source branch and using Fast-Forward Merge.

Sign Git Commits with GPG

Last updated：2024-09-05 16:17:13

CODING supports GPG signature verification for Git commits. Verified commit records will be tagged as "Verified" to ensure the reliability of code submitters and enhance code security.

Generate GPG Key Pair

1. Download and install GPG. If you are using macOS, you can run the following command using the brew package manager:

   ```
   brew install gpg
   ```

2. Run the following command to generate a GPG key pair (public/private key):

   ```
   gpg --full-gen-key
   ```

   In some scenarios, such as using Windows Gpg4win or other macOS versions, use the `gpg --gen-key` command to generate a key pair.
   This is an interactive command. You will need to select the algorithm type, specify the validity period of the key, enter your real name and email, set a password, etc., based on the prompts.
   - **Key Type**: Select the key type to use or press Enter to choose the default RSA and RSA.
   - **Elliptic Curve**: Press Enter to select the default elliptic curve `Curve 25519`.
   - **Validity period**: Specify the validity period of the key as needed, or press Enter to select the default never expires.
   - **Email address**: This should be the email address configured in your CODING account.
3. Run the following commands to list the created GPG keys (the email address in the command should be the one specified in step 2):

   ```
   gpg --list-secret-keys --keyid-format LONG "your_email"
   ```

4. Copy the GPG key ID that starts with `sec`. For example, copy `4AEA00A342C24CA3` in the example:

   ```
   sec   ed25519/4AEA00A342C24CA3 2021-09-14 [SC]
         6DE3507E82DEB6E8828FAAC34AEA00A342C24BD4
   uid                 [ ultimate ] your_name "your_email"
   ssb   cv25519/812B586FD245B560 2021-09-14 [E]
   ```

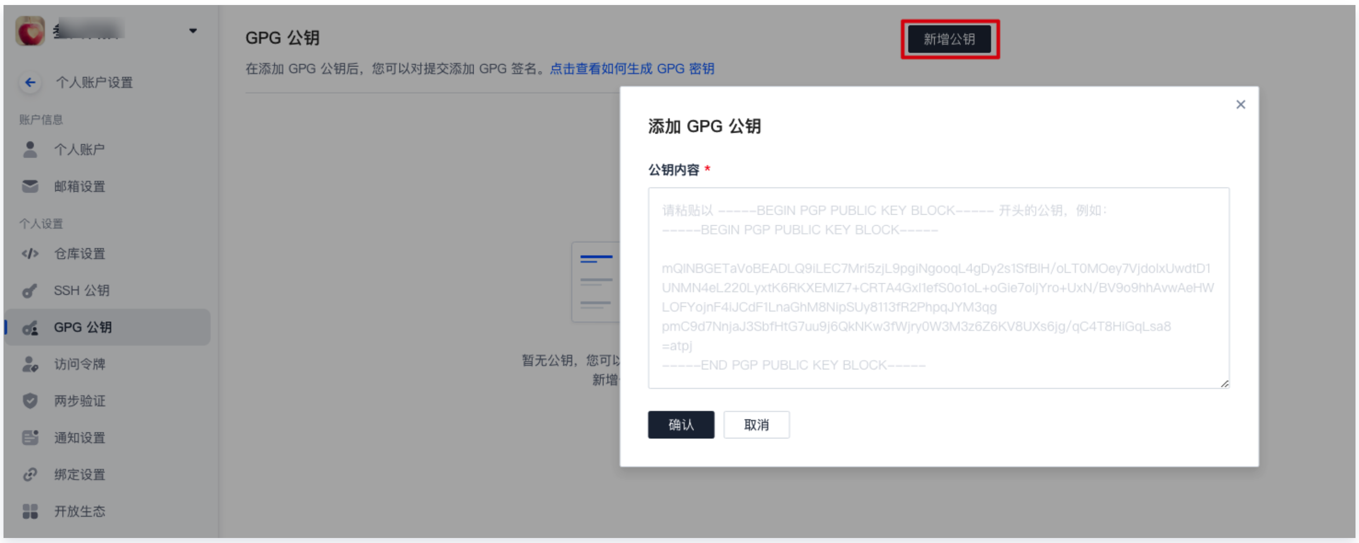
5. Use the copied ID to export the public key (using the ID mentioned above as an example):

   ```
   gpg --armor --export 4AEA00A342C24CA3
   ```

After generating the public key, you can add it to your CODING account.

Add Public Key to Personal Account Settings

1. After logging in to CODING, click the **Personal Account Settings** option in the lower left corner of the page.
2. Select **GPG Public Key** from the left sidebar to enter the Public Key Management page.
3. Click **Add Public Key**, paste the exported GPG Public Key into the content box, and confirm.



GPG 公钥

在添加 GPG 公钥后，您可以对提交添加 GPG 签名。点击查看如何生成 GPG 密钥

添加 GPG 公钥

公钥内容 *

```
请粘贴以 -----BEGIN PGP PUBLIC KEY BLOCK----- 开头的公钥，例如：
-----BEGIN PGP PUBLIC KEY BLOCK-----

mQINBGETaVoBEADLQ9iLEC7Mri5zjL9pgiNgooqL4gDy2s1SfBIH/oLT0MOey7VjdolxUwdtD1
UNMN4eL220LyxtK6RKXEMIZ7+CRTA4GxI1efS0o1oL+oGie7olJYro+UxN/BV9o9hhAvwAeHW
LOFYojnF4lJCdF1LnaGhM8NipSUy8113fR2PhpqJYM3qg
pmC9d7NnjaJ3SbfHtG7uu9j6QkNKw3fWjry0W3M3z6Z6KV8UXs6jg/qC4T8HiGqLsa8
=atpj
-----END PGP PUBLIC KEY BLOCK-----
```

After the public key is successfully added, the verification status of the email address, key ID, and subkey will be displayed.

Note:
If the email address shows an unverified status, it means the email is not configured in the CODING account. Please add the email in **Personal Account Settings > Email Settings**.

Associate with the local Git repository

1. Run the following commands to list the created GPG keys (the email address in the command should be the one specified when generating the key):

```
gpg --list-secret-keys --keyid-format LONG "your_email"
```

2. Copy the GPG key ID that starts with `sec`. In the example below, copy `4AEA00A342C24CA3`:

```
sec   ed25519/4AEA00A342C24CA3 2021-09-14 [SC]
      6DE3507E82DEB6E8828FAAC34AEA00A342C24BD4
uid                 [ ultimate ] your_name "your_email"
ssb   cv25519/812B586FD245B560 2021-09-14 [E]
```

3. Configure the key in the local Git repository to sign commit submissions:

```
git config --global user.signingkey 4AEA00A342C24CA3
```

You have now successfully associated the created GPG key with the local Git repository. Sign the Git commit message when making local changes to verify the authenticity of the submitter.

Sign Git commits

When running the Git commit command, the -S parameter is required.

1. When you have finished editing the code locally and need to submit changes, add the -S parameter to the git commit command:

   ```
   git commit -S -m "your_commit_message"
   ```

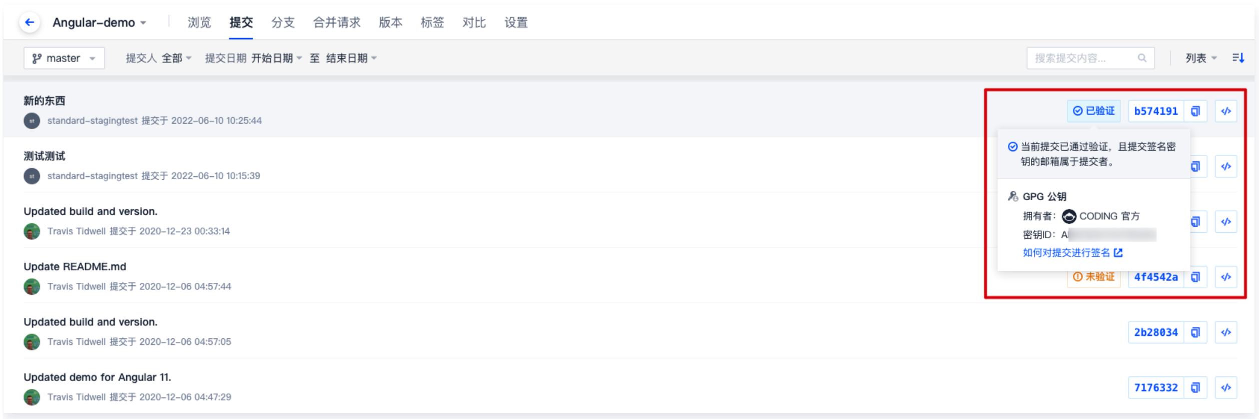
   If you do not want to enter the -S flag every time, you can use the following command to set Git to automatically sign commits:

   ```
   git config --global commit.gpgsign true
   ```

2. If prompted to enter a password, provide the password set when generating the GPG key.

Signature verification

After pushing the signed commit to the CODING code repository, you can check the commit verification on the repository's **commit** page to see if the signature was successful.



新的东西
standard-stagingtest 提交于 2022-06-10 10:25:44
b574191

当前提交已通过验证，且提交签名密钥的邮箱属于提交者。

GPG 公钥
拥有者： CODING 官方
密钥ID：A[redacted]

测试测试
standard-stagingtest 提交于 2022-06-10 10:15:39

Updated build and version.
Travis Tidwell 提交于 2020-12-23 00:33:14

Update README.md
Travis Tidwell 提交于 2020-12-06 04:57:44
4f4542a

Updated build and version.
Travis Tidwell 提交于 2020-12-06 04:57:05
2b28034

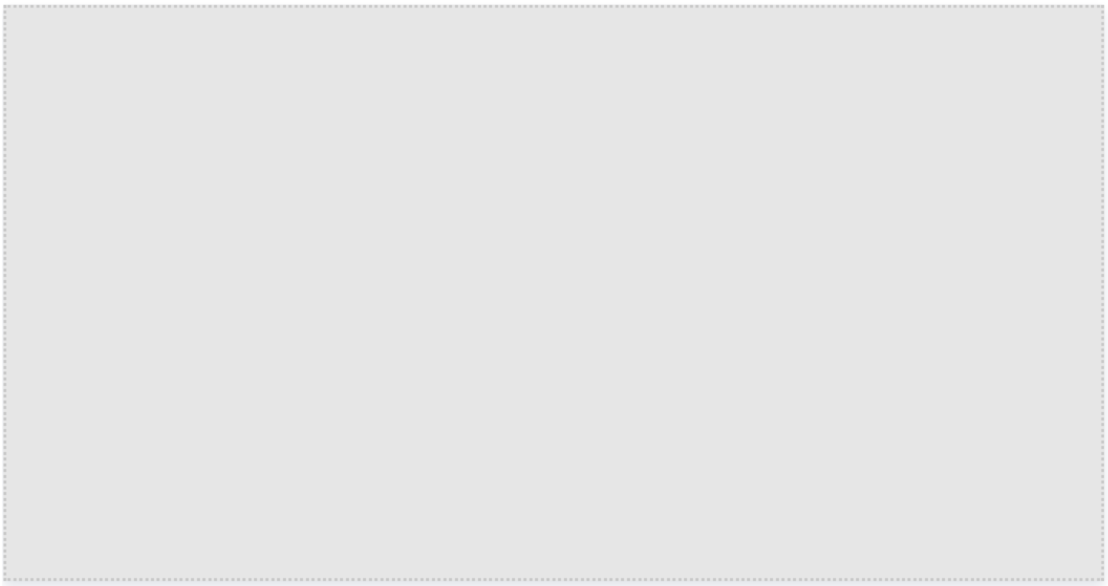
Updated demo for Angular 11.
Travis Tidwell 提交于 2020-12-06 04:47:29
7176332

The explanations for the commit verification statuses are as follows:

Verification status	Description
Verified	Signed with a GPG private key, the corresponding public key is in the CODING account, and the public key email is verified.
Unverified	Signed with a GPG private key, but the corresponding public key is not in the CODING account, or the public key email is not verified (if an unverified email appears, please add the email in **Personal Account Settings** > **Email Settings**).
No verification status Tag	Did not use a GPG private key for the signature.

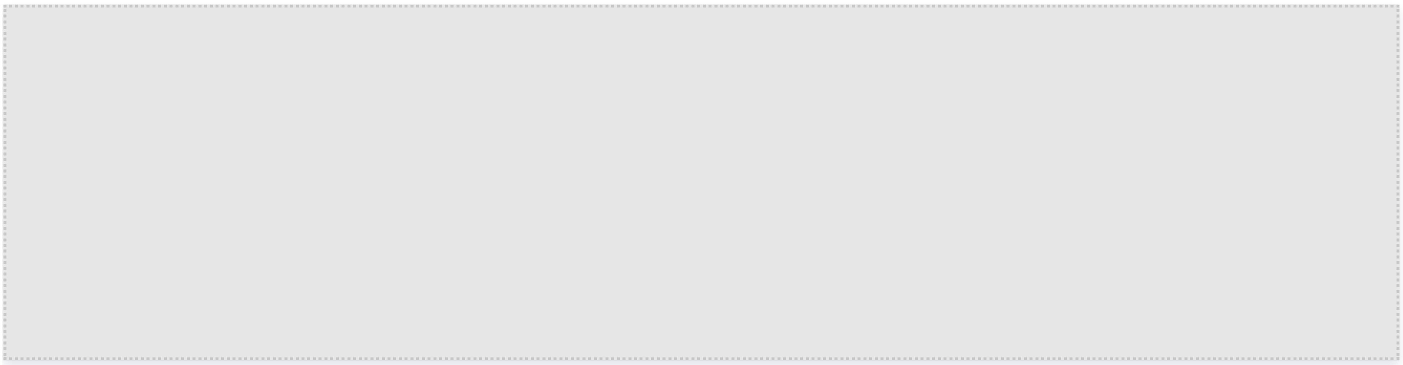
Delete GPG Public Key

If your GPG public key is at risk of being compromised or you no longer use GPG signatures, you can delete the public key in **Personal Account Settings** > **GPG Public Key**.



After the public key is deleted:

- Verified commits will become unverified.
- Commits still signed with the GPG private key (i.e., using `git commit -S -m`) will become unverified.
- Unsigned commits (i.e., using `git commit -m`) will not be verified and will have an unverified status tag.



Note:
If Git automatic signing is configured, you can run the command `git config --global commit.gpgsign false` to disable automatic signing. Otherwise, after deleting the GPG public key, commits pushed to the remote repository will still show an 'Unverified' status.

GPG signature error handling

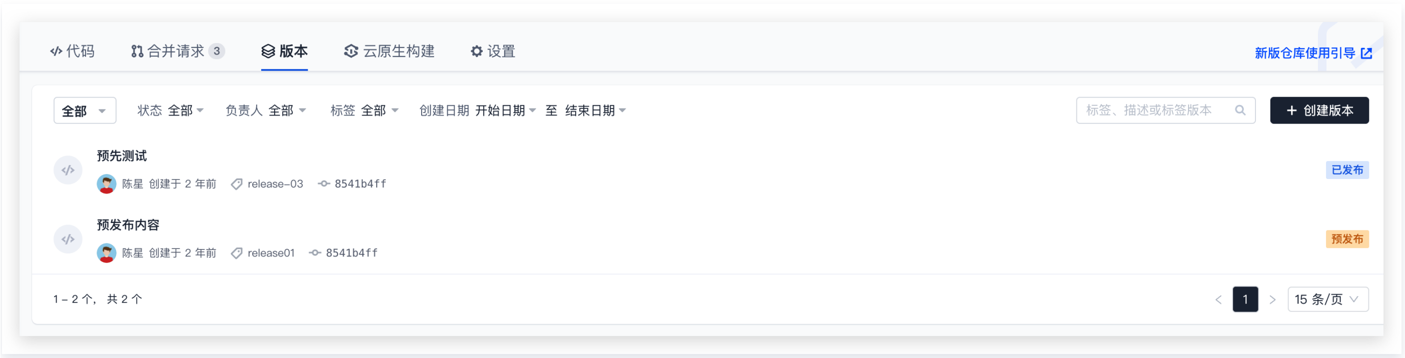
If, after completing all the steps mentioned above, you encounter the following error when signing commits using `git commit -S -m`, you can refer to Solving GPG Signature Failures to modify the related configurations.

```
error: gpg failed to sign the data
fatal: failed to write commit object
```

Versions and Tags
Code Version

Last updated：2024-09-05 16:17:27

A code version can be understood as a snapshot of the code repository at a certain point in time. In the code repository management list, click the specified code repository to enter its detail page, and then click **Versions** to enter the Management Release Page.



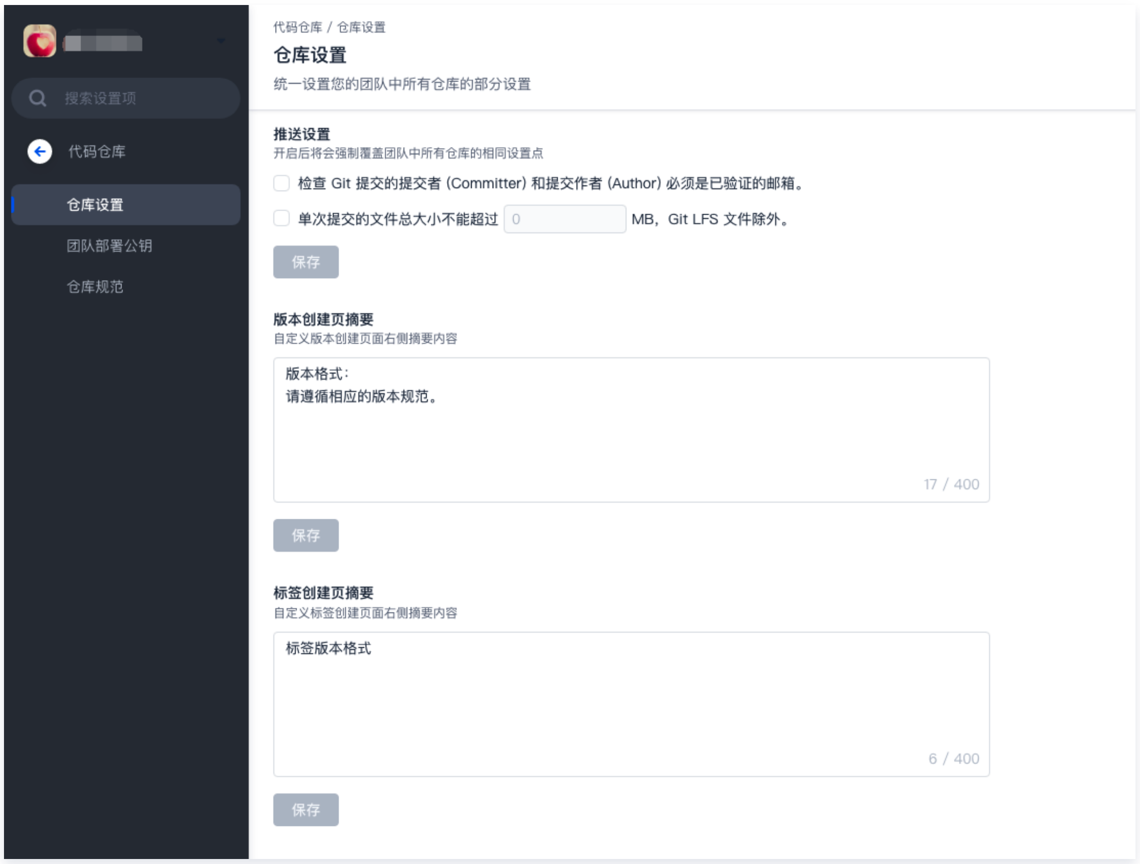
预先测试
陈星 创建于 2 年前 release-03 8541b4ff
已发布

预发布内容
陈星 创建于 2 年前 release01 8541b4ff
预发布

The version release list displays code versions released in a project with their tag names and committed versions by descending order of creation time.

Create Code Version

1. On the version management page, click **Create Version** in the upper-right corner.
2. Enter a tag version, release title, and version description. You can upload files smaller than 100 MB and associate resources in the project (such as tasks, files, Wiki pages, and merge requests). Only new tag names are allowed and you need to select a source for new tag versions (branch, tag, or revision number).
3. Abstract content supports modification by definition and can be used as a team bulletin to inform how to fill in a compliant version number. The team leader or admin can go to **Team Settings Center > Feature Settings > Repository Settings** to define the abstract content.



代码仓库 / 仓库设置

仓库设置

统一设置您的团队中所有仓库的部分设置

推送设置

开启后将会强制覆盖团队中所有仓库的相同设置点

检查 Git 提交的提交者 (Committer) 和提交作者 (Author) 必须是已验证的邮箱。

单次提交的文件总大小不能超过 0 MB，Git LFS 文件除外。

版本创建页摘要

自定义版本创建页面右侧摘要内容

版本格式：
请遵循相应的版本规范。

17 / 400

标签创建页摘要

自定义标签创建页面右侧摘要内容

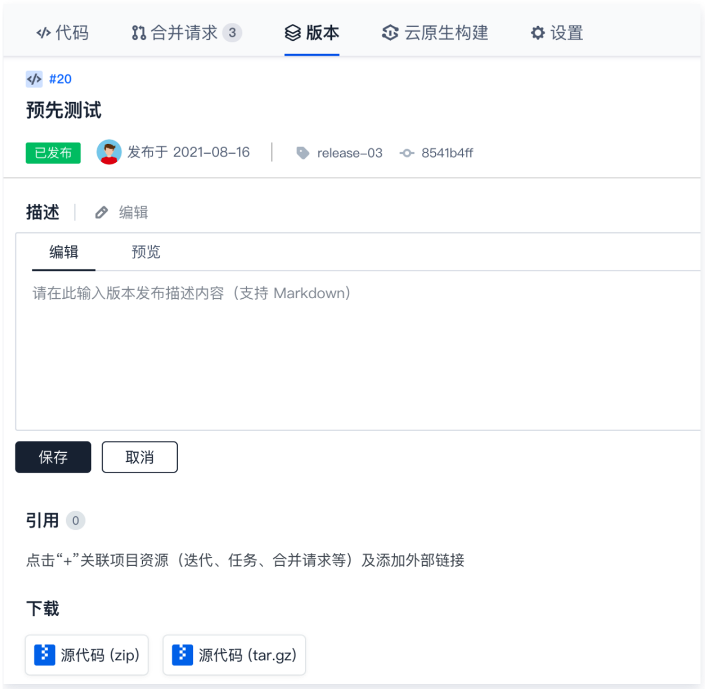
标签版本格式

6 / 400

4. After entering the above information, you can mark the version as a pre-release, and then create the code version release.

Edit Code Version

In the Version Release List, click any code version to enter its detail page, then the version release creator or project admin can click **Edit Version Description** to edit the version information.



#20

预先测试

已发布　发布于 2021-08-16　release-03　8541b4ff

描述　编辑

请在此输入版本发布描述内容（支持 Markdown）

引用 0

点击"+"关联项目资源（迭代、任务、合并请求等）及添加外部链接

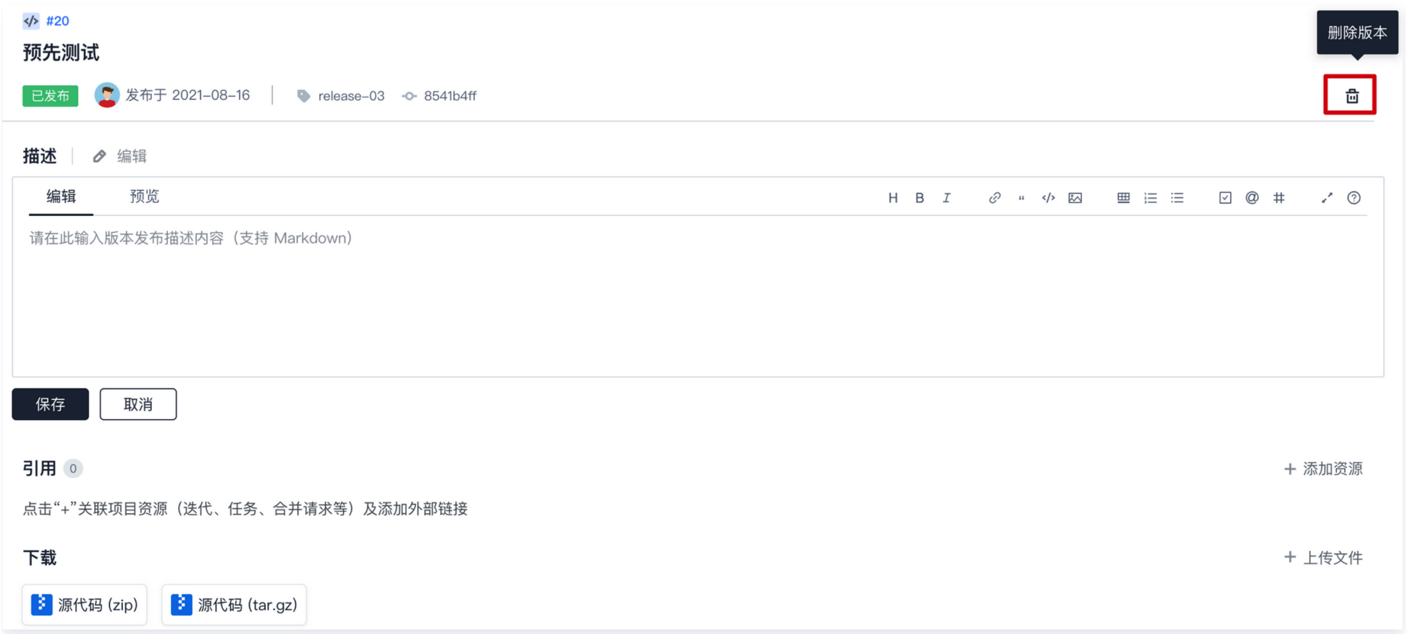
下载

源代码 (zip)　源代码 (tar.gz)

Delete Code Version

On the details page of a code version, the release creator or project admins can delete the version by clicking on the Delete icon.

Note:
You can also delete the linked version tags when deleting a version. You can only use or create a tag with the same name after you have deleted the version tag.


#20
预先测试
已发布 发布于 2021-08-16 release-03 8541b4ff
删除版本
描述 编辑
编辑 预览
请在此输入版本发布描述内容（支持 Markdown）
保存 取消
引用 0 + 添加资源
点击"+"关联项目资源（迭代、任务、合并请求等）及添加外部链接
下载 + 上传文件
源代码 (zip) 源代码 (tar.gz)

Set the Default Release Branch

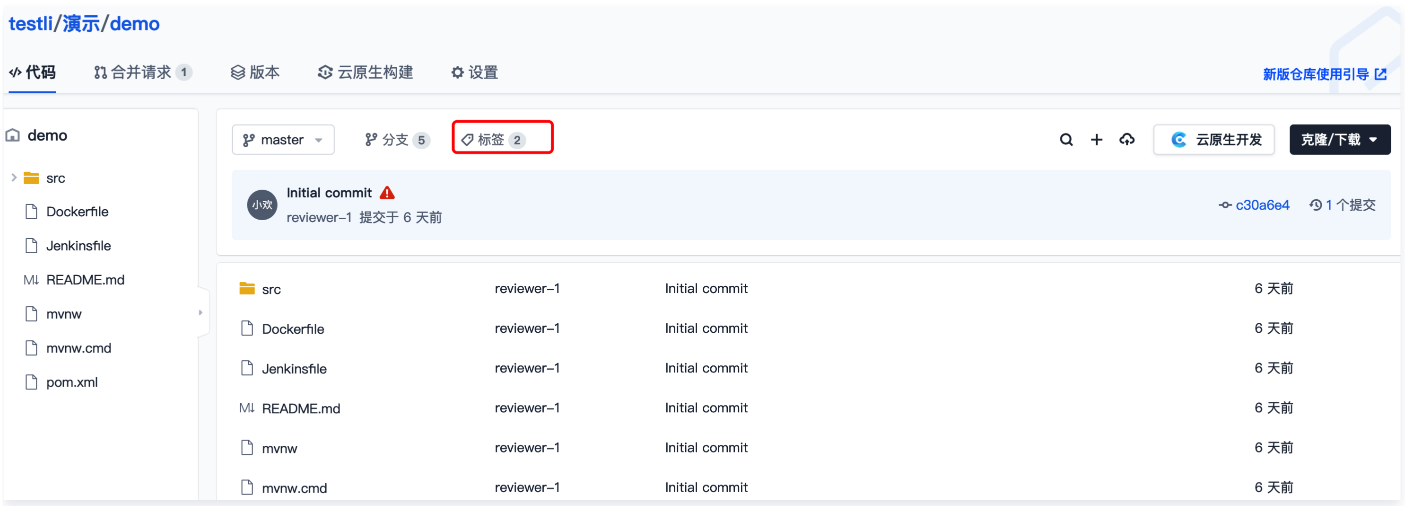
Go to the code repository's **Settings > Version Release** page, where project admins can specify the default branch selected for version releases.

Code Tag

Last updated：2024-09-05 16:18:56

Overview

Log in to CODING DevOps Console > Code Repository. After clicking **Use Now**, continue to click the target code repository in the code repository management list to enter its details page. Then click **Tag** to enter the Code Tag management list.


testli/演示/demo

代码 合并请求 1 版本 云原生构建 设置 新版仓库使用引导

master | 分支 5 | 标签 2 | 云原生开发 | 克隆/下载

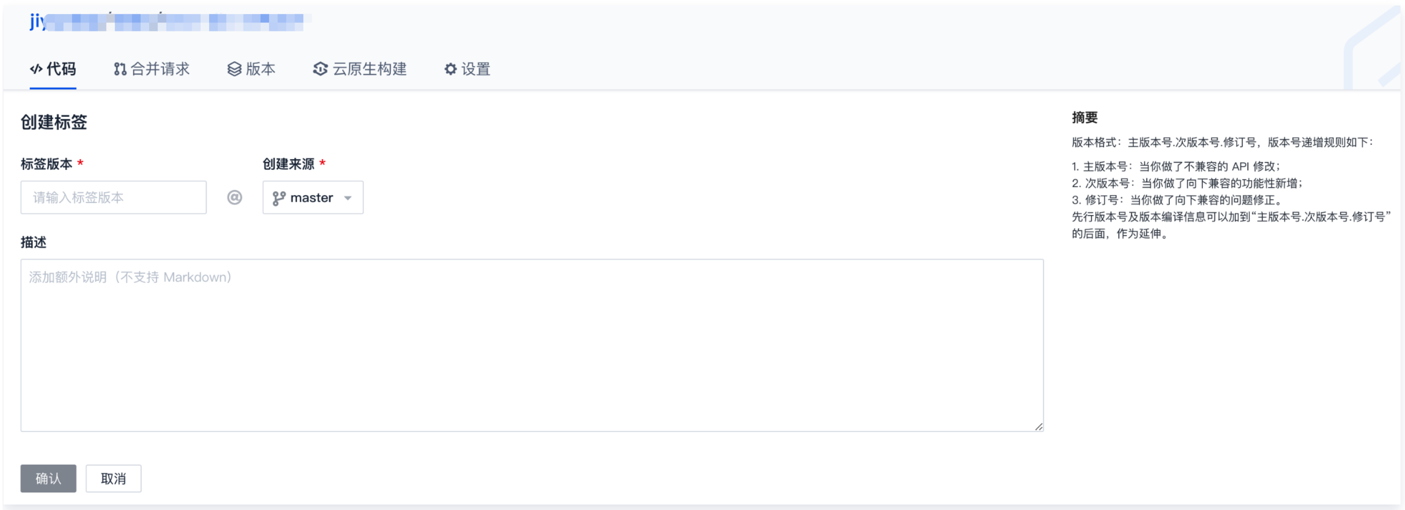
Initial commit
reviewer-1 提交于 6 天前 | c30a6e4 | 1 个提交

src	reviewer-1	Initial commit	6 天前
Dockerfile	reviewer-1	Initial commit	6 天前
Jenkinsfile	reviewer-1	Initial commit	6 天前
README.md	reviewer-1	Initial commit	6 天前
mvnw	reviewer-1	Initial commit	6 天前
mvnw.cmd	reviewer-1	Initial commit	6 天前

The Tag list displays all Tags in the repository in descending order of creation. The Tag list shows the Tag name, Tag description, and the corresponding version, providing download entries for zip and tar.gz files and an option to delete the Tag. Click the Tag name or version number to go to the corresponding code version detail page.

Create New Tag

In the Tag management list, click **Create Tag** in the upper right, enter a Tag name, and select the code version (branch, Tag, revision number) corresponding to the Tag to create a new Tag. The "Summary" displayed on the right side of the page can serve as a bulletin board to inform team members on how to write code Tags in a unified format.


ji[redacted]

代码 合并请求 版本 云原生构建 设置

创建标签

标签版本 *: 请输入标签版本 @ 创建来源 *: master

描述: 添加额外说明（不支持 Markdown）

确认 | 取消

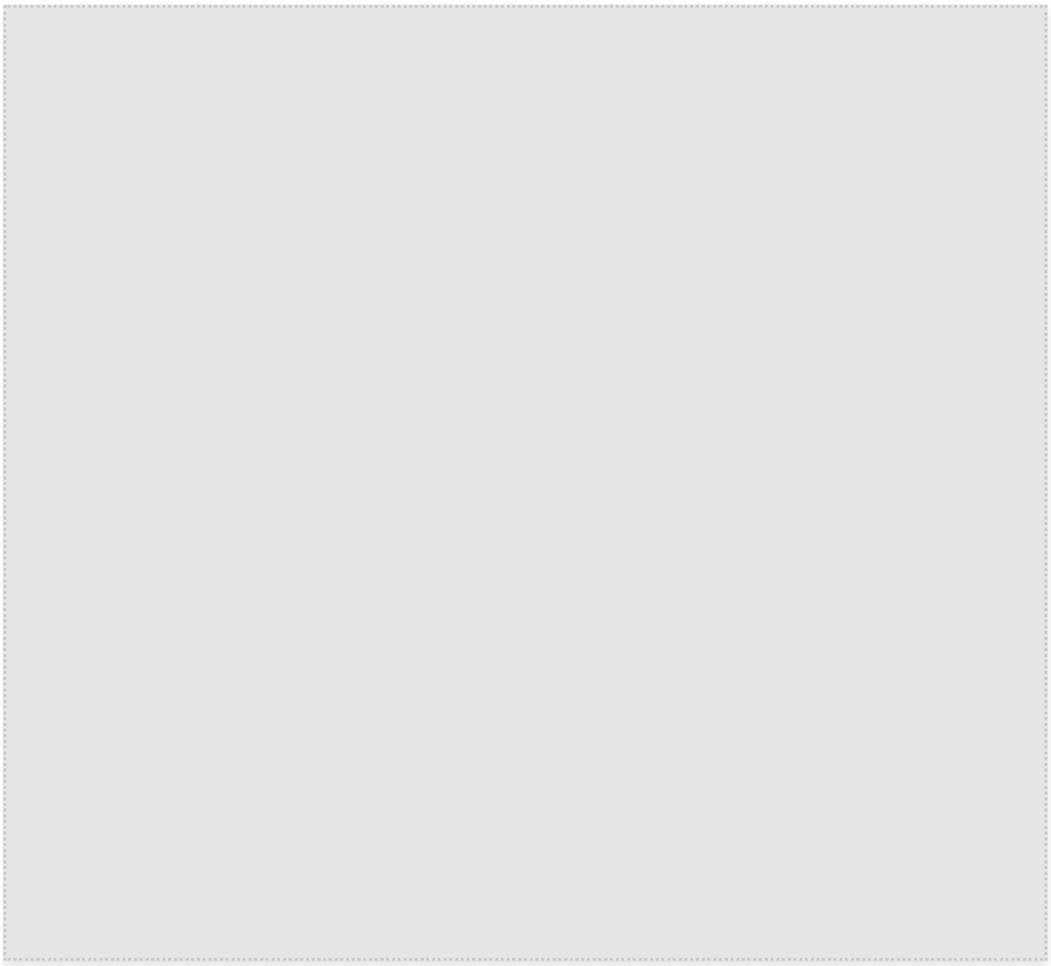
摘要

版本格式：主版本号.次版本号.修订号，版本号递增规则如下：

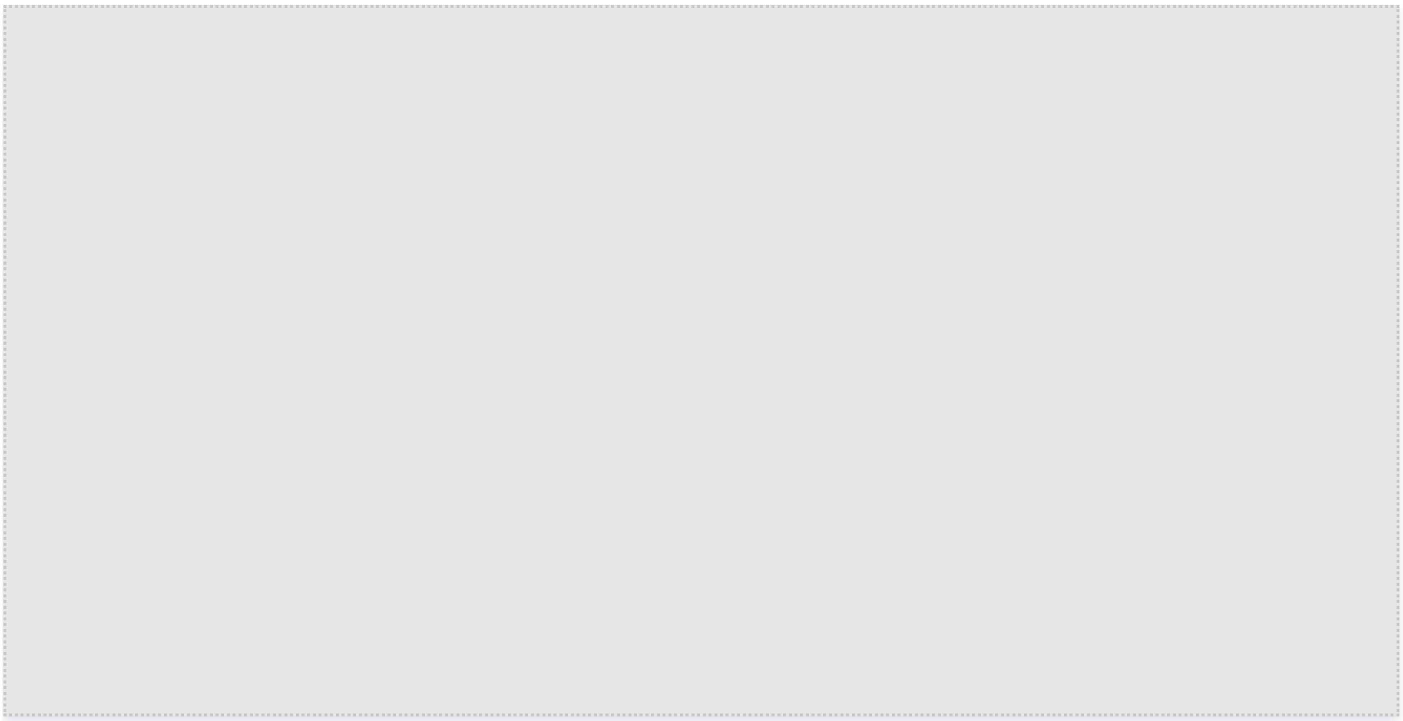
1. 主版本号：当你做了不兼容的 API 修改；
2. 次版本号：当你做了向下兼容的功能性新增；
3. 修订号：当你做了向下兼容的问题修正。

先行版本号及版本编译信息可以加到“主版本号.次版本号.修订号”的后面，作为延伸。

The team leader/administrator can go to **Team Settings Center** > **Feature Settings** > **Code Repository** > **Repository Settings** to customize the content in the Definition summary.

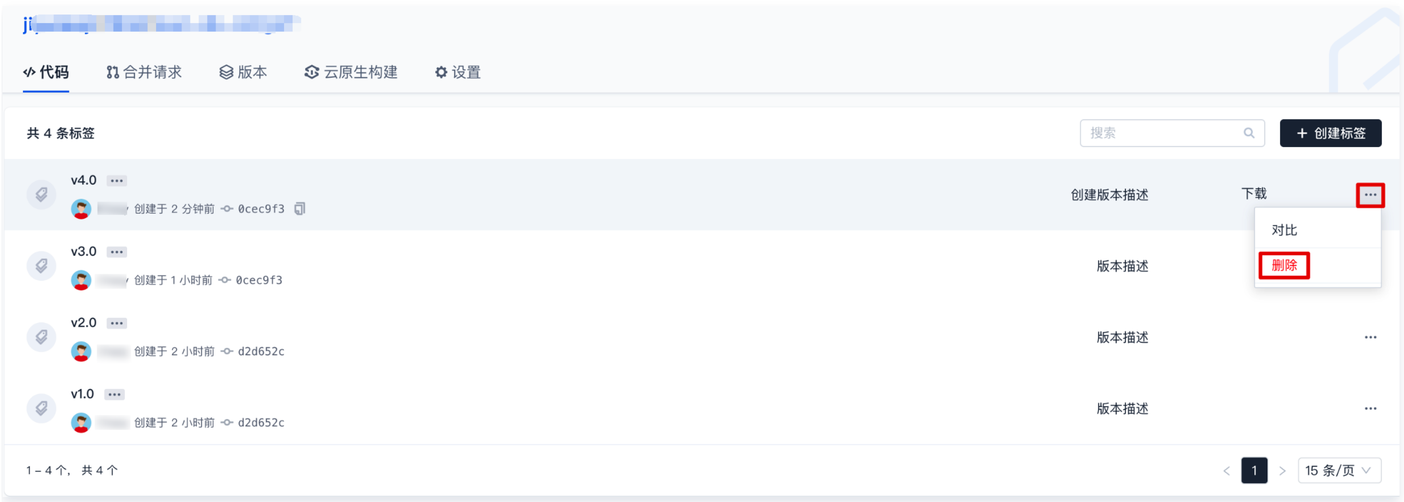


If you want Code Tags to strictly follow team standards, you can go to **Code Repository** > **Settings** > **Code Tag** page and click Set Protected Tags to restrict members from creating or modifying Code Tags of specific formats.



Deleting Tag

On the **Tag** page, the Tag creator or administrator can only delete code Tags that are not associated with any version.

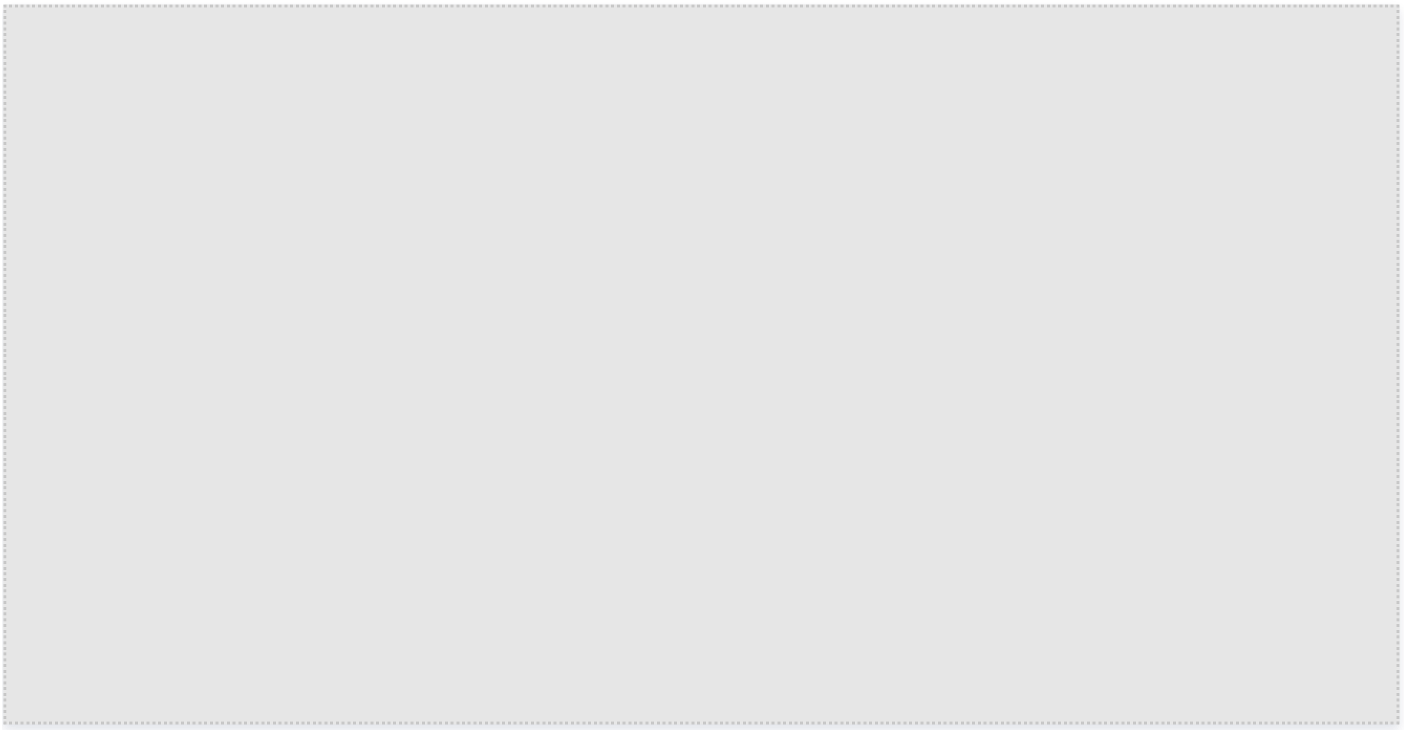

代码　合并请求　版本　云原生构建　设置

共 4 条标签　搜索　+ 创建标签

v4.0 ··· 创建于 2 分钟前 0cec9f3	创建版本描述	下载	···
v3.0 ··· 创建于 1 小时前 0cec9f3	版本描述		对比 删除
v2.0 ··· 创建于 2 小时前 d2d652c	版本描述		···
v1.0 ··· 创建于 2 小时前 d2d652c	版本描述		···

1 – 4 个，共 4 个　1　15 条/页

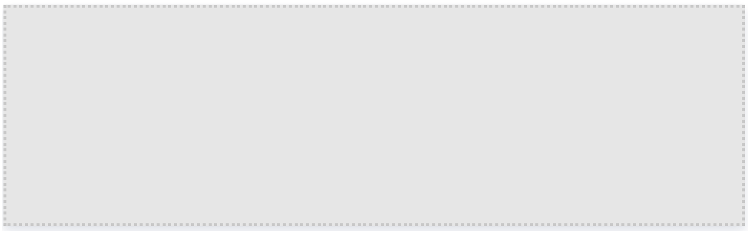
If any Tag is associated with a version release, it can only be deleted by deleting the corresponding version on the **Version** list page. If you do not want the Tag to be deleted, the project administrator can go to **Code Repository > Settings > Code Tag** page and check whether Tag deletion is allowed. If Tag deletion is not allowed, all project members will be unable to delete Tags via the command line, and the Tag deletion feature will not be provided on the webpage.



Set Protected Tags

Protected Tags are mainly used to standardize the creation, update, or deletion of Tags by specific members. After enabling protected Tags, a Tag administrator must be set, and only the administrator is allowed to create Tags that match the Tag rules. When the *-release branch protection rule is set, non-admin users will receive the following prompt when pushing Tag xxx-release via Git:

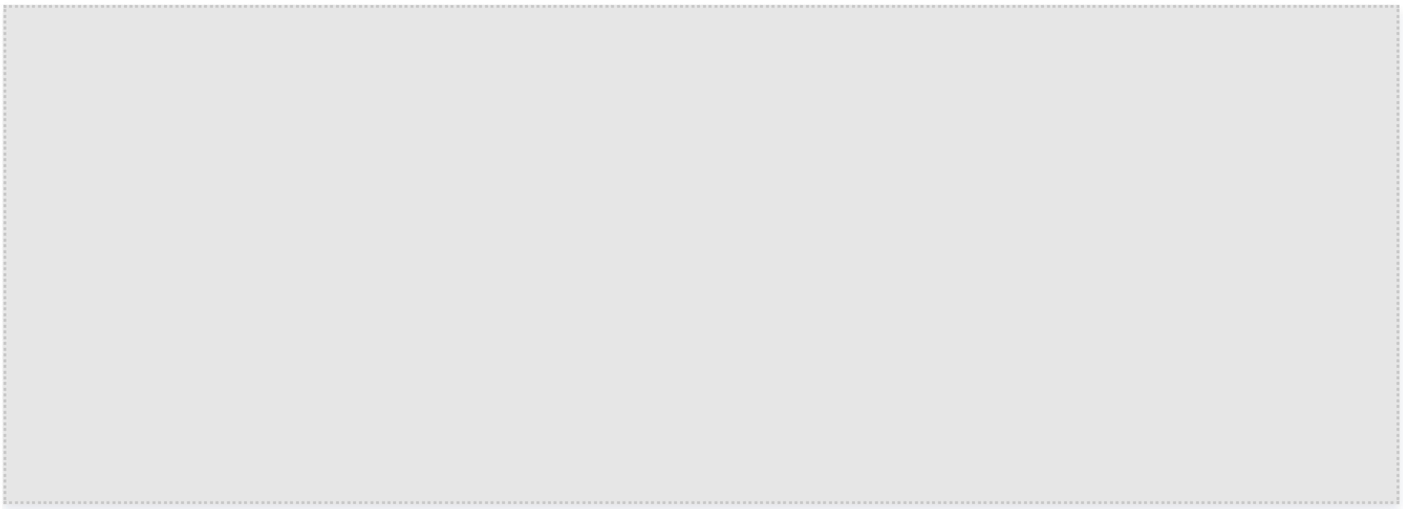
They will also fail to create Tags or new versions on the Web end.



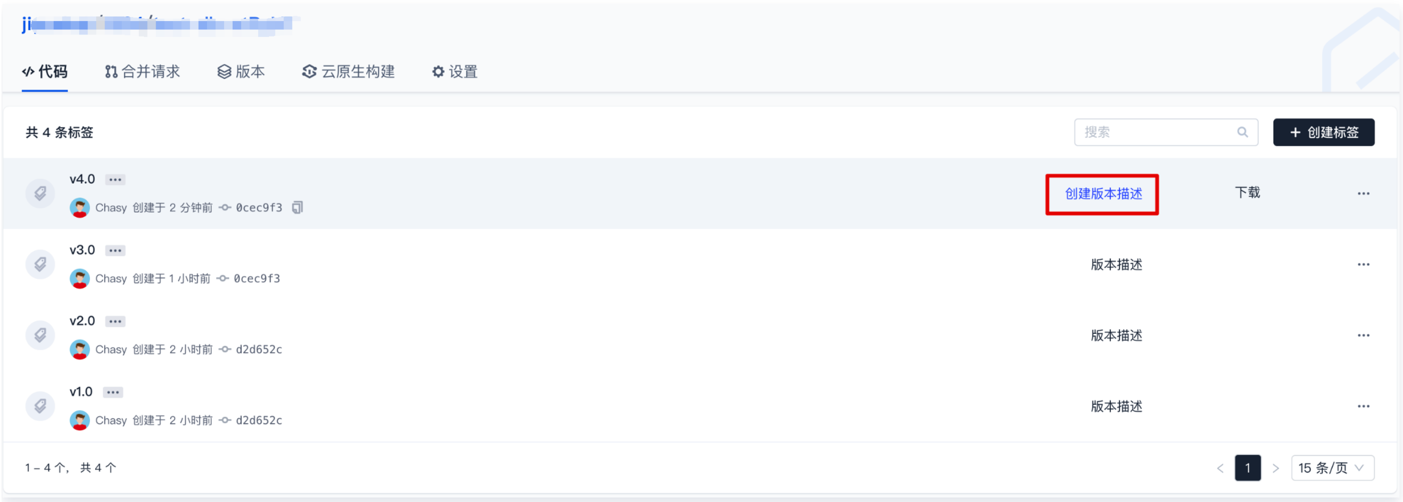
For example, a team may use Tags as a trigger for CI builds, pushing a Tag with the format `v1.0-release` in the production branch as a release command. Setting protected Tags allows only the Tag administrator to create this type of Tag and complete the release action, maintaining the cleanliness and standardization of version sequences within the code repository.

Viewing Version Information

Code Tags can represent a specific code version. Click **Version Description** to view the release details of that version. Click **Edit Version Description** to edit it.



If a Tag does not correspond to any version release, click **Create Version Description** to quickly create a release version for the Tag.



共 4 条标签

v4.0 Chasy 创建于 2 分钟前 0cec9f3
v3.0 Chasy 创建于 1 小时前 0cec9f3
v2.0 Chasy 创建于 2 小时前 d2d652c
v1.0 Chasy 创建于 2 小时前 d2d652c

1 – 4 个，共 4 个