

CodeArts Pipeline

User Guide

Issue 01
Date 2023-11-30



Copyright © Huawei Technologies Co., Ltd. 2023. All rights reserved.

No part of this document may be reproduced or transmitted in any form or by any means without prior written consent of Huawei Technologies Co., Ltd.

Trademarks and Permissions



HUAWEI and other Huawei trademarks are trademarks of Huawei Technologies Co., Ltd.

All other trademarks and trade names mentioned in this document are the property of their respective holders.

Notice

The purchased products, services and features are stipulated by the contract made between Huawei 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.

Security Declaration

Vulnerability

Huawei's regulations on product vulnerability management are subject to "Vul. Response Process". For details about the policy, see the following website:<https://www.huawei.com/en/psirt/vul-response-process>
For enterprise customers who need to obtain vulnerability information, visit:<https://securitybulletin.huawei.com/enterprise/en/security-advisory>

Contents

1 Pipeline Management.....	1
1.1 Logging In to CodeArts Pipeline.....	1
1.2 Creating or Copying a Pipeline.....	2
1.3 Configuring a Pipeline.....	4
1.4 Executing a Pipeline.....	11
1.5 Viewing a Pipeline.....	11
1.6 Managing Groups.....	12
1.7 Configuring a Pipeline Template.....	14
1.8 Managing Parameters.....	16
1.8.1 Background.....	16
1.8.2 Configuring Parameters.....	16
1.8.3 Using Parameters.....	17
2 Permissions.....	20
2.1 Introduction.....	20
2.2 Tenant-level Permissions.....	21
2.3 Resource-level Permissions.....	22
3 Rules and Policies.....	24
3.1 Overview.....	24
3.2 Configuring a rule.....	24
3.3 Tenant-level Policies.....	25
3.4 Project-level Policies.....	27
3.5 Policies and Pipelines.....	28
4 Extensions.....	29
4.1 Overview.....	29
4.2 Official Extensions.....	30
4.3 Custom Extensions.....	32
4.3.1 Getting Started.....	32
4.3.2 Registering a Custom Extension.....	33
4.3.3 Developing Extensions Using Low Code.....	38
4.4 Extensions and Policies.....	41
4.5 Jenkins Tasks.....	41
4.6 KubernetesRelease Extension.....	42

4.6.1 Introduction.....	42
4.6.2 Blue-Green Upgrade.....	43
4.6.3 CCE Rolling Upgrade.....	47
5 Microservice Management.....	52
5.1 Microservice Entry.....	52
5.2 Creating a Microservice.....	53
5.3 Viewing a Microservice.....	53
6 Change Management.....	55
6.1 Change Entry.....	55
6.2 Creating Changes.....	56
6.3 Viewing Changes.....	56
6.4 Changes and Pipelines.....	57
7 Service Endpoints.....	61

1 Pipeline Management

1.1 Logging In to CodeArts Pipeline

This section describes how to access **Pipelines** tab page and how to perform related operations on the page.

Procedure

- Step 1** Log in to the CodeArts homepage.
- Step 2** On the top navigation bar, choose **Services > Pipeline**.
- View the pipelines.

Parameter	Description
Name	Project name/pipeline name.
Last Executed	Information about the most recently executed pipeline, including the branch, and ID and description of the latest code commit.
Execution Type	Execution mode and executor of the latest pipeline.
Workflow	Displays the scheduling process and execution status (successful, failed, running, or stopped) of the pipeline.
Started & Lasted	Displays the start time and duration of the last execution.
Operation	Click ▶ in the Operation column to execute the pipeline. Click ⋮ to edit, copy, or delete the pipeline and view the execution history (creation and deleting). Click ☆ to follow the pipeline. After the pipeline is followed, the icon changes to ★. You can click the icon again to unfollow the pipeline.

 NOTE

- By default, all users can view the pipeline task list.
- Click the drop-down list box next to **Create Pipeline** to filter pipelines by **All pipelines**, **My created pipelines**, or **My executed pipelines**.
- You can enter a pipeline name in the search box.
- Click **Create Pipeline** to create a pipeline. For details, see [Creating or Copying a Pipeline](#).
- (Optional) Click a pipeline name to view the pipeline execution history.
- (Optional) Click the name of a project to which the pipeline belongs to access the project-specific pipeline list.

Step 3 Switch to the **Templates** tab page.

You can view and manage system and custom templates. For details, see [Configuring a Pipeline Template](#).

----End

1.2 Creating or Copying a Pipeline

Procedure

Step 1 [Log in to CodeArts Pipeline](#).

Step 2 On the **Pipelines** tab page, click **Create Pipeline**.

On the **Basic Information** page, select a project, enter the pipeline name, select a pipeline source, and configure related parameters. [Table 1-1](#) describes the pipeline source and related parameters.

Table 1-1 Parameter description

Pipeline Source	Parameter	Description
Repo provides comprehensive code hosting services for enterprises and Git-based online code hosting services for software developers.	Repository	Select an available source code repository.
	Default Branch	Default branch used when a pipeline is manually or periodically executed.
	CodeArts Repo HTTPS Authorization	Configure authorization endpoints to elevate repository operation permissions. This function is used for microservice change pipelines and some repository operation extensions.

Pipeline Source	Parameter	Description
	Alias	If an alias is specified, predefined parameters are generated for the repository and displayed on the Parameter Configuration tab page. If a repository alias is configured when a user selects Repo as the pipeline source, the system-predefined parameter corresponding to the repository is generated on the Predefined Parameters tab page. Otherwise, the system-predefined parameter is not generated.
	Description	Description of the pipeline.
Git The repository configured in the service endpoint can be accessed through the common Git service endpoint.	Service Endpoints	Select an existing Git service endpoint or click Create to create a Git service endpoint .
	Repository	Select an available source code repository.
	Default Branch	Default branch used when a pipeline is manually or periodically executed.
	Description	Description of the pipeline.

 **NOTE**

- If a pipeline task does not need to be associated with a code repository, select **None** for **Pipeline Source**.

Step 3 Configure the information and click **Next**.

On the **Template** page, select a template to quickly create a pipeline. You can also select **Blank Template** to directly create a task from scratch.

Step 4 Select a template and click **OK**. On the **Task Orchestration** tab page, [configure a pipeline](#) and click **Save**.

Step 5 (Optional) Clone a pipeline in one of the following ways:

- On the **Pipelines** tab page, search for the target pipeline, click **...** in the **Operation** column, and click **Copy**.
- Click the name of the target pipeline, on the **Execution History** page that is displayed, click **...** in the upper right corner and click **Copy**.
- Click the name of the target pipeline, then click the **Pipeline Details** tab or click the execution ID to go to the **Pipeline Details** page. Click **...** in the upper right corner and click **Copy**.

On the pipeline copy page, you can add configurations based on the original pipeline as required. For details, see [Configuring a Pipeline](#).

----End

1.3 Configuring a Pipeline

Procedure

Step 1 [Log in to CodeArts Pipeline](#).

Step 2 On the **Pipelines** tab page, search for the target pipeline, click **...** in the **Operation** column, and click **Edit**.

On the displayed page, you can modify the basic information (name and description), [task orchestration](#), [parameter configuration](#), [execution plan](#), [permissions](#), and [notifications](#).

Step 3 Click **Save**.

----End

Task Orchestration

On the **Task Orchestration** tab page, you can configure the pipeline source, stages, pass conditions, and tasks.

- **Configuring a Pipeline Source**

Click the area where the repository is located in the pipeline source stage. The **Edit Pipeline Source** dialog box is displayed. You can modify the pipeline source information.

- **Configuring a Stage**







On the **Task Orchestration** tab page, click  or [+ Stage](#) to add a stage to the pipeline. After a stage is added, you can edit, delete, copy, or move the stage.

Table 1-2 Configuring a stage




Operation	Description
Editing a stage	Click  . In the dialog box displayed, you can configure the stage name and whether the stage is always running. NOTE Always Running: Option Yes will select all the tasks in this stage and execute them, which cannot be canceled. Option No will select the tasks in this stage but you can cancel their execution.
Deleting a stage	Click  and confirm the deletion as prompted.
Copying a stage	Click  to copy a pipeline stage.

Operation	Description
Sorting stages	Click  to adjust the stage sequence.
Setting stage entry	Click  . In the dialog box displayed, configure the stage entry type. <ul style="list-style-type: none"> • Automatic (default): The pipeline automatically enters this stage and continues to run. • Manual: The pipeline can continue to run only after manual confirmation.

- **Configuring a Task**


After a stage is added, you can add tasks to each stage. After a task is added, you can edit, copy, delete, or move the task.

Table 1-3 Configuring a task


Operation	Description
Adding a task	<ul style="list-style-type: none"> • Click + Job to add a task to an empty stage. • Click  under a task to add a task orchestrated with the task in serial mode. • Click + Parallel job to add a task orchestrated with an existing task in parallel mode. <p>NOTE</p> <ul style="list-style-type: none"> • Serial execution: Tasks are executed in sequence. For example, build tasks and deployment tasks must be executed sequentially. • Parallel execution: Tasks are executed at the same time. For example, a code check task and a build task can be executed at the same time.
Editing a task	Click a task card to edit the task.
Copying a task	Click  on the task card to copy a task orchestrated with this task in serial mode.
Deleting a task	Click  on the task card and confirm the deletion as prompted.
Sorting tasks	Click, hold, and move a task card to adjust the task sequence. <p>NOTE</p> The sequence cannot be adjusted when tasks are executed in parallel.

When you add or edit a task, a dialog box is displayed. You can configure an extension for the task.

Table 1-4 Configuring an extension for a task

Operation	Description
Adding an extension	<p>There are five types of extensions for build, code check, deployment, test, and general purposes. You can filter or search for extensions by type. Move the cursor to the extension card and click Add to add the extension to a task.</p> <p>Set parameters as required.</p> <ul style="list-style-type: none">• Enter an extension name.• Select the task to be invoked. If no proper task is available, create a task as prompted.• If the invoked task has parameters, the parameters are displayed. Set the parameters as required.• The extension name is followed by a flag. Only one extension with flag <i>Job</i> can be added to a single task. Extensions with flag <i>draft</i> indicate that they are draft extensions.• The extension for suspending a pipeline can only be added to stages that do not contain parallel tasks.
Deleting an extension	<p>Move the cursor to the extension card and click  and select Delete to delete the extension.</p>
Sorting extensions	<p>Click, hold, and move an extension card to adjust the extension sequence.</p>

- **Configuring a Pass Condition**

Click  **Pass Conditions** under a stage. In the dialog box displayed, move the cursor to the pass condition cards and click **Add** to add a pass condition for the current stage and a policy for the condition.

- Currently, only standard policy pass conditions can be added.
- Policy: Add standard policy pass conditions. You can select those already created for the current project or tenant.

A policy is a set of rules. Each rule corresponds to a condition template of the output metric value in the extension. By predefining a policy, you can easily apply the same pass condition to multiple pipelines. For details, see [Overview](#).

 **NOTE**

- A pass condition can be set exclusively for each stage of a pipeline and is valid only for the corresponding stage.
- Multiple pass conditions can be configured for the same stage.

Setting Parameters

Switch to the **Parameter Configuration** tab page, and add parameters for the pipeline. For details, see [Managing Parameters](#).

NOTE

Pipeline parameters include custom and system-predefined parameters. The custom parameter types include string, enumeration, and auto-increment.

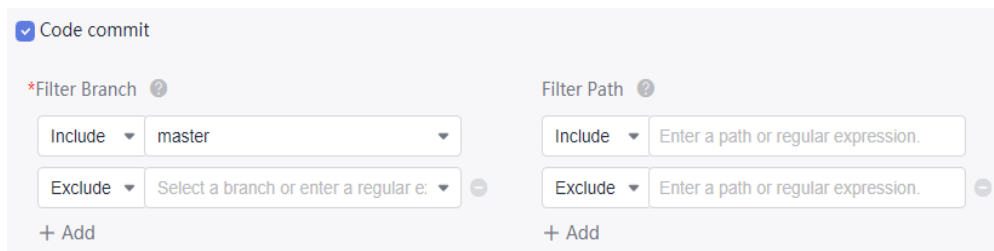
Configuring an Execution Plan

Switch to the **Execution Plan** tab page, and configure an execution plan (event trigger or scheduled task) for the pipeline. Triggering events include code commits, merge request, and tag creation.

- Triggered upon code commits (Repo)

The related pipeline is automatically executed when code is committed in an associated code repository, and the related branch and path of changed files meet inclusion and exclusion policies.

 - Branch inclusion: If the target branch to which the code is committed is included, the matching is successful.
 - Branch exclusion: If the target branch to which the code is committed is excluded, the matching fails.
 - Path inclusion: If any changed file (if path exclusion is configured, the changed file must not be excluded) is included, the matching is successful.
 - Path exclusion: If all changed files are excluded, the matching fails.



- Triggered upon merge request (Repo)

Merge request event triggering: monitors events triggered by an MR. The related pipeline is automatically executed when code is committed in an associated, updated, merge, or re-opened event in the associated code repository, and the related branch and path of changed files meet inclusion and exclusion policies.

 - Branch inclusion: If the requested target branch is included, the matching is successful.
 - Branch exclusion: If the requested target branch is excluded, the matching fails.
 - Path inclusion: If any changed file (if path exclusion is configured, the changed file must not be excluded) is included, the matching is successful.
 - Path exclusion: If all changed files are excluded, the matching fails.

The screenshot shows the configuration for a 'Merge Request' trigger. It includes an 'Event' section with checkboxes for 'Create', 'Update', 'Merge', and 'Reopen', where 'Create', 'Update', and 'Reopen' are checked. Below this is a '*Filter Branch' section with two rows: one for 'Include' with a dropdown set to 'master', and another for 'Exclude' with a dropdown set to 'Select a branch or enter a regular e:'. There is also a 'Filter Path' section with two rows for 'Include' and 'Exclude', both with dropdowns set to 'Enter a path or regular expression.'. Each row has a '+ Add' button below it.

- Triggered upon tag creation (CodeArts Repo)
The related pipeline is automatically executed when a tag is created in an associated code repository and the created tag meets inclusion and exclusion policies.
 - Tag inclusion: If the tag created in the code repository is included in the tag, the matching is successful.
 - Tag exclusion: If a tag created in the code repository is in the exclusion tag, the matching fails.

The screenshot shows the configuration for a 'Tag creation' trigger. It features a '*Filter Tag' section with two rows: one for 'Include' with a dropdown set to 'Include' and a text input containing an asterisk (*), and another for 'Exclude' with a dropdown set to 'Exclude' and a text input containing 'Enter a tag or regular expression.'. A '+ Add' button is located at the bottom left.

NOTE

- The branch is matched first, and then the path (if configured) is matched. If the matching is successful, the pipeline is triggered.
 - Branch exclusion takes precedence over branch inclusion. That is, when the target branch is included and excluded at the same time, the matching fails.
 - Path exclusion takes precedence over path inclusion. That is, the excluded paths are matched first. If not all the changed files are excluded, the included paths are matched. If the included path is not configured, the matching is successful. If the included path is configured and any of the changed files is included, the matching is successful.
 - Tag exclusion takes precedence over tag inclusion. That is, if a tag is included and excluded at the same time, the matching fails.
- **Scheduled Execution**
Click to add a scheduled task. Click **Enable**, set date and time, and click **OK**. The trigger takes effect after the pipeline is saved. The pipeline is executed at the specified time.

*Enable



*Run On

- Sunday Monday Tuesday Wednesday Thursday Friday
 Saturday

*Time

00:00 (UTC+08:00) Beijing, Chongqing, Hong Kong, ...

00	00
01	01
02	02
03	03
04	04
05	05
06	06
07	07

NOTE

You can add a maximum of 10 scheduled tasks.

Managing Permissions

Switch to the **Permissions** tab page, and configure permissions for the pipeline as required, including role permissions and user permissions.

- If the role permissions are not changed, they are the same as those in the project settings.
- The permissions of the project creator and pipeline creator cannot be changed.
- User permissions take precedence over role permissions.

NOTE

By default, user permissions are automatically synchronized after role permissions are configured for a user. If user permissions are configured, the user permissions overwrite the role permissions of the user.

Role Permissions

You can click or to specify whether a role has the permissions to view, execute, edit, and delete pipelines.

Role Permission		Member Permission			
Role	View	Execute	Edit	Delete	
Project creator	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
Pipeline creator	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
Project manager	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
Developer	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Test manager	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Tester	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Participant	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Viewer	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Operation manager	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

User Permissions

You can click or to specify whether a user has the permissions to execute, edit, and delete pipelines.

Role Permissions		User Permissions					
User	Role	Alias	Enterprise User	View	Execute	Edit	Delete
	Pipeline creator	pipeline		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

NOTE

- The view permission allows you to view the pipeline running details.
- The edit permission allows you to orchestrate pipeline tasks.

Notification Subscription

Switch to the **Notifications** tab page and configure event notifications for the pipeline as required. When a pipeline is deleted, fails to run, or is successfully run, or configurations are updated, you can configure pop-up notifications and email to notify related personnel.

Click to enable the notification or click to disable it.

Name	Notification Method		Default Users		
Pipeline deleted	<input checked="" type="checkbox"/> Pop-up Notifications	<input type="checkbox"/> Email	<input checked="" type="checkbox"/> Creator	<input checked="" type="checkbox"/> Executor	<input checked="" type="checkbox"/> Follower
Failed to run the pipeline	<input checked="" type="checkbox"/> Pop-up Notifications	<input type="checkbox"/> Email	<input checked="" type="checkbox"/> Creator	<input checked="" type="checkbox"/> Executor	<input checked="" type="checkbox"/> Follower
Pipeline run successfully	<input checked="" type="checkbox"/> Pop-up Notifications	<input type="checkbox"/> Email	<input checked="" type="checkbox"/> Creator	<input checked="" type="checkbox"/> Executor	<input checked="" type="checkbox"/> Follower
Pipeline configurations updated	<input checked="" type="checkbox"/> Pop-up Notifications	<input type="checkbox"/> Email	<input checked="" type="checkbox"/> Creator	<input checked="" type="checkbox"/> Executor	<input checked="" type="checkbox"/> Follower


NOTE

- By default, pop-up notifications are sent, and email notifications are not sent.
- You can click in the upper right corner of the pipeline homepage and view the notification messages on the **Notice** dialog box.

1.4 Executing a Pipeline

Procedure

Step 1 [Log in to CodeArts Pipeline.](#)

Step 2 On the **Pipelines** tab page, click  in the **Operation** column.

Step 3 In the displayed **Execution Configuration** dialog box, perform the following operations:

- **Pipeline Source:** Select the branch or tag of the pipeline source as required.
- **Runtime Parameters:** If runtime parameters have been set for the pipeline, set the parameters as required and then save them. For details, see [Using Parameters](#).
- **Execution Stages:** Select one or more tasks to execute in the pipeline. By default, all tasks are executed.

 **NOTE**

If **Always Run** is set to **Yes** for a stage during configuration, tasks in this stage are selected by default and cannot be canceled.

- **Description:** Enter the debugging information about the execution.

Step 4 After the configuration is complete, click **Run** to view the pipeline execution progress and task execution status in real time on the pipeline execution details page.

- Click **Stop** in the upper right corner to stop the task.
- Pipelines can be executed in parallel. You can click **Run** to continue executing a pipeline.

Step 5 After the execution is complete, you can view the execution result.

----End

1.5 Viewing a Pipeline

On the pipeline details page, you can view basic pipeline information and perform operations on pipelines as needed.

Procedure

Step 1 [Log in to CodeArts Pipeline.](#)

Step 2 On the **Pipelines** tab page, search for the target pipeline and click the task name. The pipeline **Execution History** page is displayed.

You can click the time filter in the upper right corner to view execution records by time. By default, executions in the past 7 days are displayed. You can switch to 14 days and 31 days.

 NOTE

During the first execution, the execution histories page is empty. Records are generated only after the first execution.

Step 3 Click the execution ID to access the **Pipeline Details** page.

Click **...** in the upper right corner to edit, copy, and delete pipelines, and view pipeline operation history (creation and editing operations) as required.

Perform the following operations on the **Pipeline Details** page:

Operation	Description
Retry	If the task fails, you can click Retry in the upper right corner to continue the task.
Run	Click Run in the upper right corner to execute the pipeline again with the latest configuration and generate an execution record.
Download	Click Download next to Output to download the build package automatically built using the pipeline. NOTE <ul style="list-style-type: none">• A build package is available only for build tasks.• If there are multiple build packages, click Download All.• Only the latest 10 build packages are displayed. To download other build packages, switch to the Release Repos page.
View logs	Click a task card to view the task log. NOTE No log will be generated for tasks of DelayedExecution and PipelineSuspension .
More operations	Click ... in the upper right corner of the page to edit , copy , and delete pipelines, and view pipeline operation history (creation and editing operations) as required. NOTE By default, only project managers and creators (project creators or pipeline creators) can delete pipelines. You can configure permissions for different roles.

----End

1.6 Managing Groups


Application Scenarios

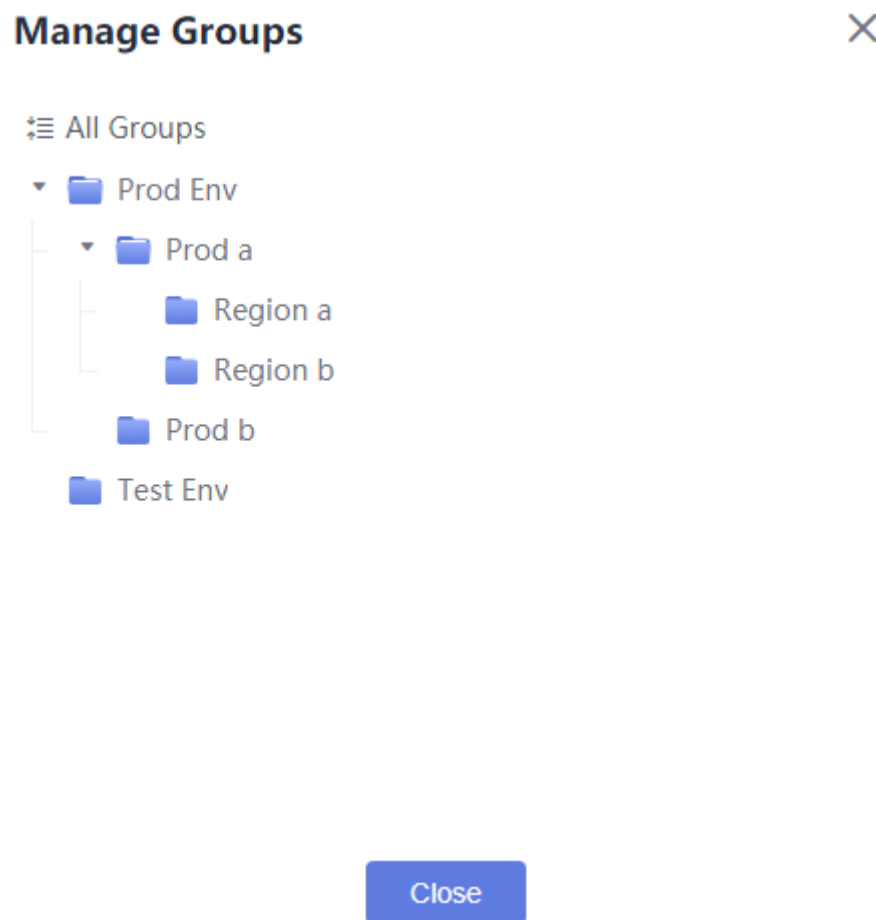
You can group and manage pipelines with the same features in a unified manner, improving operation efficiency. For example, you can group pipelines into production and test by environment or into scheduled build, development self-test, integration test, and production and deployment by R&D stage.




Prerequisites

- Only project creators and project managers can manage groups.
- This grouping function applies only to pipeline jobs of the new version.




Procedure

1. Go to the pipeline page of the project.
2. Click **All Groups** to expand the pipeline group panel.
3. Click . The **Manage Groups** dialog box is displayed.



4. Move the cursor to the row where **All Groups** is located and click  to add a group.
5. Enter the group name. Click  to confirm group creation or click  to cancel group creation.

After a group is created, you can perform the following operations:

- Click  in the row where the group is located to create a subgroup. You can create a maximum of three levels of subgroups.
- Click  in the row where the group is located to change the group name.
- Click  in the row where the group is located to move or delete the group.

 NOTE

Ungroup is also automatically generated for ungrouped pipelines along with the first created group.

6. Click **Close** to return to the pipeline list page after all groups are created. You can move pipelines to the corresponding group as required.
 - a. Select the pipeline to be grouped. The following dialog box is displayed at the bottom of the page.



- b. Click **Move To**. The **Move Group** dialog box is displayed. You can move the selected pipeline to the corresponding group.

1.7 Configuring a Pipeline Template

You can configure a pipeline template to quickly generate a pipeline. You can access the template page in the following ways.

- Homepage: Access CodeArts Pipeline. On the **Pipelines** tab page that is displayed, switch to the **Templates** tab page.
- Project: Access the pipeline of a project and choose **More > Templates** in the upper right corner.






 NOTE

The template in **Templates** corresponds to the template selected during **pipeline creation**.

Template Types

CodeArts Pipeline supports system and custom templates.

The operations on the template list are as follows:

Operation	Description
	Click this icon and the Create Pipeline page is displayed.
	Click this icon to follow a template. After a template is followed, the icon changes to  . You can click  to unfollow the template.
	<ul style="list-style-type: none">• Click this icon and select Edit. On the Task Orchestration tab page that is displayed, you can edit the template as required.• Click this icon and select Copy. On the Task Orchestration tab page that is displayed, you can copy the template as required.• Click this icon and select Delete to delete the template as prompted.

NOTE

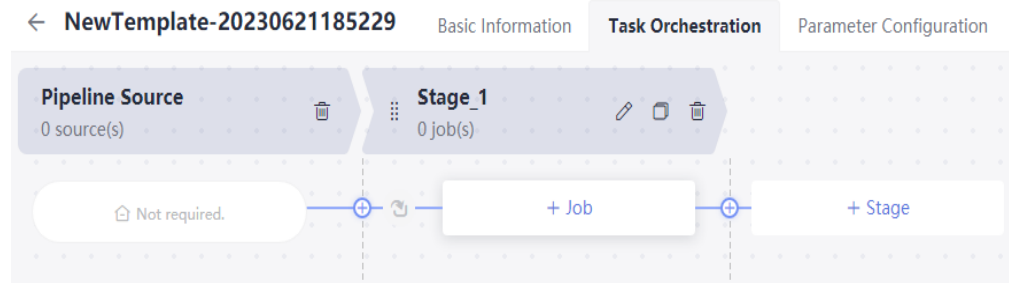
System templates are used to copy or generate pipelines. They cannot be edited or deleted.

Creating or Configuring a Template

1. Access the **Templates** tab page.
2. Click **Create Pipeline Template**. The **Task Orchestration** tab page is displayed.


Configure basic template information, template stages/tasks, and template parameters as required.

- **Basic Information:** Specify the template name (mandatory), language (Java, Python, Node.js, Go or None (default)), and description (optional).
- **Pipeline Source:** You do not need to set this parameter.
- **Stage Entry:** Stage entry is not supported during template orchestration.
- **Task Orchestration:** Currently, pipeline stages and some extensions can be added to a pipeline template. After tasks such as build, code check, deployment, and API test are configured in the template, corresponding tasks will be created when a pipeline is created using this template.
- **Parameter Configuration:** Switch to the **Parameter Configuration** tab page and add parameters to the template as required. Pipeline template parameters include custom and predefined parameters. Custom parameters include string, enumeration, and auto-increment types. For details about how to configure parameters, see [Managing Parameters](#).



3. After configuring the information, click **Save**.

Creating a Pipeline

1. On the **Templates** page, search for the target template, and click  in the **Operation** column.
2. On the **Basic Information** page, select a project, enter a pipeline name, select a pipeline source as needed, set related parameters, and click **OK**.
3. The **Task Orchestration** page is displayed.
 - You can customize the template content or directly click **Save**.
 - For the pipeline generated using the template, tasks mounted to it are automatically generated by the system. To mount custom tasks to the pipeline, manually add them after the pipeline is generated.
4. After configuring the information, click **Save**.

1.8 Managing Parameters

1.8.1 Background

Pipeline parameters are transferred to each task to implement data transition between tasks. You only need to configure pipeline parameters to streamline data of build, deployment, and API test tasks. The options are as follows:

- **Custom Parameters:** Add parameters as required. Parameter types include string, enumeration, and auto-increment.
- **Predefined Parameters:** The parameter values are generated and cannot be deleted or modified.

 **NOTE**

- For Repo pipeline sources, if you enter a repository alias, repository-related predefined parameters are generated based on predefined parameters.

1.8.2 Configuring Parameters

You can create and configure pipeline parameters.

Procedure


Step 1 [Log in to CodeArts Pipeline](#).

Step 2 Search for the target pipeline, click **...** in the **Operation** column, and click **Edit**.

Step 3 Switch to the **Parameter Configuration** tab page.

The following parameters are provided.

Basic Information	Description
Create now	Click Create now in the parameter list to add a parameter.
Name	Parameter name. You can change the parameter name. NOTE The name of a custom parameter cannot be the same as that of a predefined parameter.
Parameter Type	Parameter types include string, auto-increment, and enumeration.
Default	Enter or select a parameter value.
Runtime Setting	If Runtime Setting is enabled, the value of this parameter can be changed during pipeline execution.

Basic Information	Description
Private Parameter	If a parameter is private, the system encrypts the input for storage and only decrypts the parameter when using it. Private parameters are not displayed in run logs.
Parameter Description	Description of a parameter.
Operation	Click  to delete a parameter.

Step 4 Add and configure parameters.


Click **Create now** in the parameter list area to set the name, type (**String** by default), and value of a new parameter. You can also set it as a private parameter or enable or disable runtime setting.

- **String**

The parameter value is a string. You can customize the value in the **Default** column, and enable **Private Parameter** or **Runtime Setting**.

- **Enumeration**

After you select **Enumeration**, the **Enumeration** dialog box is displayed. You can set **Optional value**.

After the setting is complete, click the **Default** drop-down list box to select a value. Alternatively, click  next to the parameter value to change the value.

- **Auto-increment**

The parameter value is a string and defaults to **1.0.0**. If such a parameter is referenced by a task in the pipeline, the value of this parameter is incremented by 1 each time the pipeline is run.

 **NOTE**

If the value of an auto-increment parameter does not end with a digit, the value does not increase automatically after an execution.

----End

1.8.3 Using Parameters

This section describes how to configure the **releaseversion** parameter in a pipeline and transfer the parameter to a build task. The operations for other tasks are the same.

Procedure

Step 1 Create a build task.

Step 2 On the **Parameter Configuration** tab page, add the **releaseversion** parameter, set the default value, and enable **Runtime Setting**.

Step 3 On the **Build Actions** tab page, select **Upload to Release Repos** and set **Release Version** as a reference parameter. After you enter **\$** in the text box, a parameter list is displayed. Select the **releaseversion** parameter created in the previous step.

Step 4 Save the build task.

Step 5 Create a blank template pipeline, add the **Build** extension and select the build task created in previous steps. The new build task parameter **releaseversion** is displayed.

Step 6 Move the cursor to the **releaseversion** parameter to set it as a pipeline parameter. Alternatively, click **OK**, switch to the **Parameter Configuration** tab page, create the pipeline parameter **releaseversion**, set **Type** to **Auto-increment** or **String**, set a default value, and enable **Runtime Setting**.

Name	Type	Default	Runtime Setting	Private Parameter	Description	Operation
releaseversion	String	1.0.1	<input checked="" type="checkbox"/>	<input type="checkbox"/>		

Step 7 Switch back to the **Task Orchestration** tab page, and edit the added build task. Reference the pipeline **releaseversion** parameter to the build task **releaseversion** parameter using **\$**.

 **NOTE**

- Only text parameters for which **Runtime Setting** is enabled are displayed.
- The parameter reference format is "\${ParameterName}". *ParameterName* specifies the pipeline parameter name. After you enter \$ in the parameter text box, the parameter list is automatically displayed.
- You can move the cursor to a parameter name to quickly set the parameter as a pipeline parameter and directly reference the parameter.

Step 8 Save the information and click **Save and Run**. In the displayed dialog box, you can view the running configuration.

For the runtime parameter, the value is the default value specified when you add the parameter. You can change the value as required. When the pipeline is run, the parameter value used in the build task is the value you set here.

Step 9 Click **Run** to execute the pipeline.

----End

2 Permissions

2.1 Introduction

In CodeArts Pipeline, permissions are controlled at three layers from top to bottom to manage user behavior of different modules and granularities.

Layer	Module	Description
Tenant (all accounts)	Extensions, tenant-level policies, tenant-level rules, and pipeline templates	Controls resources of all modules under a tenant. It is configured in IAM and takes effect in all projects of the tenant.
Project	Pipeline, project-level policy, microservice, change	Controls module resources of a specific project. You can configure it in project settings. The configuration takes effect for all resources of the entire project.
Resource	Pipeline	Controls the operation permission of a specific pipeline. The permission can be configured by project member or role in the pipeline editing state.

2.2 Tenant-level Permissions

An administrator can use IAM to configure tenant-level rules, tenant-level policies, extensions, and pipeline templates for specified users.

Configuration Method

1. Use a tenant account or another authorized account to log in to CodeArts. Click the avatar in the upper right corner and choose **IAM** to access the IAM console.
2. In the navigation pane, choose **User Groups**. On the displayed page, create a user group or select an existing user group, and click **Authorize**.

Select Pipeline service to view the following policies:

Policy Name	Description
CloudPipeline Tenant Rules FullAccess	Controls whether a user has full permissions for tenant-level rules of CodeArts Pipeline.
CloudPipeline Tenant Rule Templates FullAccess	Controls whether a user has full permissions for tenant-level policies of CodeArts Pipeline.
CloudPipeline Tenant Extensions FullAccess	Controls whether a user has full permissions for extensions of CodeArts Pipeline.
CloudPipeline Tenant Pipeline Templates FullAccess	Controls whether a user has full permissions for templates of CodeArts Pipeline.

3. Select the required policy, click **Next**, and set the minimum authorization scope for the user group.
4. Add a specified user to a user group through user authorization or user group management.

NOTE

In addition to system-defined policies, tenants can also [create custom policies](#) to grant permissions.

Policy Management

Log in to CodeArts, click the avatar in the upper right corner, choose **All Account Settings > Policy Management**, and view **Rules** and **Policies**.

- In IAM, rule and policy settings correspond to permissions **cloudpipeline:rule:update** and **cloudpipeline:ruletemplate:update** respectively. An administrator can use the built-in system-defined policies

CloudPipeline Tenant Rules FullAccess and **CloudPipeline Tenant Rule Templates FullAccess** to authorize them in a unified manner or customize policies to authorize them separately.

- Common users under a tenant can view all data after choosing **Policy Management > Rules**. Authorized users can view and manage all tenant-level rules.
- Common users under a tenant can view all data after choosing **Policy Management > Policies**. Authorized users can view and manage all tenant-level policies.

Extensions

Log in to CodeArts and choose **Services > Extensions**.

- The extension permission corresponds to **cloudpipeline:extensions:update** in IAM. An administrator can use system-defined policies **CloudPipeline Tenant Extensions FullAccess** or custom policies to authorize users.
- Common users of a tenant can view all extensions on the page. Authorized users can view and manage all extensions of the tenant.


Pipeline Templates

Log in to CodeArts, choose **Services > Pipeline**, and click **Templates**.

- The pipeline template permission corresponds to **cloudpipeline:pipelinetemplate:update** in IAM. An administrator can use system-defined policies **CloudPipeline Tenant Pipeline Templates FullAccess** or custom policies to authorize users.
- Common users of a tenant can create templates and view all templates. However, they can manage only the templates created by themselves. Authorized users can view and manage all templates of the tenant.

2.3 Resource-level Permissions

On the **Permissions** tab of CodeArts Pipeline, you can configure permissions for a single pipeline by role or user.

<u>Role Permissions</u>	User Permissions				
Role	View	Execute	Edit	Delete	
Project creator	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
Pipeline creator	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
Project manager	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
Developer	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Test manager	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Tester	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Participant	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Viewer	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Operation manager	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Configuring Permissions by Role

- By default, role permissions are consistent in a pipeline and the related project. If project role permissions are changed, pipeline role permissions will be changed accordingly.
- The project creator, pipeline creator, and project manager can change pipeline role permissions.
- However, after pipeline role permissions are changed, project role permissions are not longer changed accordingly, because pipeline permissions prevail over project permissions.

Configuring Permissions by User

- By default, user and role permissions are consistent in a pipeline. If pipeline role permissions are changed, pipeline user permissions will be changed accordingly.
- The project creator, pipeline creator, and project manager can change pipeline user permissions.
- However, after pipeline user permissions are changed, pipeline role permissions are not longer changed accordingly, because user permissions prevail over role permissions.

3 Rules and Policies

3.1 Overview

CodeArts Pipeline provides unified pass condition management capabilities, uses rules and policies to associate extensions with pipelines, and implements stage-based pass verification.

- **Rules:** Comparison relationships and threshold conditions can be set for policies based on the output thresholds of extensions. Rules are finally applied to pipeline as pass conditions.
- **Policy:** Policies can be displayed and selected during pipeline orchestration. You can use these policies as pass conditions to control pipeline execution. Policies are classified into tenant-level policies and project-level policies.

Users can apply policies to pipelines as pass conditions to help efficiently manage projects and ensure high-quality delivery.

3.2 Configuring a rule

Rules are tenant-level resources and can be referenced and configured in all tenant- or project-level policies of the current tenant.

Rule Entry

1. On any service page, click the avatar icon in the upper right corner and choose **All Account Settings** from the drop-down list.
2. Click **Policy Management > Rules**. The rule management page is displayed.


Adding Rules

Click **Create Rule**. On the displayed **Create Rule** page, set parameters as required and click **Confirm** to create a tenant-level rule.

Parameter	Description
-----------	-------------


Name	Rule name, which is generated based on the current time by default.
Type	Rule type, which corresponds to the extension type.
Extension	Name of the extension bound to the rule. Only the extension version for which the metric threshold is configured can be selected.
Version	Version of the extension bound to the rule. Only the extension version for which the metric threshold is configured can be selected.
Threshold Configuration	Extension output threshold configuration automatically based on the selected extension version. Based on the configured relationship, the extension output can be parsed into the number or text. After the rule is configured, the configured relationship is referenced by the policy and finally configured in the pipeline as a pass condition.

Modifying Rules

On the rule management page, click  in the **Operation** column of the target rule. On the displayed page, edit the rule and click **Save**.

- The rule type cannot be modified.
- After a rule is edited, all policies that reference the rule are automatically modified.

Deleting Rules

On the rule management page, click  in the **Operation** column of the target rule. On the displayed page, confirm the information and click **Delete** to delete the rule.

NOTE

After a rule is deleted, all policies that reference the rule automatically cancel the reference.

3.3 Tenant-level Policies

Tenant-level policies are tenant-level resources. They can be configured as pass conditions of all pipelines under the current tenant.

Policy Entry

1. On any service page, click the avatar icon in the upper right corner and choose **All Account Settings** from the drop-down list.
2. Click **Policy Management > Policies**. The policy management page is displayed.

 **NOTE**

By default, a system policy exists. You can view and use the policy, but cannot edit or delete it.

Creating Policies

Click **Create Policy**. The **Create Policy** page is displayed. Enter a policy name, select a rule, and click **Confirm** to generate a tenant-level policy.


 **NOTE**

A maximum of 20 rules can be selected for a policy.

The rules selected in the policy are displayed in the right pane. You can perform the following operations on each rule:

- **Switch**: You can click the switch in the upper right corner of each rule to enable/disable the selected rule. After the rule is disabled, the corresponding pipeline pass condition is automatically disabled and no interception or verification is performed.
- **Edit**: Click **Detail** in the upper right corner of the rule to view the details. Click **Edit** in the upper right corner to edit the rule as needed.


Editing Policies

On the tenant-level policy management page, click  in the **Operation** column of the target policy. On the displayed page, edit the policy and click **Save**.

 **NOTE**

The page for editing a policy is similar to that for creating a policy.


Copying Policies

On the tenant-level policy management page, click  in the **Operation** column of the target policy and select **Copy**. The policy copy page is displayed.

 **NOTE**

The page for copying a policy is similar to that for creating a policy.

Deleting Policies

On the tenant-level policy management page, click  in the **Operation** column of the target policy and select **Delete**. On the displayed page, confirm the information and click **Delete** to delete the policy.

 **NOTE**

When you delete a policy, the system displays a message indicating the number of pipelines that reference the policy. Once the policy is deleted, the corresponding pipeline execution will fail.

Disabling Policies

On the tenant-level policy management page, click the switch on the right of the target policy to disable it.

NOTE

If a policy is referenced and disabled, the system displays a message showing the number of pipelines that reference the policy. Once the policy is disabled, the pass condition in the policy will be ignored during pipeline execution.

3.4 Project-level Policies

Policy Entry

Go to the pipeline list of any project, click **More** in the upper right corner, and select **Policies** from the drop-down list. The project-level policy management page is displayed.

Creating Policies

Click **Create Policy**. The **Create Policy** page is displayed. Enter a policy name, select a rule, and click **Confirm** to generate a tenant-level policy.


NOTE

A maximum of 20 rules can be selected for a policy.

The rules selected in the policy are displayed in the right pane. You can perform the following operations on each rule:

- **Switch:** You can click the switch in the upper right corner of each rule to enable/disable the selected rule. After the rule is disabled, the corresponding pipeline pass condition is automatically disabled and no interception or verification is performed.
- **Edit:** Click **Detail** in the upper right corner of the rule to view the details. Click **Edit** in the upper right corner to edit the rule as needed.


Editing Policies

On the project-level policy management page, click  in the **Operation** column of the target policy. On the displayed page, edit the policy and click **Save**.

NOTE

The page for editing a policy is similar to that for creating a policy.


Copying Policies

On the project-level policy management page, click  in the **Operation** column of the target policy and select **Copy**. The policy copy page is displayed.

NOTE

The page for copying a policy is similar to that for creating a policy.

Deleting Policies

On the project-level policy management page, click  in the **Operation** column of the target policy and select **Delete**. On the displayed page, confirm the information and click **Delete** to delete the policy.

NOTE

When you delete a policy, the system displays a message indicating the number of pipelines that reference the policy. Once the policy is deleted, the corresponding pipeline execution will fail.

Disabling Policies




On the project-level policy management page, click the switch on the right of the target policy to disable it.

NOTE

If a policy is referenced and disabled, the system displays a message showing the number of pipelines that reference the policy. Once the policy is disabled, the pass condition in the policy will be ignored during pipeline execution.

Tenant-level Policies

On the project-level policy management page, click **Tenant Level Policies** in the upper right corner. Then, you can view, copy, or inherit a tenant-level policy as required.

- View: Click  in the **Operation** column to view the tenant-level policy.
- Copy: Click  in the **Operation** column. On the displayed page, edit the policy and click **Save**.
- Inherit: Click  in the **Operation** column. On the displayed page, edit the policy and click **Save**.

NOTE

For a project-level policy generated through inheritance, the rule configured in the parent policy cannot be canceled or disabled in the child policy.

3.5 Policies and Pipelines

Tenant-level and project-level policies can be used as pass conditions to control pipeline execution.

Tenant-level policies can be applied to pipelines of all projects under the current tenant, and project-level policies can be applied to all pipelines under the current project.

1. Select a pipeline of a project and click **Edit**. In the pipeline stage, click **Pass Conditions**. On the displayed page, click **Add** to add a pass condition.
2. After the condition is added, select a required policy on the right and click **OK**.
3. After the pass condition is configured, run the pipeline. During pipeline execution, the configured policy can be invoked to determine whether to exit.

4 Extensions

4.1 Overview

What Is an Extension?

CodeArts Pipeline has a collection of built-in extensions for you to orchestrate pipelines covering build, check, deploy, and test, aiming to create a continuous and efficient development and delivery process.

How to Access the Extension List?

- Entry 1: Log in to CodeArts and choose **Services** > **Extensions** from the top navigation bar.
- Entry 2: Go to the pipeline list of a project, create or edit a pipeline, and click **More Extensions** in the upper right corner to go to the extension page.

How to View Details of an Extension?

The **Extensions** page displays all available extensions. Click the card of an extension to view details.

How to Use an Extension?

All extensions displayed on the **Extensions** page can be used when you edit pipelines under a tenant.

1. When adding or editing a task during pipeline orchestration, you can view the list of existing extensions on the page displayed on the right.
2. In this extension list, select an extension, click **Add** to use it in the pipeline.
3. Configure the extension to use. For details, see [Task Orchestration](#).

4.2 Official Extensions

Type	Name	Description
Build	Build	Calls CodeArts Build capabilities. CodeArts Build provides an easy-to-use, cloud-based build platform that supports multiple programming languages, helping you achieve continuous delivery, with shorter period and higher efficiency. With CodeArts Build, you can create, configure, and run build tasks with a few clicks. It also supports automation of code retrieval, build, and packaging, as well as real-time status monitoring. View details.
	Build-Template	This extension can be configured only in a pipeline template. When a pipeline is generated from the template, the extension automatically creates a Build task and configures the task in the generated pipeline.
Test	TestPlan	Calls CodeArts TestPlan capabilities. CodeArts TestPlan is a one-stop test platform that allows developers to conduct API and performance tests and manage these tests in the cloud. Developed on the basis of the DevOps Agile testing concept, CodeArts TestPlan helps developers improve management efficiency and deliver high-quality products. View details.
	TestPlan-Template	This extension can be configured only in a pipeline template. When a pipeline is generated from the template, the extension automatically creates a TestPlan task and configures the task in the generated pipeline.
Deploy	Deploy	Calls CodeArts Deploy capabilities. CodeArts Deploy allows you to visually deploy applications in VMs or containers by using Tomcat, Spring Boot, and other templates or flexibly orchestrated atomic actions. It standardizes your deployment environment and processes by integrating with CodeArts Pipeline. View details.
	Deploy-Template	This extension can be configured only in a pipeline template. When a pipeline is generated from the template, the extension automatically creates a Deploy task and configures the task in the generated pipeline.
	KubernetesRelease	Deploys container images to Huawei Cloud containerized applications. It supports grayscale release on Application Service Mesh (ASM) and container deployment on Cloud Container Engine (CCE).

Type	Name	Description
Check	Check	Calls CodeArts Check capabilities. CodeArts Check is a cloud-based management service that checks code quality. Developers can easily perform static code and security checks in multiple languages and obtain comprehensive quality reports. CodeArts Check also provides suggestions on fixing code defects and trend analysis, effectively controlling quality and reducing costs. View details .
	Check-Template	This extension can be configured only in a pipeline template. When a pipeline is generated from the template, the extension automatically creates a Check task and configures the task in the generated pipeline.
	BranchCheck	Specifies the target branch. If the current running branch lags behind the specified branch, the pipeline fails to run.
General	CreateTag	Creates and pushes tags for code repositories.
	Subpipeline	Configures and calls other pipeline tasks in a project.
	JenkinsTask	Calls Jenkins tasks to customize actions. For details, see Jenkins Tasks .
	DelayedExecution	Pauses pipeline for a period or until a specified time. You can manually resume or terminate it, and perform extra delay for a maximum of three times.
	ManualReview	Creates manual review tasks by assigning one person or one group.
	GitClone	Clones the code repositories configured in the pipeline source, which can be used together with shell commands and Maven build.
Microservice	CreateReleaseBranch	Creates a release branch based on the default branch of Microservice for change pipelines. This extension is automatically configured by change pipelines and manual configuration is not supported.
	MergeReleaseBranch	Merges a feature branch into a release branch during the running of a change pipeline. This extension is automatically configured by change pipelines and manual configuration is not supported.
	MergeDefaultBranch	Merges a release branch into the default branch of a microservice during the running of a change pipeline. This extension is automatically configured by change pipelines and manual configuration is not supported.

Type	Name	Description
Pass Condition	Pass-Condition-Standard-Policies	Select a standard extension policy for access control interception.

4.3 Custom Extensions

4.3.1 Getting Started


Medium- and large-sized enterprises normally have their own pipeline tools, but in-house assets such as CI/CD and open-source tools, after migrated to the cloud, may fail to be inherited or reused and thus lead to repetitive construction. CodeArts Pipeline provides a collection of standard pipeline extensions for enterprises to quickly connect legacy tools to the extension platform, or to develop and release their own extensions to share and co-deploy, accelerating cloudification. CodeArts Pipeline builds a visualized, low-code, and open extension market so that you can focus on the ecosystem, reusability, and customization of DevOps extensions. Custom extensions are available only to enterprises.

This section describes how to develop a log printing extension.

- [Step 1: Register a Custom Extension](#)
- [Step 2: Configure a Custom Extension](#)

Step 1: Register a Custom Extension

1. Log in to CodeArts.
2. Choose **Services** > **Extensions** from the top navigation bar.

3. Click  to register an extension.
 - Basic information
 - **Icon:** icon of the extension to display. If none is uploaded, a system icon is generated. The icon can be in PNG, JPEG, or JPG format, with a file size less than or equal to 512 KB. The recommended size is 128 x 128 pixels.
 - **Name:** name of the extension.
 - **Unique Identifier:** ID of the extension. Once set, this parameter cannot be changed.
 - **Type:** type of the extension, which can be **Build**, **Check**, **Test**, **Deploy**, or **Normal**. Once set, this parameter cannot be changed.
 - **Description:** purposes and functions of the extension. This parameter can be modified later.

- Version information

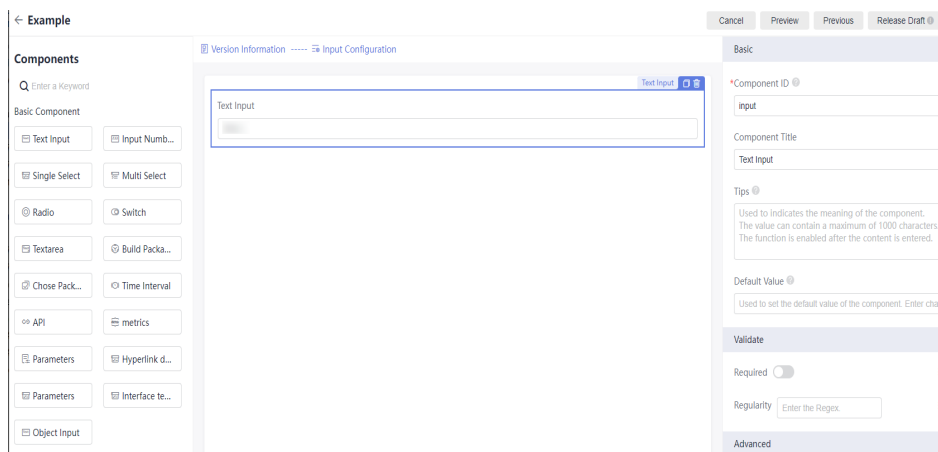
Enter the shell command in the **Shell Command** area to implement service logic. In the following figure, the output will be the controls displayed on the **Input Configuration** page and pipeline system parameters.

```
1 echo ===== begin =====
2 # Fill in the extension business logic here
3
4 # Use environment variables to get the parameters from interface
5 echo ${input}
6 # Use environment variables to get the pipeline running parameters
7 echo ${PIPELINE_ID}
8
9 echo ===== end =====
```

- Input configuration

Drag and drop widgets to generate forms and connect pipeline data at lower development costs.

The following describes how to configure a single-line text box and change its unique ID to **input**.



4. Release the extension.

After the design is complete, click **Release** or **Release Draft**. Drafts can be deleted.

Step 2: Configure a Custom Extension

1. Configure a custom extension on a pipeline.

After the custom extension is developed, create or edit a pipeline, add the extension, and enter **Low-code input** in the single-line input box.

2. Execute the pipeline to execute the extension.

After the execution is complete, the extension log contains **Low-code input** and pipeline system variables.

4.3.2 Registering a Custom Extension

1. Log in to the CodeArts homepage.
2. Choose **Services > Extensions** from the top navigation bar.

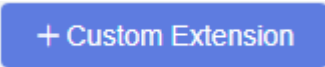
- Click  to register an extension.
- Set basic information. For details, see [Table 4-1](#).

Table 4-1 Parameter description

Parameter	Description
Icon	Icon of the extension to display. If none is uploaded, a system icon is generated. The icon can be in PNG, JPEG, or JPG format, with a file size less than or equal to 512 KB. The recommended size is 128 x 128 pixels.
Name	Name of the extension.
Unique Identifier	ID of the extension. Once set, this parameter cannot be changed.
Type	Type of the extension, which can be Build , Check , Test , Deploy , or Normal . Once set, this parameter cannot be changed.
Description	Purposes and functions of the extension. This parameter can be modified later.

- Click **Next** and configure version information. For details, see [Table 4-2](#).

 **NOTE**

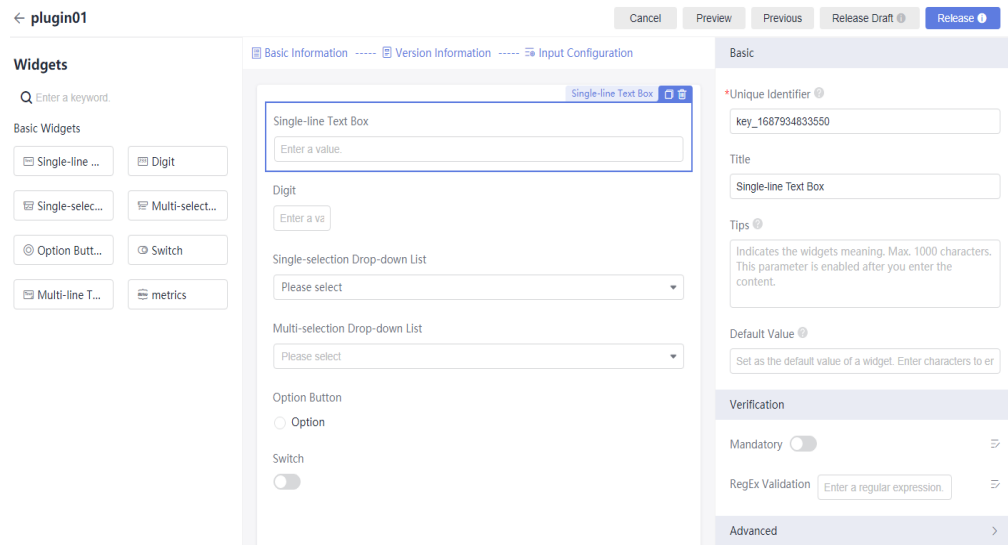
This page displays the version information of the extension, which cannot be modified later.

Table 4-2 Parameter description

Parameter	Description
Version	Version of the extension, in X.X.X format. Each digit ranges from 0 to 99.
Description	Describes the main functions or updates of the specified version.
Execution Environment	Only System Container can be set currently.
System Container	System container where the extension command is executed.
Shell Command	Commands to execute when the pipeline calls the extension. NOTE The commands indicate the actual service logic implementation process of the extension. For more input and output configurations, see Developing Extensions Using Low Code .

- Click **Next** and set the input.

Drag and drop widgets to generate forms and connect pipeline data at lower development costs. Multiple preset controls are available, including Single-line Text Box, Digit, Single-selection Drop-down List, Multi-selection Drop-down List, Option Button, Switch, and Multi-line Text Box.



Drag the widgets to the middle area. Click a widget and the configurations are displayed on the right. For details about the parameter description, see [Table 4-3](#).

Table 4-3 Parameter description

Category	Property	Description	Widget
Basic	Unique Identifier	ID of the widget, which is unique and used to obtain widget input during extension logic implementation.	All
	Title	Name of the widget for identification, which will be displayed on the pipeline task orchestration page.	All
	Tips	Description of the widget, which will be displayed as preview.	All
	Accuracy	Number of decimal places supported by the control value. The value ranges from 0 to 4.	Digit
	Default Value	Sets the default value of the component.	Single-line Text Box, Digit, Switch, Multi-line Text Box

Verif icati on	Man dator y	Whether the widget content is mandatory. Error messages can be set.	Single-line Text Box, Digit, Single-selection Drop-down List, Multi-selection Drop-down List, Option Button, Multi-line Text Box
	RegE x Valid ation	Sets the widget content verification. Error messages can be set.	Single-line Text Box, Digit, Single-selection Drop-down List, Multi-selection Drop-down List, Multi-line Text Box
Opti ons	Custo m	Options supported by the widget: <ul style="list-style-type: none"> Label: content displayed on the page during extension configuration. Value: value delivered when the extension is running. In addition to manual configuration, set the options by: <ul style="list-style-type: none"> API: Configure the webapi to obtain options. For details about the API parameters, see Table 4-4. Context: information from the pipeline source or build tasks 	Single-selection Drop-down List, Multi-selection Drop-down List, Option Button
Adv ance d	Displ ay	Whether the widget is visible. Conditions can be set.	All
	Disab le	Whether to disable the widget. Conditions can be set.	All

Table 4-4 Parameter description

Parameter	Description
Linked Attribute	Establish the association between other components and APIs. Parameters can be transferred. When the value of a component is updated, the new value is used to call APIs again.
URL	Only HTTPS protocol is supported.

Returned Data Path	<p>The widget used must be list data. In the following response body example, the returned data path is result.parameters.</p> <pre>{ "result": { "total": 2, "parameters": [{ "id": 3353753, "name": "parameters01" }, { "id": 3353697, "name": "parameters02" }] }, "status": "success" }</pre>
Option Value	Set this parameter to the value of the corresponding field in the returned data path. This parameter is delivered when the extension is running.
Option Name	Set this parameter to the value of the corresponding field in the returned data path. This parameter is displayed on the extension configuration page.
Remote Search	After this function is enabled, you can add a remote search field. The entered value is used as the value of the remote search field to call the APIs again.

You can also use the metrics widget to configure extension output:

- Drag the metrics widget to the middle area. This widget is not displayed by default and is used only for parsing output metrics after pipeline execution is complete.
- Configure the threshold property of the metrics control. The threshold information can be referenced in rules or policies through this extension and then used in a pipeline.

Threshold ×

Group: 🗑️

Key	Name	Value	
<input type="text"/>	<input type="text"/>	<input type="text"/>	🗑️



[+ Add](#)

[+ Add](#)

[OK](#) [Cancel](#)

Parameters:

Parameter	Description
Group	Threshold group, which is user-defined. Configure the extension and use it in a rule/policy for a pipeline.
Key	Parses the <code>\${STEP_ID}_metrics.json</code> output file of the extension code logic. After the pipeline is executed, the file is matched based on the value of this parameter.
Name	Name to display as a threshold check item here, and then display in a policy and a pipeline exit condition.
Value	Default condition for a threshold check.

7. Click **Release** or **Release Draft** to complete the registration.
Both draft release and official release are supported.
 - Official release:
Extensions officially released have an independent version number. All members of the current tenant can use this extension version in a pipeline.
 - Draft release
 - Extension drafts can be configured in a pipeline to debug. After the debug is complete, the drafts can be officially released for other members of the current tenant to use.
 - All draft versions are marked with .
 - Only one draft is allowed. If there is already a draft, no more versions can be created until you officially release or delete the draft.
 - Drafts can be directly deleted.
8. (Optional) Update the extension.
 - a. After the extension is registered, click its card to view details.
 - b. Click **New Version**. Or, copy an existing extension version and click  to view the historical configuration. Then modify the extension based on a version copy.

4.3.3 Developing Extensions Using Low Code

When registering an extension or creating an extension version, you can enter shell commands to implement service logic, which usually involves interaction with all kinds of data during pipeline execution. This section describes the code implementation of extension development through input and output.

Data Input of Custom Extensions

The information obtained during extension development mainly consists of three parts: low-code GUI input, pipeline running parameters and other information.

- Low-code GUI input: Obtain the low-code GUI output through environment variables, for example, `echo ${Control ID}`.

- Pipeline running parameters: Some pipeline running parameters are delivered to environment variables. Currently supported environment variables are listed as follows:

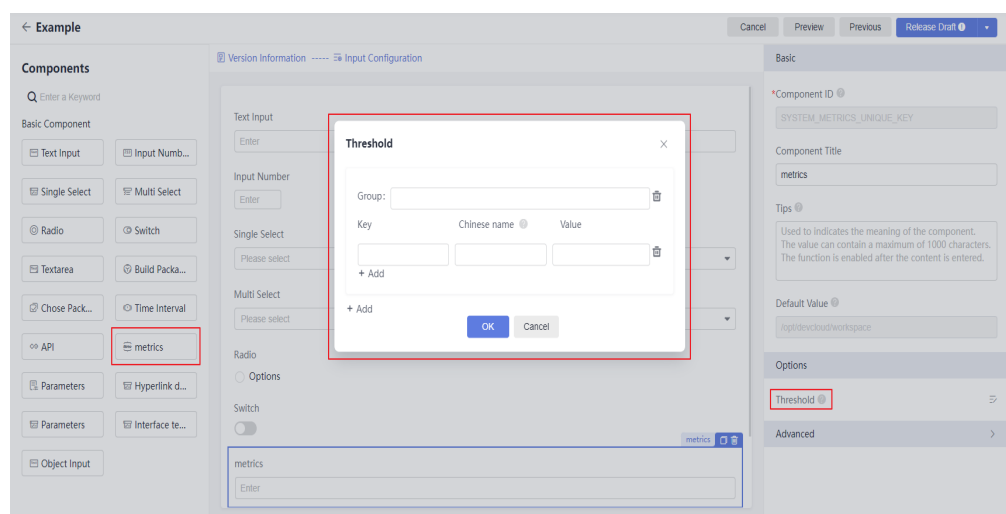
Variable	Description
STEP_NAME	Step name of the pipeline.
STEP_ID	Step ID of the pipeline.
PLUGIN_VERSION	Version of the extension.
PIPELINE_ID	Pipeline ID.
PIPELINE_RUN_ID	Pipeline execution ID.
PLUGIN_NAME	Extension name.
PROJECT_ID	Project ID.
JOB_ID	Task ID of the pipeline.

- Other information is obtained from the interaction with external data through git, wget, and curl.

Data Output of Custom Extensions

Once executed, the custom extension can read file information in a specified path and obtain the metric data output.

1. On the low-code GUI, drag the metrics widget to configure the output threshold of the extension.



2. During development, the `${STEP_ID}_result.json` and `${STEP_ID}_metrics.json` files are exported to the specified path so that metric values can be parsed after the extension is executed.

Table 4-5 Output files

File	Description
/var/devcloud/workspace/temp-mnt/result/\${STEP_ID}_result.json	The output is a text file in {"par1":123, "par2":456} format. After the pipeline is executed, the result is displayed as the corresponding task result. NOTE Only check extensions can display the result.
/var/devcloud/workspace/temp-mnt/result/\${STEP_ID}_metrics.json	The output is a text file in {"par1":123, "par2":456} format. The metrics control is also used. After the extension is executed, the threshold of the metrics control and the actual file content <code>\${STEP_ID}_metrics.json</code> are parsed and compared for pipeline exit. Precautions: <ul style="list-style-type: none"> • During parsing, empty key values will be ignored. • If the corresponding key value cannot be found in the <code>\${STEP_ID}_result.json</code> file, the specified threshold value is used.

Example: **par1** and **par2** metrics for exit interception and **par3** and **par4** for task result display. The sample code is as follows:

```
# Optionally, construct the extension output.
```

```
echo '{"par1":100,"par2":200}' > /var/devcloud/workspace/temp-mnt/result/${STEP_ID}_metrics.json
echo '{"par3":300,"par4":400}' > /var/devcloud/workspace/temp-mnt/result/${STEP_ID}_result.json
```

3. After the extension is run, click the corresponding step card to view the output.

Red Line	
Check Items	Result
par3	300
par4	400

If a policy is configured for the current extension and applied to the pipeline pass condition, click the pass condition to view the interception.

CustomExtension / CustomExtension				
Group				
Check Items	Status	Current ...	Compare	Threshold
par1	Success	100	=	100
par2	Success	200	=	200

4.4 Extensions and Policies

Some extensions can also be added to policies by configuring rules. You can then apply the policies to pass conditions during pipeline orchestration.

Currently, the following official extensions support rule configuration:

- Check: invokes CodeArts Check capabilities and returns the total number of issues (critical/major/minor/suggestion).

Check Items	Relationship	Default Threshold	Enabled
Critical	=	numeraica 0	<input checked="" type="checkbox"/>
Major	=	numeraica 0	<input checked="" type="checkbox"/>
Minor	=	numeraica 0	<input checked="" type="checkbox"/>
Suggestion	=	numeraica 0	<input checked="" type="checkbox"/>
Total	=	numeraica 0	<input checked="" type="checkbox"/>

- TestPlan: invokes CodeArts TestPlan capabilities and returns the API test pass rate.

Check Items	Relationship	Default Threshold	Enabled
API-test-pass-rate	=	numeraica 0.5	<input checked="" type="checkbox"/>

NOTE

Custom extensions configured with the **metrics** widget can also be used to configure rules.


4.5 Jenkins Tasks

Background

CodeArts Pipeline supports calling the Jenkins tasks provided by third-party services to extend your experience.

Method

1. On the CodeArts Pipeline homepage, search for the target pipeline, click **...** in the **Operation** column and select **Edit**. On the **Task Orchestration** page displayed, add **JenkinsTask**.



JenkinsTask
Jenkins tasks can be called to extend and implement custom actions.

[Tips](#)

***Name**

***Jenkins Endpoint** [Create One](#) | [Refresh](#)

***JobName**

params

Asynchronous

Description

2. Set task parameters as required.

Parameter	Description
Jenkins Endpoint	Jenkins endpoint to connect, which can be configured in service endpoints. If none is available, create one.
JobName	Name of the Jenkins task to call, for example, Api_Guard .
params	Running parameters of a Jenkins task, which are transferred through the pipeline during calling, for example, <code>{'serviceVersion':\${releaseVersion}}</code> .
Asynchronous	Whether the configuration is asynchronous.
Description	Task description.

4.6 KubernetesRelease Extension

4.6.1 Introduction

During pipeline configuration, you can use CodeArts Deploy to deploy container images in Huawei Cloud containerized applications in a pipeline. It supports grayscale release on Application Service Mesh (ASM) and container deployment on Cloud Container Engine (CCE), which correspond to blue-green upgrade and CCE rolling upgrade, respectively.

- [Blue-Green Upgrade](#)

- [CCE Rolling Upgrade](#)

4.6.2 Blue-Green Upgrade

Kubernetes provides the blue-green upgrade policy. Users who have purchased ASM can upgrade Deployment in a cluster in blue-green mode.




- In Kubernetes, a service can be associated with Deployment instances of multiple versions through LabelSelector.
- VirtualService can be used to further control the routing policy between traffic and Deployment instances of a specific version.
- The blue-green release starts. Kubernetes creates a Deployment instance of a new version (blue cluster) based on the current Deployment instance (green cluster). By default, the blue cluster inherits the configuration of the green cluster. You can update the configuration on the CCE console before traffic diversion.
- After the blue cluster is created, update the DestinationRule resource object of the ASM to generate routing table information.
- When the traffic diversion starts, user traffic is automatically switched from the green cluster to the blue cluster.
- After the verification, click **Released** to remove the original green cluster and end the upgrade task.

Prerequisites



- The service has been brought online on CCE, and the service and Deployment have been created.
- VirtualService and DestinationRule corresponding to the service have been created in ASM.

Procedure

1. After creating a pipeline and configuring image build, add a **KubernetesRelease** task to the pipeline and set related parameters. For details about the parameters, see [Table 4-6](#).

KubernetesRelease   

KubernetesRele...


KubernetesRelease 
① Tips

Deploy the container image to Huawei Cloud container applications. Support graysca...
[Expand](#)


***Name**

***Region**

***CCE Cluster**

***Namespace**

Use IAM to improve permissions

***Deployment Strategy** 

Rolling Update Blue/Green Deployment


***Service Name**


***VirtualService**

***DestinationRule**

***Image URL**

Grayscale Version

***Grayscale Workload Upgrading Timeout (minutes)** 

***Route Traffic Waiting Time (minutes)** 

Add extension

Table 4-6 Parameter description

Parameter	Description
Name	Enter the name of the upgrade. The default name is KubernetesRelease .
Region	Select a region for deployment.
CCE Cluster	Select the Kubernetes cluster applied on CCE.
Namespace	Select the namespace of the Kubernetes cluster on CCE.
Use IAM to improve permissions	If you do not have the permission to execute an API, this parameter enables you to obtain the temporary AK/SK of the parent user to execute the CCE API through IAM.

Parameter	Description
Deployment Strategy	The deployment strategy includes Rolling Update and Blue/Green Deployment (select Blue/Green Deployment in this section). <ul style="list-style-type: none">• Rolling Update: rolling release for a Deployment• Blue/Green Deployment: ASM-based blue-green release
Service Name	Select the service to be upgraded.
VirtualService	Select the target VirtualService. Log in to the ASM console, choose Mesh Configuration > Istio Resource Management and filter the target namespace and Istio resources.
DestinationRule	Select the target DestinationRule. Log in to the ASM console, choose Mesh Configuration > Istio Resource Management and filter the target namespace and Istio resources.
Image URL	URL of the image to be upgraded. You can enter a prepared image URL as the target image, for example, swr.cn-north-1.myhuaweicloud.com/demo/springboot-helloworld:v1.1 . You can also use <code>\${}</code> to reference pipeline parameters, for example, swr.cn-north-1.myhuaweicloud.com/demo/springboot-helloworld:\${version} .
Grayscale Version	If you enable it, use the keyword "version" in the label to distinguish the official version from the grayscale version. The version number must be the same as the subsets object name in DestinationRule and is used as an identifier for grayscale traffic distribution.
Grayscale Workload Upgrading Timeout (minutes)	If the grayscale Deployment cannot be created within the timeout interval, the upgrade will fail.
Route Traffic Waiting Time (minutes)	After the waiting time, all user traffic is switched to the grayscale Deployment.

2. After the configuration, run the pipeline. Click the task card and select **Task Result** in the displayed dialog box. You can view the detailed upgrade process on the **Release Sheet** page.

The page displays information such as the sheet ID, executor, start time, end time, release status, basic information, release details, and release logs.

The screenshot shows the 'KubernetesRelease' interface. The 'Task Result' tab is active, displaying the 'Release Sheet' page. The page is titled 'Release Sheet' and is in an 'Executing' state. It shows the following information:

- Sheet ID: 939... | Executor: intl | Start Time: 2023-03-08 18:06:47 GMT+08:00 | End Time: --
- Basic Information:**
 - Previous Workload: blue-green-test | Current Workload: blue-green-test-v5767580
 - Workload Type: Deployment | Version number: v5767580
 - Namespace: default | Grayscale Workload Upgrading Timeout (minutes): 30
 - Image: [redacted] | Route Traffic Waiting Time (minutes): 1
- Release Detail:**
 - Release Log
 - Two cards showing deployment versions: 'v1' (Number of copies: 2) and 'v5767580' (Number of copies: 2, Online).
- Pod Information Table:**

Name	Status	Pod IP	Create Time
blue-green-test-7d6bbc...	Running	[redacted]	2023-02-15 16:37:58 GMT+08:00
- Basic Information (Pod):**
 - Instance Name: blue-green-test-7d6bbc557... | Pod IP: [redacted]
 - Status: Running | Host IP: [redacted]
 - Create Time: 2023-02-15 16:37:58 GMT+08:00
- Key Event:**
 - Message: The event is only saved for one hour, Kubernetes will automatically clear the data after the time.
 - Table with columns: K8s Component Na..., Event Type, K8s Event, First Occurrence T..., Recent Occurrenc...
 - Message: No records found.

- To end the upgrade, click **Released** to bring the old Deployment offline.
- To roll back a release, click **Rollback** to switch the traffic to the old Deployment and bring the new one offline. The **Rollback Sheet** page is displayed. The content of the rollback sheet is similar to that of the release sheet.
- Click **Release Status** to refresh the release status.

Basic Information

This area displays the names, types, namespaces, versions, images, timeout interval, and traffic switching waiting time of the old and new Deployments.

Release Detail

This area displays the number of Deployment copies, basic Deployment information, and key events of the old and new Deployments.

The **Online** icon indicates the Deployment of the current user traffic. You can click the card to view the pod information corresponding to the Deployment.

- Basic Information
 - **Instance Name:** pod name.

- **Status:** running status of the pod.
- **Pod IP:** IP address of the pod.
- **Host IP:** IP address of the node where the pod is located.
- **Create Time:** time when the instance is created.
- Key Event
This area displays the key events of the pod, including the Kubernetes component name, event type, Kubernetes event, first occurrence time, and recent occurrence time. These can be used to locate a pod fault.

Release Log

Switch to the **Release Log** tab page to view the detailed logs.

4.6.3 CCE Rolling Upgrade

KubernetesRelease supports CCE rolling upgrade as well as rolling release of Deployment in CCE/Kubernetes clusters. Supported rolling release types include:

- Upgrade only the Deployment images in a cluster.
- Use the native YAML files in your code repository for a CCE cluster. YAML templates can be used as parameters to facilitate reuse.

Prerequisites

The Deployment to upgrade has been created in a CCE cluster.

Procedure

1. After creating a pipeline and configuring image build, add a **KubernetesRelease** task to the pipeline and set related parameters. For details about the parameters, see [Table 4-7](#).

KubernetesRelease ✎ 🗑️ ✕

KubernetesRelease 🔔 📄 Tips

Deploy the container image to Huawei Cloud container applications. Support graysca...
[Expand](#)

*Name
KubernetesRelease

*Region
Sao Paulo1

*CCE Cluster
release-test

*Namespace
default

Use IAM to improve permissions

*Deployment Strategy ?
 Rolling Update Blue/Green Deployment

*Deploy Mode ?
 Image Upgrade YAML Deployment

*Workload ?
zfc-roll-1

*Container
container-1

*Image
[Image field with placeholder text]

Table 4-7 Parameter description

Parameter	Description
Name	Enter the name of the upgrade. The default name is KubernetesRelease .
Region	Select a region for deployment.
CCE Cluster	Select the Kubernetes cluster applied on CCE.
Namespace	Select the namespace of the Kubernetes cluster on CCE.
Use IAM to improve permissions	If you do not have the permission to execute an API, this parameter enables you to obtain the temporary AK/SK of the parent user to execute the CCE API through IAM.
Deployment Strategy	The deployment strategy includes Rolling Update and Blue/Green Deployment (select Rolling Update in this section). <ul style="list-style-type: none">● Rolling Update: rolling release for deployment● Blue/Green Deployment: ASM-based blue-green release

Parameter		Description
Deploy Mode		Options: Image Upgrade and YAML Deployment <ul style="list-style-type: none"> • Image Upgrade: upgrades the deployment images in a cluster. • YAML Deployment: uses the native YAML files in your code repository for a CCE cluster. YAML templates can be used as parameters to facilitate reuse.
Deploy Mode set to Image Upgrade .	Workload	Select a deployment to upgrade.
	Container	Select a container to upgrade.
	Image	Path of the image to upgrade, for example, swr.cn-north-1.myhuaweicloud.com/demo/springboot-helloworld:v1.1 . You can also use \${} to reference pipeline parameters, for example, swr.cn-north-1.myhuaweicloud.com/demo/springboot-helloworld:\${version} .
Deploy Mode set to YAML Deployment .	Repository	Code repository where the YAML file is located.
	Branch	Branch where the YAML file is located.
	YAML path	YAML file path.
	Variables	Click Add to add parameters. In a YAML file, {{}} can be used to reference parameters. The KubernetesRelease extension dynamically renders the parameters configured here to the YAML file.

2. After the extension is configured, run the pipeline and go to the release sheet page to view the upgrade process.

The page displays information such as the sheet ID, executor, start time, end time, release status, upgrade/rollback information, and release details.

KubernetesRelease

Task Log **Task Result**

KubernetesRelease

Release Sheet Executed [Release Status](#) [Rollback](#)

Sheet ID: 438e... | Executor: intl | Start Time: 2023-03-08 18:09:41 GMT+08:00 | End Time: 2023-03-08 18:09:46 GMT+08:00

Basic Information

Workload Name zfc-roll-1 | Number of copies 1/1
Workload Type Deployment | Version number v1.1
Namespace default | Create Time 2023-02-15 15:38:17 GMT+08:00
Image

Release Detail

Name	Status	Pod IP	Create Time
zfc-roll-1-6c88c97c48-...	Running	1...	2023-03-08 18:09:39 GMT+08:00

Basic Information

Instance Name zfc-roll-1-6c88c97c48-5pq... | Pod IP
Status Running | Host IP 1...
Create Time 2023-03-08 18:09:39 GMT+08:00

Key Event

The event is only saved for one hour, Kubernetes will automatically clear the data after the time.

K8s Component Na...	Event Type	K8s Event	First Occurrence T...	Recent Occurrenc...
---------------------	------------	-----------	-----------------------	---------------------

No records found.

- To roll back a release, click **Rollback** to switch the traffic to the old Deployment and bring the new one offline. The **Rollback Sheet** page is displayed. The content of the rollback sheet is similar to that of the release sheet.
- Click **Release Status** to refresh the release status.

Basic Information

This area displays the name, type, number of copies, namespace, version, and image of the Deployment to upgrade.

Release Detail

This area displays the pod and event information of the Deployment.

- Pod information
 - **Instance Name:** pod name.
 - **Status:** running status of the pod.
 - **Pod IP:** IP address of the pod.
 - **Host IP:** IP address of the node where the pod is located.

- **Create Time:** time when the instance is created.
- **Key Event**





This area displays the key events of the pod for troubleshooting, including the Kubernetes component name, event type, Kubernetes events, and the time when the key events first and last occurred.

5 Microservice Management

5.1 Microservice Entry

Procedure

1. Log in to the CodeArts homepage.
2. Go to the target project and choose **CICD > Pipeline**.
3. Click **Microservice**. The microservice list page is displayed, showing information about all microservices in the project.

Parameter	Description
Microservice	Microservice name.
Created By	Name of the user who creates a microservice.
Created At	Time when a microservice is created. You can move the cursor to the Create At column and click  to sort microservices by creation time.
Status	Current status of a microservice. After a microservice is created, it automatically changes to Activated .
Operation	Click  to follow the microservice. After the microservice is followed, the icon changes to  . You can click the icon again to unfollow the microservice. Also, you can click  to delete the microservice. NOTE After a microservice is deleted, all change records and pipelines of the microservice will be deleted. Exercise caution when performing this operation.

- Click **Create Microservice** to create a microservice. For details, see [Creating a Microservice](#).

- You can enter a microservice name in the search box to search for the microservice.
- Click a microservice name to view microservice details. For details, see [Viewing a Microservice](#).

5.2 Creating a Microservice

Procedure

1. Access the microservice.
2. On the microservice list page, click **Create Microservice**.
3. On the page that is displayed, set microservice parameters. For details, see [Table 5-1](#).

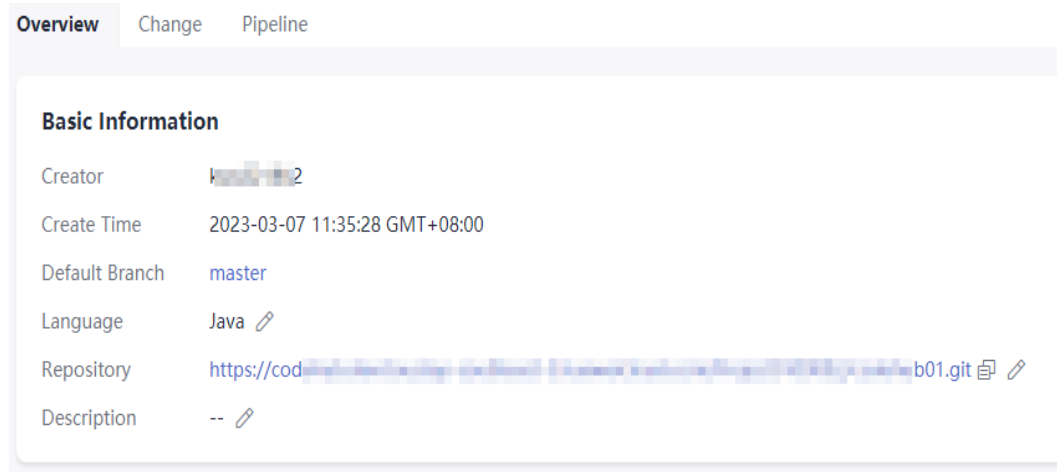
Table 5-1 Parameter description

Parameter	Description
Project	Project to which the microservice belongs, which cannot be changed.
Microservice name	Microservice name.
Pipeline Source	Source code repository. Only Repo is supported.
Repository	Created code repository.
Default Branch	Branch used when a microservice pipeline is manually or periodically executed. NOTE After the change pipeline is executed, all changed feature branches will be merged into the default branch.
Language	Development languages for the microservice.
Description	Microservice description.

4. After setting all parameters, click **OK**. The microservice is created.

5.3 Viewing a Microservice

On the microservice page, click a microservice name to view details.



- **Overview**

Information such as the creator, creation time, and code source of the microservice is displayed. You can edit the microservice language, associated code repository, and description as required.

When you change the code repository, if there are changes that are not closed or there are running pipelines under the microservice, the **Data Processing** dialog box is displayed. In that case, close all changes and stop all running pipelines.

- **Changes**

Manage microservice changes. For details, see [Change Management](#).

- **Pipelines**

Manage pipelines in a microservice. Differences between microservice pipelines and common pipelines are as follows:

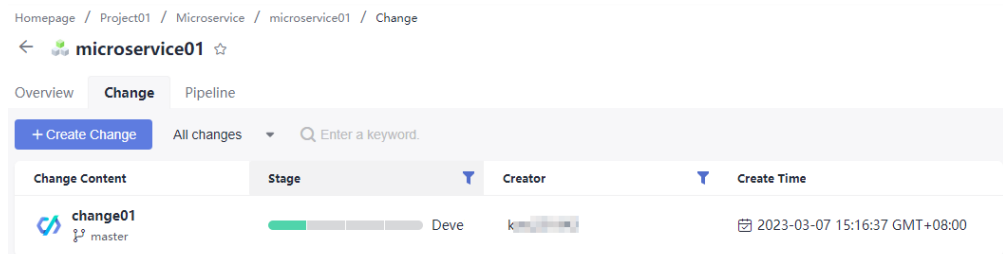
- In a microservice, the code repository cannot be changed when a pipeline is created. By default, the code repository is the same as that bound to the microservice.
- If you modify the code repository of a microservice, the code repositories configured for all pipelines of the microservice are automatically modified.
- In a microservice, you can create a change pipeline to associate microservice changes and release changed resources. For details about the change pipeline, see [Changes and Pipelines](#).

6 Change Management

6.1 Change Entry

Procedure

1. Log in to the CodeArts homepage.
2. Go to the target project and choose **CICD > Pipeline**.
3. Click **Microservice**. The microservice list page is displayed.
4. Click a microservice name. The **Overview** page is displayed.
5. Switch to the **Changes** tab page to view all change information.



Parameter	Description
Change Content	Change subject and associated branch information.
Stage	Status of the change stage. You can move the cursor to the Stage column and filter changes based on the stage status.
Created By	Name of the user who creates the change.
Created At	Creation time of the change.
My changes/All changes	Select My changes or All changes to filter changes. My changes displays changes created by the current user. All changes displays all changes of the microservice.

- Click **Create Change** to create a change. For details, see [Creating Changes](#).
- Click the drop-down list next to **Create Change** and select **All changes** or **My changes** to filter changes. **All changes** displays all changes of the microservice. **My changes** displays changes created by the current user.
- You can enter a keyword in the search box to search for the change content.
- Click the change name to view the change details. For details, see [Viewing Changes](#).

6.2 Creating Changes

Procedure

1. Go to the change list page.
2. Click **Create Change**.
3. On the displayed page, enter the basic information. For details, see [Table 6-1](#).

Table 6-1 Parameter description

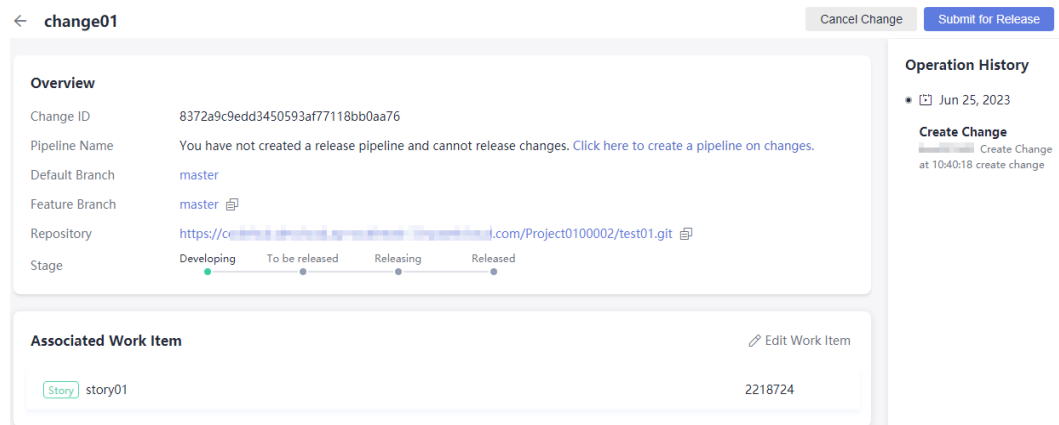
Parameter	Description
Change Subject	Enter a change subject.
Repository	Name of the repository associated with the microservice, which cannot be changed.
Branch	Pull a new branch from the default branch or associate with an existing branch.
Associated Work Item	Select started or ongoing work items in associated projects in CodeArts Req.

4. After setting all parameters, click **OK**.

6.3 Viewing Changes

Go to the change list page and click a change name. The change details page is displayed.

The page displays the change overview, associated work items, and operation history. You can also submit the change for release, exit from the release, and cancel the change.



The following describes how to submit a change for release, exit from the release, and cancel the change.

- **Submit for Release**

For a change in the **Developing** stage, click **Submit for Release** on the change details page. The **Submit for Release** dialog box is displayed.

- If no change pipeline exists under the microservice, create a change pipeline as prompted. For details, see [Changes and Pipelines](#).
- If a change pipeline exists under the microservice, click **OK** to submit the change to the change pipeline.

After the change is submitted for release, the change status changes from **Developing** to **To be released**.

- **Exit Release**

For a change in the **To be released** or **Releasing** stage, click **Exit Release** on the change details page.

In the displayed dialog box, click **OK**. The change is removed from the release list of the change pipeline, and the change status changes back to **Developing**.

 **NOTE**

For a change in the **Releasing** stage, if the change pipeline is running, you can exit the release only after the change pipeline running is complete or stopped.

- **Cancel Change**

For a change in the **Developing** stage, click **Cancel Change** on the change details page.

In the displayed dialog box, click **OK**. The change status changes to **Canceled**.

6.4 Changes and Pipelines

In microservices, you can create pipeline resources, set them to change pipelines, and associate them with microservice change resources. Compared with a common pipeline, a change pipeline has the following features:

- A microservice can have only one change-triggered pipeline.

- When the change pipeline is running, an integration branch is automatically created. After successful execution, the branch content is merged to the master branch.
- After successful execution, the change status of the published pipeline is automatically updated.
- Only one pipeline instance can run at one time.
- The change pipeline cannot be triggered by an event or a schedule task.

Creating a Change Pipeline

1. [Access the microservice.](#)
2. In the microservice list, click a microservice. The **Overview** page is displayed.
3. Switch to the **Pipelines** tab.
4. Click **Create Pipeline**. On the page that is displayed, set basic information. For details, see [Table 6-2](#).

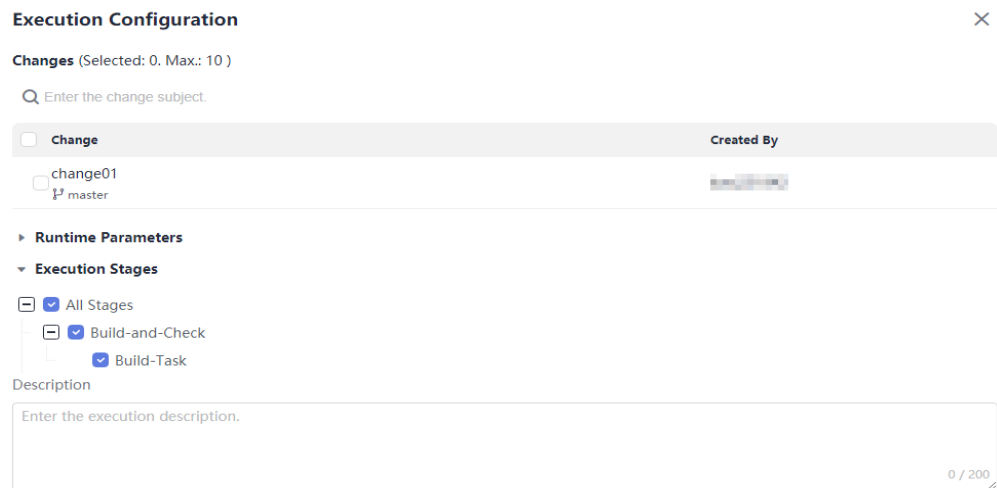
Table 6-2 Parameter description

Parameter	Description
Project	Project to which the microservice belongs, which cannot be changed.
Name	Pipeline name, which is generated based on the current time by default.
Pipeline Source	Only Repo is supported.
Repository	Name of the repository associated with the microservice, which cannot be changed.
Default Branch	Default branch associated with the microservice, which cannot be changed.
CodeArts Repo HTTPS Authorization	Configure authorization endpoints to elevate repository operation permissions. This function is used for microservice change pipelines and some repository operation extensions.
Alias	If an alias is specified, predefined parameters are generated for the repository and displayed on the Parameter Configuration tab page.
Change-based Trigger	A pipeline for which the Change-based Trigger option is enabled is a change pipeline. In the microservice pipeline list, the change pipeline is marked as Based on Changes . NOTE A microservice can have only one change-triggered pipeline.
Description	Description of the pipeline.

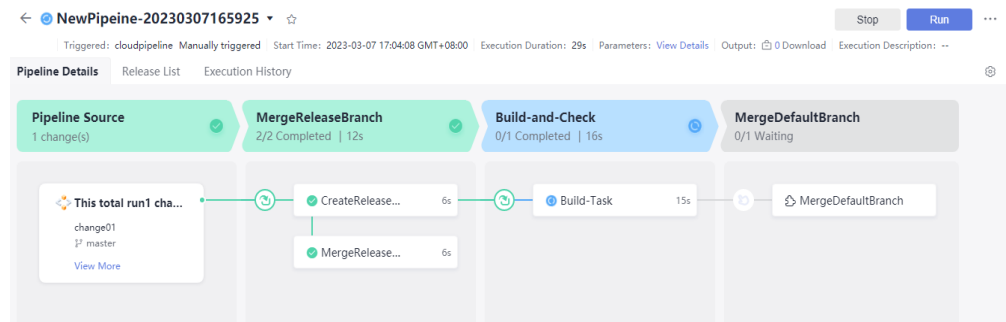
5. After setting all the parameters, click **Next**. On the page that is displayed, select a template as required to quickly create a task. You can also select **Blank Template**.
6. After selecting a template, click **OK**, and then click **Save**.

Executing a Change Pipeline

1. [Access the microservice.](#)
2. In the microservice list, click a microservice. The **Overview** page is displayed.
3. Switch to the **Pipelines** tab page.
4. Locate the change pipeline and click the pipeline name. The pipeline details page is displayed.
5. Click **Run** in the upper right corner. In the displayed **Execution Configuration** dialog box, perform the following operations:



- **Changes:** The change list displays the changes in **To be released** or **Releasing** stage. Select one or more changes when running the change pipeline.
 - **Runtime Parameters:** If runtime parameters have been set for the pipeline, set the parameters as required and then save them. For details, see [Using Parameters](#).
 - **Execution Stages:** Select one or more tasks to execute in the pipeline. By default, all tasks are executed.
 - **Description:** Enter the debugging information about the execution.
6. After the configuration is complete, click **Run**. The change pipeline execution details page is displayed.




When the change pipeline is running, the **MergeReleaseBranch** and **MergeDefaultBranch** stages will be added by default.

- **MergeReleaseBranch:** The change pipeline automatically pulls a new branch from the default branch, integrates all feature branches of the current change into the new branch, and runs the pipeline based on the integration branch.
- **MergeDefaultBranch:** The integration branch is merged back to the default branch.

Viewing the Result

After the execution is complete, you can view the execution result.

When the change pipeline is successfully executed, the status of all changes selected for pipeline running is changed to **Released**.

- Click a pipeline name to go to its details page.
 - Click **View More** on the pipeline source card. In the dialog box that is displayed, view the changes selected for pipeline running.
 - Click a change name in the list to go to the change details page.
- Click the **Releases** tab.
 - The release list page displays all changes in the **To be released** and **Releasing** stages.
 - You can enter a keyword in the search box to search for a change.
 - Click  in the **Operation** column. In the dialog box that is displayed, click **OK**. The change is removed from the release list of the change pipeline, and the change status changes back to **Developing**.

NOTE

For a change in the **Releasing** stage, if the change pipeline is running, you can exit the release only after the change pipeline running is complete or stopped.

7 Service Endpoints

A service endpoint is an extension of CodeArts. It enables CodeArts to connect to third-party services.

During the pipeline configuration, you can connect to a Kubernetes cluster to perform deployment tasks, a Docker repository to manage Docker images, or create an IAM user endpoint to allow the IAM user to perform tasks using the AK/SK of your account.

Prerequisites

- All roles can create service endpoints.
- Project creators and project managers can view all service endpoints in projects and set permissions.
- Ensure that the third-party services configured in the service extension point can be accessed through the public network without restrictions.

Creating a Docker Repository Service Endpoint

You can use a Docker repository service endpoint to connect to a Docker image repository. After the connection is successful, you can perform operations on Docker images.

1. In a project, choose **Settings > General > Service Endpoints**.
2. Click **Create Endpoint** and choose **Docker repository** from the drop-down list.
3. In the **Create Service Endpoint** dialog box, set the parameters.

Table 7-1 Parameter description

Parameter	Description
Service Endpoint Name	Name of the service endpoint.
Repository Address	Address of the Docker image repository to connect (HTTP or HTTPS address).

Parameter	Description
Username	Username for connecting to the Docker image repository.
Password	Password for connecting to the Docker image repository.

4. Click **OK**.

Creating a Kubernetes Service Endpoint

You can use a Kubernetes service endpoint to connect to a Kubernetes cluster. After the connection is successful, you can deliver deployment tasks to the Kubernetes cluster.

1. In a project, choose **Settings > General > Service Endpoints**.
2. Click **Create Endpoint** and choose **Kubernetes** from the drop-down list.
3. In the **Create Service Endpoint** dialog box, set the parameters.

Table 7-2 Parameter description

Parameter	Description
Service Endpoint Name	Name of the service endpoint.
Kubernetes URL	API server address of the Kubernetes cluster to connect (HTTP or HTTPS address).
Kubeconfig	The Kubeconfig file contains information about clusters, users, namespaces, and authentication mechanisms. Kubectl uses kubeconfig to select a cluster and communicate with the Kubernetes APIs. This file supports multiple clusters, users, and authentication mechanisms. For details, see Organizing Cluster Access Using kubeconfig Files . NOTE If a CCE cluster is used, you can obtain the Kubeconfig file by referring to Connecting to a Cluster Using kubectl .

4. Click **OK**.

Creating a Git Service Endpoint

After connecting to a Git repository, you can obtain the repository and branch information.

1. In a project, choose **Settings > General > Service Endpoints**.
2. Click **Create Endpoint** and choose **Git repository** from the drop-down list.
3. In the **Create Service Endpoint** dialog box, set the parameters.

Table 7-3 Parameter description

Parameter	Description
Service Endpoint Name	Name of the service endpoint.
Git Repository URL	Web URL of a Git repository (HTTPS address).
Username	Username used for logging in to the Git repository.
Password or Access Token	Password or access token used for logging in to the Git repository.

4. Click **OK**.

Creating an IAM User Service Endpoint

You can use an IAM user endpoint to delegate your AK/SK to the IAM user that needs to perform tasks. The user can obtain the token of your account through the AK/SK to perform tasks required higher permissions.

1. In a project, choose **Settings > General > Service Endpoints**.
2. Click **Create Endpoint** and choose **IAM user** from the drop-down list.
3. In the **Create Service Endpoint** dialog box, set the parameters.

Table 7-4 Parameter description

Parameter	Description
Service Endpoint Name	Name of the service endpoint.
Access Key Id	Access Key ID (AK). How Do I Obtain an Access Key (AK/SK)?
Secret Access Key	Secret Access Key (SK). See How Do I Obtain an Access Key (AK/SK)?

4. Click **OK**.

Creating a CodeArts Repo HTTPS Service Endpoint

The CodeArts Repo HTTPS endpoint is used to authorize CodeArts to download code, create branches, merge branches, and commit code in the Repo repository. Currently, it is mainly used for the microservice's pipeline on changes and related extensions.

1. In a project, choose **Settings > General > Service Endpoints**.
2. Click **Create Endpoint** and choose **CodeArts Repo HTTPS** from the drop-down list.
3. In the **Create Service Endpoint** dialog box that is displayed, set the parameters.

Table 7-5 Parameter description

Parameter	Description
Service Endpoint Name	Name of the service endpoint.
CodeArts Repo URL	Prefix of the clone address of the CodeArts Repo repository. The prefix can be obtained from CodeArts Repo. <ol style="list-style-type: none">Go to any code repository in the project.Obtain the HTTPS clone address of the code repository and enter https://xxx.com. NOTE When creating this endpoint during the pipeline creation, you do not need to set this parameter.
Username	HTTPS username is used to perform operations on the CodeArts Repo repository. The format is Tenant name/IAM username. Enter a complete username without spaces at the beginning or end.
Password	HTTPS password credential used to operate the CodeArts Repo repository.

 **NOTE**

Click the username in the upper right corner and choose **This Account Settings > Repo > HTTPS Password** to view and set the username and password.

- Click **OK**.

Editing or Deleting a Service Endpoint

On the **Service Endpoints** page, click a service endpoint on the list. The service endpoint details and operation buttons are displayed. Authorized users can edit and delete service endpoints.

 **NOTE**

After editing a referenced endpoint, you need to manually update the associated pipeline task.