

# CodeArts Pipeline

## User Guide

**Issue** 01  
**Date** 2024-06-18



**Copyright © Huawei Cloud Computing Technologies Co., Ltd. 2024. 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 Cloud Computing Technologies Co., Ltd.

## **Trademarks and Permissions**



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

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

## **Huawei Cloud Computing Technologies Co., Ltd.**

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

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

---

# Contents

---

<b>1 Pipelines.....</b>	<b>1</b>
1.1 Logging In to CodeArts Pipeline.....	1
1.2 Creating or Cloning a Pipeline.....	2
1.3 Configuring a Pipeline.....	4
1.4 Executing a Pipeline.....	13
1.5 Viewing a Pipeline.....	14
1.6 Managing Groups.....	15
1.7 Configuring a Pipeline Template.....	16
1.8 Configuring Pipeline Parameters.....	18
1.8.1 Background.....	18
1.8.2 Configuring Parameters.....	18
1.8.3 Using Parameters.....	21
1.9 Configuring Pipeline Execution Conditions.....	22
1.9.1 Configuration Method.....	22
1.9.2 Expressions.....	23
1.9.3 Pipeline Contexts.....	25
<b>2 Configuring Pipeline Permissions.....</b>	<b>31</b>
2.1 Introduction.....	31
2.2 Tenant-level Permissions.....	32
2.3 Project-level Permissions.....	33
2.4 Resource-level Permissions.....	39
<b>3 Rules and Policies.....</b>	<b>41</b>
3.1 Overview.....	41
3.2 Configuring a Rule.....	41
3.3 Tenant-level Policies.....	42
3.4 Project-level Policies.....	44
3.5 Using a Policy in a Pipeline.....	45
<b>4 Extensions.....</b>	<b>46</b>
4.1 Overview.....	46
4.2 Official Extensions.....	47
4.3 Official Tools.....	49
4.4 Custom Extensions.....	50

4.4.1 Registering an Extension.....	50
4.4.2 Using a Custom Extension.....	56
4.4.3 Developing Extensions Using Low Code.....	57
4.5 Basic Extensions.....	59
4.5.1 Registering a Basic Extension.....	59
4.5.2 Extension Properties.....	61
4.5.3 All inputs.....	64
4.6 Extensions and Policies.....	68
4.7 KubernetesRelease Extension.....	69
4.7.1 Introduction.....	69
4.7.2 Blue-Green Upgrade.....	70
4.7.3 CCE Rolling Upgrade.....	74
<b>5 Microservices.....</b>	<b>79</b>
5.1 Accessing Microservices.....	79
5.2 Creating a Microservice.....	80
5.3 Viewing a Microservice.....	80
<b>6 Changes.....</b>	<b>82</b>
6.1 Accessing Changes.....	82
6.2 Creating a Change.....	83
6.3 Viewing a Change.....	83
6.4 Managing Change-triggered Pipelines.....	84
<b>7 Release Environments.....</b>	<b>88</b>
7.1 Creating a Release Environment.....	88
7.2 Configuring an Environment Variable.....	90
7.3 Configuring an Environment Release Policy.....	93
7.4 Checking the Deployment Result.....	100
7.5 Using Cloud Native Release in Pipeline.....	102
<b>8 Service Endpoints.....</b>	<b>104</b>
<b>9 Key Operations Recorded by CTS.....</b>	<b>111</b>
9.1 CodeArts Pipeline Operations Recorded by CTS.....	111
9.2 Querying Real-Time Traces.....	111





# 1 Pipelines

## 1.1 Logging In to CodeArts Pipeline

This section describes how to access the pipeline list 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 execution mode, executor, branch, and ID of the latest code commit
Workflow	The scheduling process and execution status (successful, failed, running, or stopped) of the pipeline.
Started & Lasted	The start time and duration of the last execution.
Operation	Click  in the <b>Operation</b> column to execute the pipeline. Click  to edit, clone, or delete the pipeline, view the execution history (creation and deleting), and preview the pipeline. Click  to favorite the pipeline. After the pipeline is favorited, the icon changes to  . You can click the icon again to unfavorited 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 Cloning a Pipeline](#).
- (Optional) Click a pipeline name to view the pipeline execution history.
- (Optional) Click the project name of a pipeline 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 Cloning a Pipeline

### Procedure

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

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

On the displayed page, enter a pipeline name, select a project, select a code source, and set other parameters. For details, see [Table 1-1](#).

**Table 1-1** Parameter description

Code 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	Branch used when a pipeline is executed manually or at a specified time.
	Repo Endpoint	Configure endpoints to elevate permissions on repository operation. Endpoints are used for change-triggered pipelines and repository operation extensions.
	Alias	Repository alias. If an alias is set, the system parameters are generated for the repository (you can view the parameters on the parameter configuration page).

Code Source	Parameter	Description
	Description	Description of the pipeline.
GitLab	Service Endpoint	Select an available service endpoint or click <b>Create one</b> to <a href="#">create a GitLab service endpoint</a> .
	Repository	Select an available source code repository.
	Default Branch	Branch used when a pipeline is executed manually or at a specified time.
	Description	Description of the pipeline.
Git The repository configured in the service endpoint can be accessed through the common Git service endpoint.	Service Endpoint	Select an existing Git service endpoint or click <b>Create one</b> to <a href="#">create a Git service endpoint</a> .
	Repository	Select an available source code repository.
	Default Branch	Branch used when a pipeline is executed manually or at a specified time.
	Description	Description of the pipeline.

**NOTE**

- If a pipeline does not need to be associated with a code repository, select **None** for pipeline source.
- Currently, the GitLab pipeline source is available in LA-Mexico City<sup>2</sup>, LA-Sao Paulo<sup>1</sup>, and AP-Singapore.

**Step 3** Click **Next**.

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

**Step 4** After selecting a template, 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 **Clone**.
- Click the name of the target pipeline, on the displayed **Execution History** tab page, click **...** in the upper right corner and click **Clone**.
- Click the name of the target pipeline, then click the execution ID or the **Pipeline Details** tab. On the displayed page, click **...** in the upper right corner and click **Clone**.

On the displayed page, you can change configurations as needed. 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 configure the basic information (name and description), [task orchestration](#), [parameters](#), [execution plans](#), [permissions](#), and [notifications](#).


**Step 3** Click **Save**.

----End




### Task Orchestration

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



- Configuring the code source  
Click the code source area. On the displayed dialog box, you can modify the code source.
- Managing 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, clone, or move the stage.

**Table 1-2** Stage management

Operation	Description
Editing a stage	Click  . In the displayed dialog box, you can configure the stage name and whether to always run the stage. <b>NOTE</b> <b>Always Run:</b> If you select <b>Yes</b> , jobs in this stage will be executed and cannot be canceled. If you select <b>No</b> , jobs will be selected by default but can be deselected.
Deleting a stage	Click  and confirm the deletion as prompted.
Cloning a stage	Click  to clone a pipeline stage.






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

- Managing a job



You can add jobs to each stage. After a job is added, you can edit, clone, delete, or move the job.

**Table 1-3** Job management

Operation	Description
Adding a job	<ul style="list-style-type: none"><li>• Click <b>Job</b> to add a job to an empty stage.</li><li>• Click  under a job to add a serial job.</li><li>• Click <b>Parallel Job</b> to add a parallel job.</li></ul> <p><b>NOTE</b></p> <ul style="list-style-type: none"><li>• Serial execution: Jobs are executed in sequence. For example, a build job and a deployment job must be executed sequentially.</li><li>• Parallel execution: Jobs are executed at the same time. For example, a code check job and a build job can be executed at the same time.</li></ul>
Editing a job	Click a job card to edit the job.
Cloning a job	Click  on the job card to clone a serial job.
Deleting a job	Click  on the job card and confirm the deletion as prompted.
Sorting jobs	Click, hold, and move a job card to adjust the sequence. <p><b>NOTE</b> Job sequence cannot be adjusted when jobs are executed in parallel.</p>

When adding or editing a job, you can configure the job in the displayed dialog box.

**Table 1-4** Job configuration

Operation	Description
Adding an extension	<p>There are five types of extensions: build, code check, deployment, test, and normal extensions. You can filter or search for extensions by type. Move the cursor to an extension card and click <b>Add</b>.</p> <p>Set parameters:</p> <ul style="list-style-type: none"><li>• Enter an extension name.</li><li>• Select a job to be called. If no proper job is available, create a job as prompted.</li><li>• If the called job has parameters, the parameters are displayed. Set the parameters.</li><li>• The extension name is followed by a flag. You can add only one extension with flag <i>Job</i> to a single job. Extensions with flag <i>draft</i> indicate that they are draft extensions.</li><li>• The extension for suspending a pipeline can only be added to stages that do not contain parallel jobs.</li></ul>
Deleting an extension	<p>Move the cursor to an extension card, click , and select <b>Delete</b> to delete the extension.</p>
Replacing an extension	<p>Move the cursor to an extension card, click , and select <b>Replace</b> to replace the extension. Or, click <b>Replace Extension</b> above the extension name to choose another extension.</p>
Sorting extensions	<p>Click, hold, and move an extension card to adjust the extension sequence.</p>
Job configuration	<p>Set job ID, executor, and <b>execution condition</b>.</p> <ul style="list-style-type: none"><li>• The job ID should be unique.</li><li>• You can use the built-in executor or customize one.<ul style="list-style-type: none"><li>– Built-in executor: provided by CodeArts Pipeline with out-of-the-box availability.</li><li>– Custom executor: allows you to configure tools and running environments as needed. Before using a custom executor, add an agent pool. For details, see <b>Agent Pools</b>.</li></ul></li><li>• Execution conditions are the triggers for executing jobs in a pipeline.</li></ul> <p><b>NOTE</b> You only need to configure executors for non-job-level extensions.</p>

- Configuring pass conditions

Click **Pass Conditions** under a stage. In the displayed dialog box, move the cursor to the pass conditions card and click **Add**, and then configure a policy for the pass conditions.

- Only **Pass-Conditions-of-Standard-Policies** is available.
- Policy: You can select a created policy.

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 conditions to multiple pipelines. For details, see [Rules and Policies](#).

#### NOTE

- You can set exclusive pass conditions for each stage.
- You can set multiple pass conditions for one stage.

## Setting Parameters

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

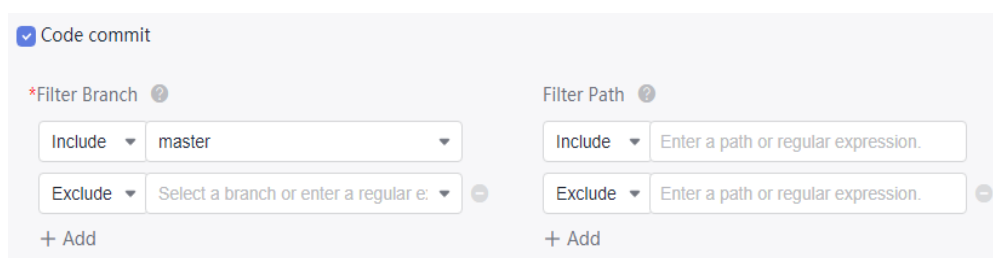
#### NOTE

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

## Configuring Execution Plans

Switch to the **Execution Plan** tab page, and configure trigger events, scheduled task, and parallel execution policy for a pipeline. Trigger events include code commit, 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 requests (Repo)

The related pipeline is automatically executed when there is an MR creation, updating, code updating, code merge, or MR reopening event in the associated code repository, and the related branch and path of changed files meet inclusion and exclusion policies.

Event description:

- **Create**: triggered upon MR creation.
- **Merge**: triggered upon code merge. This event triggers code commit event.
- **Reopen**: triggered upon MR reopening.
- **Update**: triggered upon MR content, setting, or source code update. If you enable **Code update** at the same time, the pipeline will be triggered only upon source code update.

Branch description:

- Branch inclusion: If the target branch is included, the matching is successful.
- Branch exclusion: If the 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.

Merge request

Event 

Create  Merge  Reopen

Update  Code update

\*Filter Branch 

Include

Exclude

+ Add

Filter Path 

Include

Exclude

+ Add

- Triggered upon tag creation (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, the matching is successful.
- Tag exclusion: If the tag created in the code repository is excluded, the matching fails.

✓ Tag creation

\*Filter Tag ?

Include ▼ \*

Exclude ▼ Enter a tag or regular expression. -

+ Add

NOTE

- The branch is matched first, and then the path (if configured) is matched. If the matching is successful, the pipeline will be 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 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 **Create now** to create a scheduled task. Turn on the **Enable** toggle, set the execution time and save the configuration. The pipeline will be 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 create a maximum of 10 scheduled tasks.

- Parallel execution

By default, five parallel executions are allowed in a pipeline. Excess instances will not be executed. Alternatively, you can change the maximum number of parallel instances (running and paused).

Enable **Parallel Execution**, set the execution policy for extras, and active the policy by saving the configurations.

#### Parallel Execution

Maximum pipeline instances (running and paused) allowed in a pipeline.

\* Parallel Instances

\* For Excess Instances

- Wait  
 Ignore

Parameter	Description
Parallel Instances	Maximum parallel instances, which vary by your purchases and packages.
For Excess Instances	You can choose: <ul style="list-style-type: none"><li>● <b>Wait:</b> Excess instances will wait for execution. You can view the queuing instances on the pipeline details page.<ul style="list-style-type: none"><li>- Max. 100 queuing instances per pipeline.</li><li>- Instances will not be executed after 24 hours of waiting.</li><li>- You can manually cancel the waiting.</li><li>- Configurations of instances will not be changed once they enter the queue.</li></ul></li><li>● <b>Ignore:</b> Excess instances will not be executed.</li></ul>

## Managing Permissions

Switch to the **Permissions** tab page. You can configure role permissions and user permissions for the pipeline.

- If you retain the default role permissions, 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 automatically synchronize with role permissions. If user permissions are configured, the user permissions overwrite the role permissions.

### Role Permissions

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

Role Permissions		User Permissions			Enable Project Permissions	Tips
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"/>		
Product manager	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		
System engineer	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
Committer	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		

### User Permissions

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

Role Permissions		User Permissions			Use Project-level Permissions	Tips	
User	Role	Alias	Enterprise User	View	Execute	Edit	Delete
<input type="text" value="Please enter a user, alias, or enterprise user."/>							
...	Pipeline creator	pipeline	...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

### NOTE

By default, a user with permissions to edit or execute pipelines can also view pipelines.

## Notifications

Switch to the **Notifications** tab page and configure event notifications for the pipeline.


- Internal messages  
Pop-up notifications and emails will be sent to notify related personnel of pipeline activities (deleted, failed, succeeded, updated).

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
Pipeline run failed	<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 succeeded	<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 u...	<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, only pop-up notifications will be sent.
- You can click  in the upper right corner of the pipeline homepage to check notifications in 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, set the following parameters:

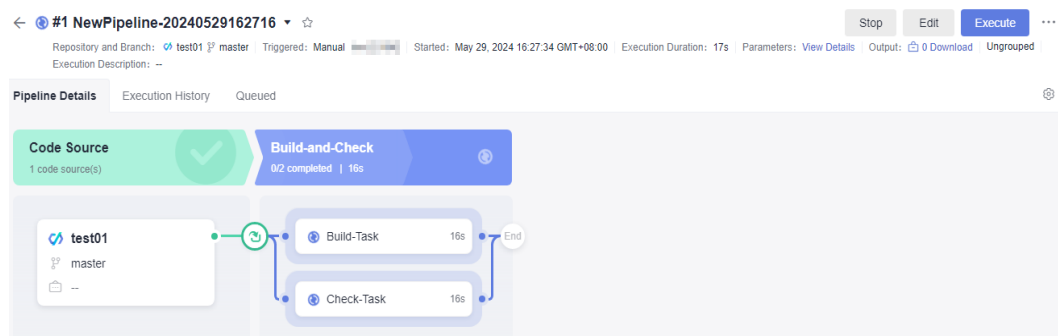
- **Code Source:** Select the branch or tag of the code source.
- **Runtime Parameters:** Set runtime parameters as needed and then save them. For details, see [Using Parameters](#).
- **Execution Stages:** Select one or more jobs to execute. By default, all jobs are executed.

**NOTE**

If **Always Run** is set to **Yes** for a stage, jobs 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 **Execute**. On the pipeline details page, you can view the execution progress and job status in real time.



- Click **Stop** in the upper right corner to stop the execution.
- Pipelines can be executed in parallel. You can click **Execute** to continue executing a pipeline. The maximum number of parallel pipeline executions varies based on your purchase (5 for basic edition, 10 for professional edition, and 15 for platinum edition).

**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.

### Procedure

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


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

You can click the time filter to view execution records by time. By default, executions in the past 7 days are displayed. You can also view executions in the past 14 days or 31 days.

 **NOTE**

Execution records are generated only after the first execution.

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

Click  in the upper right corner to edit, clone, and delete a pipeline, and check the pipeline operation histories (creation and editing).

You can perform the following operations on the **Pipeline Details** page.

Operation	Description
Retry	If the execution fails, you can click <b>Retry</b> in the upper right corner to continue the execution.
Edit	Click <b>Edit</b> to edit the pipeline by referring to <a href="#">Configuring a Pipeline</a> .
Execute	Click <b>Execute</b> in the upper right corner to execute the pipeline again with the latest configurations. An execution record can be generated.
Download	Click <b>Download</b> next to <b>Output</b> to download the automatically built packages. <b>NOTE</b> <ul style="list-style-type: none"><li>Build packages are available only for build jobs.</li><li>If there are multiple build packages, click <b>Download All</b>.</li><li>Only the latest 10 build packages are displayed. To download other build packages, go to the <b>Release Repos</b> page.</li></ul>
View logs	Click a job card to view its log. <b>NOTE</b> No log will be generated for jobs of <b>DelayedExecution</b> and <b>PipelineSuspension</b> .

Operation	Description
More operations	<p>Click <b>...</b> in the upper right corner of the page to preview, disable, delete, and <b>clone the pipeline</b>, as well as check its operation histories (creation, editing, trigger failure).</p> <p><b>NOTE</b> By default, only project managers, project creators, and pipeline creators can delete pipelines. You can configure permissions for different roles.</p>

----End

## 1.6 Managing Groups


### Scenarios

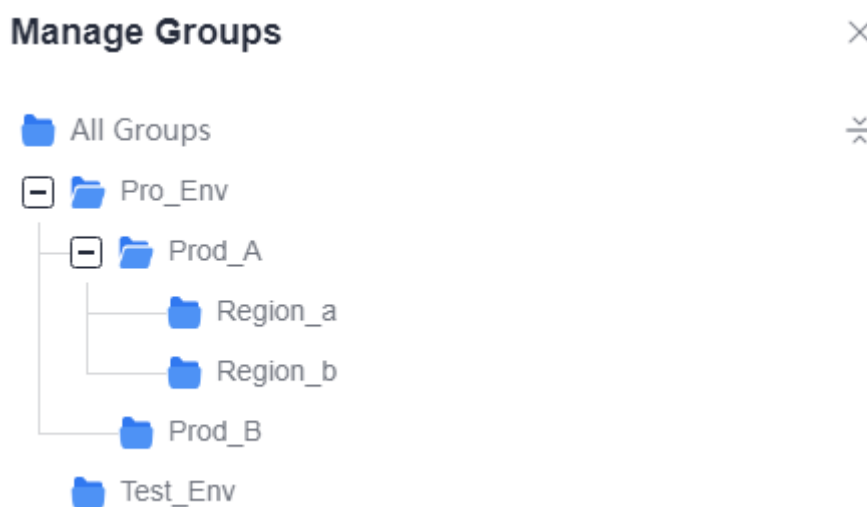
You can group pipelines by features to improve efficiency. For example, you can group pipelines into production and test by environment level, or into scheduled build, development self-test, integration test, and production and deployment by R&D stage.

### Prerequisites



- Only project creators and project managers can manage groups.
- The group function applies only to pipelines of the new version.

### Procedure




1. Go to the pipeline list page of a 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. Specify a group name. Click  to confirm group creation or click  to cancel group creation.

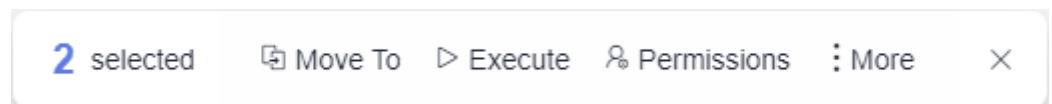
After a group is created, you can perform the following operations:

- 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

After the first group is created, **Ungrouped** is also automatically generated for ungrouped pipelines.

6. Click **Close** to return to the pipeline list page after all groups are created.
7. Select desired pipelines and perform the following operations.



- Click **Move To**. The **Move Group** dialog box is displayed. Select a group and click **Confirm**.
- Click **Execute**. In the displayed dialog box, click **OK**.
- Click **Permissions**. In the displayed dialog box, configure permissions for selected pipelines.
- Choose **More > Set Tag**. In the displayed dialog box, set tags for selected pipelines.
- Choose **More > Delete**. In the displayed dialog box, confirm the information and click **OK**.

## 1.7 Configuring a Pipeline Template

You can quickly generate a pipeline based on a template. The template page can be accessed via:

- Homepage: Access the Pipeline homepage and switch to the **Templates** tab page.
- Project: Access the pipeline list in a project and choose **More > Templates** in the upper right corner.






#### NOTE

The templates in **Templates** can be selected during [pipeline creation](#).

### Template Types

CodeArts Pipeline provides system and custom templates.

You can perform the following operations on templates.

Icon	Description
	If you click this icon, you will be redirected to the page where you can quickly create a pipeline using a template.
	Click this icon to favorite a template. After a template is favorited, the icon changes to  . You can click  to unfavorite the template.
	<ul style="list-style-type: none"><li>Click this icon and select <b>Edit</b>. On the displayed <b>Task Orchestration</b> tab page, you can edit the template.</li><li>Click this icon and select <b>Clone</b>. On the displayed <b>Task Orchestration</b> tab page, you can clone the template.</li><li>Click this icon and select <b>Delete</b> to delete the template as prompted.</li></ul>

 **NOTE**

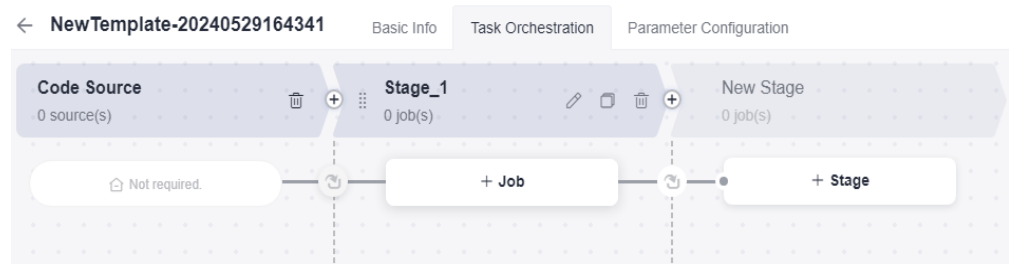
System templates are used to clone or generate pipelines. They cannot be edited or deleted.

## 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 information, stages/jobs, and parameters.

- **Basic Information:** Specify the template name (mandatory), language (Java, Python, Node.js, Go, .Net, C++, PHP), and description (optional). Language is **None** by default.
- **Code Source:** You do not need to set this parameter.
- **Admission Settings:** This function is not supported for template orchestration.
- **Task Orchestration:** Pipeline stages and some extensions can be added to a pipeline template. After jobs such as build, code check, deployment, and API test are configured in a template, corresponding jobs will be created when you create a pipeline using this template.
- **Parameter Configuration:** Switch to the **Parameter Configuration** tab page and add parameters to the template. 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 [Configuring Pipeline 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 **Create Pipeline** page, enter a pipeline name, select a project, select a code source, set other parameters, and click **OK**.
3. Access the **Task Orchestration** page.
  - You can customize the template or directly click **Save**.
  - For the template-generated pipeline, jobs mounted to it are generated by the system. To mount custom jobs to the pipeline, manually add them after the pipeline was generated.
4. After configuring the information, click **Save**.

# 1.8 Configuring Pipeline Parameters

## 1.8.1 Background

Pipeline parameters can be transferred among jobs. By configuring pipeline parameters, you can streamline data of build, deployment, and API test jobs. Parameters include:

- **Custom Parameters:** You can add parameters of string, enumeration, or auto-increment type.
- **Predefined Parameters:** You do not need to configure predefined parameters. They cannot be deleted or modified.
- **Parameter Groups:** You can associate all pipelines in the project with a parameter group.

### NOTE

- For Repo sources, if you set a repository alias, repository-related predefined parameters will be generated.
- Parameter priority: predefined parameters > custom parameters > parameter groups.
- If a pipeline is associated with multiple parameter groups and parameters with the same name exist, the value of the parameter in the last associated parameter group will be used.

## 1.8.2 Configuring Parameters

You can create and configure custom parameters and group parameters for pipelines.


### Custom Parameters

**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.

You can configure the following information.

Basic Information	Description
Create now	Click <b>Create now</b> to create a parameter.
Name	Parameter name. You can specify a parameter name. <b>NOTE</b> The specified name cannot be the same as that of a predefined parameter.
Type	Parameter types include string, auto-increment, and enumeration.
Default	You can enter or select a parameter value.
Private Parameter	If a parameter is private, the system encrypts the parameter for storage and decrypts the parameter for usage. Private parameters will not be displayed in run logs.
Runtime Setting	If <b>Runtime Setting</b> is enabled, you can change the value of this parameter during task orchestration.
Description	Description of a parameter.
Operation	You can click  to delete a parameter.

**Step 4** Create and configure parameters.

Click **Create now**, set the name, type (**String** by default), and parameter value. You can also set it as a private parameter and enable runtime setting.

- **String**

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

- **Enumeration**

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

After the setting is complete, you can click the **Default** drop-down list box to select another value. Or, you can click **Enumeration** 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 job 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

## Parameter Groups

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

**Step 2** Go to the pipeline list page and click **Parameter Groups**.

**Step 3** Click **Create Group**.


Enter a name and description. Click **Create now** to create custom parameters. For details, see [Custom Parameters](#).

• Project  
Project01

• Name  
Enter a name.

Description  
Enter a description.  
0 / 512


Name	Type	Default	Private Parameter	Description	Operation
------	------	---------	-------------------	-------------	-----------





No custom parameters yet. [Create now](#)

**Step 4** Go to the pipeline editing page, choose **Parameter Configuration > Parameter Groups**.

**Step 5** Click **Associate Now**, select a parameter group, and click **Confirm** to associate the parameter group with the pipeline.

- Expand the group to check parameter details.
- Click  in the **Operation** column to diassociate with the parameter group.

Name	Modified By	Modified	Operation
 generalVariableGroup01	pipeline	Jun 21, 2024 13:09:23 GMT+08:00	
Name	Type	Default	Description
param01	String	1.0.0	--

+ Associate with Group

----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 job.

### Procedure

**Step 1** Create a build task.

**Step 2** On the **Parameters** tab page, add the **releaseversion** parameter, set the default value, and enable **Runtime Settings**.

Name	Type	Default Value	Private Paramete	Runtime Settings	Params Description	Operation
codeBranch	String	master	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Parameters	
releaseversion	String	1.0.0	<input type="checkbox"/>	<input checked="" type="checkbox"/>		

**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.

### Upload Software Package to Release Repository

Upload a software package to Release Repo. [View User Guide](#)

\* Action Name

\* Package Location

Version

Package Name

#### NOTE

\$ will not trigger the display of parameter groups.

**Step 4** Save the build task.

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

\* Name: releaseversion  
Pipeline parameters can be referenced in tasks.

\*

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

**Step 7** Switch back to the **Task Orchestration** tab page, and edit the added build task. Use **\$** to reference **releaseversion** parameter in the build task.

\*releaseversion

#### NOTE

- Only parameters for which **Runtime Setting** is enabled are displayed.
- The parameter reference format is "**`\${ParameterName}`**". *ParameterName* indicates 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 Execute**. In the displayed dialog box, you can check the parameter information.

The parameter value is the default value specified when you add the parameter. You can change the value. If you change it, the new value will be used in the build job.

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

----End

## 1.9 Configuring Pipeline Execution Conditions

### 1.9.1 Configuration Method

Execution conditions are the prerequisites for executing jobs in a pipeline. You can configure execution conditions to control job executions.

1. Edit or add a job in pipeline.
2. In the displayed dialog box, click the job configuration card and configure the execution.

#### NOTE

Add an extension first before configuring a job.

Job Config

Build-Task

Add Extension

Task ID

ID

Task\_1

Execute

Even when previous job is not selected

When previous job succeeds

If previous job fails

Always

With expression

{{ }}

Parameter	Description
ID	This ID is unique.
Execute	<p>Set one of the following:</p> <ul style="list-style-type: none"><li>• <b>Even when previous job is not selected:</b> The current job is executed if the previous job is completed or not selected.</li><li>• <b>When previous job succeeds:</b> The current job is executed only when the previous job is successfully executed.</li><li>• <b>If previous job fails:</b> The current job is executed only when the previous job fails.</li><li>• <b>Always:</b> The current job is always executed regardless of the previous job's final state (failed, completed, canceled, or ignored).</li><li>• <b>With expression:</b> The expression is in the format of <code>\${{value}}</code> and can be any combination of contexts, operators, functions, or literals. Only if the previous job is in the completed, failed, canceled, or ignored state, and the expression's result is <b>true</b>, the current job is executed.<ul style="list-style-type: none"><li>– For details about the expression, see <a href="#">Expressions</a>.</li><li>– For details about the pipeline context, see <a href="#">Pipeline Contexts</a>.</li></ul></li></ul>

## 1.9.2 Expressions

An expression can be any combination of [contexts](#), [operators](#), [functions](#), or literals. You can use an expression as the execution condition to control job execution. Contexts can be accessed programmatically with expressions, so information such as pipeline runs, sources, variables, and jobs can be transferred within a pipeline.

## Operator

Operator	Description
.	Attribute reference
!	False
==	Equal
!=	Not equal
&&	And
	Or

### Example

If the current job is executed regardless of whether the previous job (ID: **job\_1**) succeeded or failed, the expression can be as follows:

```
${{ jobs.job_1.status == 'COMPLETED' || jobs.job_1.status == 'FAILED' }}
```

## Functions

The following functions can be used in expressions:

- **contains**
  - Format  
`contains(search, item)`
  - Description  
If *search* contains *item*, this function returns **true**. If *search* is an array, this function returns **true** if the *item* is an element in the array. If *search* is a string, this function returns **true** if the *item* is a substring of *search*.
  - Example  
**contains('abc', 'bc')** returns **true**.
- **startsWith**
  - Format  
`startsWith(searchString, searchValue)`
  - Description  
If *searchString* starts with *searchValue*, this function returns **true**.
  - Example  
**startsWith('abc', 'ab')** returns **true**.
- **endsWith**
  - Format  
`endsWith(searchString, searchValue)`
  - Description  
If *searchString* ends with *searchValue*, this function returns **true**.

- Example  
`endsWith('abc', 'bc')` returns **true**.
- **Object filter**  
You can use the `*` syntax to apply a filter and select matching items in a collection.

Example

The following is the context of jobs execution.

The filter `jobs.*.status` returns `['COMPLETED', 'FAILED']`.

Filters can be used together with the `contains` function. For the following case, `contains(jobs.*.status, 'FAILED')` will return **true**.

```
{
  "check_job": {
    "status": "COMPLETED",
    "metrics": {
      "critical": "0",
      "major": "0"
    }
  },
  "demo_job": {
    "status": "FAILED"
  }
}
```

### 1.9.3 Pipeline Contexts

Contexts are a way to access information about pipeline runs, sources, variables, and jobs. Each context is an object that contains various attributes.

- Reference format  
`${{ <context>.<attribute_name> }}`
- Pipeline contexts

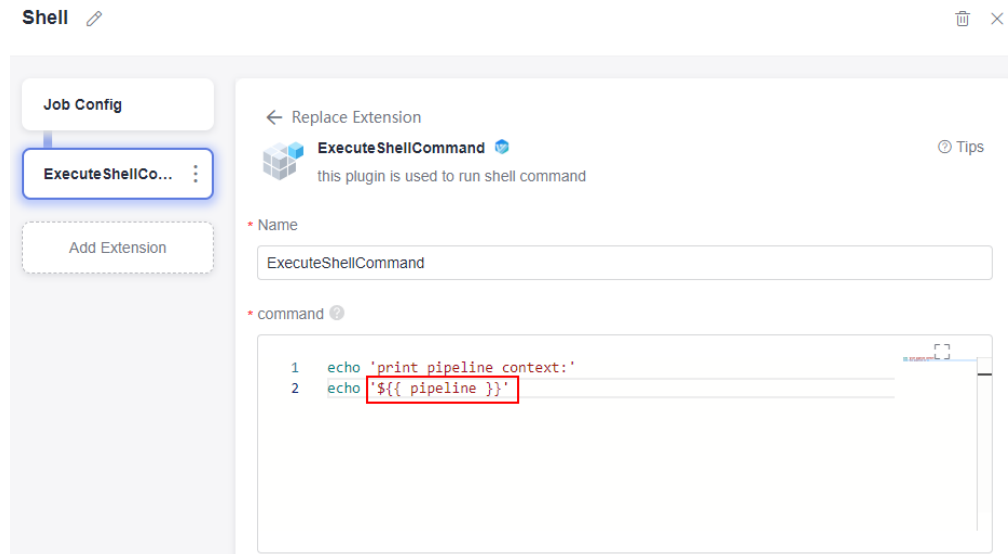
Context Name	Type	Description
pipeline	object	Information about the pipeline run.
sources	object	Information about the pipeline sources in each pipeline run.
env	object	Information about the custom parameters in each pipeline run.
jobs	object	Information about jobs that have reached the final states in each pipeline run.

## Scenarios

Most contexts can be used in any job and step of a pipeline.

- You can use contexts to specify the execution condition of a job.  
The following example shows that a job runs only when the running branch of the specified code source is **master**.  
`${{ sources.my_repo.target_branch == 'master' }}`

- You can use contexts when configuring parameters to query information.



The following expression shows how to obtain all pipeline run information.  
`{{ pipeline }}`

The following expression shows how to obtain the triggering mode of a pipeline.  
`{{ pipeline.trigger_type }}`

## pipeline Context

The pipeline context contains pipeline run information.

Name	Type	Description
pipeline	object	Information about the pipeline run. This object contains the following attributes: <b>project_id</b> , <b>pipeline_id</b> , <b>run_number</b> , <b>timestamp</b> , <b>trigger_type</b> , and <b>run_id</b> .
pipeline.project_id	string	ID of the project to which the current pipeline belongs. This string is the same as the predefined parameter <b>PROJECT_ID</b> .
pipeline.pipeline_id	string	Current pipeline ID. This string is the same as the predefined parameter <b>PIPELINE_ID</b> .
pipeline.run_number	string	Pipeline execution number. This string is the same as the predefined parameter <b>PIPELINE_NUMBER</b> .
pipeline.timestamp	string	Pipeline execution timestamp. This string is the same as the predefined parameter <b>TIMESTAMP</b> . For example, 20211222124301.
pipeline.trigger_type	string	Pipeline triggering type. This string is the same as the predefined parameter <b>PIPELINE_TRIGGER_TYPE</b> .

Name	Type	Description
pipeline.run_id	string	Pipeline execution ID. This string is the same as the predefined parameter <b>PIPELINE_RUN_ID</b> .

- Content example

The following example shows the pipeline context information contained in a manually executed pipeline.

```
{
  "project_id": "6428c2e2b4b64affa14ec80896695c49",
  "pipeline_id": "f9981060660249a3856f46c2c402f244",
  "run_number": "168",
  "timestamp": "20231016000004",
  "trigger_type": "Manual",
  "run_id": "c2f507f93510459190b543e47f6c9bec"
}
```

- Usage example

To obtain the triggering mode of the current pipeline, you can use the following syntax:

```
${{ pipeline.trigger_type }}
```

## sources Context

The sources context contains information about the pipeline sources.

Name	Type	Description
sources	object	Information about the pipeline sources in each pipeline run. This object contains the following attributes: <b>alias</b> , <b>repo_name</b> , <b>commit_id</b> , <b>commit_id_short</b> , <b>commit_message</b> , <b>repo_url</b> , <b>repo_type</b> , <b>ssh_repo_url</b> , <b>tag</b> , <b>merge_id</b> , <b>source_branch</b> , and <b>target_branch</b> .
sources.<alias>	object	Information about the pipeline source which has an alias.
sources.<repo_name>	object	Information about the pipeline source which does not have an alias but only a repository name. It contains the same information as that in <b>sources.&lt;alias&gt;</b>
sources.<alias>.commit_id	string	The last commit ID before execution. This string is the same as the predefined parameter <b>COMMIT_ID</b> .
sources.<alias>.commit_id_short	string	The last commit short ID before execution. This string is the same as the predefined parameter <b>COMMIT_ID_SHORT</b> .
sources.<alias>.commit_message	string	The last commit message before execution.

Name	Type	Description
<code>sources.&lt;alias&gt;.repo_url</code>	string	Code repository address (HTTPS). This string is the same as the predefined parameter <b>REPO_URL</b> .
<code>sources.&lt;alias&gt;.repo_type</code>	string	Type of the code repository.
<code>sources.&lt;alias&gt;.repo_name</code>	string	Name of the code repository.
<code>sources.&lt;alias&gt;.ssh_repo_url</code>	string	Code repository address (SSH).
<code>sources.&lt;alias&gt;.tag</code>	string	Tag name when the tag is triggered.
<code>sources.&lt;alias&gt;.merge_id</code>	string	Merge request ID when the merge request is triggered.
<code>sources.&lt;alias&gt;.source_branch</code>	string	Source branch name when the merge request is triggered.
<code>sources.&lt;alias&gt;.target_branch</code>	string	If the merge request is triggered, this string indicates the name of the target branch. Otherwise, this string indicates the name of the running branch.

- Content example

The following example shows the sources context information contained in a manually executed pipeline with a single code source. The alias of pipeline source is **my\_repo**.

```
{
  "my_repo": {
    "commit_id": "dedb73bb9abfdaab7d810f2616bae9d2b6632ecc",
    "commit_id_short": "dedb73bb",
    "commit_message": "maven0529 update pipeline0615.yml",
    "repo_url": "https://example.com/clsyz00001/maven0529.git",
    "repo_type": "codehub",
    "repo_name": "maven0529",
    "ssh_repo_url": "git@example.com:clsyz00001/maven0529.git",
    "target_branch": "master"
  }
}
```

- Usage example

To obtain the running branch of the pipeline, you can use the following syntax:

```
${{ sources.my_repo.target_branch }}
```



## env Context

The env context contains information about the custom parameters in each pipeline run.

Name	Type	Description
name	string	Name of a custom parameter.
value	string	Value of a custom parameter.

- Content example

The following example shows the env context information in a run, which includes two custom parameters.

```
{
  "var_1": "val1",
  "var_2": "val2"
}
```

- Usage example

To obtain the value of the custom parameter **var\_1**, you can use the following syntax:

```
${{ env.var_1 }}
```

## jobs Context

The jobs context contains information about jobs that have reached the final states in each pipeline run.

Name	Type	Description
jobs	object	Information about jobs in a pipeline. This object contains the following attributes: <b>job_id</b> , <b>status</b> , <b>outputs</b> , <b>output_name</b> , <b>metrics</b> , and <b>metric_name</b> .
jobs.<job_id>	object	Information about the job with a specified ID.
jobs.<job_id>.status	string	Execution result of a job. The value can be <b>INIT</b> , <b>QUEUED</b> , <b>RUNNING</b> , <b>CANCELED</b> , <b>COMPLETED</b> , <b>FAILED</b> , <b>PAUSED</b> , <b>IGNORED</b> , or <b>UNSELECTED</b> .
jobs.<job_id>.outputs	object	General output of an executed job.
jobs.<job_id>.outputs.<output_name>	string	The value of output_name of an executed job.
jobs.<job_id>.metrics	object	The running metric of a job.

Name	Type	Description
<code>jobs.&lt;job_id&gt;.metrics.&lt;metric_name&gt;</code>	string	The value of <code>metric_name</code> of an executed job.

- Content example

The following example shows the jobs context information in a run. There are two successfully executed jobs. The output of the **check\_job** job is two metrics, and the output of the **demo\_job** job is two general outputs.

```
{
  "check_job": {
    "status": "COMPLETED",
    "metrics": {
      "critical": "0",
      "major": "0"
    }
  },
  "demo_job": {
    "status": "COMPLETED",
    "outputs": {
      "output1": "val1",
      "output2": "val2"
    }
  }
}
```

- Usage example

To obtain the value of **output1** of **demo\_job**, you can use the following syntax:

```
${{ jobs.demo_job.outputs.output1 }}
```

# 2 Configuring Pipeline Permissions

## 2.1 Introduction

You can configure CodeArts Pipeline permissions at three levels to manage different user behaviors.

Level	Module	Permission Description
Tenant (all accounts)	Extension , tenant-level policy, tenant-level rule, and pipeline template	Permissions to manage resources of all modules under a tenant. You can configure permissions in IAM. The configuration takes effect for all projects of a tenant.
Project	Pipelines, project-level policies, microservices, environments, and changes	Permissions to manage module resources of a specific project. You can configure permissions in project settings. The configuration takes effect for all resources of a project.
Resource	Pipeline	Permissions to perform operations on a specific pipeline. You can configure role and user permissions when editing a pipeline.

## 2.2 Tenant-level Permissions

An administrator can use IAM to configure permissions on 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 on the left, choose **User Groups**. On the displayed page, create a user group or select an existing user group, and click **Authorize**.

Select the CodeArts Pipeline service to view policies, as shown in the following table.

Policy Name	Description
CloudPipeline Tenant Rules FullAccess	Full permissions on tenant-level rules of CodeArts Pipeline.
CloudPipeline Tenant Rule Templates FullAccess	Full permissions on tenant-level policies of CodeArts Pipeline.
CloudPipeline Tenant Extensions FullAccess	Full permissions on extensions of CodeArts Pipeline.
CloudPipeline Tenant Pipeline Templates FullAccess	Full permissions on 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** to manage **Rules** and **Policies**.

- Permissions on rules and policies correspond to **cloudpipeline:rule:update** and **cloudpipeline:ruletemplate:update** in IAM respectively. An administrator can use 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 can choose **Policy Management > Rules** to view all rules. Authorized users can view and manage all tenant-level rules.
- Common users can choose **Policy Management > Policies** to view all policies. Authorized users can view and manage all tenant-level policies.

## Extensions

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

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

## Pipeline Templates

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

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

## 2.3 Project-level Permissions

CodeArts allows you to configure permissions on pipeline resources for each role in a project.

### Configuration Method

1. Log in to the CodeArts homepage.
2. Click the target project name to access the project.
3. On the left navigation pane, choose **Settings > General > Service Permissions**.

On this page, the project creator and users who have the management permission can modify the permissions on pipelines of different roles in the project.

Pipeline-related resources are in CodeArts Pipeline, including pipelines, policies (project-level), microservices, environments, and changes.

 NOTE

By default, a user with permissions to edit or execute pipelines can also view pipelines.

## Pipeline Permissions

The following table lists the pipeline permissions for each role in a project in the initial state.

Role	View	Create	Execute	Edit	Delete	Group
Project creator	√	√	√	√	√	√
Project manager	√	√	√	√	√	√
Developer	√	√	√	x	x	x
Test manager	√	x	x	x	x	x
Tester	√	x	x	x	x	x
Participant	√	x	x	x	x	x
Viewer	√	x	x	x	x	x
Operation manager	√	x	x	x	x	x
Product manager	√	x	x	x	x	x
System engineer	√	√	√	√	√	√
Committer	√	√	√	x	x	x

- To copy a pipeline, you must have the permission to create a pipeline and modify the source pipeline.
- By default, role permissions in a pipeline inherit and are associated with the role permissions in the project until role permissions are modified in the pipeline.
- By default, a pipeline creator has all permissions on the pipeline.

## Policy Permissions

The following table lists the project-level policy permissions for each role in a project in the initial state.

Role	View	Create	Edit	Delete
Project creator	√	√	√	√
Project manager	√	√	√	√
Developer	√	√	√	√
Test manager	√	×	×	×
Tester	√	×	×	×
Participant	√	×	×	×
Viewer	√	×	×	×
Operation manager	√	×	×	×
Product manager	√	×	×	×
System engineer	√	√	√	√
Committer	√	√	√	√

** NOTE**

To copy a policy, you must have the permission to create a policy and modify the source policy.

## Microservice Permissions

The following table lists the microservice permissions for each role in a project in the initial state.

Role	View	Create	Edit	Delete
Project creator	√	√	√	√
Project manager	√	√	√	√
Developer	√	×	×	×
Test manager	√	×	×	×
Tester	√	×	×	×
Participant	√	×	×	×
Viewer	√	×	×	×

Role	View	Create	Edit	Delete
Operation manager	√	×	×	×
Product manager	√	×	×	×
System engineer	√	√	√	√
Committer	√	×	×	×

## Change Permissions

The following table lists the change permissions for each role in a project in the initial state.

Role	View	Create	Edit	Execute
Project creator	√	√	√	√
Project manager	√	√	√	√
Developer	√	√	√	√
Test manager	√	×	×	×
Tester	√	×	×	×
Participant	√	×	×	×
Viewer	√	×	×	×
Operation manager	√	×	×	×
Product manager	√	×	×	×
System engineer	√	√	√	√
Committer	√	√	√	√

## Environment Permissions

For release environments, the default permissions for each role in a project can be configured in a unified manner. The involved modules include the development environment, test environment, pre-release environment, and production environment.



The following table lists the microservice release permissions for each role in different environments in the initial state.

**Table 2-1** Development environment

Role	View	Create	Edit	Delete	Execute	Roll Back
Project creator	√	√	√	√	√	√
Project manager	√	√	√	√	√	√
Developer	√	√	√	√	√	√
Test manager	√	×	×	×	×	×
Tester	√	×	×	×	×	×
Participant	√	×	×	×	×	×
Viewer	√	×	×	×	×	×
Operation manager	√	√	√	√	√	√
Product manager	√	√	√	√	√	√
System engineer	√	√	√	√	√	√
Committer	√	√	√	√	√	√

**Table 2-2** Testing environment

Role	View	Create	Edit	Delete	Execute	Roll Back
Project creator	√	√	√	√	√	√
Project manager	√	√	√	√	√	√
Developer	√	×	×	×	×	×

Role	View	Create	Edit	Delete	Execute	Roll Back
Test manager	√	√	√	√	√	√
Tester	√	√	√	√	√	×
Participant	√	×	×	×	×	×
Viewer	√	×	×	×	×	×
Operation manager	√	√	√	√	√	√
Product manager	√	×	×	×	×	×
System engineer	√	×	×	×	×	×
Committer	√	√	√	√	√	√

**Table 2-3** Pre-release environment

Role	View	Create	Edit	Delete	Execute	Roll Back
Project creator	√	√	√	√	√	√
Project manager	√	√	√	√	√	√
Developer	√	×	×	×	×	×
Test manager	√	×	×	×	×	×
Tester	√	×	×	×	×	×
Participant	×	×	×	×	×	×
Viewer	×	×	×	×	×	×
Operation manager	√	√	√	√	√	√

Role	View	Create	Edit	Delete	Execute	Roll Back
Product manager	√	×	×	×	×	×
System engineer	√	×	×	×	×	×
Committer	√	√	√	√	√	√

Table 2-4 Production environment

Role	View	Create	Edit	Delete	Execute	Roll Back
Project creator	√	√	√	√	√	√
Project manager	√	√	√	√	√	√
Developer	×	×	×	×	×	×
Test manager	×	×	×	×	×	×
Tester	×	×	×	×	×	×
Participant	×	×	×	×	×	×
Viewer	×	×	×	×	×	×
Operation manager	√	√	√	√	√	√
Product manager	×	×	×	×	×	×
System engineer	√	×	×	×	×	×
Committer	√	√	√	√	√	√

## 2.4 Resource-level Permissions

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

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"/>
Product manager	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
System engineer	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Committer	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

## Configuring Role Permissions

- By default, role permissions in a pipeline are the same as the role permissions at the project level. If role permissions at the project level are changed, role permissions in a pipeline will be changed accordingly.
- The project creator, pipeline creator, and project manager can change pipeline role permissions.
- If role permissions in a pipeline are changed, role permissions at the project level will not be changed, because the resource-level permissions take precedence over the project-level permissions.

## Configuring User Permissions

- 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.
- If pipeline user permissions are changed, pipeline role permissions will not be changed, because user permissions take precedence over role permissions.

# 3 Rules and Policies

---

## 3.1 Overview

You can use pass conditions to control whether a pipeline can proceed to the next stage. Rules and policies can be used to associate extensions with pipelines.

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

You can apply policies to pipelines as pass conditions for efficient project management and high-quality delivery.

## 3.2 Configuring a Rule

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

### Rule Entry

1. On the service page, click the avatar icon in the upper right corner and choose **All Account Settings**.
2. On the left menu bar, choose **Policy Management > Rules**.

### Creating a Rule

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

Parameter	Description
Name	Rule name, which is generated based on the current time.

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	Thresholds are automatically configured based on the selected extension version. Based on the configured relationship, the extension output can be parsed into number or text. Rule configurations can be referenced by a policy and finally applied to a pipeline as pass conditions.

## Editing a Rule

On the rule management page, click  in the **Operation** column. 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 a Rule

On the rule management page, click  in the **Operation** column. On the displayed dialog box, confirm the deletion.

### 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 and can be configured in pass conditions for all pipelines under the current tenant.

### Policy Entry

1. On the service page, click the avatar icon in the upper right corner and choose **All Account Settings**.
2. On the left menu bar, choose **Policy Management > Policies**.

### NOTE

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

## Creating a Policy

Click **Create Policy**. Enter a policy name, select a rule, and click **Confirm**.


### NOTE

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

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

- **Enable/Disable:** You can click the switch in the upper right corner to enable/disable the rule. After the rule is disabled, it will not take effect in the pass conditions.
- **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.


## Editing a Policy

On the tenant-level policy management page, click  in the **Operation** column. 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 a Policy

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

### NOTE

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

## Deleting a Policy

On the tenant-level policy management page, click  in the **Operation** column and select **Delete**. On the displayed page, confirm the deletion.

### NOTE

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

## Disabling a Policy

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

### NOTE

If a policy is disabled, the system displays a message indicating the number of pipelines that use the policy. Once the policy is disabled, it will not take effect in the pass conditions.

## 3.4 Project-level Policies

### Policy Entry

Go to the pipeline list of a 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 a Policy

Click **Create Policy**. Enter a policy name, select a rule, and click **Confirm**.


#### NOTE

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

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

- **Enable/Disable:** You can click the switch in the upper right corner to enable/disable the rule. After the rule is disabled, it will not take effect in the pass conditions.
- **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 a Policy

On the project-level policy management page, click  in the **Operation** column. 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 a Policy

On the project-level policy management page, click  in the **Operation** column and select **Copy**.

#### NOTE

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

### Deleting a Policy

On the project-level policy management page, click  in the **Operation** column and select **Delete**. On the displayed page, confirm the deletion.

#### NOTE

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



## Disabling a Policy




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

### NOTE

If a policy is disabled, the system displays a message indicating the number of pipelines that use the policy. Once the policy is disabled, it will not take effect in the pass conditions.

## Tenant-level Policies

On the project-level policy management page, click **Tenant Policies** in the upper right corner. In the displayed dialog box, you can view, copy, or inherit a tenant-level policy.

- 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

Rules configured in the parent policy cannot be canceled or disabled in the inherited project-level policy.

## 3.5 Using a Policy in a Pipeline

Tenant-level and project-level policies can be configured in pass conditions to control pipeline stage execution.

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

1. Select a pipeline to edit it. Click **Pass Conditions** under a stage. On the displayed dialog box, move the cursor to the pass conditions card and click **Add**.
2. Select a policy for the pass conditions and click **OK**.
3. After the configuration, execute the pipeline. During pipeline execution, the configured policy can be called to control the stage execution.

# 4 Extensions

---

## 4.1 Overview

### Extensions

Larger enterprises tend to have their own pipeline tool chains. However, after cloud migration, assets such as self-built CI/CD tools and open-source tools cannot be inherited and end up wasted. CodeArts Pipeline solves this by allowing enterprises to quickly connect existing tools to the Pipeline service or develop their own extensions through the extension platform. CodeArts Pipeline provides a visualized, low-code, and open extension market to adapt to service requirements. CodeArts Pipeline has a collection of built-in extensions covering build, check, deployment, and test. You can use these extensions for pipeline orchestration.

Extensions can be used by tenants. An extension is a step in a pipeline job. It is the minimum running unit defined by JSON Schema and contains input parameters, service execution logic, and output content.

### Accessing 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 to create or edit a pipeline. When adding a job to a pipeline, click **More Extensions** in the upper right corner.

The extension page displays all available extensions. You can click the card of an extension to view details.

### Scenarios

- You can use official tools to develop extensions. CodeArts Pipeline allows you to compile service scripts in mainstream languages, such as Shell, Node.js, Python, and Java. For details, see [Official Tools](#). Some basic extensions can be used together with custom executors to provide more execution modes.
- You can use official extensions (such as **Kubernetes Release**, **CAE Release**) to connect to different cloud services.

- You can also customize extensions to connect to third-party CI/CD tools.

## 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. CodeArts Build also supports automated code retrieval, build, and packaging, as well as real-time status monitoring. <a href="#">Learn more.</a>
	Build-Template	This extension can be configured only in a pipeline template. When a pipeline is generated based on the template, the extension automatically creates a build job and configures the job in the generated pipeline.
Test	TestPlan	Calls CodeArts TestPlan capabilities. CodeArts TestPlan is a one-stop cloud testing platform provided for software developers. It covers test management and API tests and integrates the DevOps agile testing concepts, helping you improve management efficiency and deliver high-quality products. <a href="#">Learn more.</a>
	TestPlan-Template	This extension can be configured only in a pipeline template. When a pipeline is generated based on the template, the extension automatically creates an API test job and configures the job 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. You can also flexibly orchestrate atomic actions for deployment. CodeArts Deploy standardizes your deployment environment and processes by integrating with CodeArts Pipeline. <a href="#">Learn more.</a>
	Deploy-Template	This extension can be configured only in a pipeline template. When a pipeline is generated based on the template, the extension automatically creates a deployment job and configures the job in the generated pipeline.
	KubernetesRelease	Deploys container images to Huawei Cloud containers. It supports grayscale release based on ASM (Application Service Mesh) and containerized deployment based on CCE (Cloud Container Engine).

Type	Name	Description
	CAE Release	CAE (Cloud Application Engine) is a one-stop serverless application hosting service that enables ultra-fast deployment at low cost with simple O&M. It releases applications from source code, software packages, and image packages, with auto scaling in seconds, pay-per-use billing, and simple O&M infrastructure. The whole application lifecycle is manageable with observable metrics.
	CloudNativeRelease	Cloud native release allows you to orchestrate release policies for environments, such as rolling release and grayscale release.
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 bug fixing suggestions and trend analysis to control code quality and reduce costs. <a href="#">Learn more..</a>
	Check-Template	This extension can be configured only in a pipeline template. When a pipeline is generated based on the template, the extension automatically creates a code check job and configures the job in the generated pipeline.
	BranchCheck	Specifies the target branch. If the current running branch lags behind the specified branch, the pipeline fails to run.
Normal	CreateTag	Creates and pushes tags for code repositories.
	Subpipeline	Configures and calls other pipelines in a project.
	JenkinsTask	Calls Jenkins tasks. <b>NOTE</b> Currently, this function is available in LA-Mexico City2, LA-Sao Paulo1, and AP-Singapore.
	DelayedExecution	Pauses pipeline for a period of time or until a specified time. You can manually resume or stop a pipeline, or delay the execution 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. <b>NOTE</b> Currently, GitClone is available in LA-Mexico City2, LA-Sao Paulo1, AP-Singapore, and TR-Istanbul.

Type	Name	Description
	ExecuteShellCommand	Runs shell commands.
Microservice	CreateReleaseBranch	Creates a release branch based on the default branch of a microservice. This extension is automatically configured by a change-triggered pipeline.
	MergeReleaseBranch	Merges a feature branch into a release branch. This extension is automatically configured by a change-triggered pipeline.
	MergeDefaultBranch	Merges a release branch into the default branch of a microservice. This extension is automatically configured by a change-triggered pipeline.
Pass Conditions	Pass-Conditions-of-Standard-Policies	A standard extension policy for gate control.

## 4.3 Official Tools

Tool	Version	Default
JDK	8u372	-
	11.0.18	Yes
	17.0.2	-
nodejs	18.18.2	Yes
	20.10.0	-
go	1.18.7	Yes
	1.20.7	-
python	3.8.18	Yes
	3.12.1	-
maven	3.8.8	Yes


Built-in Tool	Description
git	2.27.0

jq	1.6
gcc	7.3.0
cmake	3.16.5
lz4	1.9.2
make	4.3
zip	3.0
unzip	6.0
curl	7.71.1
openssh	8.2
haveged	1.9.13
bzip2-devel	1.0.8
zlib	1.2.11
zlib-devel	1.2.11

## 4.4 Custom Extensions

### 4.4.1 Registering an Extension

#### Procedure

1. Log in to the CodeArts homepage.
2. Choose **Services > Extensions** from the top navigation bar.
3. Click .
4. Set basic information. For details, see [Table 4-1](#).

**Table 4-1** Parameter description

Parameter	Description
Icon	Icon of the extension. Upload an image in PNG, JPEG, or JPG format, with a file size no more than 512 KB (recommended: 128 x 128 pixels). If no image is uploaded, the system generates an icon.
Name	Name of the extension.
Unique Identifier	ID of the extension. Once set, this parameter cannot be changed.

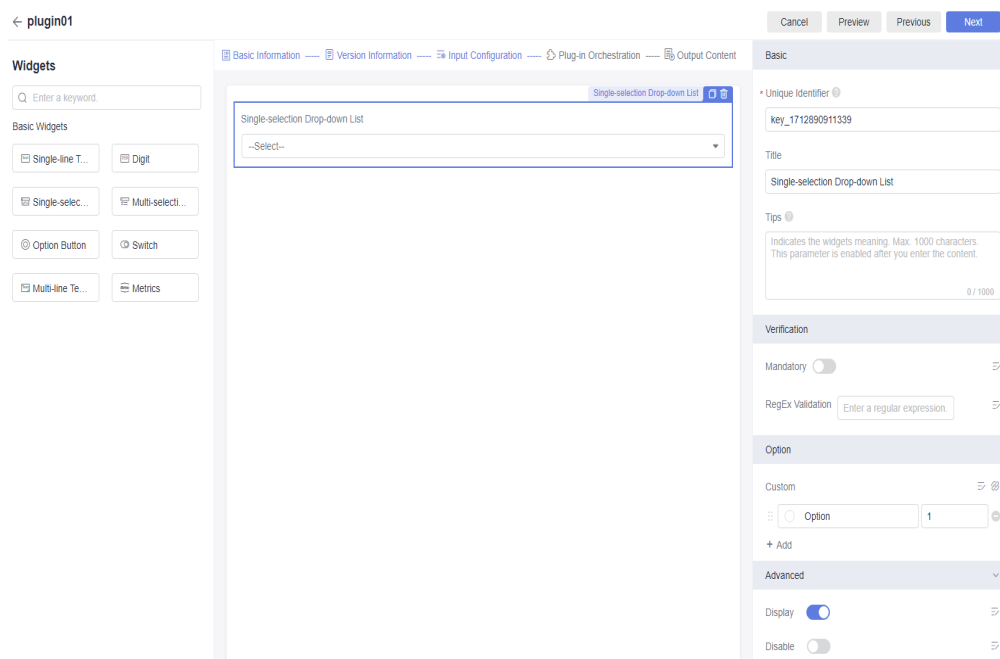
Type	Type of the extension, which can be <b>Build</b> , <b>Check</b> , <b>Test</b> , <b>Deploy</b> , or <b>Normal</b> . Once set, this parameter cannot be changed.
Description	Purposes and functions of the extension. The description can be edited.

- Click **Next**. On the **Version Information** page, set the version and description.

**NOTE**

- Version of the extension, in X.X.X format. Each digit ranges from 0 to 99.
  - Version information of the extension cannot be modified later.
- Click **Next**. The **Input Configuration** page is displayed.

You can drag and drop widgets to generate visual forms, which can streamline pipeline contexts. Multiple preset widgets are available: **Single-line Text Box**, **Digit**, **Single-selection Drop-down List**, **Multi-selection Drop-down List**, **Option Button**, **Switch**, **Multi-line Text Box**, **Metrics**, and so on.





Drag the widgets to the middle area. Click a widget and set the widget configurations on the right. For details, see [Table 4-2](#).

**Table 4-2** Parameter description

Category	Property	Description	Widget
Basic	Unique Identifier	Unique ID of the widget. The ID is used to obtain widget input.	All

	Title	Name of the widget. The name will be displayed on the pipeline job orchestration page.	All
	Tips	Description of the widget. The description will be displayed as preview.	Single-line Text Box, Digit, Single-selection Drop-down List, Multi-selection Drop-down List, Option Button, Switch, and Multi-line Text Box
	Accuracy	Number of decimal places allowed in a widget value: <b>0</b> to <b>4</b> .	Digit
	Default Value	Default value of the widget.	Single-line Text Box, Digit, Switch, Multi-line Text Box, and Metrics
Verification	Mandatory	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, and Multi-line Text Box
	Regex Validation	Verifies the widget content. You can set error messages.	Single-line Text Box, Digit, Single-selection Drop-down List, Multi-selection Drop-down List, and Multi-line Text Box
	Word limit	Max. widget characters.	Multi-line Text Box



Option	Custom	<p>Options supported by the widget:</p> <ul style="list-style-type: none"> <li>Label: content displayed on the extension configuration page.</li> <li>Value: value delivered when the extension is running.</li> </ul> <p>In addition to manual configuration, set the options by:</p> <ul style="list-style-type: none"> <li>APIs: Configure URLs to obtain option contents. Click  next to <b>Custom</b>. The API dialog box is displayed. You can configure parameters after enabling the function. For details, see <a href="#">Table 4-3</a>.</li> <li>Context: Configure data source to obtain the URL of the pipeline source or IDs of build jobs. Click  next to <b>Custom</b>. The context dialog box is displayed. You can configure parameters after enabling the function.</li> </ul>	Single-selection Drop-down List, Multi-selection Drop-down List, and Option Button
	Threshold	Configure the output metrics of the extension. After the configuration, choose <b>All Account Settings &gt; Policy Management &gt; Rules</b> to create a rule and apply the rule to the pipeline pass conditions.	Metrics
Advanced	Display	Whether the widget is visible. You can set conditions.	Single-line Text Box, Digit, Single-selection Drop-down List, Multi-selection Drop-down List, Option Button, Switch, and Multi-line Text Box
	Disable	Whether to disable the widget. Conditions can be set.	Single-line Text Box, Digit, Single-selection Drop-down List, Multi-selection Drop-down List, Option Button, Switch, and Multi-line Text Box

**Table 4-3** Parameter description

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

Linked Attribute	Associates the selected widgets with the API to transfer parameters. When a widget value is changed, the new value is used to call the API again.
URL	Only HTTPS protocol is supported.
Returned Data Path	The widget used must be list data. In the following response body example, the returned data path is <b>result.parameters</b> . <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	Enable this function to add a remote search field. For extension search, the entered value will be used as the value of the remote search field to call the API again.

You can also use the **Metrics** widget to configure the extension output:

- a. 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.
- b. Configure **Threshold** for the **Metrics** widget. The threshold information can be referenced in rules and policies and finally applied to pipelines.

**Threshold** ✕

Group:  🗑️

Key	Description <span style="font-size: 0.8em;">?</span>	Value	🗑️
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

+ Add

+ Add

OK
Cancel

The parameters are described in the following table.

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> file of the extension code logic. After the pipeline is executed, this value is used for match in the file.
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.

- Click **Next**. On the displayed page, add the **ExecuteShellCommand** extension and enter the shell command.

 **NOTE**

The commands indicate the actual service logic implementation process of an extension. For more input and output configurations, see [Developing Extensions Using Low Code](#).

- Click **Next**. On the displayed page, click **Add Configuration** to add an output. You can use the shell command in the previous step to output data.

Key	Type	Description	Source	Attribute	Operation
mykey	output	--	Custom	--	 


- Click **Release** or **Release Draft** to complete the registration.

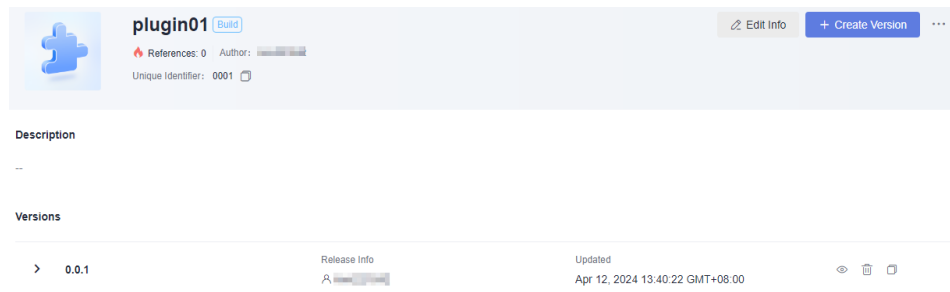
There are draft release and official release.

- Official release

An official extension has a unique version number. All members of the current tenant can use this version in a pipeline.

- Draft release

- You can configure a draft extension in a pipeline for debugging. After debugging, the draft extension can be officially released, so that other members of the current tenant can use the extension.
  - All draft versions are marked with **Draft**.
  - 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.
10. (Optional) Update the extension.
    - a. After the extension is registered, click its card to view details.
    - b. Click **Create Version**. Or, clone an existing extension version and click  to view the historical configurations. You can modify the extension based on the cloned version.

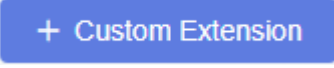


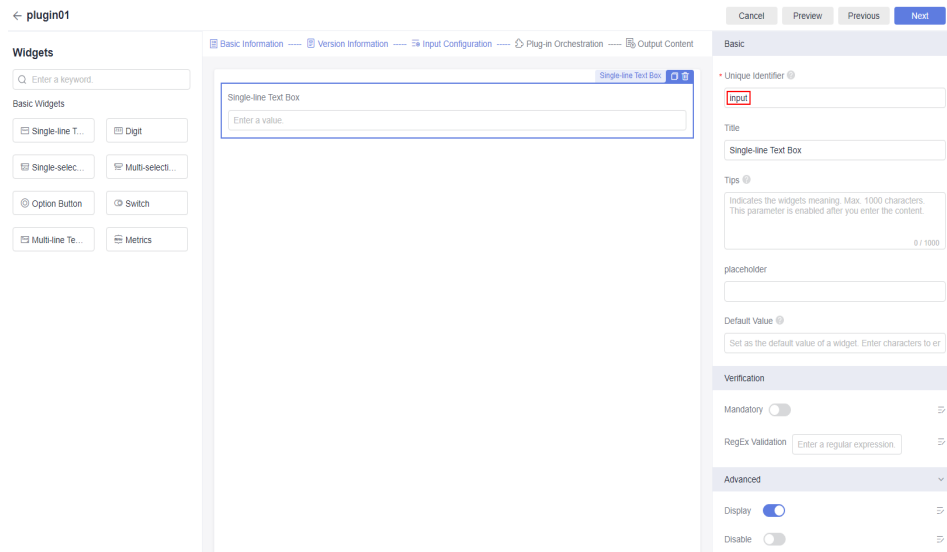
## 4.4.2 Using a Custom Extension

This section describes how to develop a log printing extension.

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

### Step 1: Register a Custom Extension

1. Log in to the CodeArts homepage.
2. Choose **Services > Extensions** from the top navigation bar.
3. Click .
4. Configure the extension information by referring to [Registering an Extension](#).
  - a. The following uses the **Single-line Text Box** widget as an example and changes its unique ID to **input**.



- b. In the extension orchestration step, use the log printing as an example to output the widget input configured on the low-code user interface and the system parameters for pipeline run. Enter the following commands in the shell command box:

```
echo ===== begin =====  
# Enter the service logic of the extension.  
# Use environment variables to obtain the widget input on the low-code user interface.  
echo ${input}  
# Use environment variables to obtain system parameters for pipeline run.  
echo ${PIPELINE_ID}  
echo ===== end =====
```

5. Release the extension.

After the configuration, click **Release** or **Release Draft**. Drafts can be deleted.

## Step 2: Execute the Custom Extension

1. Configure the custom extension for a pipeline.

Create or edit a pipeline, add the released custom extension, and enter **Low-code input** in the single-line text box.

2. Execute the pipeline.

After the execution is complete, click the extension name to view the extension log, which contains **Low-code input** and pipeline system variables.

### 4.4.3 Developing Extensions Using Low Code

When registering an extension or creating an extension version, you can use shell commands to implement service logic. The commands usually involve interaction with all kinds of data during pipeline execution. This section describes how to develop extension code through data input and output.

#### Data Input

The obtained data consists of low-code GUI input, pipeline run parameters, and other information.

- Low-code GUI input: Obtain the low-code user interface output by using environment variables, for example, **echo \${Widget ID}**.

- Pipeline run parameters: Some pipeline run parameters will be delivered to environment variables as shown in the following table.

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	Job ID of the pipeline.

- Other information: Obtain information by interacting with external data through Git, Wget, and Curl.

## Data Output

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

1. On the configuration page, 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 stored in a specified path so that metric values can be parsed.

**Table 4-4** Output files

File	Description
<code>\${RESULT_MSG_PATH}/\${STEP_ID}_result.json</code>	The output is a text file in <code>{"par1":123, "par2":456}</code> format. After the pipeline is executed, the result will be displayed as the corresponding task result. <b>NOTE</b> Only extensions of the check type can display the result.

File	Description
<code>\${RESULT_MSG_PATH}/\${STEP_ID}_metrics.json</code>	<p>The output is a text file in <code>{"par1":123, "par2":456}</code> format. The <b>Metrics</b> widget should be configured. After the extension is executed, the threshold configured for the <b>Metrics</b> widget and the content of <code>_\${STEP_ID}_metrics.json</code> are parsed for pipeline pass conditions. Note:</p> <ul style="list-style-type: none"><li>• During parsing, empty key values in the <b>Metrics</b> widget will be ignored.</li><li>• If the key value configured for the <b>Metrics</b> widget cannot be found in the <code>_\${STEP_ID}_result.json</code> file, the specified threshold value will be used.</li></ul>

Example: **par1** and **par2** for pass conditions; **par3** and **par4** for task result display. The sample code is as follows:

```
# Optionally, construct the extension output.  
echo '{"par1":100,"par2":200}' > ${RESULT_MSG_PATH}/${STEP_ID}_result.json  
echo '{"par3":300,"par4":400}' > ${RESULT_MSG_PATH}/${STEP_ID}_metrics.json
```

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

Red Line	
Check Items	Result
par3	300
par4	400

If policies are configured for the current extension and applied to the pipeline pass conditions, click the pass conditions to view the check status.

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

## 4.5 Basic Extensions

### 4.5.1 Registering a Basic Extension

**Step 1** Log in to the CodeArts homepage.

**Step 2** Choose **Services > Extensions** from the top navigation bar.

**Step 3** Click .

**Step 4** Set basic information. For details, see [Table 4-5](#).


**Table 4-5** Parameter description

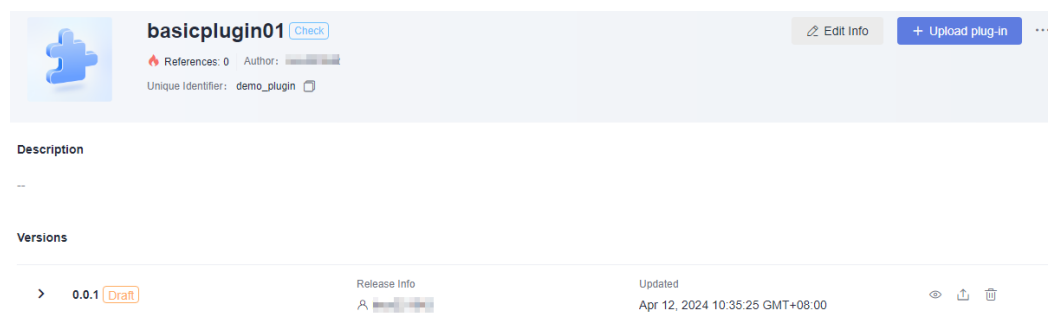
Parameter	Description
Icon	Icon of the extension. Upload an image in PNG, JPEG, or JPG format, with a file size no more than 512 KB (recommended: 128 x 128 pixels). If no image is uploaded, the system generates an icon.
Name	Name of the extension.
Unique Identifier	ID of the extension. Once set, this parameter cannot be changed.
Type	Type of the extension, which can be <b>Build</b> , <b>Check</b> , <b>Test</b> , <b>Deploy</b> , or <b>Normal</b> . Once set, this parameter cannot be changed.
Description	Purposes and functions of the extension. The description can be edited.

 **NOTE**

Some information must be the same as that of the extension uploaded in step [Step 6](#). For details, see [Extension Properties](#).

**Step 5** Click **OK**.

**Step 6** On the displayed page, click . In the displayed dialog box, select the desired extension (with input definition and execution script) and upload it. After the upload is successful, the version will be marked with **Draft**.




**Step 7** Debug the extension.

Create a pipeline. On the **Task Orchestration** page, create a job, add the registered basic extension, and set parameters.



**Step 8** Save and execute the pipeline. After the execution is complete, click the extension name to view the execution result.

**Step 9** (Optional) After debugging, publish the extension as an official version.

1. Go to the extension page.
2. Click the registered basic extension.
3. On the displayed page, click  on the right to publish the version as an official version.

The draft version can be overwritten for multiple times. However, the official version cannot be updated. You can click **Upload plug-in** in the upper right corner to upload a new version. As shown in the following figure, version **0.0.1** is an official version, and version **0.0.2** is an uploaded draft version.

----End

## 4.5.2 Extension Properties

The following uses an example to describe the properties of an extension.

### Extension Package

File structure:

```
extension.zip          # ZIP package of the extension
|-- scripts           # (Optional) Script folder for storing scripts that contain extension execution
logic.
| |-- xxx             # Script that contains extension execution logic
|-- i18n              # (Optional) Contents in multiple languages
| |-- zh-cn          # Contents in Chinese environment
|   |-- resources.json # Internationalization resources
|   |-- en-us        # Contents in English environment
|   |-- resources.json # Internationalization resources
|-- codearts-extension.json # (Mandatory) Extension execution file (in JSON format), including basic
information, inputs, and execution
```

Notes:

- The extension package must be in the ZIP format.
- The root directory of the package must contain a metadata file **codearts-extension.json**.
- During extension registration, set the **Unique Identifier** and **Type** fields to the values of **name** and **category** in the **codearts-extension.json** file, respectively.
- The **resources.json** file can be encoded only using UTF-8.

The mapping between the extension type and the value of **category** is listed in the following table.

Type	category
Build	Build
Check	Gate

Type	category
Deploy	Deploy
Test	Test
Normal	Normal

## codearts-extension.json

The following is an example of the file **codearts-extension.json**.

```
{
  "type": "Task",
  "name": "demo_plugin",
  "friendlyName": "Extension name",
  "description": "This is an extension.",
  "category": "Gate",
  "version": "0.0.2",
  "versionDescription": "Updated based on the initial version 0.0.1",
  "dataSourceBindings": [],
  "inputs": [
    {
      "label": "URL for calling the service",
      "name": "SERVICE_URL",
      "type": "input",
      "description": "Enter a URL.",
      "defaultValue": "",
      "required": true
    },
    {
      "label": "Token for calling the service",
      "name": "SERVICE_TOKEN",
      "type": "input",
      "description": "Enter a token.",
      "defaultValue": "",
      "required": true
    }
  ],
  "execution": {
    "type": "Shell",
    "target": "scripts/execution.sh"
  },
  "outputs": []
}
```

The parameters are described in the following table.

Parameter	Description
type	The value is fixed to <b>Task</b> , which indicates an extension type.
name	Same as the <b>Unique Identifier</b> field set for extension registration
friendlyName	Same as the <b>Name</b> field set for extension registration

Parameter	Description
category	Same as the <b>Type</b> field set for extension registration, which can be: <ul style="list-style-type: none"><li>• Build: corresponds to the extension of the <b>Build</b> type.</li><li>• Test: corresponds to the extension of the <b>Test</b> type.</li><li>• Gate: corresponds to the extension of the <b>Check</b> type.</li><li>• Normal: corresponds to the extension of the <b>Normal</b> type.</li><li>• Deploy: corresponds to the extension of the <b>Deploy</b> type.</li></ul>
version	Version of the extension, which consists of three numbers separated by dots (.), with each number ranges from 0 to 99. Modify this parameter only when you need to add an official version.
description	Description of the extension.
versionDescription	Description of the extension version's unique features.
dataSourceBindings	Disabled currently. Set it to [].
inputs	Extension input content. This parameter corresponds to the extension display format on the pipeline page. The values can be referenced by environment variables in service scripts.
execution	Extension execution content. The <b>type</b> field indicates the service script language, and the <b>target</b> field indicates the path to the execution file. You are advised to create a <b>scripts</b> folder and place the content under it.
outputs	Extension output content. The value can be used as the gate metrics. <b>output</b> has different display.

## inputs

The following is an example of inputs.

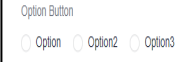
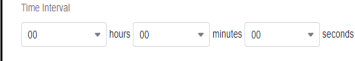
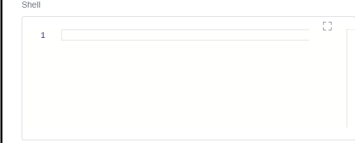

```
{
  "name": "samplestring",           # Use samplestring in a script to obtain the value
  "type": "input",                 # Different types correspond to different functions
  "description": "Sample String",  # Description of input
  "defaultValue": "00",           # Default value when the value of the required field is false
  "required": true,                # Reset defaultValue if the required field is true, or the default
  "label": "Text box",            # input information displayed on the pipeline editing page
  "validation": {
    "requiredMessage": "Enter a value.", # (Optional) The message displayed when the
    "regex": "^[a-zA-Z0-9-_\u0000-\u00ff]{1,32}$", # (Optional) RegEx validation
    "regexMessage": "Type error"       # (Optional) The message displayed when RegEx
  }
}
```

For details about all inputs, see [All inputs](#).


## 4.5.3 All inputs

### Overview

Type	Widget	Style	extendProp
input	Single-line Text Box	 A single-line text input field with the placeholder text "Enter a value."	<ul style="list-style-type: none"> <li>• visible Conditions</li> <li>• disabled Conditions</li> </ul>
inputNumber	Digit	 A digit input field with the placeholder text "Enter a value." and a numeric keypad icon.	<ul style="list-style-type: none"> <li>• visible Conditions</li> <li>• disabled Conditions</li> </ul>
switch	Switch	 A toggle switch labeled "Switch" with a slider.	<ul style="list-style-type: none"> <li>• visible Conditions</li> <li>• disabled Conditions</li> </ul>
singleSelect	Single-selection Drop-down List	 A single-selection drop-down list with the placeholder text "--Select--".	<ul style="list-style-type: none"> <li>• options</li> <li>• apiType</li> <li>• apiOptions</li> </ul>
multipleSelect	Multi-selection Drop-down List	 A multi-selection drop-down list with the placeholder text "Option" and "Option2".	<ul style="list-style-type: none"> <li>• options</li> <li>• apiType</li> <li>• apiOptions</li> </ul>
keyValuePair	Key-Value Pair	 A key-value pair input field with two text boxes labeled "Enter a value." and a "+ Add" button.	<ul style="list-style-type: none"> <li>• visible Conditions</li> <li>• disabled Conditions</li> </ul>

Type	Widget	Style	extendProp
radio	Option Button		options
timeInterval	Time Interval		<ul style="list-style-type: none"> <li>• visible Conditions</li> <li>• disabled Conditions</li> </ul>
shell	Shell		<ul style="list-style-type: none"> <li>• visible Conditions</li> <li>• disabled Conditions</li> </ul>
endpoint: <i>`\${module_id}`</i>	Endpoint		<ul style="list-style-type: none"> <li>• visible Conditions</li> <li>• disabled Conditions</li> </ul>

### Basic Fields of a Widget

Field	Description	Mandatory	Remarks
name	Unique identifier of a widget	Yes	The value must be unique.
label	Title of a widget	Yes	-
type	Type of a widget	Yes	-
defaultValue	Initial value	No	Initial default value of a widget. This field can be left blank.
description	Description of a widget	No	The info tip message next to a widget name 



Field	Description	Mandatory	Remarks
disabledConditions	Widgets are disabled if conditions are met.	No	Multiple conditions can be included: [{};{};{};...] Example: [{}comp:'key_002',symbol:'!==' , value: 'yyy'] In this example, widget A will be disabled if widget B has a unique ID of <i>key_002</i> and has a value that is not equal to (!==) <i>yyy</i> . <b>symbol</b> can be: <ul style="list-style-type: none"><li>• ===: Equal</li><li>• !==: Not equal</li><li>• empty: Empty</li><li>• notEmpty: Not empty</li></ul>
options	The fixed drop-down list. The field's type is <b>list</b> .	No	Example: [{}label:'option 1',value: 1},{label:'option 2',value: 2}]
apiType	Options in the drop-down list box: <ul style="list-style-type: none"><li>• <b>fixed</b>: The values in <b>options</b> are used as options.</li><li>• <b>api</b>: API requests, available only when <b>apiOptions</b> is configured.</li></ul>	No	If this field is left blank, <b>fixed</b> is used.

Field	Description	Mandatory	Remarks
apiOptions	JSON body, including parameters used by APIs.	No	<p>Example:</p> <pre>{ "body": {"xxx": 111}, "header": {"yyy": 222}, "linkedFields": ["key_001"], "method": "POST", "params": {"zzz": 333}, "remote": true, "remoteName": "xxx", "remoteQueryField": "body", "responseUrl": "data", "label": "name", "value": "id", "url": "https://sss/lll/mmm" }</pre> <p>JSON (parsed):</p> <pre>{  body: {xxx:111},           // Request parameters for calling an API  header: {yyy: 222},       // Request header field  params: {zzz: 333},       // Request parameters for calling an API  linkedFields: ['key_001], // This field is associated with other widgets. When the values of other widgets change, the API will be called again and the current options will be cleared.  method: 'POST',          // Request mode: POST or GET  remote: true,            // Whether to enable remote search  remoteName: 'tt',        // Field to be searched in a remote search  remoteQueryField: 'body', // Parameter passing method for the remote search field: body or params  responseUrl: 'data',     // The path of the option list obtained in the returned data  label: 'name',           // Parameter corresponding to the label in ComboBox  value: 'id',             // Parameter corresponding to the actual value in ComboBox  url: 'https://sss/lll/mmm' // API URL }</pre>

## 4.6 Extensions and Policies

Some extensions can be used for policies, which will be applied to pass conditions.

You can configure rules for the following extensions:

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



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

- **Build:** calls CodeArts Build capabilities and returns the case pass rate, total cases, and branch coverage of Maven unit tests.

Maven-Unit-Testing

Check Item	Relationship	Default Value	Enabled
PassRatio	=	Text 0	<input checked="" type="checkbox"/>
TotalCases	=	Text 0	<input checked="" type="checkbox"/>
BranchesCoverage	=	Text 0	<input checked="" type="checkbox"/>

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

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

 **NOTE**

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

## 4.7 KubernetesRelease Extension

### 4.7.1 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](#)

## 4.7.2 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...**

Add extension

 **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


\*Service Name  
blue-green-test


\*VirtualService  
blue-green-test-40001

\*DestinationRule  
blue-green-test

\*Image URL  
[https://registry.cn-hongkong.huaweicloud.com/v1/repositories/...](#)

Grayscale Version

\*Grayscale Workload Upgrading Timeout (minutes)   
30

\*Route Traffic Waiting Time (minutes)   
1

**Table 4-6** Parameter description

Parameter	Description
Name	Enter the name of the upgrade. The default name is <b>KubernetesRelease</b> .
Region	Select a region for deployment.
CCE Cluster	Select the Kubernetes cluster applied in 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 <b>Rolling</b> and <b>Blue-Green</b> (we use <b>Blue-Green</b> as an example). <ul style="list-style-type: none"><li>• <b>Rolling</b>: rolling release for deployment</li><li>• <b>Blue-Green</b>: ASM-based blue-green release</li></ul>
Service Name	The service to be upgraded.
VirtualService	Select the target VirtualService. Log in to the ASM console, choose <b>Mesh Configuration &gt; Istio Resource Management</b> and filter the target namespace and Istio resources.
DestinationRule	Select the target DestinationRule. Log in to the ASM console, choose <b>Mesh Configuration &gt; Istio Resource Management</b> 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, <b>swr.cn-north-1.myhuaweicloud.com/demo/springboot-helloworld:v1.1</b> . You can also use <b>\${}</b> to reference pipeline parameters, for example, <b>swr.cn-north-1.myhuaweicloud.com/demo/springboot-helloworld:\${version}</b> .
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.

**KubernetesRelease** Task Log Task Result

**KubernetesRelease**

**Release Sheet** Executing [Release Status](#) [Rollback](#) [Released](#)

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	...	Route Traffic Waiting Time (minutes)	1

**Release Detail** Release Log

**v1**  
Number of copies: 2

**v5767580** Online  
Number of copies: 2

Name	Status	Pod IP	Create Time
blue-green-test-7d6bbc...	Running	1...	2023-02-15 16:37:58 GMT+08:00

**Basic Information**

Instance Name	blue-green-test-7d6bbc557...	Pod IP	1...
Status	Running	Host IP	1...
Create Time	2023-02-15 16:37:58 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 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.

### 4.7.3 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](#).

The screenshot displays the configuration interface for the 'KubernetesRelease' extension. On the left, there is a sidebar with a 'KubernetesRele...' button and an 'Add extension' button. The main area contains the following fields and options:

- Name:** A text input field containing 'KubernetesRelease'.
- Region:** A dropdown menu with 'Sao Paulo1' selected.
- CCE Cluster:** A dropdown menu with 'release-test' selected.
- Namespace:** A dropdown menu with 'default' selected.
- Use IAM to improve permissions:** A toggle switch that is currently turned off.
- Deployment Strategy:** Radio buttons for 'Rolling Update' (selected) and 'Blue/Green Deployment'.
- Deploy Mode:** Radio buttons for 'Image Upgrade' (selected) and 'YAML Deployment'.
- Workload:** A dropdown menu with 'zfc-roll-1' selected.
- Container:** A dropdown menu with 'container-1' selected.
- Image:** A text input field containing a long alphanumeric string.

**Table 4-7** Parameter description

Parameter	Description
Name	Enter the name of the upgrade. The default name is <b>KubernetesRelease</b> .
Region	Select a region for deployment.
CCE Cluster	Select the Kubernetes cluster applied in 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 <b>Rolling</b> and <b>Blue-Green</b> (we use <b>Rolling</b> as an example). <ul style="list-style-type: none"><li>● <b>Rolling:</b> rolling release for deployment</li><li>● <b>Blue-Green:</b> ASM-based blue-green release</li></ul>

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

## 5.1 Accessing Microservices

### Procedure

1. Log in to the CodeArts homepage.
2. Go to the target project and choose **CICD > Pipeline**.
3. Click **Microservices**. The microservice list with microservice information is displayed.

Parameter	Description
Microservice	Microservice name.
Creator	Name of the user who creates the microservice.
Created	Time when a microservice is created. You can move the cursor to the <b>Created</b> column and click  to sort microservices by creation time.
Status	Status of a microservice. After a microservice is created, it is in an activated status.
Operation	Click  to follow a 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. <b>NOTE</b> After a microservice is deleted, all changes and pipelines in the microservice will be deleted.

- 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 the 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 displayed page, set parameters. For details, see [Table 5-1](#).

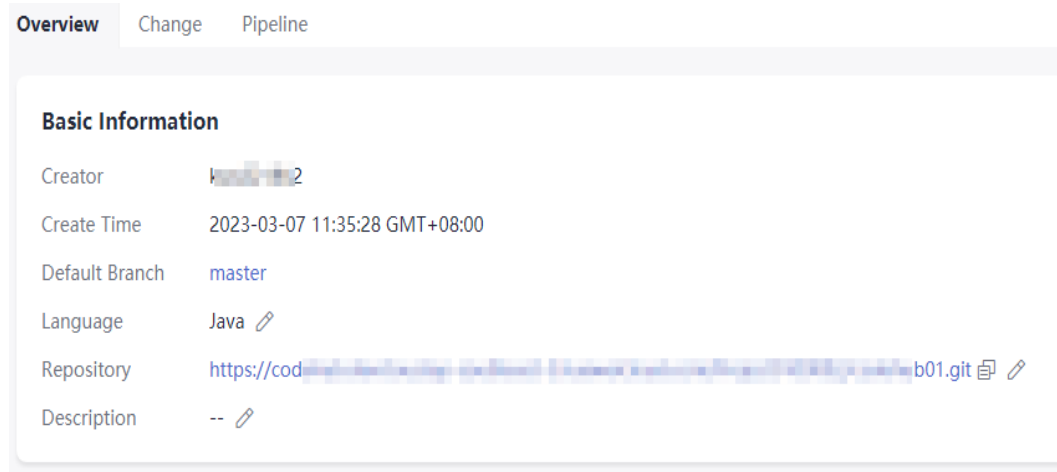
**Table 5-1** Parameter description

Parameter	Description
Project	Project to which the microservice belongs. The project cannot be changed.
Microservice name	Microservice name.
Pipeline Source	Source code repository. Only <b>Repo</b> is supported.
Repository	Created code repository.
Default Branch	Branch used when a pipeline is executed manually or at a specified time. <b>NOTE</b> After the pipeline is triggered by changes, all changed feature branches will be merged into the default branch.
Language	Development language for the microservice.
Description	Microservice description.

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

## 5.3 Viewing a Microservice

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



- **Overview**

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

When you change the code repository, if there are unclosed changes or running pipelines in the microservice, the **Data Processing** dialog box is displayed. In that case, close all changes and stop all running pipelines.

- **Changes**

You can manage changes in the microservice. For details, see [Changes](#).

- **Pipelines**

You can manage pipelines in the microservice. Pipelines in a microservice have the following features:

- 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-triggered pipeline to release changes. For details, see [Managing Change-triggered Pipelines](#).

# 6 Changes

## 6.1 Accessing Changes

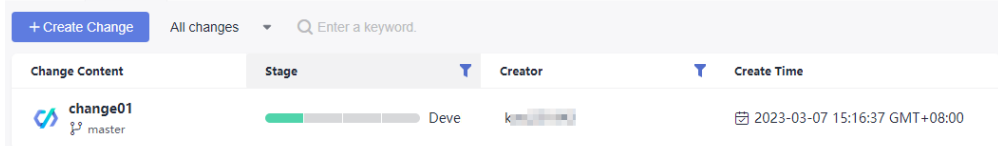
### Procedure

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

Homepage / Project01 / Microservice / microservice01 / Change

← microservice01 ☆

Overview **Change** Pipeline



Parameter	Description
Change	The name and associated branch of the change.
Stage	Status of the change. You can move the cursor to the <b>Stage</b> column and filter changes based on the stage.
Creator	Name of the user who creates the change.
Created	Creation time of the change.
My changes/All changes	Select <b>My changes</b> or <b>All changes</b> to filter changes. <b>My changes</b> displays changes created by the current user. <b>All changes</b> displays all changes in the microservice.

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

## 6.2 Creating a Change

### 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

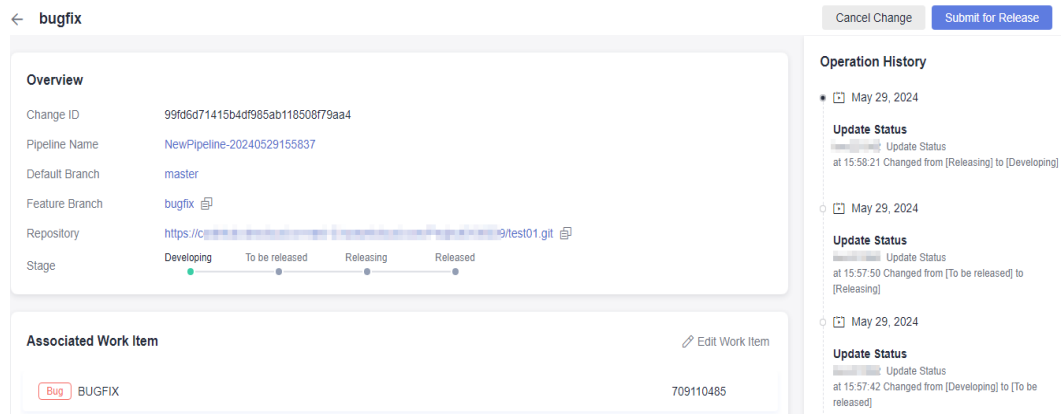
Parameter	Description
Change Subject	Enter a change name.
Repository	Name of the repository associated with the microservice, which cannot be changed.
Branch	Pull a new branch from the default branch or associate with an existing branch.
Associated Work Item	Select started or ongoing work items in CodeArts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

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

## 6.3 Viewing a Change

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

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



The following describes how to submit a change for release, exit 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-triggered pipeline exists in the microservice, create one as prompted. For details, see [Managing Change-triggered Pipelines](#).
- If there is a change-triggered pipeline in the microservice, click **OK** to submit the change.

After the change is submitted, 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 exits release, and the change status changes to **Developing**.

**NOTE**

For a change in the **Releasing** stage, if the change-triggered pipeline is running, you can exit the release only after the pipeline run 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 Managing Change-triggered Pipelines

In microservices, you can create change-triggered pipelines to associate them with change resources. A change-triggered pipeline has the following features:

- A microservice can have only one change-triggered pipeline.
- An integration branch is automatically created during the execution of the change-triggered pipeline. After successful execution, the branch content is merged to the master branch.



- After successful execution, the change status is automatically updated.
- Only one pipeline instance can run at one time.
- The change-triggered pipeline cannot be triggered by an event or at a specified time.

## Creating a Change-triggered Pipeline

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

**Table 6-2** Parameter description

Parameter	Description
Project	Project to which the microservice belongs. The project 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. The repository cannot be changed.
Default Branch	The default branch associated with the microservice. The default branch cannot be changed.
Repo HTTPS Authorization	Configure endpoints to elevate permissions on repository operation. Endpoints are used for change-triggered pipelines and repository operation extensions.
Alias	If an alias is set, the system parameters are generated for the repository (you can view the parameters on the parameter configuration page).
Change-based Trigger	If <b>Change-based Trigger</b> is enabled for a pipeline, this pipeline is marked with <b>Based on Changes</b> . <b>NOTE</b> A microservice can have only one change-triggered pipeline.
Description	Description of the pipeline.

5. After setting all parameters, click **Next**. On the displayed page, select a template or select **Blank Template**.
6. Click **OK**, and then click **Save**.

## Executing a Change-triggered Pipeline

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

**Execution Configuration** ×

Changes (Selected: 0. Max: 10)

Q Enter the change subject.

Change	Created By
<input type="checkbox"/> bugfix 🔗 bugfix	

▶ Runtime Parameters

▼ Execution Stages

All Stages

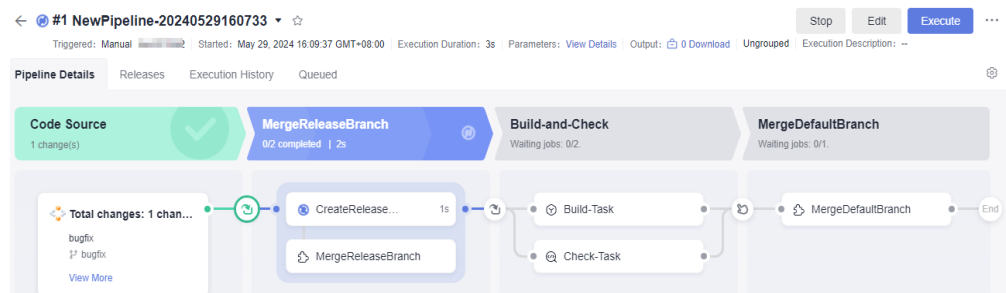
- Build-and-Check
  - Build-Task
  - Check-Task

Description

Enter the execution description.

0 / 1024

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




When the change-triggered pipeline is running, the **MergeReleaseBranch** and **MergeDefaultBranch** stages are added by default.

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

## Viewing the Result

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

After the pipeline is successfully executed, the status of all selected changes is changed to **Released**.

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

### NOTE


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

# 7 Release Environments

## 7.1 Creating a Release Environment

### Environment Entry

1. Log in to CodeArts.
2. Click a project name to access the project.
3. Choose **CICD > Release** to access the environment list page.

Parameter	Description
Environment Name	Unique environment identifier in the project.
Resource Type	Type of resources needed by services
Environment Level	Available environment types: development, test, pre-release, and production.
Creator	Creator of the environment.
Created	Time when the environment was created.
Operation	Click  in the <b>Operation</b> column. In the displayed dialog box, enter the environment name and click <b>Delete</b> .

- Click **Create Environment** to [create an environment](#).
- Click an environment name to [view the details](#).

### Creating a Release Environment

1. On the environment list page, click **Create Environment**.
2. On the displayed page, enter basic information. For details, see [Table 7-1](#).

**Table 7-1** Parameter description

Parameter	Description
Project	Project to which the environment belongs. The project cannot be changed.
Environment Name	Unique environment identifier within the microservice. The name cannot be changed once the environment was created.
Resource Type	<b>CCE, UCS, and K8s</b> are available. They support different deployment extensions. <ul style="list-style-type: none"><li>• CCE: a type of Kubernetes cluster encapsulated by Huawei Cloud. Select this type if you want to use Huawei Cloud resources. <a href="#">Learn more</a>.</li><li>• UCS: a type of Kubernetes cluster encapsulated by Huawei Cloud for multi-cloud deployment. Select this type if you want to deploy clusters on multiple clouds. <a href="#">Learn more</a>.</li><li>• K8s: the native Kubernetes cluster. Select this type if you want to use self-built clusters or third-party clusters.</li></ul>
Publish User	Options: <b>Current User</b> or <b>Other Users</b> . You need to use the service endpoint to obtain permissions of <b>Other Users</b> .
Region	This parameter is required when you select <b>CCE</b> for <b>Resource Type</b> . Select a region for deployment.
Cluster	This parameter is required when you select <b>CCE</b> for <b>Resource Type</b> . Select the <a href="#">purchased Kubernetes clusters</a> in Cloud Container Engine (CCE).
Association Type	This parameter is required when you select <b>UCS</b> for <b>Resource Type</b> . Select the associated UCS resource.
Fleets	This parameter is required when you select <b>UCS</b> for <b>Resource Type</b> . <a href="#">Select a fleet</a> .
Kubernetes Endpoint	This parameter is required when you select <b>K8s</b> for <b>Resource Type</b> . Select a <a href="#">created Kubernetes endpoint</a> to access cluster resources with credential.
Environment Level	Available environment types: development, test, pre-release, and production.
Description	Enter the description of the environment.

3. After setting all parameters, click **OK**. The environment information page is displayed.

## Viewing a Release Environment

1. On the environment list page, click an environment name.
2. On the displayed page, you can check the basic information of the environment, as shown in the following table (using CCE as an example).

Parameter	Description
Resource Type	Resource types associated with the environment.
Service Endpoint	Service endpoint of CCE resources.
Cluster Region	Kubernetes cluster region applied in CCE.
Cluster ID	Kubernetes cluster ID applied in CCE.
Variable Version	Version number of the environment variable in the current environment.

3. Switch tabs to view [environment variables](#), [release policies](#), and [deployment history](#).

## 7.2 Configuring an Environment Variable


### Background

You can use `${variable name}` to reference an environment variable when creating or editing a release policy, or use `{{variable name}}` to reference an environment variable in YAML files. Environment variables include:

- Custom variables: can be added as needed. Currently, only variables of the string type are supported.
- Default variables: system parameters, which cannot be deleted or modified. Default variables include **ARTIFACT**, **TIMESTAMP**, and **PROJECT\_ID**.

### Configuring a Variable

1. Log in to CodeArts.
2. Click a project name to access the project.
3. Choose **CICD** > **Release** to access the environment list page.
4. Click an environment name, the **Environment Information** tab is displayed. Switch to the **Environment Variable** tab.
5. Click **Edit Variable** to add a variable and set parameters.


Parameter	Description
Variable	Variable name, which can be customized.
Type	Only the string type is supported.
Value	The current value of a variable, or empty if you are adding a new variable.
Change Value	Set a new variable value here if you are editing a variable.
Description	The description of a variable.
Private Variable	If a parameter is private, the system encrypts the parameter for storage and decrypts the parameter for usage. Private parameters will not be displayed in run logs.
Operation	Click  in the <b>Operation</b> column to delete a variable.

You can also add, modify, and delete variables as required.

6. After setting all parameters, click **Save**.
7. Confirm the information on the dialog box that is displayed, enter the remarks, and click **OK**.

## Historical Version

On the **Environment Variable** page, you can click **Versions** to view variable versions.

- Click a version name to view the variable details.
- Click  in the **Operation** column to compare the current version with a specified version.

## Using a Variable

You can use environment variables in the following scenarios:

- When creating or editing a release policy, you can use `${variable name}` to reference an environment variable in the YAML path, for example, the workload YAML path in the rolling upgrade task.

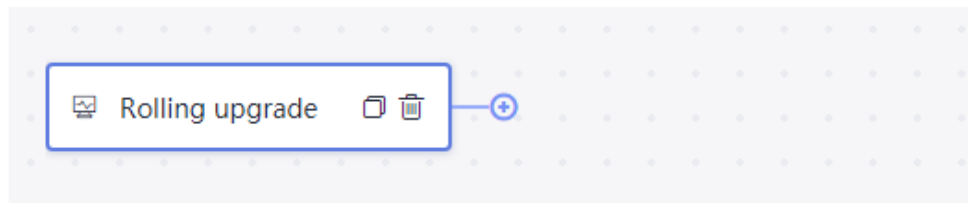
\*Policy

RollingUpgrade


Description

Releases deployments

**Task Orchestration**



**Rolling upgrade**

\*YAML path of workload 

`${deployment.yaml}`

- Use `{{variable name}}` to reference an environment variable in the YAML configuration file associated with the release policy.



```
1  apiVersion: apps/v1
2  kind: Deployment
3  metadata:
4    name: rolling-test
5    labels:
6      run: rolling-test
7    namespace: default
8  spec:
9    replicas: 1
10   selector:
11     matchLabels:
12       run: rolling-test
13   template:
14     metadata:
15       labels:
16         run: rolling-test
17     spec:
18       containers:
19         - name: main
20           image: {{ARTIFACT}}
21           ports:
22             - containerPort: 8080
23           env:
24             - name: TIMESTAMP
25               value: {{TIMESTAMP}}
26             - name: PROJECT_ID
27               value: {{PROJECT_ID}}
28             - name: COMPONENT_ID
29               value: {{COMPONENT_ID}}
30       resources:
31         limits:
32           cpu: 250m
33           memory: 512Mi
34         requests:
35           cpu: 250m
36           memory: 512Mi
```

## 7.3 Configuring an Environment Release Policy

### Creating a Policy

You can add atomic extensions and edit release policies based on the preset **RollingUpgrade** or **GrayscaleUpgrade** templates.

- Step 1** Log in to CodeArts.
- Step 2** Click a project name to access the project.
- Step 3** Choose **CICD > Release** to access the environment list page.
- Step 4** Click an environment name.
- Step 5** On the displayed page, click the **Release Policy** tab.
- Step 6** Click **+** next to **Custom Policies**, on the displayed dialog box, select a policy template and click **OK**.

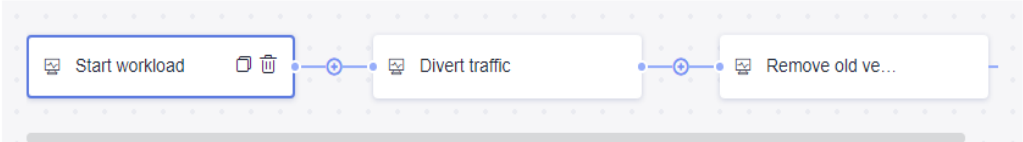
**Step 7** Enter the basic information and orchestrate extensions. For details, see [Atomic Extensions](#).

**Basic Information** Cancel Save Save and Apply

\* Policy  
GrayscaleUpgrade

Description  
Grayscale release based on ASM or Service

**Plugin Orchestration**



**Start workload**


\* Deploy Mode ?  
 Image Upgrade  YAML Deployment

\* Namespace  
--Select--

\* Service ?  
--Select--

Grayscale Version

**Step 8** Click **Save** after the configuration.

**Step 9** Find the created policy on the left and click  to apply it. The applied policy will be marked as **In-use**.

----End

## Atomic Extensions

CodeArts Release provides the following five extensions for rolling upgrade and grayscale upgrade:

### Rolling upgrade

**Rolling upgrade** supports image upgrade and YAML deployment.

- Image upgrade: Replace the container image in the workload.

**Rolling upgrade**\* Deploy Mode  Image Upgrade  YAML Deployment

\* Namespace

\* Workload

\* Container

**Table 7-2** Parameter description

Parameter	Description
Namespace	Namespace to which the service to be upgraded belongs.
Workload	Relative path of the YAML file. <ul style="list-style-type: none"><li>• The current directory is the root directory of the code branch.</li><li>• Only one YAML file is supported.</li><li>• You can use <i>\${variable name}</i> in a YAML path to reference an environment variable, and <i>{{variable name}}</i> in a YAML file to reference an environment variable.</li></ul>
Container	Container to be upgraded in the workload.

- YAML deployment: Use the YAML file to deploy or upgrade the workload.

**Rolling upgrade**

\* Deploy Mode [?](#)

Image Upgrade  YAML Deployment

\* Repo Type

Repo

\* Repo Uri

--Select--

\* Branch

--Select--

\* YAML path of workload [?](#)

deployment.yaml

**Table 7-3** Parameter description

Parameter	Description
Repo Type	Type of the code repository.
Repository	The code repository configured in the microservice.
Branch	The branch configured in the microservice.
YAML Path of Workload	YAML path of the workload to be upgraded. Enter a relative path of the YAML file. <ul style="list-style-type: none"><li>The current directory is the root directory of the code branch.</li><li>Only one YAML file is supported.</li><li>You can use <math>\\${variable\ name}</math> in a YAML path to reference an environment variable, and <math>\{\{variable\ name\}\}</math> in a YAML file to reference an environment variable.</li></ul>

### Start workload

**Start workload** supports image upgrade and YAML deployment.

- Image creation: Replace the container image in the workload, create a workload with the same online configuration, and update only the build product (image package).

**Start workload**



\* Deploy Mode 

Image Upgrade  YAML Deployment


\* Namespace

--Select--

\* Service 

--Select--

Grayscale Version

\* Grayscale Version Number 

Enter a value.

**Table 7-4** Parameter description

Parameter	Description
Namespace	Namespace to which the service to be upgraded belongs.
Service	Service to be upgraded.
Grayscale Version	Enable the parameter.
Grayscale Version Number	The version number will be used as the traffic diversion identifier. Enter <b><code>\${TIMESTAMP}</code></b> to reference the system environment variable.

- YAML deployment: Use the YAML file to deploy or upgrade the workload.

**Start workload**



\* Deploy Mode 

Image Upgrade  YAML Deployment

\* Namespace

--Select--

\* Service 

--Select--

\* Repo Type


Repo

\* Repo Url

--Select--

\* Branch

--Select--


\* YAML path of workload 

deployment.yaml

**Table 7-5** Parameter description

Parameter	Description
Namespace	Namespace to which the service to be upgraded belongs.
Service	The service in a cluster namespace. Only one workload must be associated with the service.
Repo Type	Type of the code repository.
Repository	The code repository configured in the microservice.
Branch	The branch configured in the microservice.
YAML Path of Workload	Relative path of the YAML file. <ul style="list-style-type: none"><li>• The current directory is the root directory of the code branch.</li><li>• Only one YAML file is supported.</li><li>• You can use <math>\\${variable\ name}</math> in a YAML path to reference an environment variable, and <math>\{\{variable\ name\}\}</math> in a YAML file to reference an environment variable.</li></ul>

## Divert traffic

**Divert traffic**\* Traffic type Service blue-green release 

Traffic diversion includes:

- Service blue-green release: All traffic will be switched to the new workload (gray load).
- ASM grayscale release: Use ASM (Application Services Mesh) VirtualService and DestinationRule configurations to control access traffic, perform grayscale diversion based on traffic proportion and request headers. ASM must be installed in the cluster.

**Remove old version**

This extension automatically takes down the old workload associated with the service. No configurations are required.

**Manual check**

You can approve or reject the deployment policy through manual check. If the policy is approved, the pipeline will continue to run. If the policy is rejected, the pipeline will stop.

**Manual check**

\* Timeout Processing

- Check failed and release flow terminated  
 Check result ignored and release flow continues

\* Check Duration

00  hours 00  minutes

Description

Enter a value.

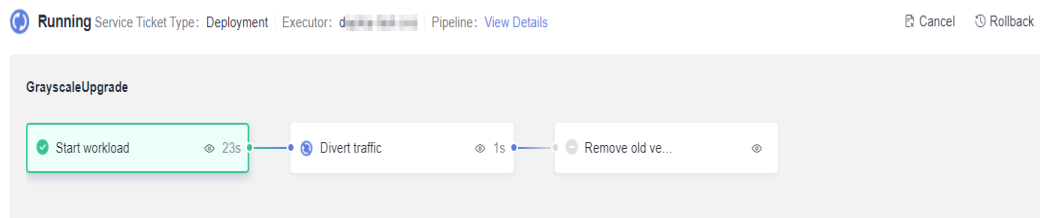
**Table 7-6** Parameter description

Parameter	Description
Timeout Processing	<ul style="list-style-type: none"><li>• Check failed and release flow terminated: Pipeline will pause before manual check. If the policy is not approved within the specified period, the pipeline will stop.</li><li>• Check result ignored and release flow continues: Pipeline will pause before manual check. If the policy is not approved within the specified period, the pipeline will continue to run.</li></ul>
Check Duration	The waiting time for check. The value ranges from 1 minute to 59 minutes and 59 seconds.
Description	(Optional) Enter a description.

## 7.4 Checking the Deployment Result

- Step 1** Log in to CodeArts.
- Step 2** Click a project name to access the project.
- Step 3** Choose **CICD > Release** to access the environment list page.
- Step 4** Click an environment name.
- Step 5** On the displayed page, click the **Deployment History** tab.
- Step 6** Click a service ticket name to check details. The details page displays the information of release flow, atomic extension, and some basic information.

### Release flow



- Release flow displays information such as the execution result, service ticket type, executor, pipeline, and release policy. You can click an atomic extension to check its details.
- Cancel: You can click **Cancel** to cancel the release.
- Retry: If the release fails or the release is canceled, you can click **Retry** to retry the release.
- Rollback: Click **Rollback**, in the displayed dialog box, if you confirm the rollback, the release will be canceled and the service state will be restored to its pre-release state. The rollback ticket page will be displayed.


### NOTE

Rollback can be performed anytime. If the current deployment version does not meet your expectation, you can quickly restore the environment to the previous one through rollback.

### Basic information

Basic information includes environment name, policy, service endpoint, variable version, image, start time, and end time.

### Atomic extension release

The release detail of atomic extension is displayed. You can click  **Refresh** to refresh the details.



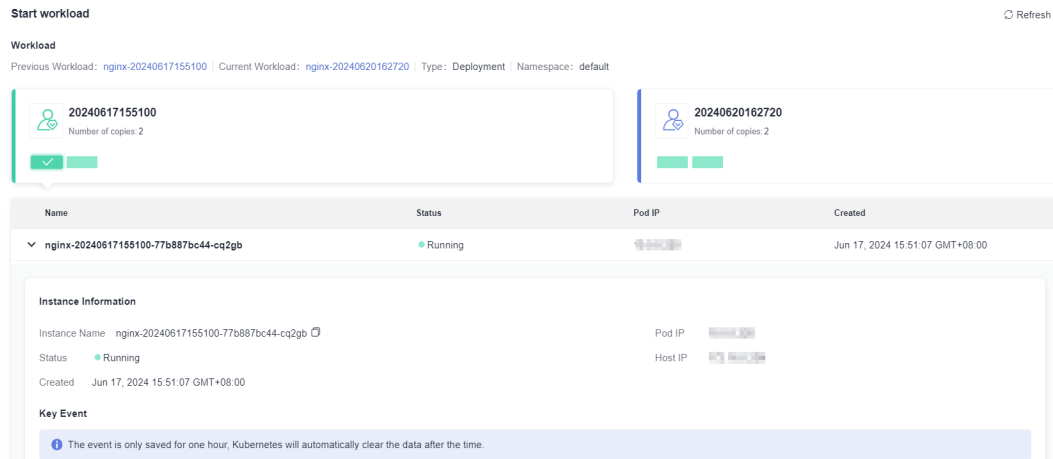


Table 7-7 Atomic extension release

Extension	Release Information
Rolling upgrade	<p>The details page displays the workload to be upgraded, instance information, and key event.</p> <ul style="list-style-type: none"> <li>• Workload Workload name, type, namespace, and creation time</li> <li>• Instance information Instance name, status, pod IP address, host IP address (IP address of the node where the pod is located), and creation time.</li> <li>• Key event K8s component name, event type, K8s event, first occurrence time, and recent occurrence time. Key event information can help you to locate pod faults.</li> </ul>
Start workload	<p>The details page displays workload to be upgraded, pod information, and key event. You can click the version cards to check the previous or the current workload information.</p> <ul style="list-style-type: none"> <li>• Workload Previous workload name, current workload name, type, and namespace</li> <li>• Instance information Instance name, status, pod IP address, host IP address (IP address of the node where the pod is located), and creation time.</li> <li>• Key event K8s component name, event type, K8s event, first occurrence time, and recent occurrence time. Key event information can help you to locate pod faults.</li> </ul>
Divert traffic	The details page displays the service name, old version number, new version number, and namespace.
Remove old version	The details page displays the workload name, workload type, and namespace.

Extension	Release Information
Manual check	The details page displays check duration, operation time, description, and status.

----End

## 7.5 Using Cloud Native Release in Pipeline

Cloud native release allows you to orchestrate release policies for environments in CCE clusters (such as rolling upgrade release and grayscale release).

You can use the cloud native release extension in a pipeline to trigger the configured release policy.

### Procedure

1. When creating or editing a pipeline, add the **CloudNativeRelease** extension for a job. For details, see [Table 7-8](#).

**MicroserviceRelease** 📘 Tips

CodeArts Release capabilities can be called on the pipeline for deployment. CodeArts...  
[Expand](#)

\*Name

\*Environment Level

\*Environment

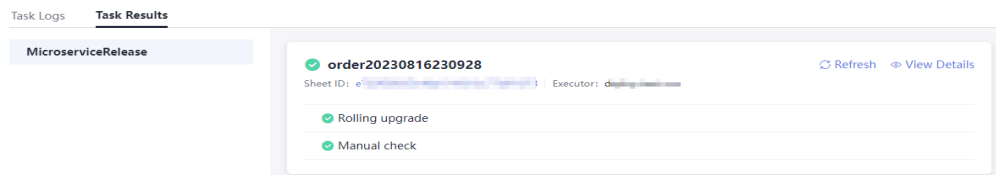
\*Artifact Path

**Table 7-8** Parameter description

Parameter	Description
Name	Extension name
Environment Level	Release environment type (microservice environment). Available environment types: development, test, pre-production, and production.
Environment	Microservice environment. For details, see <a href="#">Creating a Release Environment</a> .

Parameter	Description
Artifact Path	Image path for microservice deployment. Example: <b>swr.example.com/demo/springboot-helloworld:v1.1</b> . You can use <code>\${}</code> to reference pipeline parameters. Example: <b>swr.example.com/demo/springboot-helloworld:\${version}</b> . <b>NOTE</b> SoftWare Repository for Container (SWR) is recommended. You can <a href="#">build an image and push it to SWR</a> through CodeArts Build.

- Execute the pipeline after the configuration is complete.
- Click the task card to view the **Task Logs** and **Task Results**.



- **Task Logs:** displays real-time log information and running status.
- **Task Results:** displays basic task information, including the service ticket name, ticket ID, and trigger person.

Click the service ticket ID or the **View Details** button to go to the details page. For details, see [Checking the Deployment Result](#).

# 8 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 GitHub repository to obtain project source code, the Jenkins service to execute Jenkins tasks, a Kubernetes cluster for deployment, a Nexus repository to add private Maven repository information, or a Docker repository to manage Docker images, or create an IAM user endpoint to delegate the AK/SK of your account to the desired IAM user.

## Prerequisites

- By default, the project manager and project creator have all permissions for service endpoints, and other roles have only the read permission.
- Ensure that the third-party services configured in the service endpoint can be accessed through the public network without restrictions.

## Creating a Docker Repository Service Endpoint

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

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

**Table 8-1** Parameter description

Parameter	Description
Service Endpoint Name	Name of the service endpoint.
Repository Address	Address of the Docker image repository (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 Jenkins Service Endpoint

You can use a Jenkins service endpoint to connect to Jenkins. After the connection is successful, you can perform operations on Jenkins tasks.

### NOTE

Currently, this function is available in LA-Mexico City2, LA-Sao Paulo1, and AP-Singapore.

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

**Table 8-2** Parameter description

Parameter	Description
Service Endpoint Name	Name of the service endpoint.
Server URL	Address of the Jenkins service. Currently, only public network addresses such as <code>http://&lt;IP_address&gt;:&lt;port_number&gt;</code> and <code>https://&lt;IP_address&gt;:&lt;port_number&gt;</code> are supported.
Username	Username for logging in to the Jenkins service.
Password	Password for logging in to the Jenkins service.

### NOTE

You can click **Verify and OK** to check whether the username and password can be connected to the Jenkins server.

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. Access a project, choose **Settings > General > Service Endpoints**.

2. Click **Create Endpoint** and select **Kubernetes** from the drop-down list.
3. In the **Create Service Endpoint** dialog box, set parameters.

**Table 8-3** Parameter description

Parameter	Description
Service Endpoint Name	Name of the service endpoint.
Kubernetes URL	API server address of the Kubernetes cluster (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 <a href="#">Organizing Cluster Access Using kubeconfig Files</a> . <b>NOTE</b> If a CCE cluster is used, you can obtain the Kubeconfig file by referring to <a href="#">Connecting to a Cluster Using kubectl</a> .

4. Click **OK**.

## Creating a Nexus Repository Service Endpoint

You can use a Nexus repository endpoint to add private Maven repository information.

### NOTE

Currently, this function is available in LA-Mexico City2, LA-Sao Paulo1, and AP-Singapore.

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

**Table 8-4** Parameter description

Parameter	Description
Service Endpoint Name	Name of the service endpoint.
Repository URL	Nexus repository public network address (HTTP or HTTPS address).
Username	Username for logging in to the Nexus repository.
Password	Password for logging in to the Nexus repository.

4. Click **OK**.

## Creating a GitLab Repository Service Endpoint

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

### NOTE

Currently, this function is available in LA-Mexico City<sup>2</sup>, LA-Sao Paulo<sup>1</sup>, and AP-Singapore.

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

**Table 8-5** Parameter description

Parameter	Description
Service Endpoint Name	Name of the service endpoint.
GitLab URL	Public network address (HTTP or HTTPS address) of the official or self-built GitLab repository.
Username	Username for logging in to the GitLab repository.
Access Token	Obtain an access token of the GitLab repository and enter it for authentication.

4. Click **OK**.

## Creating a GitHub Service Endpoint

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

### NOTE

Currently, this endpoint is available in LA-Mexico City<sup>2</sup>, LA-Sao Paulo<sup>1</sup>, AP-Singapore, and TR-Istanbul.

1. Access a project, choose **Settings > General > Service Endpoints**.
2. Click **Create Endpoint** and select **GitHub repository** from the drop-down list.
3. In the **Create Service Endpoint** dialog box, select an authentication mode and set other parameters.
  - OAuth authentication

**Table 8-6** Parameter description

Parameter	Description
Service Endpoint Name	Name of the service endpoint.
Authentication Mode	In OAuth authentication mode, log in to GitHub for manual authorization.

- Access token authentication

**Table 8-7** Parameter description

Parameter	Description
Service Endpoint Name	Name of the service endpoint.
Authentication Mode	Access token authentication is used.
Access Token	<b>Obtain an access token</b> and enter it for authentication.

4. Click **OK**.

## Creating a Git Service Endpoint

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

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

**Table 8-8** Parameter description

Parameter	Description
Service Endpoint Name	Name of the service endpoint.
Git Repository URL	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. The user can obtain the token of your account through the AK/SK to perform tasks with higher permissions.

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

**Table 8-9** Parameter description

Parameter	Description
Service Endpoint Name	Name of the service endpoint.
Access Key Id	Access Key ID (AK). For details, see <a href="#">How Do I Obtain an Access Key (AK/SK)?</a>
Secret Access Key	Secret Access Key (SK). For details, see <a href="#">How Do I Obtain an Access Key (AK/SK)?</a>

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 used for change-triggered pipelines and related extensions.

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

**Table 8-10** 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"><li>1. Go to any code repository in the project.</li><li>2. Copy the HTTPS address of the code repository and enter <b>https://example.com</b>.</li></ol> <p><b>NOTE</b> When creating a pipeline in the Pipeline homepage, you do not need to set this parameter for the Repo endpoint.</p>

Parameter	Description
Username	HTTPS username used to push code to and pull code from Repo. HTTPS username can be the tenant name or IAM username. Enter a complete username without spaces.
Password	HTTPS password used to push code to and pull code from Repo.

 **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.

4. Click **OK**.

## Editing or Deleting a Service Endpoint

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

 **NOTE**

After editing an endpoint used by a pipeline, you need to manually update the pipeline.

# 9 Key Operations Recorded by CTS

## 9.1 CodeArts Pipeline Operations Recorded by CTS

CodeArts Pipeline: provides visualized, custom automatic delivery pipelines to shorten the delivery period and improve efficiency.

With CTS, you can record pipeline operations for later query, audit, and backtracking.

**Table 9-1** CodeArts pipeline operations that can be recorded by CTS

Operation	Resource Type	Event Name
Executing a pipeline task	pipelineTask	executePipelineTask
Editing a pipeline task	pipelineTask	updatePipelineTask
Creating a pipeline task	pipelineTask	createPipelineTask
Deleting a pipeline task	pipelineTask	deletePipelineTask
Stopping a pipeline task	pipelineTask	stopPipelineTask

## 9.2 Querying Real-Time Traces

### Scenarios

After you enable CTS and the management tracker is created, CTS starts recording operations on cloud resources. After a data tracker is created, the system starts recording operations on data in OBS buckets. CTS stores operation records generated in the last seven days.

This section describes how to query and export operation records of the last seven days on the CTS console.


- [Viewing Real-Time Traces in the Trace List of the New Edition](#)




- [Viewing Real-Time Traces in the Trace List of the Old Edition](#)

## Constraints


- Traces of a single account can be viewed on the CTS console. Multi-account traces can be viewed only on the **Trace List** page of each account, or in the OBS bucket or the **CTS/system** log stream configured for the management tracker with the organization function enabled.
- You can only query operation records of the last seven days on the CTS console. To store operation records for more than seven days, you must configure an OBS bucket to transfer records to it. Otherwise, you cannot query the operation records generated seven days ago.
- After performing operations on the cloud, you can query management traces on the CTS console 1 minute later and query data traces on the CTS console 5 minutes later.



## Viewing Real-Time Traces in the Trace List of the New Edition

1. Log in to the management console.
2. Click  in the upper left corner and choose **Management & Governance** Management & Deployment > **Cloud Trace Service**. The CTS console is displayed.
3. Choose **Trace List** in the navigation pane on the left.
4. On the **Trace List** page, use advanced search to query traces. You can combine one or more filters.
  - **Trace Name:** Enter a trace name.
  - **Trace ID:** Enter a trace ID.
  - **Resource Name:** Enter a resource name. If the cloud resource involved in the trace does not have a resource name or the corresponding API operation does not involve the resource name parameter, leave this field empty.
  - **Resource ID:** Enter a resource ID. Leave this field empty if the resource has no resource ID or if resource creation failed.
  - **Trace Source:** Select a cloud service name from the drop-down list.
  - **Resource Type:** Select a resource type from the drop-down list.
  - **Operator:** Select one or more operators from the drop-down list.
  - **Trace Status:** Select **normal**, **warning**, or **incident**.
    - **normal:** The operation succeeded.
    - **warning:** The operation failed.
    - **incident:** The operation caused a fault that is more serious than the operation failure, for example, causing other faults.
  - **Enterprise Project ID:** Enter an enterprise project ID.
  - **Access Key:** Enter an access key ID, including temporary access credentials and permanent access keys.
  - Time range: Select **Last 1 hour**, **Last 1 day**, or **Last 1 week**, or specify a custom time range.

5. On the **Trace List** page, you can also export and refresh the trace list, and customize the list display settings.
  - Enter any keyword in the search box and press Enter to filter desired traces.
  - Click **Export** to export all traces in the query result as an .xlsx file. The file can contain up to 5000 records.
  - Click  to view the latest information about traces.
  - Click  to customize the information to be displayed in the trace list. If **Auto wrapping** is enabled () , excess text will move down to the next line; otherwise, the text will be truncated. By default, this function is disabled.
6. For details about key fields in the trace structure, see [Trace Structure](#) section "Trace References" > "Trace Structure" and [Example Traces](#) section "Trace References" > "Example Traces".
7. (Optional) On the **Trace List** page of the new edition, click **Go to Old Edition** in the upper right corner to switch to the **Trace List** page of the old edition.

## Viewing Real-Time Traces in the Trace List of the Old Edition

1. Log in to the management console.
2. Click  in the upper left corner and choose **Management & Governance** **Management & Deployment** > **Cloud Trace Service**. The CTS console is displayed.
3. Choose **Trace List** in the navigation pane on the left.
4. Each time you log in to the CTS console, the new edition is displayed by default. Click **Go to Old Edition** in the upper right corner to switch to the trace list of the old edition.
5. Set filters to search for your desired traces. The following filters are available:
  - **Trace Type, Trace Source, Resource Type, and Search By:** Select a filter from the drop-down list.
    - If you select **Resource ID** for **Search By**, specify a resource ID.
    - If you select **Trace name** for **Search By**, specify a trace name.
    - If you select **Resource name** for **Search By**, specify a resource name.
  - **Operator:** Select a user.
  - **Trace Status:** Select **All trace statuses, Normal, Warning, or Incident**.
  - Time range: You can query traces generated during any time range in the last seven days.
  - Click **Export** to export all traces in the query result as a CSV file. The file can contain up to 5000 records.
6. Click **Query**.
7. On the **Trace List** page, you can also export and refresh the trace list.
  - Click **Export** to export all traces in the query result as a CSV file. The file can contain up to 5000 records.

- Click  to view the latest information about traces.
8. Click  on the left of a trace to expand its details.

Trace Name	Resource Type	Trace Source	Resource ID	Resource Name	Trace Status	Operator	Operation Time	Operation
createDockerConfig	dockerlogincmd	SWR	-	dockerlogincmd	normal		Nov 16, 2023 10:54:04 GMT+08:00	View Trace

request	
trace_id	
code	200
trace_name	createDockerConfig
resource_type	dockerlogincmd
trace_rating	normal
api_version	
message	createDockerConfig, Method: POST Url=/v2/manage/utils/secret, Reason:
source_ip	
domain_id	
trace_type	ApiCall

9. Click **View Trace** in the **Operation** column. The trace details are displayed.

**View Trace** ×

```
{
  "request": "",
  "trace_id": "",
  "code": "200",
  "trace_name": "createDockerConfig",
  "resource_type": "dockerlogincmd",
  "trace_rating": "normal",
  "api_version": "",
  "message": "createDockerConfig, Method: POST Url=/v2/manage/utils/secret, Reason:",
  "source_ip": "",
  "domain_id": "",
  "trace_type": "ApiCall",
  "service_type": "SWR",
  "event_type": "system",
  "project_id": "",
  "response": "",
  "resource_id": "",
  "tracker_name": "system",
  "time": "Nov 16, 2023 10:54:04 GMT+08:00",
  "resource_name": "dockerlogincmd",
  "user": {
    "domain": {
      "name": "",
      "id": ""
    }
  }
}
```

10. For details about key fields in the trace structure, see [Trace Structure](#) section "Trace References" > "Trace Structure" and [Example Traces](#) section "Trace References" > "Example Traces" in the *CTS User Guide*.
11. (Optional) On the **Trace List** page of the old edition, click **New Edition** in the upper right corner to switch to the **Trace List** page of the new edition.