CodeArts Repo

FAQ

Issue	01
Date	2024-11-07

HUAWEI CLOUD COMPUTING TECHNOLOGIES CO., LTD.

Trademarks and Permissions

Notice

The purchased products, services and features are stipulated by the contract made between Huawei Cloud and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

Huawei Cloud Computing Technologies Co., Ltd.

Address: Huawei Cloud Data Center Jiaoxinggong Road
Qianzhong Avenue
Gui'an New District
Gui Zhou 550029
People's Republic of China

Website: https://www.huaweicloud.com/intl/en-us/

Contents

1 Authentication

1.1 TLS Handshake Fails and the Error Message "ssl handshake failure" Is Displayed

Symptom

Run the following command on the local host to establish a TLS1.0 connection with the specified server and obtain the certificate information:

```
openssl s_client -connect test.com:443 -tls1
```

The following error message is displayed.

```
CONNECTED(00000003)
140155533838224:error:1409442E:SSL routines:ssl3_read_bytes:tlsv1 alert protocol version:s3_pkt.c:1493:SSL
alert number 70
140155533838224:error:1409E0E5:SSL routines:ssl3_write_bytes:ssl handshake failure:s3_pkt.c:659:
---
no peer certificate available
---
No client certificate CA names sent
---
SSL handshake has read 7 bytes and written 0 bytes
---
New, (NONE), Cipher is (NONE)
Secure Renegotiation IS NOT supported
Compression: NONE
Expansion: NONE
No ALPN negotiated
SSL-Session:
    Protocol  : TLSv1
    Cipher    : 0000
    Session-ID:
    Session-ID-ctx:
    Master-Key:
    Key-Arg   : None
    Krb5 Principal: None
    PSK identity: None
    PSK identity hint: None
    Start Time: 1720443876
    Timeout   : 7200 (sec)
    Verify return code: 0 (ok)
---
```

Analysis

CodeArts Repo supports TLS1.2 and TLS1.3.

Solution

Step 1 Run the following command on the Git Bash client to check your Git version:

```
git --version
```

Step 2 If your Git is earlier than 2.6.0, upgrade it to the **latest version**. The latest Git supports TLSv1.2 by default. If your Git version is 2.6.0 or later, specify the TLS protocol version:

```
openssl s_client -connect test.com:443 -tls1_2
```

----End

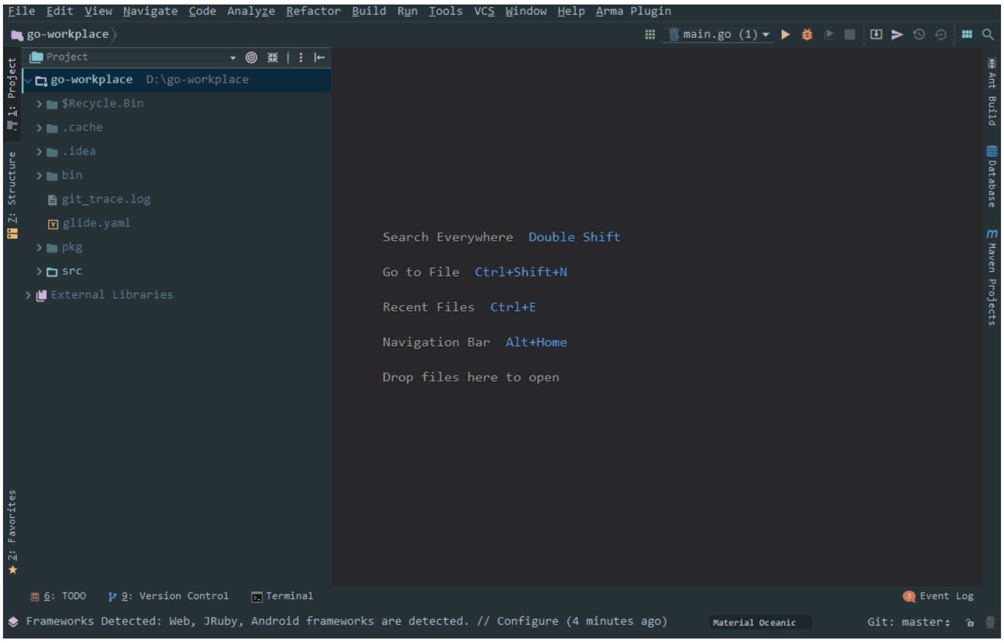
Table 1-1 Known incompatible clients

Client	Description
Git Credential Manager for Windows< v1.14.0	TLSv1.2 is not supported. Upgrade the Git client to the **latest version**. Select to install the Git Credential Manager during the upgrade.
Git on Red Hat 5	● TLSv1.2 is not supported. ● Upgrade to a later version.
Git on Red Hat 6	Upgrade to 6.8 or a later version.
Git on Red Hat 7	Upgrade to 7.2 or a later version.
JDK7	TLSv1.0 is used in JDK 7 by default. Any client, such as Eclipse JGit, running on an earlier version of JDK is affected. This can be resolved by enabling TLSv1.2 in JDK 7.
JDK6 or earlier	● TLSv1.2 is not supported. ● Upgrade to a later version.
Visual Studio	● Visual Studio comes with specific versions of Git for Windows and Git Credential Manager for Windows (GCM). ● Microsoft has updated the latest version of Visual Studio 2017 to work with the TLSv1.2 Git server. ● We recommend that Visual Studio users upgrade to the latest version by clicking the in-product notification flag or checking for updates directly from the IDE. ● Microsoft has provided additional guidance on the Visual Studio developer community support forum.

Third-party Git GUIs

If you use a third-party Git client, such as Eclipse, JetBrains, SourceTree, VSCode, Tower, or TortoiseGit, you are advised to download the latest Git Bash and use the native Git instead of the built-in type in the tool.

- Settings in IDEA



Search Everywhere Double Shift

Go to File Ctrl+Shift+N

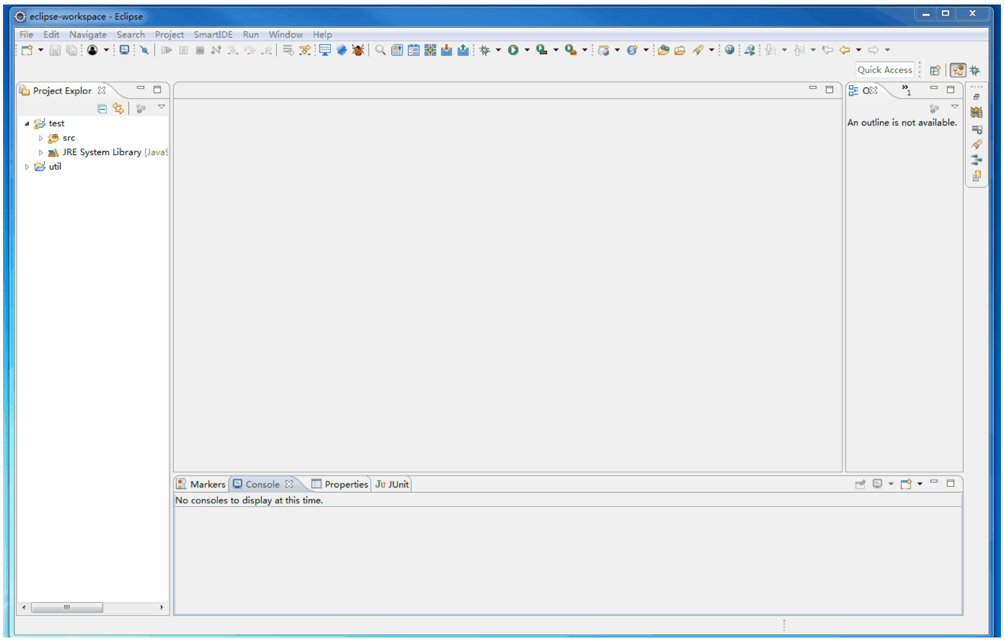
Recent Files Ctrl+E

Navigation Bar Alt+Home

Drop files here to open

Frameworks Detected: Web, JRuby, Android frameworks are detected. // Configure (4 minutes ago)

- Settings in Eclipse


eclipse-workspace - Eclipse
File Edit Navigate Search Project SmartIDE Run Window Help
Quick Access
Project Explor
test
src
JRE System Library [JavaS
util
Markers
Console
Properties
JUnit

An outline is not available.

No consoles to display at this time.

1.2 Upgrading the SSH Function of CodeArts Repo

Some insecure encryption algorithms have been removed from the SSH function of CodeArts Repo.

Key Exchange (KEX) and Message Authentication Code (MAC) no longer support the following algorithms:

KEX:

- diffie-hellman-group18-sha512
- diffie-hellman-group-exchange-sha1
- diffie-hellman-group-exchange-sha256

MAC:

- hmac-md5
- hmac-md5-96
- hmac-sha1-etm@openssh.com
- hmac-sha1-96-etm@openssh.com
- hmac-md5-etm@openssh.com
- hmac-md5-96-etm@openssh.com
- umac-64@openssh.com
- umac-128@openssh.com

Upgrade your code commit tool (such as Git Bash, Eclipse, and Idea) to the latest version. By default, the new version uses algorithms that are more secure.

If you have any questions, contact technical support.

1.3 How Can I Set Multiple SSH Keys on My Computer?

Scenario

Developers generate only one public SSH key named **id_rsa** and submit it to multiple websites.

However, one public SSH key cannot be shared by two accounts in a website. In this case, you need to generate two SSH keys.

Procedure

Step 1 Generate two different SSH keys in your local Git repository.

```
ssh-keygen -t rsa -C "email"
Generating public/private rsa key pair.
Enter file in which to save the key (~/.ssh/id_rsa):<Type two file names before pressing Enter.>
Enter passphrase(empty for no passphrase):<Enter your custom password before pressing Enter.>
```

NOTE

To generate two SSH keys, name two different file names **github_1** and **github_2**.

Step 2 Upload the two keys to the two accounts respectively.

Print the content of the ***.pub** file and paste it to the service website. Take note of the two usernames and their mappings to the keys.

Step 3 Edit the **~/.ssh/config** file.

```
Host dc_1
HostName *******.com
```

```
IdentityFile ~/.ssh/dc_1
PreferredAuthentications publickey
User username1
Host dc_2
HostName *******.com
IdentityFile ~/.ssh/dc_2
PreferredAuthentications publickey
User username2
```

Note that **Host** and **HostName** should be set to different values.

- **HostName**: Enter the service address.
- **Host**: Enter an alias for each key. The aliases will be used when you access the service.
- **IdentityFile**: Enter the location of each SSH key file. SSH key files can be stored in any directory you want.

Step 4 Read and write code.

In cases where only one SSH key is set, the SSH URL obtained from the service web page can be directly used to communicate with the service.

```
git@*****.com:name/repo.git
```

With two SSH keys on your computer, you need to replace the service address in the URL with a corresponding alias configured previously based on which account you are using.

```
git@dc_1:name/repo.git or git@dc_2:name/repo.git
```

----End

2 Member Permissions

2.1 How Do I Quickly Add Members to All Code Repositories?

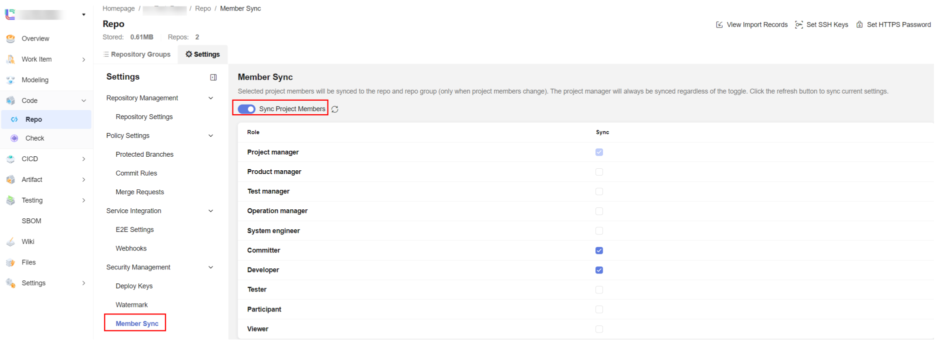
Scenario

There are too many repositories to configure for new members in a project group and the configuration is complex and difficult to manage. It is complex to configure permissions in each repository for new project members.

Solution

- Solution 1: Syncing Members in CodeArts Repo

Selected project members will be synced to the repo and repo group (only when project members change). The project manager will always be synced regardless of the toggle. Click the refresh button to sync current settings.



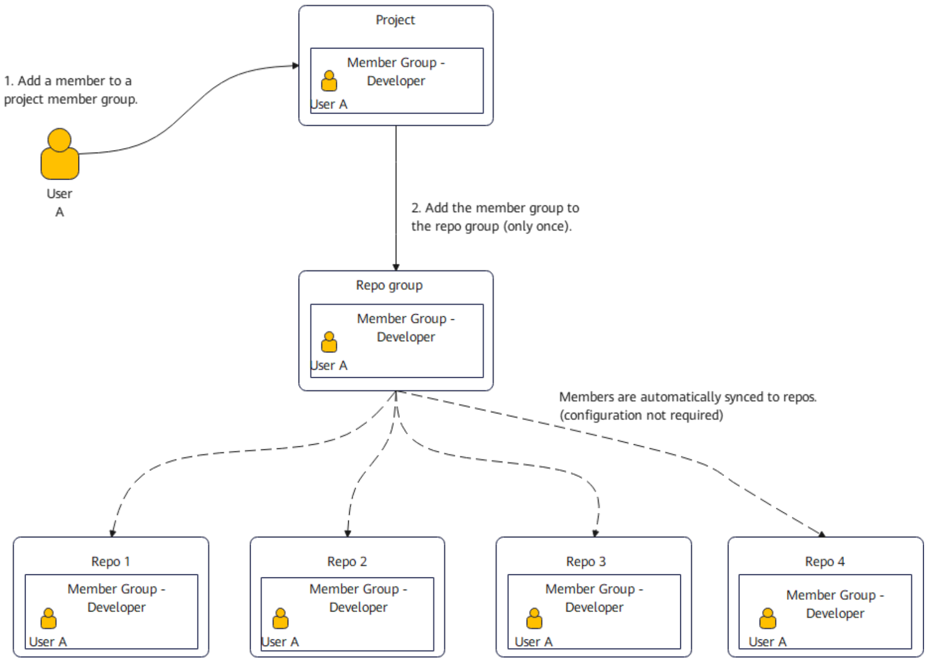
Member Sync

Selected project members will be synced to the repo and repo group (only when project members change). The project manager will always be synced regardless of the toggle. Click the refresh button to sync current settings.

● Sync Project Members

Role	Sync
Project manager	☑
Product manager	□
Test manager	□
Operation manager	□
System engineer	□
Committer	☑
Developer	☑
Tester	□
Participant	□
Viewer	□

- Solution 2: Managing Members by Group



The procedure is as follows:

Step 1 Define member groups in the project.

Step 2 The repository group is used to manage repositories hierarchically. Repositories are created under the repository group.

Step 3 After a project member group is added to the repository group, new members in the project member group are automatically synced to the repository group. Add members to all repositories.

----End

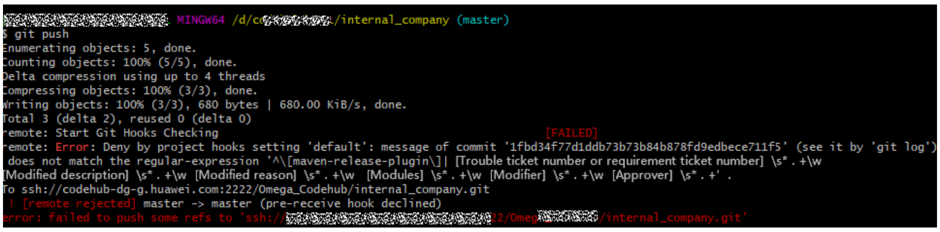
3 Uploading and Downloading Code

3.1 "Error: Deny by project hooks setting 'default': message of commit" Is Reported When Code Repository Is Pushed from the Local Host to CodeArts Repo

Symptom

If the format of the code file to be pushed to the remote repository is not standard and the ticket number and modifier are not provided, the error information shown in the following figure is displayed.

Figure 3-1 Error information displayed when code is pushed

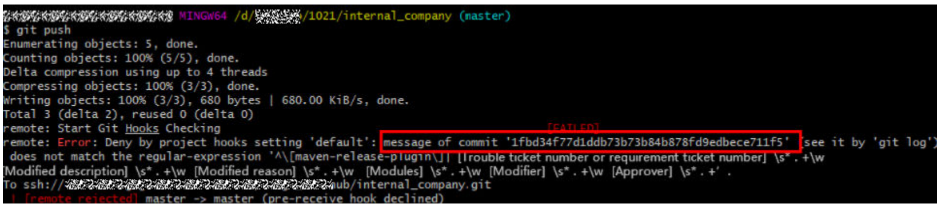


```
[redacted] MINGW64 /d/co[redacted]/internal_company (master)
$ git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 680 bytes | 680.00 KiB/s, done.
Total 3 (delta 2), reused 0 (delta 0)
remote: Start Git Hooks Checking                                    [FAILED]
remote: Error: Deny by project hooks setting 'default': message of commit '1fbd34f77d1ddb73b73b84b878fd9edbece711f5' (see it by 'git log')
 does not match the regular-expression '^\[maven-release-plugin\]| [Trouble ticket number or requirement ticket number] \s* . +\w
[Modified description] \s* . +\w  [Modified reason] \s* . +\w  [Modules] \s* . +\w  [Modifier] \s* . +\w  [Approver] \s* . +' .
To ssh://codehub-dg-g.huawei.com:2222/Omega_Codehub/internal_company.git
 ! [remote rejected] master -> master (pre-receive hook declined)
error: failed to push some refs to 'ssh://[redacted]22/Omeg[redacted]/internal_company.git'
```

Analysis

The commit message does not match the regular-expression: ^\[maven-release-plugin\]|[Trouble ticket number or requirement ticket number]\s*.+\W[Modified description]\s*.+\W[Modified reason]\s*.+\W[Modules] \s*.+\W[Modifier]\s*.+\W[Approver]\s*.+

Figure 3-2 Commit message when code is pushed



```
[redacted] MINGW64 /d/[redacted]/1021/internal_company (master)
$ git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 680 bytes | 680.00 KiB/s, done.
Total 3 (delta 2), reused 0 (delta 0)
remote: Start Git Hooks Checking                                    [FAILED]
remote: Error: Deny by project hooks setting 'default': message of commit '1fbd34f77d1ddb73b73b84b878fd9edbece711f5' (see it by 'git log')
 does not match the regular-expression '^\[maven-release-plugin\]| [Trouble ticket number or requirement ticket number] \s*.+\w
[Modified description] \s*.+\w  [Modified reason] \s*.+\w  [Modules] \s*.+\w  [Modifier] \s*.+\w  [Approver] \s*.+' .
To ssh://[redacted]ub/internal_company.git
 ! [remote rejected] master -> master (pre-receive hook declined)
```

Solution

- To modify the latest incorrect commit message, perform the following operations.

Step 1 Perform the following operations to modify the latest incorrect commit message:

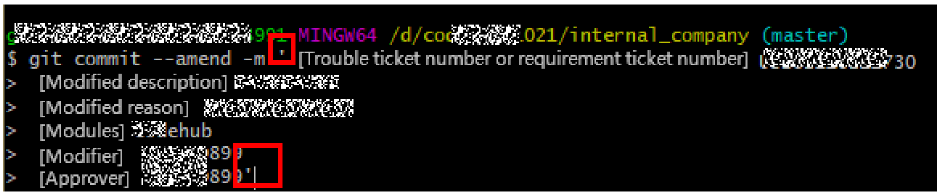
Right-click the **.git** hidden folder and choose **Open Git Bash here** from the shortcut menu to open Git Bash.

Step 2 Run the **git log** command to find the incorrect commit message.

Step 3 Run the **git commit --amend -m** command to access the vi text editor.

Step 4 Run the **git commit --amend -m '** command, enter the information in the first line, and press **Enter**. Enter the information in the second line, and repeat the process to the last line. Add the other single quotation mark at the end of the last line, and press **Enter**. The modification is successful.

Figure 3-3 Modifying the commit message



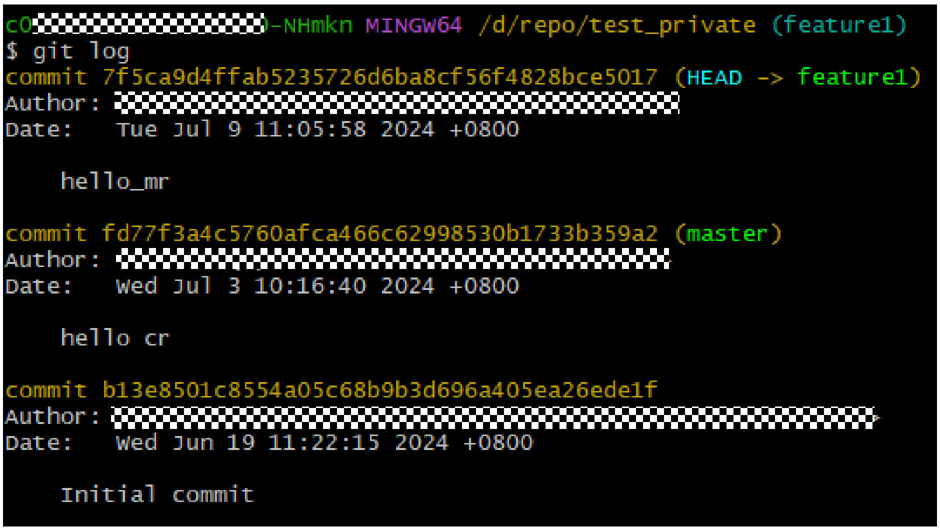
```
[redacted]901 MINGW64 /d/co[redacted].021/internal_company (master)
$ git commit --amend -m '[Trouble ticket number or requirement ticket number] [redacted]30
>   [Modified description] [redacted]
>   [Modified reason] [redacted]
>   [Modules] [redacted]ehub
>   [Modifier] [redacted]89
>   [Approver] [redacted]89'
```

----End

- To modify the incorrect commit message that is not the latest, perform the following operations.

Step 1 To modify the second-to-last commit message, perform the following operations:

Run the **git log** command on Git Bash to search for incorrect commit messages.



```
c0[redacted]-NHmkn MINGW64 /d/repo/test_private (feature1)
$ git log
commit 7f5ca9d4ffab5235726d6ba8cf56f4828bce5017 (HEAD -> feature1)
Author: [redacted]
Date:   Tue Jul 9 11:05:58 2024 +0800

    hello_mr

commit fd77f3a4c5760afca466c62998530b1733b359a2 (master)
Author: [redacted]
Date:   Wed Jul 3 10:16:40 2024 +0800

    hello cr

commit b13e8501c8554a05c68b9b3d696a405ea26ede1f
Author: [redacted]
Date:   Wed Jun 19 11:22:15 2024 +0800

    Initial commit
```

Step 2 Run the **git reset --hard commitID** command to roll back to the previous commit ID before this one.

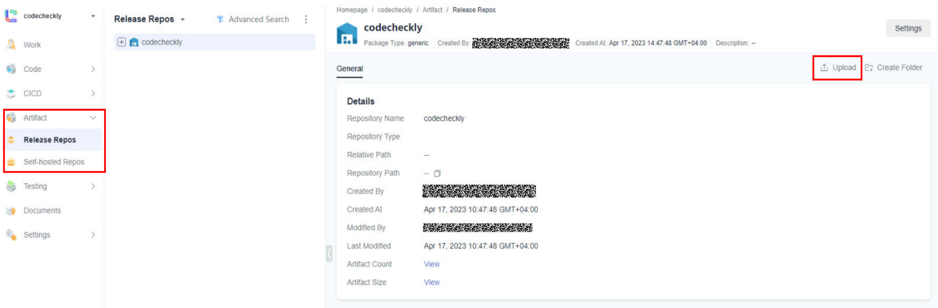
Step 3 Run **Step 3** and **Step 4** to modify the commit message.

----End

3.2 Pushing Binary Files to CodeArts Repo Failed

CodeArts Repo supports binary file commit control. Check whether binary file commit is prohibited for your repos.

You are not advised storing binary files in the CodeArts Repo repository. Otherwise, the performance and stability will be affected. You are advised to upload the binary file to **CodeArts Artifact** for storage.



codecheckly

Package Type: generic Created By: [redacted] Created At: Apr 17, 2023 14:47:48 GMT+04:00 Description: --

Details

Repository Name codecheckly
Repository Type
Relative Path --
Repository Path --
Created By [redacted]
Created At Apr 17, 2023 10:47:48 GMT+04:00
Modified By [redacted]
Last Modified Apr 17, 2023 10:47:48 GMT+04:00
Artifact Count View
Artifact Size View

3.3 Error "'origin' does not appear to be a git repository..." Is Reported When the Git Push Command Is Executed

Symptom

Error **'origin' does not appear to be a git repository...** is reported when you run the following command:

```
git push --set-upstream origin feature1
```

Analysis

origin does not appear to be a git repository.

Solution

Verify the name and path of the remote repository, delete the false name and add another remote repository. Run the following commands:

Step 1 Check the details of the remote repository, including its name and associated address.

```
git remote -v
```

Step 2 Delete the false origin repository.

```
git remote remove origin
```

Step 3 Add another remote repository address.

```
git remote add origin
```

Step 4 Commit the code file to the master branch of the remote code repository again.

```
git push -u origin master
```

----End

3.4 Error "The requested URL returned error: 401" Is Reported When HTTPS Is Used to Clone Code in CentOS

Symptom

Error **The requested URL returned error: 401** is reported when HTTPS is used to clone code in CentOS.

Analysis

The built-in Git version of CentOS is 1.7.1 or earlier.

Solution

Step 1 Run the following command in Git Bash to check the Git version provided by the system: If the version is 1.7.1 or earlier, go to Step 2.

```
git --version
```

Step 2 Run the following command to uninstall the Git provided by CentOS:

```
yum remove git
```

Step 3 Download the latest version from the **Git website** and add Git to the environment variables.

1. Download the Git source code package from GitHub. Replace the version number with the actual one.

```
wget https://github.com/git/git/archive/Version number.tar.gz
```

2. Decompress the source code package.
```
tar zxvf Version number.tar.gz
```
3. Switch to the Git source code directory containing the decompressed files.
```
cd git-Version number
```
4. Generate the configuration file.
```
make configure
```
5. Configure the Git installation path and code conversion library.
```
./configure --prefix=/usr/local/git --with-iconv=/usr/local/libiconv
```
6. Compile the Git source code and documents.
```
make all doc
```
7. Install Git and its documents.
```
make install install-doc install-html
```
8. Add the path of the Git executable file to the system environment variables.
```
echo "export PATH=$PATH:/usr/local/git/bin" >> /etc/bashrc
```
9. Apply the environment variables.
```
source /etc/bashrc
```

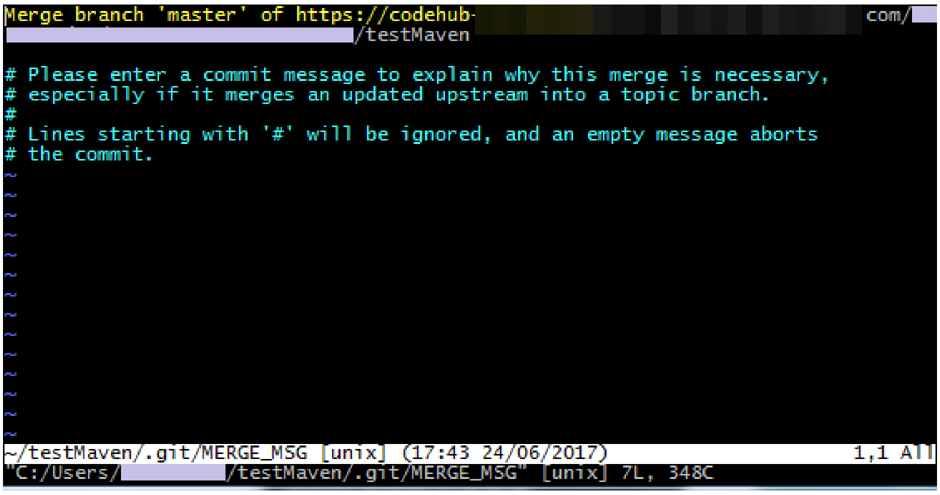
----End

3.5 Error "Merge branch 'master' of https://test.com Please Enter a commit" Is Reported When Pulling Code Using the Git Pull Command

Symptom

Error **Merge branch 'master' of https://test.com Please enter a commit message to explain why this merge is necessary...** is reported when pulling code using the **git pull** command, as shown in the following figure.

Figure 3-4 Error message



```
Merge branch 'master' of https://codehub-[redacted]com/[redacted]
[redacted]/testMaven

# Please enter a commit message to explain why this merge is necessary,
# especially if it merges an updated upstream into a topic branch.
#
# Lines starting with '#' will be ignored, and an empty message aborts
# the commit.
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~/testMaven/.git/MERGE_MSG [unix] (17:43 24/06/2017)                1,1 All
"C:/Users/[redacted]/testMaven/.git/MERGE_MSG" [unix] 7L, 348C
```

Analysis

The code in CodeArts Repo is different from the code in your local repository. Therefore, when **git pull** is executed, the remote code will be merged to the local code. The dialog box displayed asks if you confirm the merge, and prompts you to enter a commit message.

Solution

Go to the local repository directory and run the following command in Git Bash:

Step 1 Cancel the merge.

```
git merge --abort
```

Step 2 Merge the master branch of the remote repository (origin) to the current branch:

```
git merge origin/master
```

----End

3.6 Message "fatal: refusing to merge unrelated histories" Is Displayed

Symptom

The message **refusing to merge unrelated histories** is displayed after **git pull** is executed.

Analysis

The objects of the merge attempt are two unrelated repositories or branches.

Solution

Run **git pull origin master --allow-unrelated-histories** for force merge.

3.7 How Do I Prevent Files Containing Secrets from Being Pushed to CodeArts Repo?

Background

- You can go to the repository details page, choose **Settings** > **Commit Rules**, and select **Files containing secrets cannot be committed**.
- CodeArts Repo will refuse push attempts when the name of file to be pushed matches the regular expressions listed below.

NOTICE

This rule does not apply to files that have been pushed.

The following table lists the regular expressions for restricting pushes of files containing secrets and provides some file name examples.

Regular Expression	File Name Example
(ssh|config)\/(personal|server)_(rsa|dsa|ed\d+|ecdsa)	ssh_server_rsa
_rsa$	id_rsa
_dsa$	id_dsa
_ed25519$	id_ed25519
_ecdsa$	id_ecdsa
\.(pem|key)$	secret.key privatekey.pem
"[._]history$"	.bash_history

4 Migrating Repositories

4.1 When a Git-based Remote Repository Is Imported to CodeArts Repo, an Error Message "Accessing the remote repository timed out. Check the network." Is Displayed.

Symptom

When a Git-based remote repository is imported to CodeArts Repo, the import takes more than 30 minutes, and an error message "Accessing the remote repository timed out. Check the network." is displayed.

Analysis

The code repository is too large or the network is poor.

Solution

Step 1 Download a repository from the source repository address. Go to the code repository to be downloaded and copy the HTTPS address.

Step 2 Open the Git Bash client and clone the code repository to the local computer:

```
git clone --bare <source_repository_address>
```

Step 3 Associate the cloned code repository and push it to CodeArts Repo.

1. Go to the CodeArts Repo homepage, click **New Repository**, and select an existing project from the **Project** drop-down list box or create a project.
2. Set **Repository Type** to **Common**, enter related parameters, deselect **Generate README** and set **.gitignore Programming Language** to create a code repository. The homepage of the code repository is displayed.
3. Choose **Clone/Download** > **Clone with HTTPS** in the upper right corner and copy the HTTPS address.
4. Open the Git Bash client and push the local code repository to the created code repository:

```
git push --mirror <Created_CodeArts Repo_repository_address>
```

When running commands, enter the HTTPS account and password of CoeArts Repo.

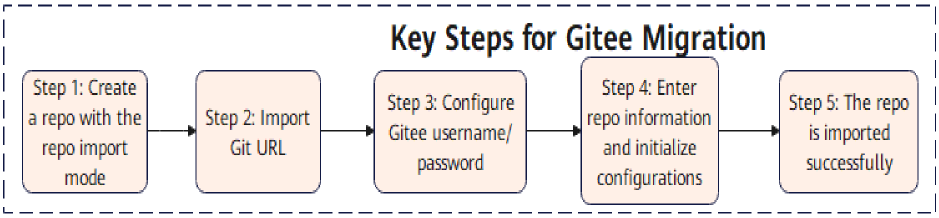
----End

NOTE

- If your local code repository has branches and tags, they will be pushed to the created code repository.
- After the push is successful, go to the code repository details page of CodeArts Repo to check whether the pushed code repository is complete. If the problem persists, contact Huawei Cloud technical support.

4.2 How Do I Migrate a Gitee Repo?

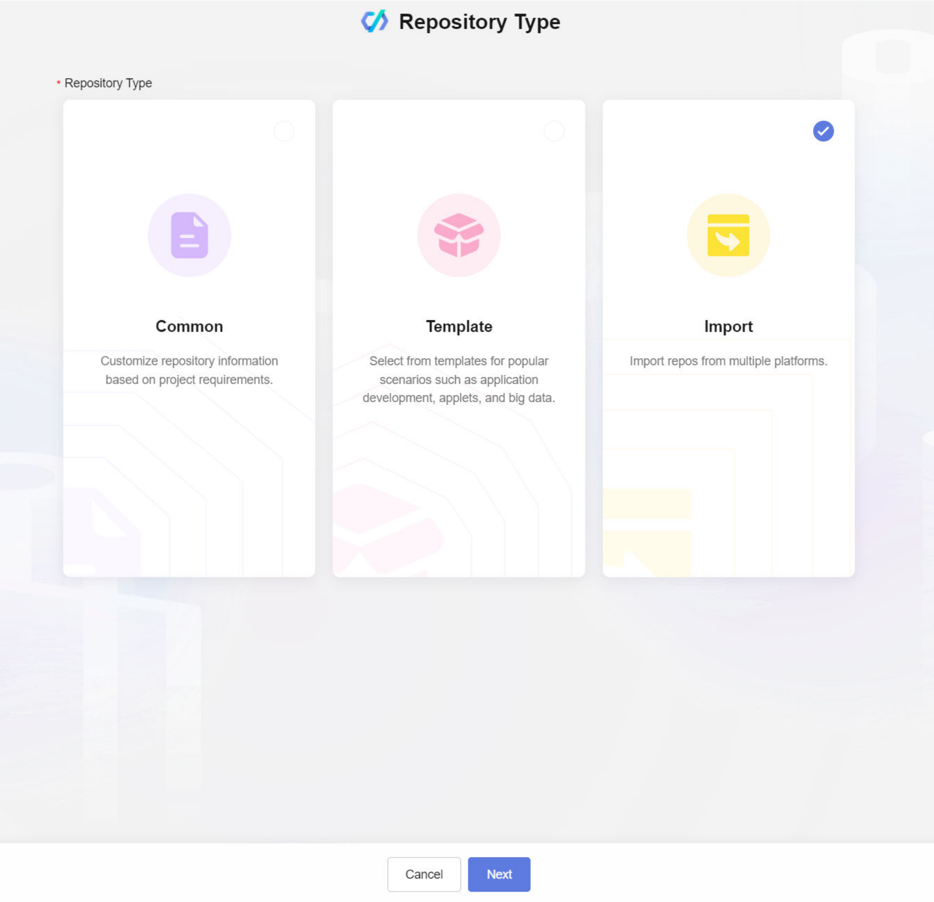
Migration Procedure


Key Steps for Gitee Migration
Step 1: Create a repo with the repo import mode
Step 2: Import Git URL
Step 3: Configure Gitee username/ password
Step 4: Enter repo information and initialize configurations
Step 5: The repo is imported successfully

The migration tool can migrate data related to repo files, such as branches, tags, commits, and source files of code repositories. Gitee repo data, such as members, PRs, and comments, cannot be migrated.

Procedure

Step 1 Create a repo and select the repo import mode. Log in to CodeArts Repo, and choose **New Repository > Import**.



Repository Type

* Repository Type

Common

Customize repository information based on project requirements.

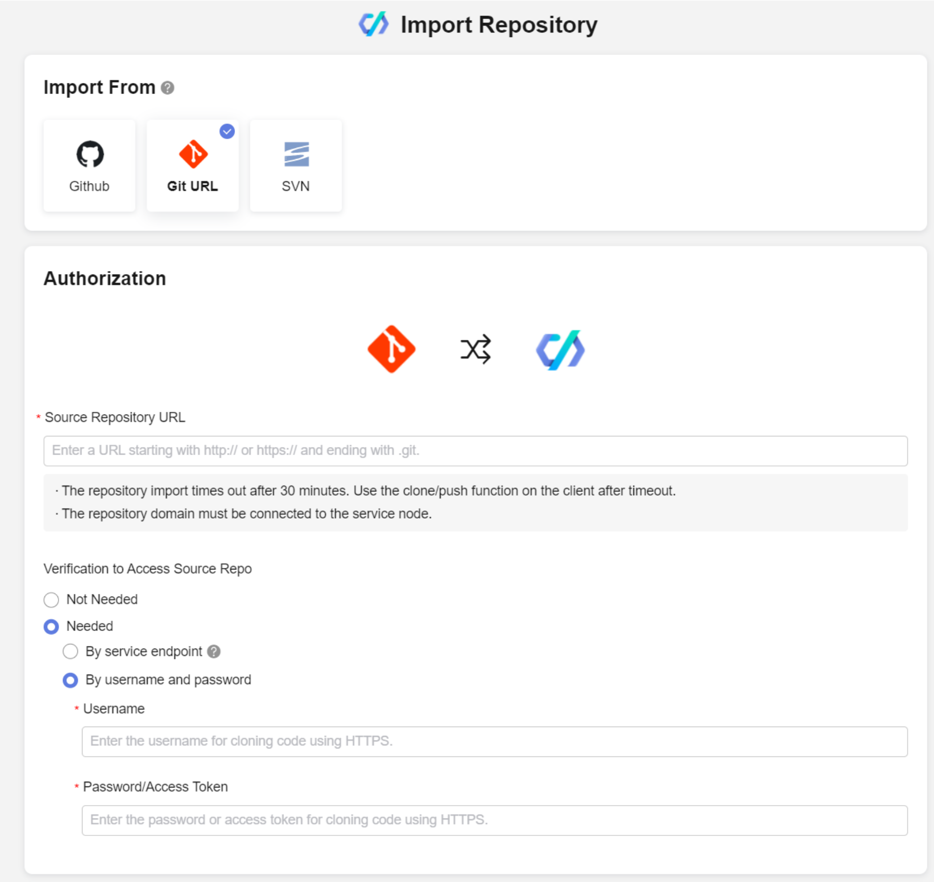
Template

Select from templates for popular scenarios such as application development, applets, and big data.

Import

Import repos from multiple platforms.

Step 2 Select **Git URL** and enter a Gitee URL.



Import Repository

Import From

Authorization

* Source Repository URL

Enter a URL starting with http:// or https:// and ending with .git.

· The repository import times out after 30 minutes. Use the clone/push function on the client after timeout.
· The repository domain must be connected to the service node.

Verification to Access Source Repo

○ Not Needed
● Needed
○ By service endpoint
● By username and password

* Username

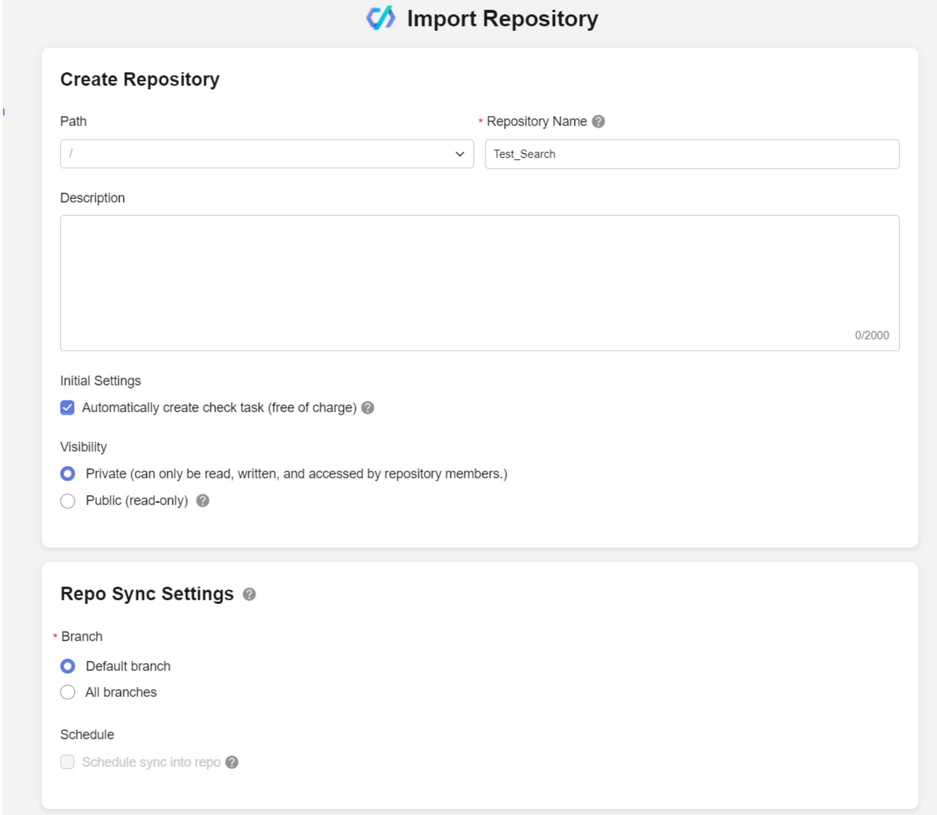
Enter the username for cloning code using HTTPS.

* Password/Access Token

Enter the password or access token for cloning code using HTTPS.

Step 3 Configure the username and password of Gitee. **Username**: username for logging in to Gitee, which is usually a mobile number. **Password**: password for logging in to Gitee.

Step 4 Enter the repository information and initialize the configuration.



Import Repository

Create Repository

Path: /

* Repository Name: Test_Search

Description:

0/2000

Initial Settings

☑ Automatically create check task (free of charge)

Visibility

● Private (can only be read, written, and accessed by repository members.)

○ Public (read-only)

Repo Sync Settings

* Branch

● Default branch

○ All branches

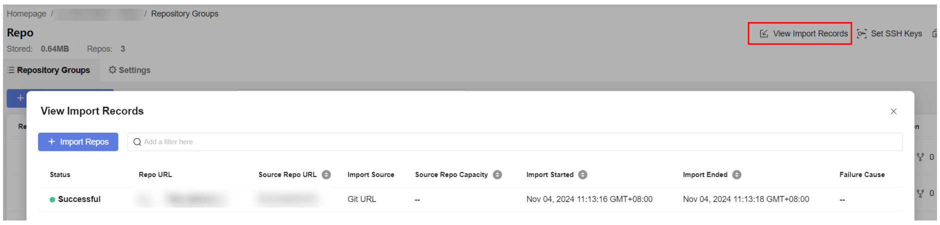
Schedule

□ Schedule sync into repo

Step 5 Enter the key information about the imported repo.

- **Path**: Select the root directory of the repo to be imported.
- **Repository Name**: Name of the imported repo.
- **Visibility**: Private or public.
- **Schedule**: If this is enabled, the imported repository is an image repository. The repository cannot commit code and can only be synced from the source repository periodically. The repository is updated every 24 hours, and the updated content is the content of the source repository 24 hours ago.
- **Branch**: Import the default branch or all branches of the source repo.
- **Automatically create check task**

Step 6 The repo is imported successfully. On the CodeArts Repo homepage, click **View Import Records** in the upper right corner to view key information such as the import status and failure cause. As shown in the following figure, after the repo is successfully imported, the repo is successfully created.



Homepage / [redacted] / Repository Groups

Repo

Stored: 0.64MB Repos: 3

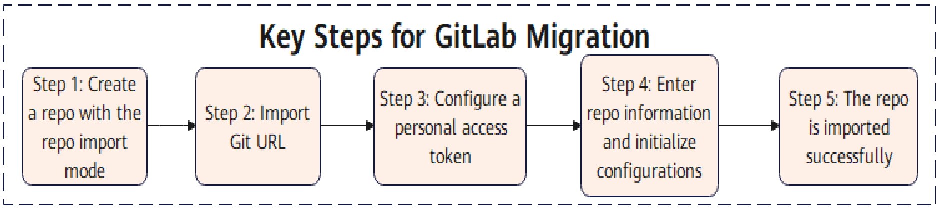
View Import Records

Status	Repo URL	Source Repo URL	Import Source	Source Repo Capacity	Import Started	Import Ended	Failure Cause
● Successful	[redacted]	[redacted]	Git URL	--	Nov 04, 2024 11:13:16 GMT+08:00	Nov 04, 2024 11:13:18 GMT+08:00	--

----End

4.3 How Do I Migrate a GitLab Repo?

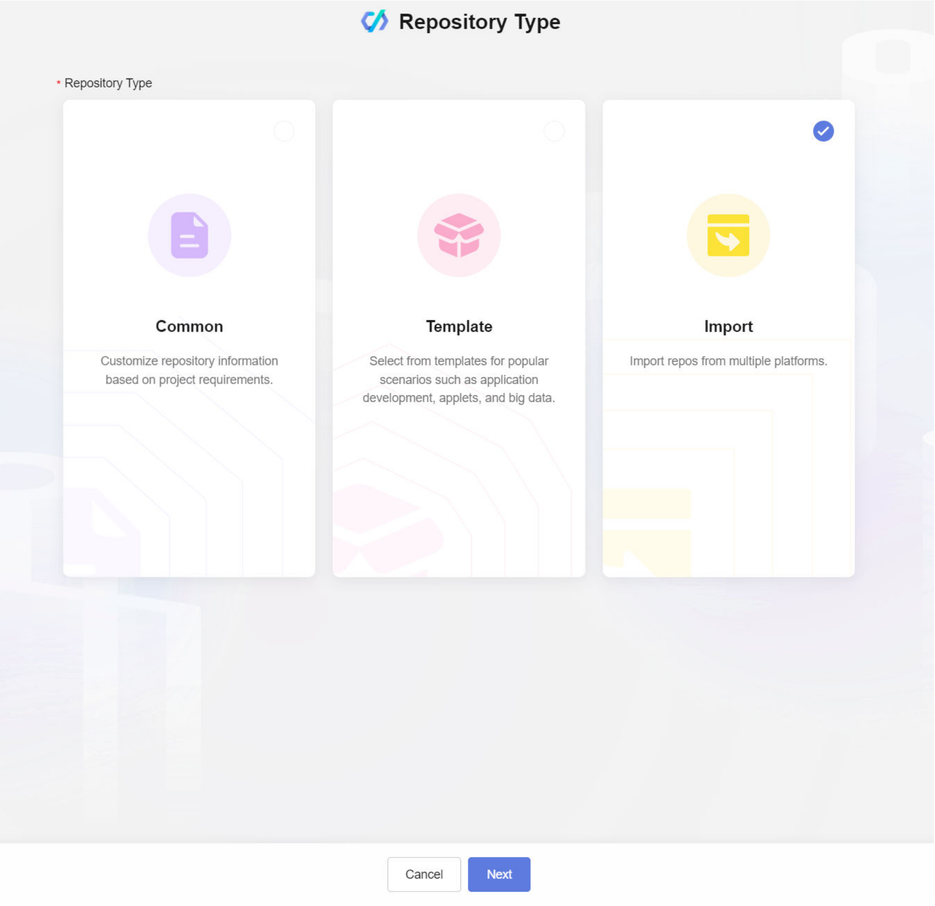
Migration Procedure


Key Steps for GitLab Migration
Step 1: Create a repo with the repo import mode
Step 2: Import Git URL
Step 3: Configure a personal access token
Step 4: Enter repo information and initialize configurations
Step 5: The repo is imported successfully

The migration tool can migrate data related to repo files, such as branches, tags, commits, and source files of code repositories. GitLab repo data, such as members, PRs, and comments, cannot be migrated.

Procedure

Step 1 Log in to CodeArts Repo, and choose **New Repository** > **Import**.


Repository Type

* Repository Type

Common
Customize repository information based on project requirements.

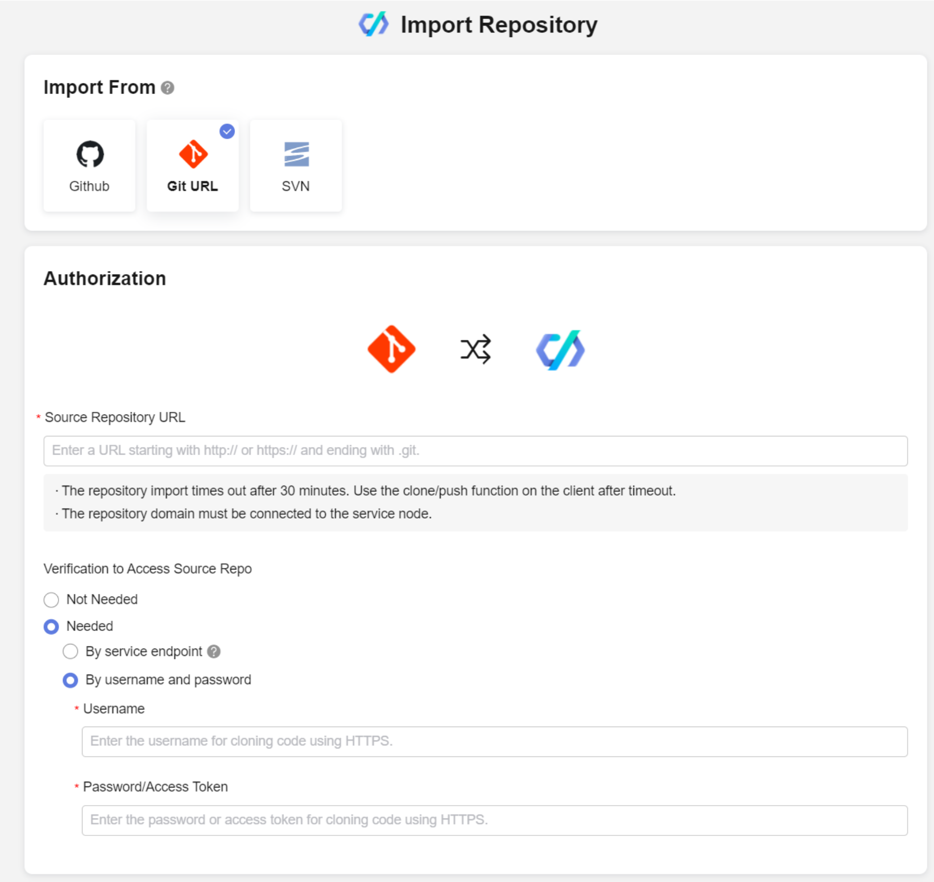
Template
Select from templates for popular scenarios such as application development, applets, and big data.

Import
Import repos from multiple platforms.

Step 2 Select **Git URL** and enter a GitLab URL.

Step 3 Configure the GitLab username and personal access token.

- **Username**: This field is not used. You can enter any character string, for example, **test**.
- **AccessToken**: Click the avatar in the upper right corner of GitLab, choose **Preferences** > **User Settings** > **Access Tokens**, and select the **read_repository** permission.



Import Repository

Import From

Authorization

* Source Repository URL

Enter a URL starting with http:// or https:// and ending with .git.

· The repository import times out after 30 minutes. Use the clone/push function on the client after timeout.
· The repository domain must be connected to the service node.

Verification to Access Source Repo

○ Not Needed
● Needed
○ By service endpoint
● By username and password

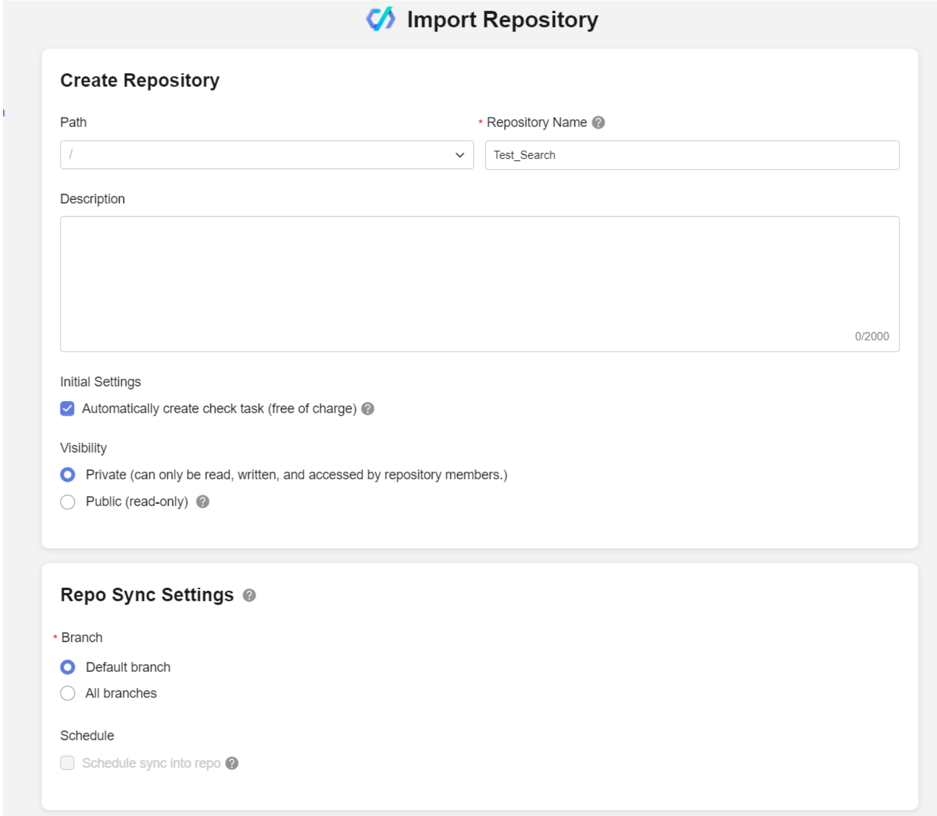
* Username

Enter the username for cloning code using HTTPS.

* Password/Access Token

Enter the password or access token for cloning code using HTTPS.

Step 4 Enter the repository information and initialize the configuration.



Import Repository

Create Repository

Path: /

* Repository Name: Test_Search

Description

0/2000

Initial Settings

☑ Automatically create check task (free of charge)

Visibility

● Private (can only be read, written, and accessed by repository members.)

○ Public (read-only)

Repo Sync Settings

* Branch

● Default branch

○ All branches

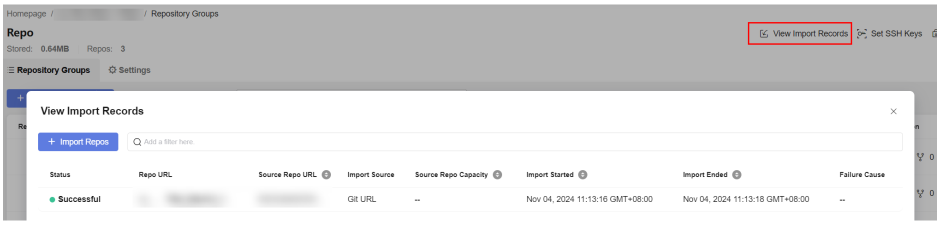
Schedule

□ Schedule sync into repo

Step 5 Enter the key information about the imported repo.

- **Path**: Select the root directory of the repo to be imported.
- **Repository Name**: Name of the imported repo.
- Visibility: Private or public.
- **Schedule**: If this is enabled, the imported repository is an image repository. The repository cannot commit code and can only be synced from the source repository periodically. The repository is updated every 24 hours, and the updated content is the content of the source repository 24 hours ago.
- **Branch**: Import the default branch or all branches of the source repo.
- **Automatically create check task**

Step 6 The repo is imported successfully. You can view key information such as the import status and failure cause in the import record list. As shown in the following figure, after the repo is successfully imported, the repo is successfully created.



Homepage / [redacted] / Repository Groups

Repo

Stored: 0.64MB Repos: 3

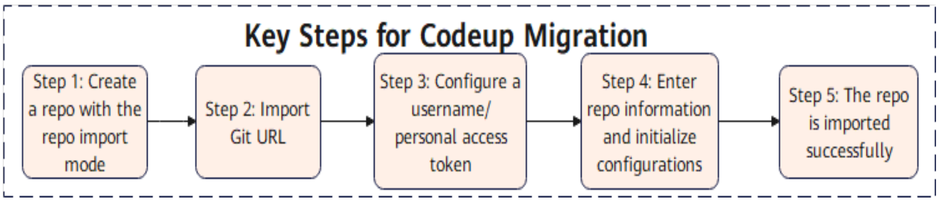
View Import Records

Status	Repo URL	Source Repo URL	Import Source	Source Repo Capacity	Import Started	Import Ended	Failure Cause
● Successful	[redacted]	[redacted]	Git URL	--	Nov 04, 2024 11:13:16 GMT+08:00	Nov 04, 2024 11:13:18 GMT+08:00	--

----End

4.4 How Do I Migrate a Codeup Repo?

Migration Procedure



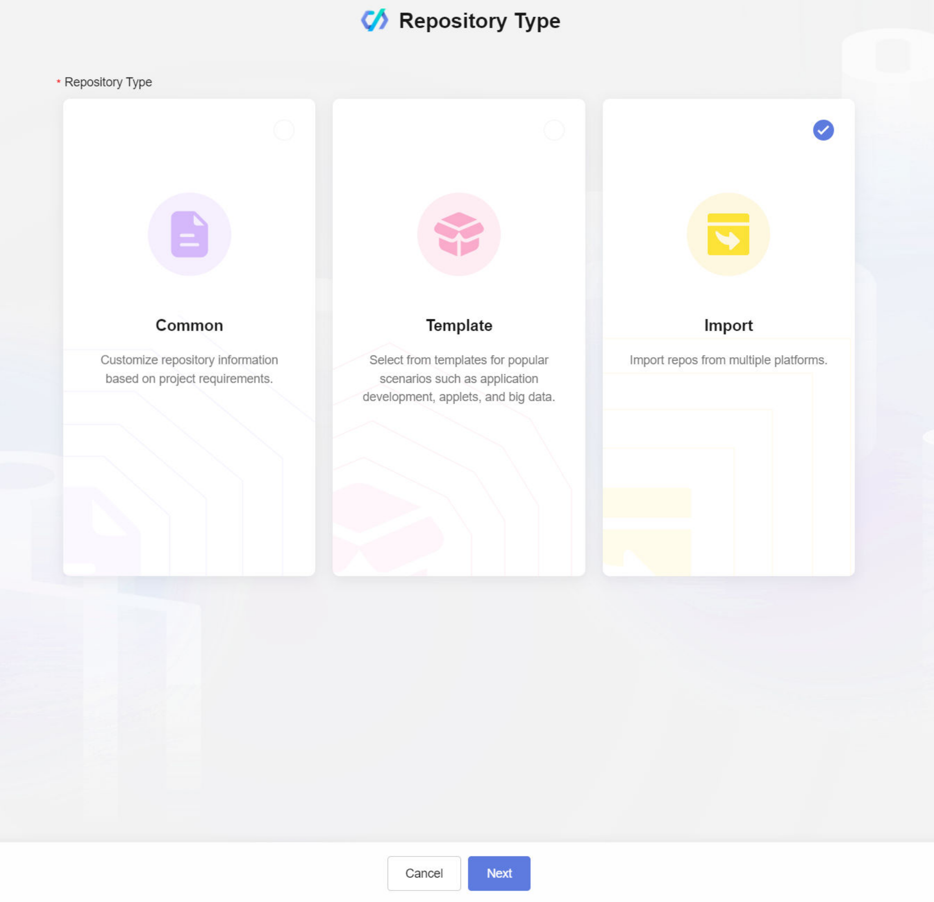
Key Steps for Codeup Migration

Step 1: Create a repo with the repo import mode → Step 2: Import Git URL → Step 3: Configure a username/ personal access token → Step 4: Enter repo information and initialize configurations → Step 5: The repo is imported successfully

The migration tool can migrate data related to repo files, such as branches, tags, commits, and source files of code repositories. Codeup repo data, such as members, PRs, and comments, cannot be migrated.

Procedure

Step 1 Create a repo and select the repo import mode. Log in to CodeArts Repo, and choose **New Repository > Import**.



Repository Type

* Repository Type

○ Common

Customize repository information based on project requirements.

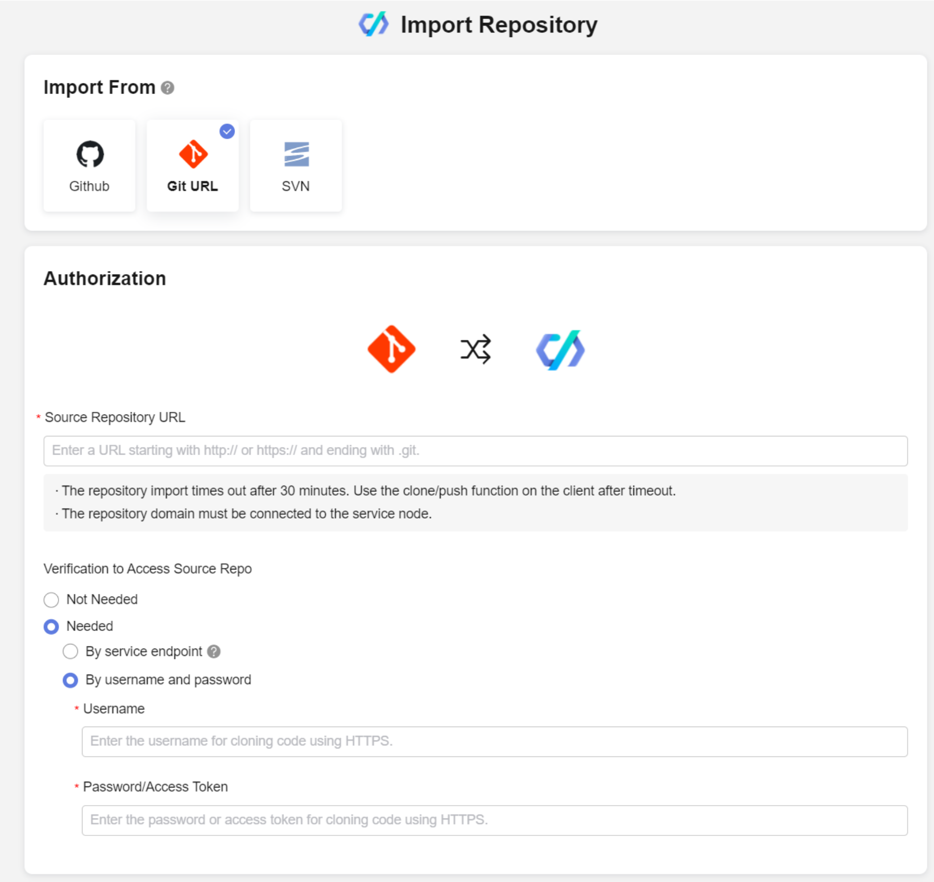
○ Template

Select from templates for popular scenarios such as application development, applets, and big data.

● Import

Import repos from multiple platforms.

Step 2 Select **Git URL** for **Import**. Enter the repo URL of Codeup.



Import Repository

Import From

Authorization

* Source Repository URL

Enter a URL starting with http:// or https:// and ending with .git.

· The repository import times out after 30 minutes. Use the clone/push function on the client after timeout.
· The repository domain must be connected to the service node.

Verification to Access Source Repo

○ Not Needed
● Needed
○ By service endpoint
● By username and password

* Username

Enter the username for cloning code using HTTPS.

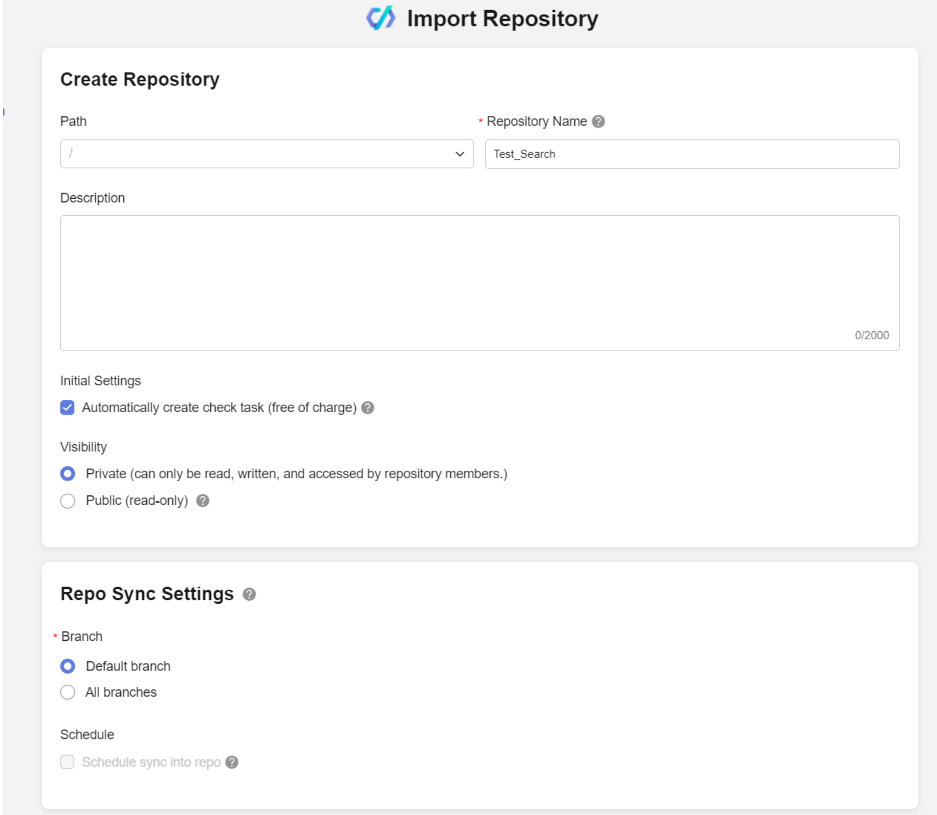
* Password/Access Token

Enter the password or access token for cloning code using HTTPS.

Step 3 Configure the Codeup username and personal access token. **Username**: username for logging in to the Codeup. Generally, the user name is a mobile number. **AccessToken**: **Codeup** > **Personal Settings** > **Personal Access Token** > **New Access Token**. Select the **read-only** permission for the repository in code management.

Step 4 Enter the repository information and initialize the configuration.

- **Path**: Select the root directory of the repo to be imported.
- **Repository Name**: Name of the imported repo.
- **Visibility**: Private or public.
- **Schedule**: If this is enabled, the imported repository is an image repository. The repository cannot commit code and can only be synced from the source repository periodically. The repository is updated every 24 hours, and the updated content is the content of the source repository 24 hours ago.
- **Branch**: Import the default branch or all branches of the source repo.
- **Automatically create check task**



Import Repository

Create Repository

Path
/

* Repository Name
Test_Search

Description

0/2000

Initial Settings
☑ Automatically create check task (free of charge)

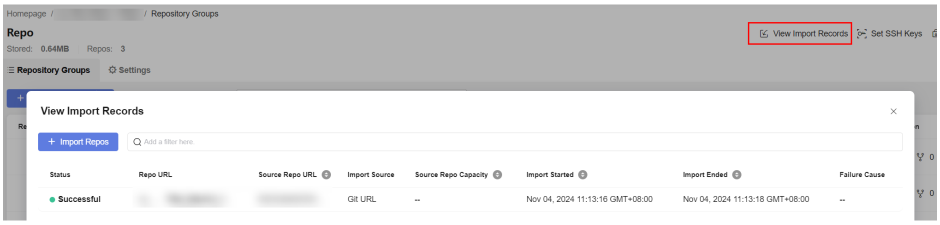
Visibility
● Private (can only be read, written, and accessed by repository members.)
○ Public (read-only)

Repo Sync Settings

* Branch
● Default branch
○ All branches

Schedule
□ Schedule sync into repo

Step 5 The repo is imported successfully. You can view key information such as the import status and failure cause in the import record list. After the repo is imported, check whether the repo is created successfully.



Homepage / [redacted] / Repository Groups

Repo
Stored: 0.64MB | Repos: 3

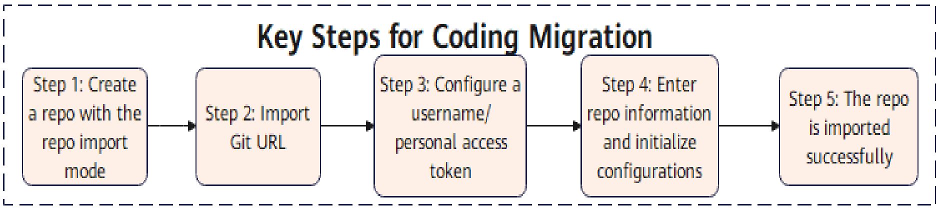
View Import Records

Status	Repo URL	Source Repo URL	Import Source	Source Repo Capacity	Import Started	Import Ended	Failure Cause
● Successful	[redacted]	[redacted]	Git URL	--	Nov 04, 2024 11:13:16 GMT+08:00	Nov 04, 2024 11:13:18 GMT+08:00	--

----End

4.5 How Do I Migrate a Coding Repo?

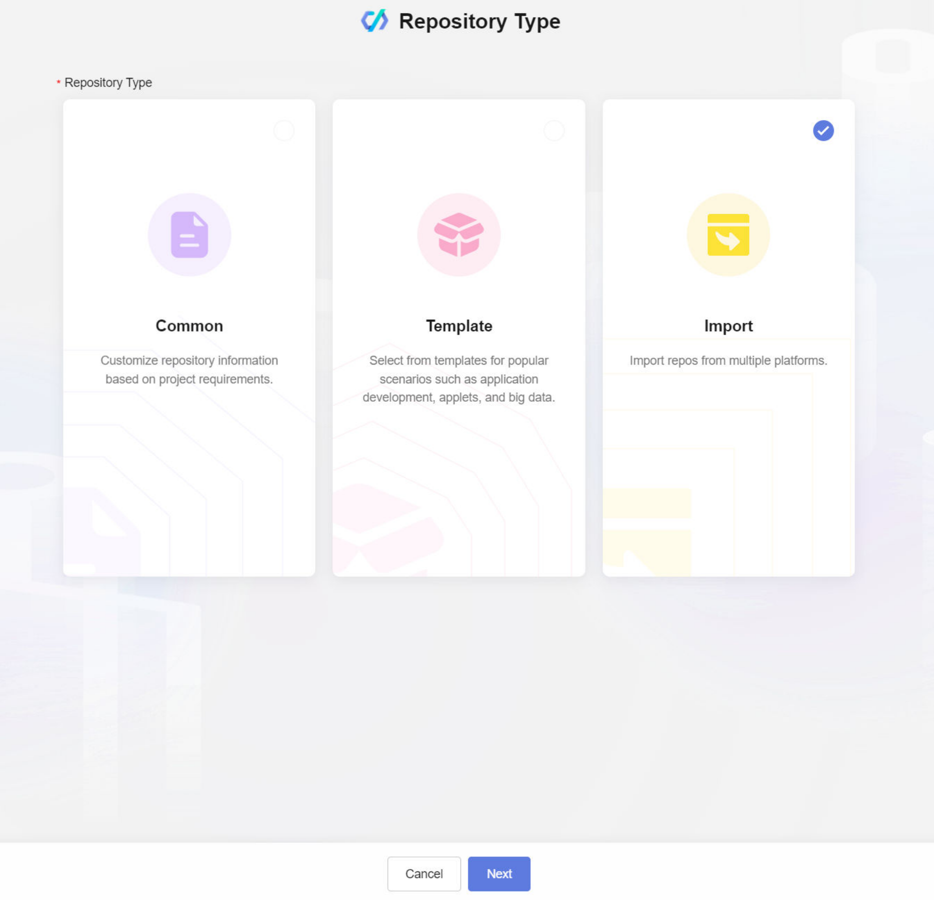
Migration Description


Key Steps for Coding Migration
Step 1: Create a repo with the repo import mode
Step 2: Import Git URL
Step 3: Configure a username/ personal access token
Step 4: Enter repo information and initialize configurations
Step 5: The repo is imported successfully

The migration tool can migrate data related to repo files, such as branch, tag, commit, and source files of code libraries. Coding repo data, such as members, PRs, and comments, cannot be migrated.

Procedure

Step 1 Create a repo and select the repo import mode. Log in to CodeArts Repo, and choose **New Repository > Import**.



Repository Type

* Repository Type

○ Common

Customize repository information based on project requirements.

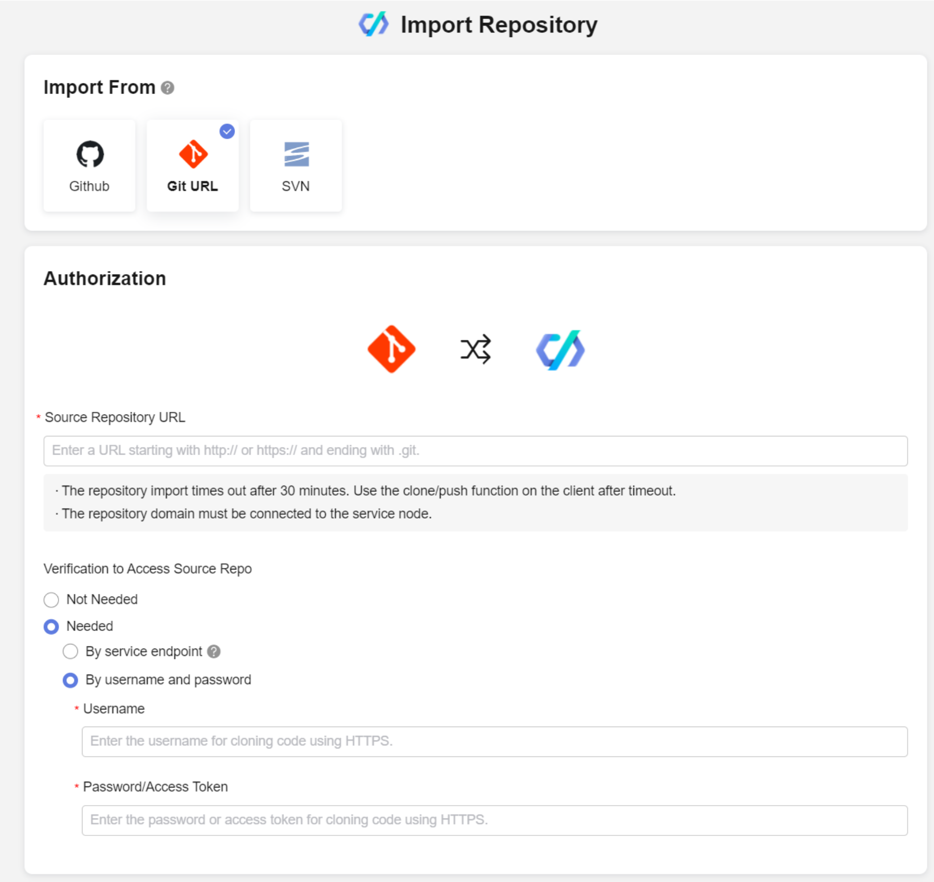
○ Template

Select from templates for popular scenarios such as application development, applets, and big data.

● Import

Import repos from multiple platforms.

Step 2 Select **Git URL** and enter a Coding URL.



Import Repository

Import From

Authorization

* Source Repository URL

Enter a URL starting with http:// or https:// and ending with .git.

· The repository import times out after 30 minutes. Use the clone/push function on the client after timeout.
· The repository domain must be connected to the service node.

Verification to Access Source Repo

○ Not Needed

● Needed

○ By service endpoint

● By username and password

* Username

Enter the username for cloning code using HTTPS.

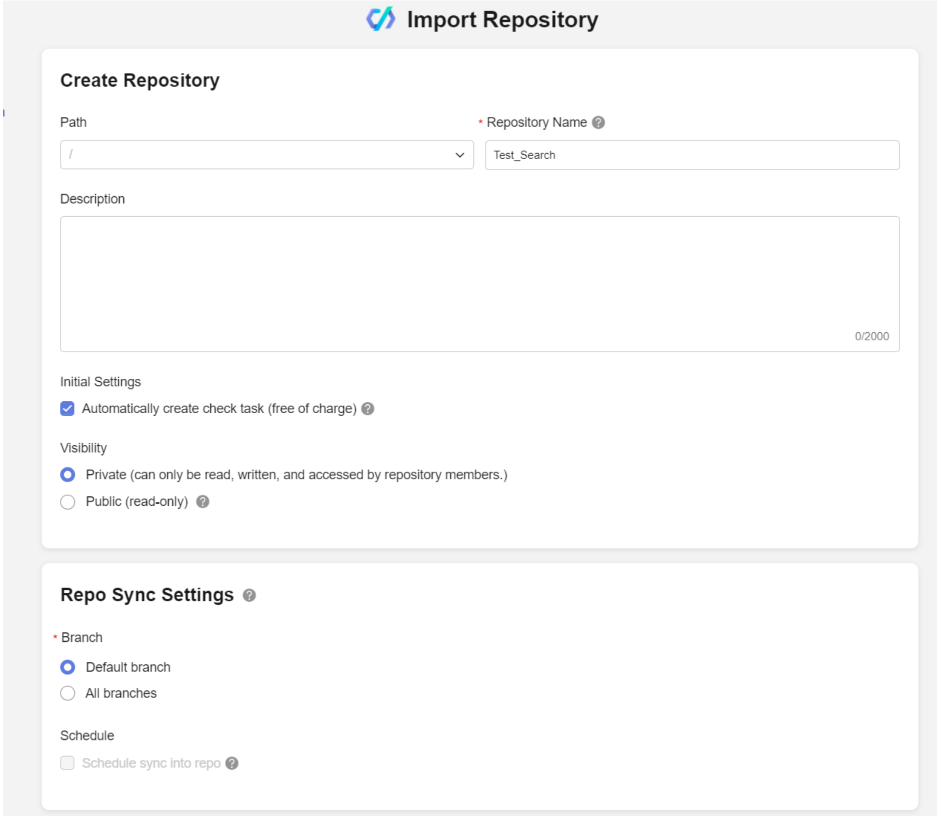
* Password/Access Token

Enter the password or access token for cloning code using HTTPS.

Step 3 Configure the Coding username and personal access token. Username: username for logging in to the Coding. Generally, the username is a mobile number. AccessToken: Choose **Coding** > **Personal Account Settings** > **Access Token** > **Create Access Token**, and make the repository read-only.

Step 4 Enter the repository information and initialize the configuration.

- **Path**: Select the root directory of the repo to be imported.
- **Repository Name**: Name of the imported repo.
- **Visibility**: Private or public.
- **Schedule**: If this is enabled, the imported repository is an image repository. The repository cannot commit code and can only be synced from the source repository periodically. The repository is updated every 24 hours, and the updated content is the content of the source repository 24 hours ago.
- Branch: Import the default branch or all branches of the source repo.
- **Automatically create check task**



Import Repository

Create Repository

Path: /

* Repository Name: Test_Search

Description: 0/2000

Initial Settings

☑ Automatically create check task (free of charge)

Visibility

● Private (can only be read, written, and accessed by repository members.)

○ Public (read-only)

Repo Sync Settings

* Branch

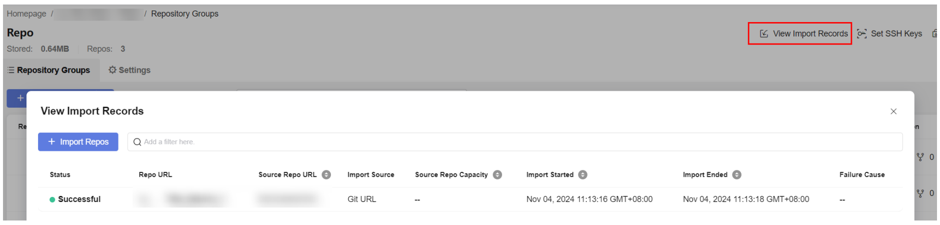
● Default branch

○ All branches

Schedule

□ Schedule sync into repo

Step 5 The repo is imported successfully. You can view key information such as the import status and failure cause in the import record list. After the repo is imported, check whether the repo is created successfully.



Homepage / [redacted] / Repository Groups

Repo

Stored: 0.64MB Repos: 3

View Import Records

Status	Repo URL	Source Repo URL	Import Source	Source Repo Capacity	Import Started	Import Ended	Failure Cause
● Successful	[redacted]	[redacted]	Git URL	--	Nov 04, 2024 11:13:16 GMT+08:00	Nov 04, 2024 11:13:18 GMT+08:00	--

----End

5 Merge Request

5.1 Why Is a Message Displayed Indicating Not Authorized When Merging an MR?

Symptom

On the MR details page, when a user clicks **Merge**, a message is displayed indicating **no permissions**.

Analysis

You must have the **Push** and **Merge** permissions for the target branch.

Solution

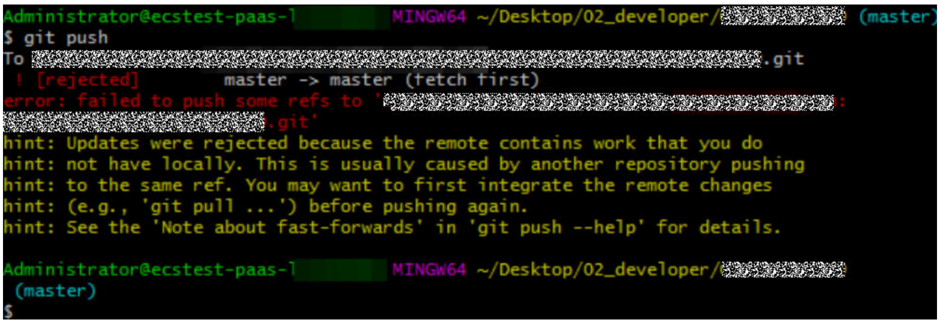
- If the target branch is a common branch, check whether you have these two permissions. If not, contact the administrator to add the permissions.
- If the target branch is a protected branch, go to the **Settings** page, choose **Policy Settings > Protected Branches**, and check whether you have the **Push** and **Merge** permissions. If not, contact the administrator to modify the branch configuration.

5.2 Error "failed to push some refs to '....git'" Is Reported When the Merge Request Is Committed Locally

Symptom

If two people modify the same line of code simultaneously, a code commit conflict may arise when the code is pushed to CodeArts Repo. As a result, the push fails and an error message "failed to push some refs to '....git'" is displayed, as shown in the following figure.

Figure 5-1 Error message displayed when pushing code



```
Administrator@ecstest-paas-l[redacted] MINGW64 ~/Desktop/02_developer/[redacted] (master)
$ git push
To [redacted].git
 ! [rejected]        master -> master (fetch first)
error: failed to push some refs to '[redacted]:
[redacted].git'
hint: Updates were rejected because the remote contains work that you do
hint: not have locally. This is usually caused by another repository pushing
hint: to the same ref. You may want to first integrate the remote changes
hint: (e.g., 'git pull ...') before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.

Administrator@ecstest-paas-l[redacted] MINGW64 ~/Desktop/02_developer/[redacted]
 (master)
$
```

Analysis

A conflict occurs when the same line of the same file is modified (the current version of the local repository is different from that of the CodeArts Repo).

Solution

To resolve a code commit conflict, pull the remote repository to the working directory in the local repository. Git will merge the changes and display the conflicting file content that cannot be merged. Then, modify the conflicting content and push it to the remote repository again (by running the **add**, **commit**, and **push** commands in sequence).

Modify the conflicting file carefully. If necessary, negotiate with the other member to resolve the conflict and avoid overwriting the code of other members by mistake.

NOTE

git pull combines **git fetch** and **git merge**. The following describes the operations in detail.

```
git fetch origin master # Pull the latest content from the master branch of the remote host.
git merge FETCH_HEAD    # Merge the latest content into the current branch.
```

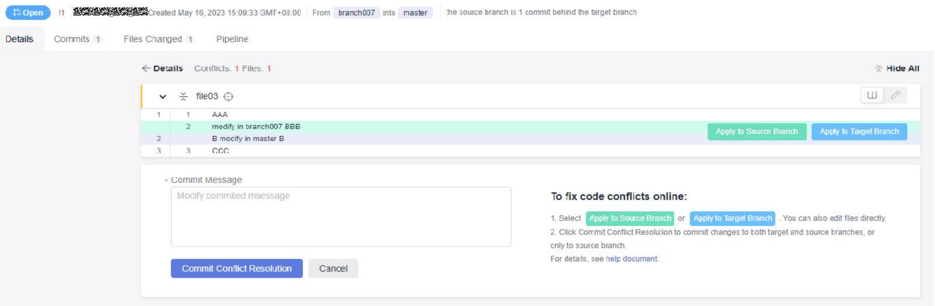
During merge, a message indicating that the merge fails due to a conflict is displayed.

Resolving Merge Conflicts on the Console

- **Fix them online** (recommended for small code volume)

  a. Click **Fix them online**. A code conflict is displayed, as shown in the following figure.

     On this page, you can directly select the **Apply to Source Branch** or **Apply to Target Branch** to select the modification of one party as the final repaired content.



Open !1 [redacted] Created May 16, 2023 15:09:33 GMT+08:00 From branch007 into master the source branch is 1 commit behind the target branch

Details Commits 1 Files Changed 1 Pipeline

← Details Conflicts. 1 Files. 1 Hide All

file03

```
1 1 AAA
  2 modify in branch007 BBB
2   B mocify in master B
3 3 CCC
```

* Commit Message

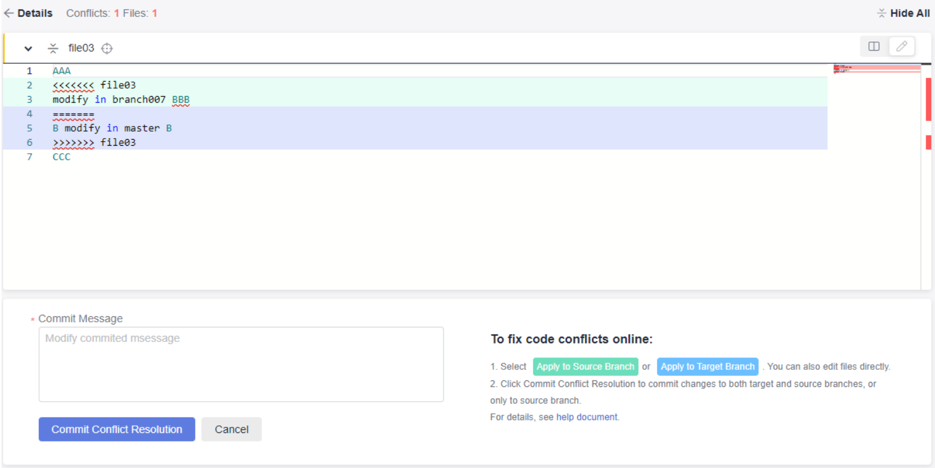
Modify commited msessage

To fix code conflicts online:

1. Select Apply to Source Branch or Apply to Target Branch . You can also edit files directly.
2. Click Commit Conflict Resolution to commit changes to both target and source branches, or cnly to source branch.

For details, see help document.

b. If the situation is complex and the problem cannot be solved by simple direct overwriting, click ✎ to enter the manual editing mode, as shown in the following figure.



```
AAA
<<<<<<< file03
modify in branch007 BBB
=======
B modify in master B
>>>>>>> file03
CCC
```

To fix code conflicts online:

1. Select Apply to Source Branch or Apply to Target Branch . You can also edit files directly.
2. Click Commit Conflict Resolution to commit changes to both target and source branches, or only to source branch.

For details, see help document.

c. Manually modify the code to resolve the conflict and commit the changes.

NOTE

Enter a commit message.

In the preceding figure, the following signs are used for conflict display and separation: <<<<, >>>>, and ====. Delete the lines where the signs are located when modifying code.

5.3 How Do I Resolve Local Code File Conflicts?

Symptom

On the details page of an MR, the message **Unresolved** is displayed.

Analysis

The same file is modified by two users and a conflict is reported when an MR is merged.

Solution

Step 1 Update the code and switch to the source branch of this MR.

```
git fetch origin
git checkout -b feature_010 origin/feature_010
```

Step 2 Merge the target branch into the source branch.

```
git merge origin/master
```

Step 3 Manually resolve conflicts locally as prompted.

Step 4 Commit code to the remote repository after conflicts are resolved.

```
git add .
git commit -m'Commit information'
git push origin feature_010
```

Step 5 Refresh the page and continue to review the MR.

----End

6 Fork Sync

6.1 How Do I Sync Code from a Primary Database to a Derived Database Generated by Fork?

Symptom

In CodeArts Repo, you can fork the code repository of the primary repository to other projects for improved collaborative development. However, when the fork mode is used for development, the following problem may occur: After the primary repository (source project) is updated, you need to manually update the forked repository.

You can perform the following operations to sync the code of the primary library to the derived library generated by fork.

For example:

Primary repository address: https://test.com/f2e22eeb1b8c43cfb764765f5e3ff039/AlphaGo/TestService.git

Forked repository address: https://test.com/f2e22eeb1b8c43cfb764765f5e3ff039/Roger/TestService.git

Branch name: master

Solution

Step 1 Clone the forked repo in your personal space to the local host.

```
git clone https://test.com/f2e22eeb1b8c43cfb764765f5e3ff039/Roger/TestService.git
cd TestService
```

Step 2 Add the remote original repository (primary repository) to the local host. (You can run the **git remote -v** command to view the remote repository list.)

```
git remote -v
```

Step 3 If there is no remote original repository, add the following information:

```
git remote -v
origin  https://test.com/f2e22eeb1b8c43cfb764765f5e3ff039/Roger/TestService.git (fetch)
origin  https://test.com/f2e22eeb1b8c43cfb764765f5e3ff039/Roger/TestService.git (push)
```

Step 4 Check the remote repository list.

```
git remote -v
origin  https://test.com/f2e22eeb1b8c43cfb764765f5e3ff039/Roger/TestService.git (fetch)
origin  https://test.com/f2e22eeb1b8c43cfb764765f5e3ff039/Roger/TestService.git (push)
main https://test.com/f2e22eeb1b8c43cfb764765f5e3ff039/AlphaGo/TestService.git (fetch)
main https://test.com/f2e22eeb1b8c43cfb764765f5e3ff039/AlphaGo/TestService.git (push)
```

Step 5 Obtain the latest branch code of the original repository (primary repository) to the local host and merge the code of the two versions.

```
git pull main master
```

Step 6 Sync the latest merged code to the fork repository.

```
git push origin master
```

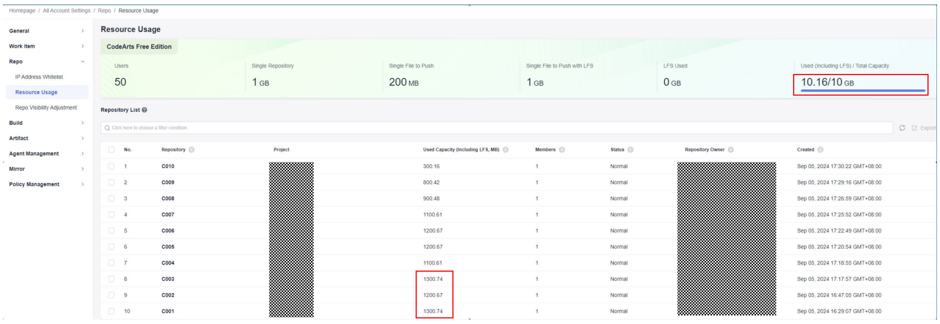
----End

7 Repository Capacity

7.1 Insufficient Remaining Repository Capacity

Scenario

If the capacity warning notification is configured and enabled, a notification will be sent when the repository capacity usage reaches a specified threshold. You can click the avatar and choose **All Account Settings** > **Repo** > **Resource Usage** to view the capacity usage. As shown in the following figure, the repo capacity usage has reached 10.16 GB, exceeding the maximum capacity.



Users 50
Single Repository 1 GB
Single File to Push 200 MB
Single File to Push with LFS 1 GB
LFS Used 0 GB
Used (Including LFS) / Total Capacity 10.16/10 GB

No.	Repository	Project	Used Capacity (Including LFS, MB)	Members	Status	Repository Owner	Created
1	C010	[redacted]	300.16	1	Normal	[redacted]	Sep 05, 2024 17:30:22 GMT+08:00
2	C009	[redacted]	800.42	1	Normal	[redacted]	Sep 05, 2024 17:29:16 GMT+08:00
3	C008	[redacted]	900.48	1	Normal	[redacted]	Sep 05, 2024 17:26:59 GMT+08:00
4	C007	[redacted]	1100.61	1	Normal	[redacted]	Sep 05, 2024 17:25:52 GMT+08:00
5	C006	[redacted]	1200.67	1	Normal	[redacted]	Sep 05, 2024 17:22:49 GMT+08:00
6	C005	[redacted]	1200.67	1	Normal	[redacted]	Sep 05, 2024 17:20:54 GMT+08:00
7	C004	[redacted]	1100.61	1	Normal	[redacted]	Sep 05, 2024 17:18:55 GMT+08:00
8	C003	[redacted]	1300.74	1	Normal	[redacted]	Sep 05, 2024 17:17:57 GMT+08:00
9	C002	[redacted]	1200.67	1	Normal	[redacted]	Sep 05, 2024 16:47:05 GMT+08:00
10	C001	[redacted]	1300.74	1	Normal	[redacted]	Sep 05, 2024 16:29:07 GMT+08:00

Analysis

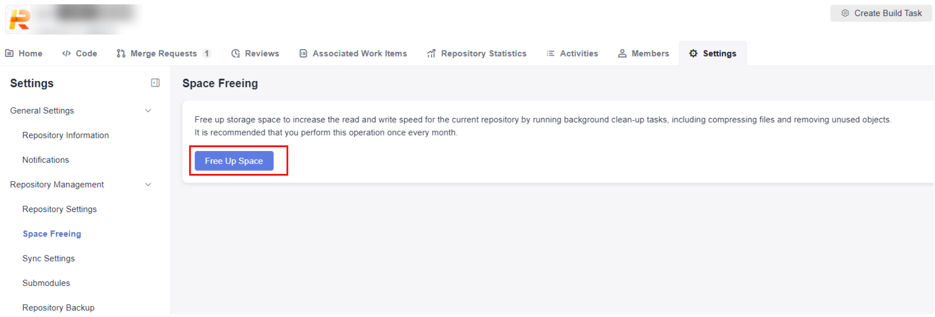
Some files may have occupied repository space.

Solution

- Method 1: Compress the repository to reduce the capacity usage. The procedure is as follows:

Step 1 Click the avatar and choose **All Account Settings** > **Repo** > **Resource Usage** to view the repositories with large capacity usage.

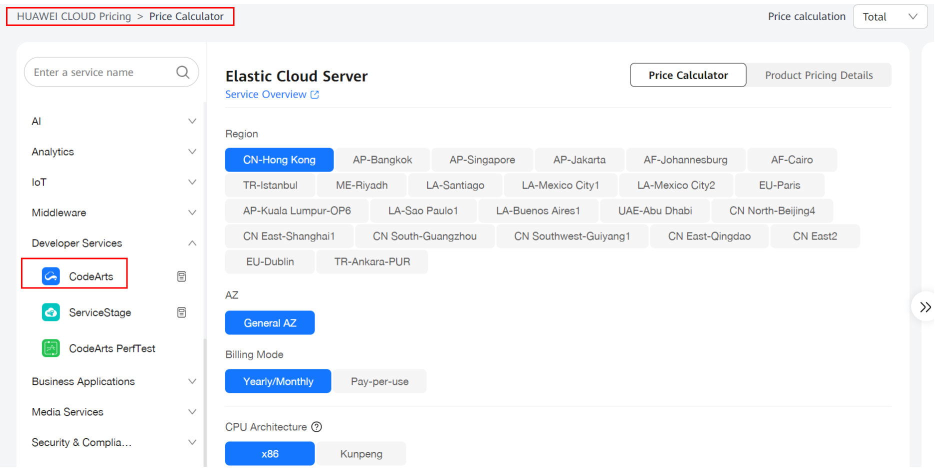
Step 2 Go to the repository homepage where a large amount of space is occupied, choose **Settings** > **Repository Management** > **Space Freeing**, and compress the repository that occupies a large amount of space to reduce the repository capacity.



Free up storage space to increase the read and write speed for the current repository by running background clean-up tasks, including compressing files and removing unused objects.
It is recommended that you perform this operation once every month.

----End

- Method 2: Purchase a storage expansion package for more capacity. On the Huawei Cloud official website, choose **Pricing > Price Calculator > Developer Services** and select the region which you want to purchase the package for.



8 FAQs

Does CodeArts Repo Support SVN?

No. CodeArts Repo is a Git-based distributed version control service that enables easier remote collaboration.

Does CodeArts Repo Support One-Click Import of External Git Sources?

Yes.

CodeArts Repo supports import of Git repositories from the following sources:

- bitbucket.org
- code.aliyun.com
- coding.net
- git.qcloud.com
- gitee.com
- github.com
- gitlab.com
- visualstudio.com
- xiaolvyun.baidu.com

Does CodeArts Repo Support Batch Download of Multiple Repository Packages?

No. CodeArts Repo does not support batch download or upload of multiple code repositories. You need to perform operations on each code repository one by one. To back up local repositories, an administrator can use Shell or batch processing commands to download multiple repositories.

How Do I Obtain the Path for Storing Downloaded Code in CodeArts Repo?

- If Git is installed in the default path and you open Git Bash from the start menu, downloaded code is stored in the default path **C:/User/*XX user***.
- If you open Git Bash by right-clicking in a directory and choosing **Git bash Here**, downloaded code is stored in this directory.

How Do I Obtain the Clone Address of a Repository in CodeArts Repo?

Step 1 On the CodeArts Repo homepage, click the name of a repository in the repository list. The repository details page is displayed.

Step 2 Click **Clone/Download** in the right navigation bar. Click **Clone with SSH** to obtain the SSH protocol address. Click **Clone with HTTPS** to obtain the HTTPS protocol address.

----End

Does the Repository Support Online Decompression When Uploading ZIP Archives?

No. Online decompression is no longer supported. You are advised to decompress the file package locally and use the Git command to upload the files.

Does the Repository Support Private or Public Conversions?

Yes. On the repository details page, click the **Settings** tab, and choose **General Settings** > **Repository Information** > **Visibility**.

Can All Users Upload and Download Code with the Same SSH Key?

No. An SSH key is used to establish a secure connection between a computer and CodeArts Repo. Different users should configure an SSH key on their own computers before connecting to CodeArts Repo via SSH.