CodeArts

Getting Started

Issue	01
Date	2024-11-15

HUAWEI CLOUD COMPUTING TECHNOLOGIES CO., LTD.

Trademarks and Permissions

Notice

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

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

Contents

1 Guidance

Table 1-1 Service guidance

Service	Guidance
Overall process	• Deploying a Code Development Pipeline on ECS • Deploying a Code Development Pipeline on CCE
CodeArts Req	• Creating a Scrum Project and Work Item • Creating an IPD-System Device Project and Work Item
CodeArts Repo	• Developing Java Code in a Scrum Project • Configuring CodeArts Repo Policies by Administrator
CodeArts Pipeline	Generating a Software Package and Deploying It on a Host Through CodeArts Pipeline
CodeArts Check	Checking Code from CodeArts Repo
CodeArts Build	• Building with Ant and Uploading the Package to a Release Repo • Building with CMake and Uploading the Package to a Release Repo • Building with Maven, Uploading the Software Package, and Pushing the Image
CodeArts Artifact	• Uploading Software Packages to Release Repos • Uploading Components to Maven Repository
CodeArts Deploy	Creating a Tomcat Application Using CodeArts Deploy and Deploying It on an ECS
CodeArts TestPlan	Executing a Test Plan and Viewing the Report
CodeArts PerfTest	CodeArts PerfTest Getting Started

2 Deploying a Code Development Pipeline on ECS

This section describes how to use the built-in code repository of CodeArts to develop, build, and deploy projects for continuous delivery.

This chapter uses Elastic Cloud Server (ECS) for traditional software package deployment.

To use container-based deployment, see **Deploying a Code Development Pipeline on CCE**.

Preparations

1. You have signed up for Huawei Cloud. If you do not have a HUAWEI ID, create one by referring to **Registering a HUAWEI ID and Enabling Huawei Cloud Services**.
2. You have **purchased an ECS**. The following table lists the mandatory configurations. You can select the configurations that are not listed in the table based on the site requirements. After the purchase is complete, add inbound rules for ports 22 and 8080 by referring to **Configuring Security Group Rules**.

Table 2-1 Configuring ECS

Category	Parameter	Suggestion
Basic Settings	Billing Mode	Select **Pay-per-use**.
Instance	CPU Architecture	Select **x86**.
	Specifications	Select 2 vCPUs and 4 GiB or higher.

Category	Parameter	Suggestion
OS	Image	Choose **Public image** > **CentOS** > **CentOS 7.6 64bit(10GiB)**.
Public Network Access	EIP	Select **Auto assign**.
	Billed By	Select **Bandwidth**.
Instance Management	Login Mode	Select **Password**.
	Password	Enter a password.

Enabling CodeArts Free Edition

Step 1 Go to the **Buy CodeArts** page.

Step 2 Select **Free**, read and agree to the statement, and click **Subscribe**.

Step 3 Check the subscription record on the **CodeArts** page.

----End

Creating a Project

A project is the basis for using services on CodeArts. Subsequent operations can be performed only after a project is created.

Step 1 Click **Access Service** on the CodeArts console.

Step 2 Click **Create Project**, select **Scrum**, enter the project name **Demo**, and click **OK**.

----End

Creating a Code Repository

You can use a code repository to manage project code versions. This section describes how to use the built-in template **Java Web Demo** to create a code repository.

Step 1 In the navigation pane, choose **Code** > **Repo**.

Step 2 Click **New Repository**, select **Template**, and click **Next**.

Step 3 On the page displayed, select **Java Web Demo** and click **Next**.

Step 4 Enter the code repository name **Web-Demo** and click **OK**.

----End

Checking Code

You can use CodeArts Check to perform static code check and control code quality.

Step 1 In the navigation pane, choose **Code** > **Check**. The automatically created task **Web-Demo-check** is displayed on the page.

Step 2 Click ▷ in the row where the task is located to start the task.

Step 3 When ✓ is displayed, the task is successfully executed. Click the task name, go to the **Overview** tab page, and view the check result.

If the task fails, check and fix errors based on the message displayed on the page.

----End

Building and Archiving Software Packages

You can use CodeArts Build to compile the source code of the software into a target file, packs the configuration file and resource file, and archives them to a release repo.

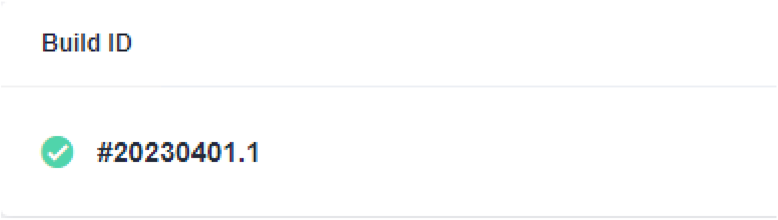
Step 1 In the navigation pane, choose **CICD** > **Build**. The automatically created build task **Web-Demo-build** is displayed.

Step 2 Click ▷ in the row where the task is located to start the task. If a dialog box is displayed, confirm the parameter settings and click **Confirm**.

Step 3 When ✓ is displayed, the task is successfully executed. Click the task name. On the **Build History** page that is displayed, find the **Build ID** of the latest build in the list and record the ID.

If the build fails, rectify the fault based on the failed action information and error information in logs.

Figure 2-1 Build ID

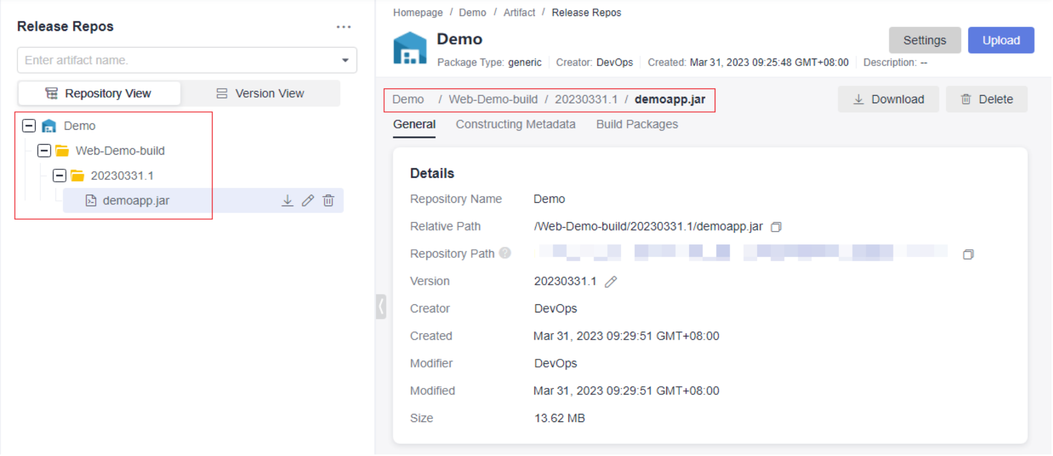


Build ID
✓ #20230401.1

Step 4 Choose **Artifact** in the navigation pane and click the **Release Repos** tab.

In the repository named after the project, go to the folder named after the build task and the folder named after the build number in sequence to find the generated software package **demoapp.jar**.

Figure 2-2 Viewing the software package



Demo

Package Type: generic | Creator: DevOps | Created: Mar 31, 2023 09:25:48 GMT+08:00 | Description: --

Demo / Web-Demo-build / 20230331.1 / **demoapp.jar**

Details

Repository Name	Demo
Relative Path	/Web-Demo-build/20230331.1/demoapp.jar
Repository Path	[redacted]
Version	20230331.1
Creator	DevOps
Created	Mar 31, 2023 09:29:51 GMT+08:00
Modifier	DevOps
Modified	Mar 31, 2023 09:29:51 GMT+08:00
Size	13.62 MB

----End

Deploying the Build Package

You can use CodeArts Deploy to deploy software packages in a release repo to a VM and run the software.

Step 1 Configure the target host.

1. In the navigation pane, choose **Settings** > **General** > **Basic Resources**.
2. Click **Create Host Cluster**, configure the following information, and click **Save**.

Table 2-2 Creating a host cluster

Parameter	Suggestion
Cluster Name	Enter **host-group**.
OS	Select **Linux**.
Host Connection Mode	Select **Direct Connection**.
Execution Resource Pool	Select **Official**.

3. Click **Add Host**, configure the following information, and click **OK**.

Table 2-3 Adding a host

Parameter	Suggestion
Add Hosts by	Select **Adding IP**.

Parameter	Suggestion
Host Name	You are advised to keep this same as the name of an ECS purchased in **Preparations**.
IP	Enter the elastic IP address of an ECS purchased in **Preparations**.
Authorization	Select **Password**.
Username	Enter **root**.
Password	Enter the password set when you purchase the ECS in **Preparations**.
SSH Port	Enter **22**.

4. A host record is displayed on the page. If the **Verification Result** column shows successful, the host is added.

   If the host fails to be added, check its configuration based on the failure details.

Step 2 Choose **CICD** > **Deploy** from the navigation pane. The automatically created application **Web-Demo-deploy** is displayed on the page.

Step 3 Click ··· and choose **Edit**.

Step 4 Click the **Environment Management** tab and configure the host environment.

1. Click **Create Environment**, configure the following information, and click **Save**.

Table 2-4 Creating an environment

Parameter	Suggestion
Environment	Enter **demo-env**.
Resource Type	Select **Host**.
OS	Select **Linux**.

2. Click **Import Host**. In the displayed dialog box, select the host cluster and host configured in **Step 1** and click **Import**.
3. A message is displayed, indicating that the import is successful. Close the window.

Step 5 Click the **Deployment Actions** tab and configure information.

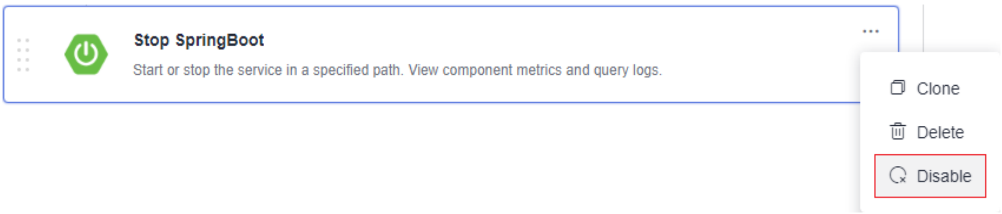
- **Install JDK**: Check that the JDK version is **openjdk-1.8.0**.
- **Select Deployment Source**: Set the parameters based on the following table.

Table 2-5 Deployment source configuration

Parameter	Value
Source	Click **Build task**.
Build Task	Select **Web-Demo-build**.
Download Path	Enter **/usr/local/${package_name}/**.

- **Stop Spring Boot**: When you perform this action for the first time, it fails because no service is running on the target host. To disable this action, click ··· on the action card and choose **Disable**.

Figure 2-3 Disabling "Stop Spring Boot"


Stop SpringBoot
Start or stop the service in a specified path. View component metrics and query logs.
···
Clone
Delete
Disable

- **Health Test via URLs**: This action is optional and disabled in this example.

Step 6 Click the **Parameters** tab and set parameters by referring to the following table.

Parameter Name	Value
host_group	Select the environment **demo-env** added in **Step 4**.
package_url	This parameter is not required. Click 🗑 in the same row to delete it.
service_port	Enter **8080**.
package_name	Enter **demoapp**.

Step 7 Click **Save & Deploy**. If a dialog box is displayed, confirm the parameter settings and click **OK**.

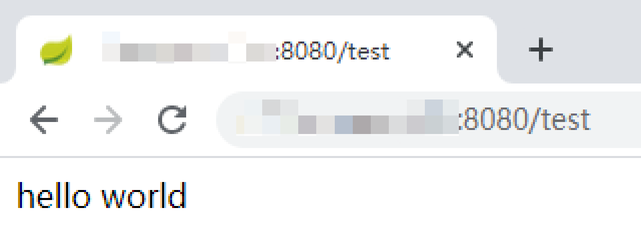
Wait until ✓ Successful is displayed on the page. If the deployment fails, rectify the fault based on the failed action information and error information in logs.

Step 8 View the deployment result.

Open a new browser page and enter the access address **http://IP:8080/test**. **IP** is the elastic IP address of the ECS purchased in **Preparations**.

If the following information is displayed, the deployment is successful.

Figure 2-4 Deployment result



hello world

----End

Configuring a Pipeline

You can use a CodeArts pipeline to connect code check, build, and deployment tasks. When code changes, the pipeline is automatically triggered for continuous delivery.

Step 1 Choose **CICD** > **Pipeline** from the navigation pane. On the **Pipelines** tab, the automatically created pipeline **Web-Demo-pipeline** is displayed.

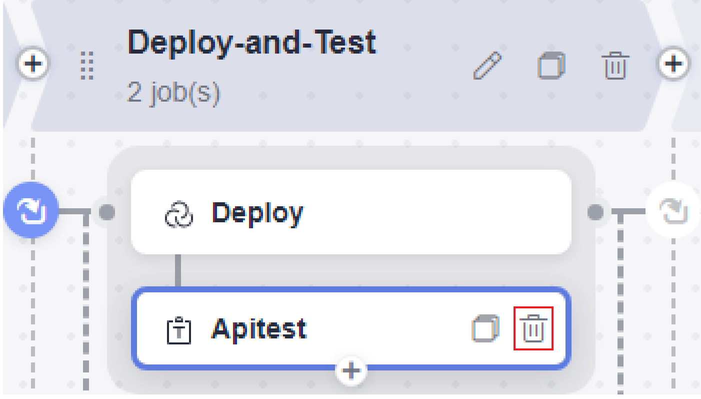
Step 2 Click ··· and choose **Edit**.

Step 3 On the **Task Orchestration** tab page, configure the pipeline.

1. APITest is not involved in this document. Therefore, remove the API test task from the pipeline.

   Click 🗑 next to the **Apitest** job. In the dialog box that is displayed, click **OK**.

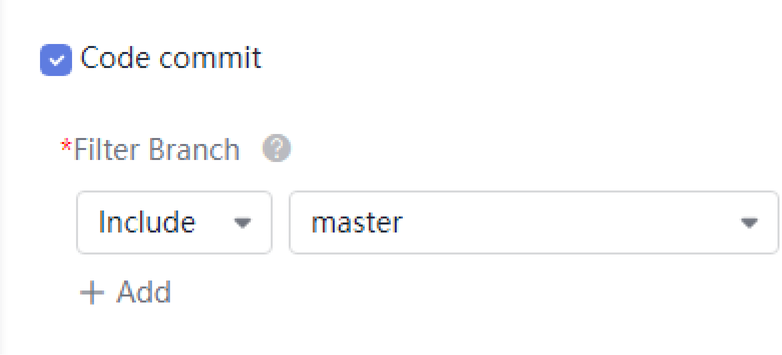
   Figure 2-5 Deleting a job

2. Click the **Deploy** job, associate build task **Build**, and set other parameters based on the parameter settings in **Deploying the Build Package**.

Step 4 Click the **Execution Plan** tab, select **Code commit**, select **master** from the branch filter drop-down list.

Figure 2-6 Configuring the execution plan



☑ Code commit

*Filter Branch

Include | master

+ Add

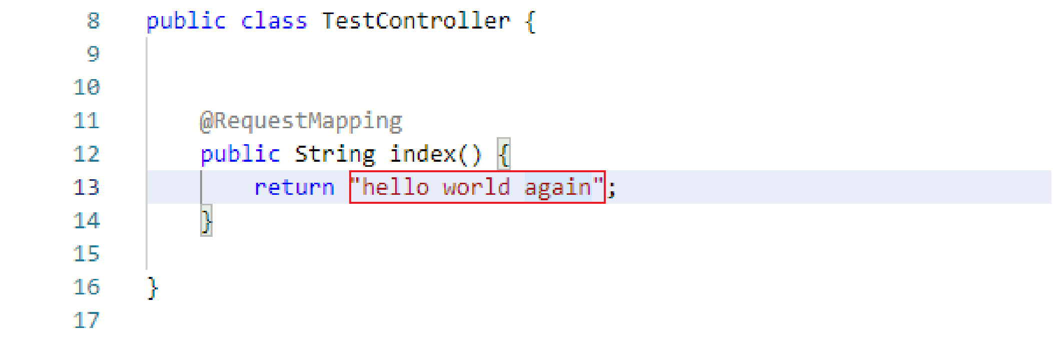
Step 5 Click **Save**.

Step 6 Go to **Deploy**, edit the deployment actions, and enable **Stop Spring Boot**.

Step 7 Go to the code repository and search for and open the **TestController.java** file.

Click ✎, change **hello world** to **hello world again**, submit the information, and click **OK**.

Figure 2-7 Modifying code



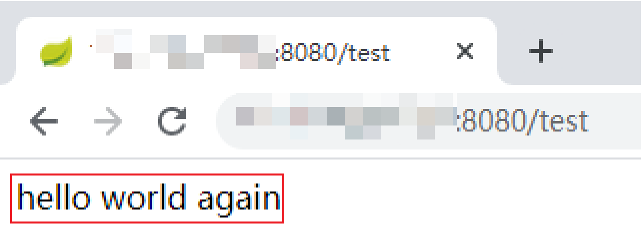
```
public class TestController {


    @RequestMapping
    public String index() {
        return "hello world again";
    }

}

```

Step 8 Return to the Pipeline page. You can see that the pipeline is running.

When ✓ is displayed, access **http://IP:8080/test** again. The following figure shows the access result.

If the task fails to be executed, check the failure cause. You can open the step details page to view the task logs and rectify the fault based on the logs.

Figure 2-8 Pipeline execution result



hello world again

----End

Follow-up Operations

To avoid incurring unnecessary fees, you can release any unused resources once you have completed this example. The following resources can be released.

Table 2-6 Releasing resources

Resource	Releasing Method
CodeArts project	Choose **Settings** > **General** > **Basic Information**, click **Delete Project**, and follow the prompts to delete the project.
ECS	Log in to the ECS console, locate the ECS to be deleted in the list, choose **More** > **Delete**, and follow the prompts to delete the ECS.

NOTICE

Released resources cannot be restored. Exercise caution when performing this operation.

Related Information

The check, build, deployment, and pipeline tasks used in this section are provided by the repo template.

You can create tasks for your own project by referring to the following instructions.

Table 2-7 Task creation instructions

Service	Method
CodeArts Check	See **Creating a Task to Check Code from Repo**.
CodeArts Build	See **Creating a Build Task**.
CodeArts Deploy	See **Creating an Application**.
CodeArts Pipeline	See **Creating a Pipeline**.

3 Deploying a Code Development Pipeline on CCE

This section describes how to use the built-in code repository of CodeArts to develop, build, and deploy projects for continuous delivery.

This chapter uses Cloud Container Engine (CCE) for container-based deployment.

To use traditional software package deployment, see **Deploying a Code Development Pipeline on ECS**.

Preparations

1. You have signed up for Huawei Cloud. If you do not have a HUAWEI ID, create one by referring to **Registering a HUAWEI ID and Enabling Huawei Cloud Services**.
2. You have **purchased a CCE cluster**. The cluster is configured according to **Table 3-1** and **Table 3-2**. Default values can be retained for configurations not listed in the tables.

Table 3-1 Cluster configurations

Category	Parameter	Suggestion
Basic Settings	Type	Select **CCE Standard Cluster**.
	Billing Mode	Select **Pay-per-use**.
	Cluster Name	Enter a name.
	Cluster Version	You are advised to select the latest version.
Network Settings	Network Model	Select **VPC network**.

Category	Parameter	Suggestion
	VPC	Select a VPC. If no proper VPC is available in the list, click **Create VPC** to create one.
	Default Node Subnet	Select a subnet. If no proper subnet is available in the list, click **Create Subnet** to create one.
	Container CIDR Block	Select **Auto select**.

Table 3-2 Node configurations

Category	Parameter	Suggestion
Node Configuration	Billing Mode	Select **Pay-per-use**.
	Node Type	Select **Elastic Cloud Server (VM)**.
	Specifications	Select 2 vCPUs and 4 GiB or higher.
	Container Engine	Select **Docker**.
	OS	Select **Public image** > **CentOS 7.6**.
	Login Mode	Select **Password**.
	Password	Enter a password.
Network Settings	Node IP	Select **Random**.
	EIP	Select **Do not use**.

3. You have created an organization named **web-demo** in SWR. For details, see **Creating an Organization**.

NOTE

If the system displays a message indicating that the organization already exists, use another name.

Enabling CodeArts Free Edition

Step 1 Go to the **Buy CodeArts** page.

Step 2 Select **Free**, read and agree to the statement, and click **Subscribe**.

Step 3 Check the subscription record on the **CodeArts** page.

----End

Creating a Project

A project is the basis for using services on CodeArts. Subsequent operations can be performed only after a project is created.

Step 1 Click **Access Service** on the CodeArts console.

Step 2 Click **Create Project**, select **Scrum**, enter the project name **Demo**, and click **OK**.

----End

Creating a Code Repository

You can use a code repository to manage project code versions. This section describes how to use the built-in template **Java Web Demo** to create a code repository.

Step 1 In the navigation pane, choose **Code** > **Repo**.

Step 2 Click **New Repository**, select **Template**, and click **Next**.

Step 3 On the page displayed, select **Java Web Demo** and click **Next**.

Step 4 Enter the code repository name **Web-Demo** and click **OK**.

----End

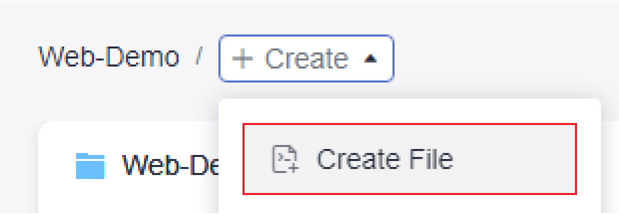
Preparing a Dockerfile

A Dockerfile is a text file that contains the instructions and descriptions required for building an image. For details about Dockerfile, see the **Docker official website**.

Step 1 Click a repository name to go to the repository.

Step 2 Click **Create** above the file list. Select **Create File** from the drop-down list.

Figure 3-1 Creating a file



Step 3 Enter the file name **Dockerfile** and then enter the following code:

```
FROM openjdk:8-alpine
ADD target /demo
COPY ./target/demoapp.jar /demo
CMD ["java","-jar","/demo/demoapp.jar"]
```

Step 4 Enter a commit message and click **OK**.

----End

Building and Pushing an Image

Use a build task to compile the software source code into an image and push and archive the image to SWR.

Step 1 In the navigation pane, choose **CICD** > **Build**.

Step 2 Click **Create Task** and configure task information.

1. **Basic Information**: Configure the following information and click **Next**.

Table 3-3 Basic information

Parameter	Suggestion
Name	Enter **Web-Demo-docker**.
Code Source	Select **Repo**.
Repository	Select **Web-Demo**.
Branch	Select **master**.

2. **Select Template**: Select **Blank Template** and click **OK**.

Step 3 Configure build actions.

1. Click **Add Build Actions**, find **Build with Maven** in the list, and click **Add**.
2. Click **Add Action**. In the action list, find **Build Image and Push to SWR**. Click **Add**.
3. Configure **Build Image and Push to SWR** by referring to the following table. (Retain the default values for the fields not listed in this table.)

Table 3-4 Configuring image information

Parameter	Suggestion
Organization	Enter the name (**web-demo**) of the organization created in **Preparations**.
Image Tag	v1.0.0

Step 4 After the configuration is complete, click **Save and Run**.

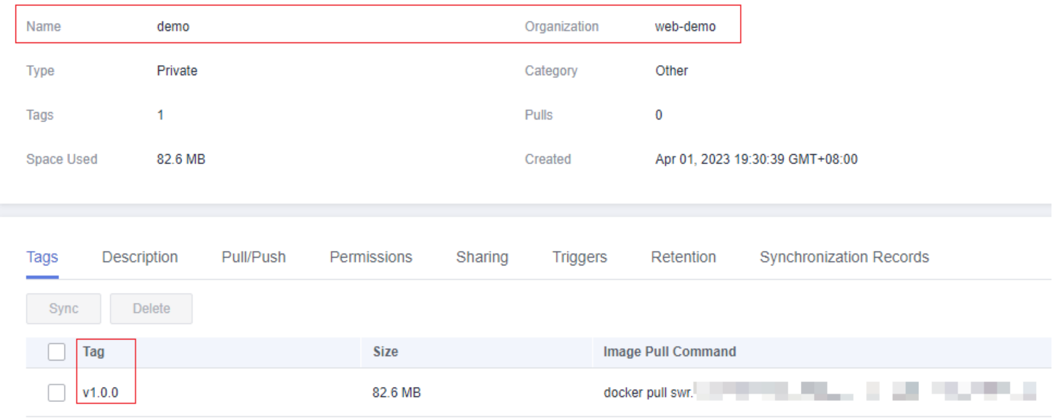
When ✓ is displayed, the task is successfully executed. If the build fails, rectify the fault based on the failed action information and error information in logs.

Step 5 Log in to the SWR console. In the navigation pane, choose **My Images**.

There is a record whose **Name** is **demo** and **Organization** is **web-demo**.

Click the image name to view details. The image version is **v1.0.0**.

Figure 3-2 Viewing images



Name	demo	Organization	web-demo
Type	Private	Category	Other
Tags	1	Pulls	0
Space Used	82.6 MB	Created	Apr 01, 2023 19:30:39 GMT+08:00

Tag	Size	Image Pull Command
v1.0.0	82.6 MB	docker pull swr.[redacted]

----End

Creating a Workload

Create a Deployment on CCE to deploy and run the demo image.

Step 1 Log in to the CCE console and click the cluster purchased in **Preparations** to go to the details page.

Step 2 Choose **Workloads** in the navigation pane, and click **Create Workload**.

Step 3 Complete the configurations by referring to the following table and click **Create Workload**.

Table 3-5 Creating workload

Category	Parameter	Suggestion
Basic Info	Workload Type	Select **Deployment**.
	Workload Name	Enter **web-demo**.
	Pods	Enter **1**.
Container Settings	Image Name	Click **Select Image**. In the dialog box that is displayed, select **demo** and click **OK**.
	Pull Policy	Select **Always**.
	Image Tag	Select **v1.0.0**.
Advanced Settings	Upgrade Mode	Set **Upgrade Mode** to **Replace upgrade**.

Step 4 When the creation is complete, click **View Workload Details** to go back to the details page. There is a record on the **Pods** tab.

If the pod status is **Running**, click the **Access Mode** tab, click **Create Service**, configure the service by referring to the following table, and click **OK**.

If the instance status is abnormal, rectify the fault by referring to **Workload Abnormalities**.

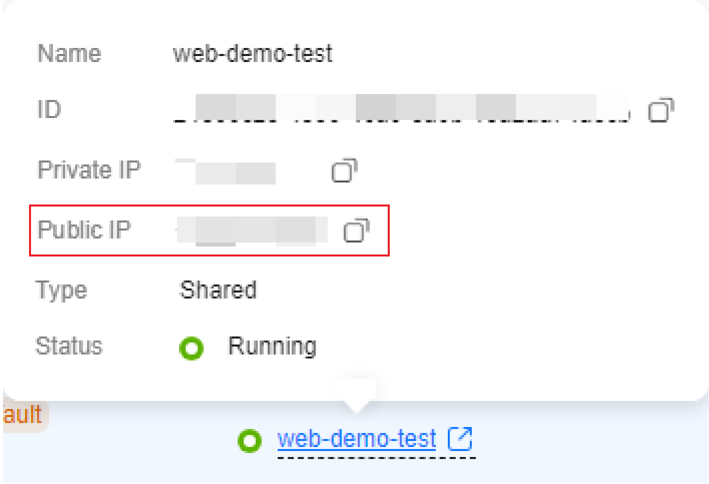
Table 3-6 Configuring access mode

Parameter	Suggestion
Service Name	Enter **web-demo**.
Service Type	Select **LoadBalancer**.
Service Affinity	Select **Cluster-level**.
Load Balancer	● Choose **Shared** > **Auto create**. ● **Instance Name**: Enter **web-demo-test**. ● **EIP**: Select **Auto create**.
Port	● Set **Protocol** to **TCP**. ● Set **Container Port** to **8080**. ● Set **Service Port** to **8080**.

Step 5 Check the record displayed in the list.

If ○ web-demo-test ↗ is displayed in the record, hover over the load balancer name in the **Access Type** column, and copy the public IP address in the pop-up window.

Figure 3-3 Copying the access address



Name web-demo-test

ID [redacted]

Private IP [redacted]

Public IP [redacted]

Type Shared

Status ○ Running

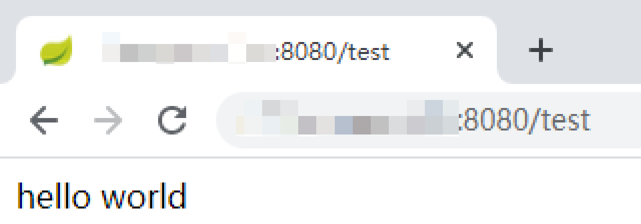
ault

○ web-demo-test ↗

Step 6 Open a new browser page and enter **http://IP:8080/test** in the address box. Replace **IP** with the public network address copied in **Step 5**.

If the following information is displayed, the workload is running properly.

Figure 3-4 Deployment result



[redacted]:8080/test

[redacted]:8080/test

hello world

----End

Deploying an Image

You can create applications on Deploy to automatically deploy images.

Step 1 Return to the CodeArts page and choose **CICD** > **Deploy** in the navigation pane.

1. Click **Create Application**.
2. Enter **web-demo-k8s** for **Name** and click **Next**.
3. Select the **Blank Template** and click **OK**.

Step 2 Search for and add action **Kubernetes Quick Deployment (CCE Cluster)**. Configure this action by referring to the following table.

Table 3-7 Configuring deployment actions

Parameter	Suggestion
Region	Select the region where the cluster located.
Cluster Name	Select the cluster purchased in **Preparations**.
Namespace	Select **default**.
Workload	Select **web-demo**.
Container	Select a container name configured when **Create Workload** is selected.

Step 3 Click **Save & Deploy**.

If ✓ Successful is displayed, the test is successful. If the deployment fails, rectify the fault based on the failed action information and error information in logs.

----End

Configuring a Pipeline to Automatically Update Image Deployment

Configure a pipeline to integrate the code repository, build, and deployment. When a code commit action occurs in the code repository, the pipeline is automatically executed for continuous delivery.

Step 1 Choose **CICD** > **Pipeline** from the navigation pane.

Step 2 Click **Create Pipeline** and configure the pipeline.

1. **Basic Information**: Configure the following information and click **Next**.

Table 3-8 Pipeline basic information

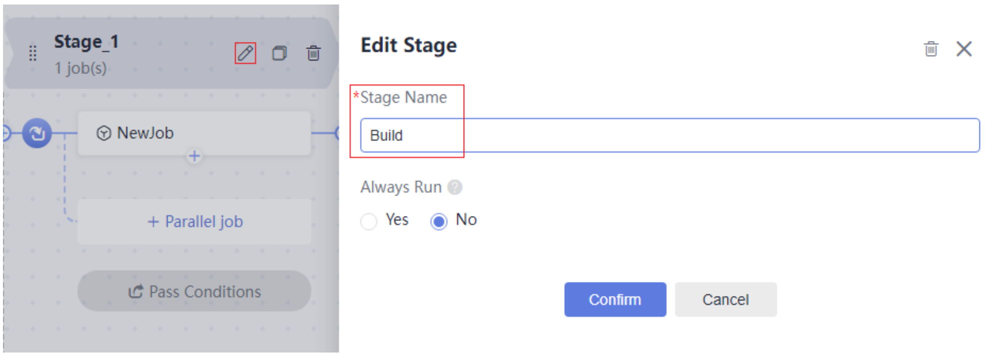
Parameter	Suggestion
Name	Enter **pipeline-web-demo**.
Code Source	Select **Repo**.
Repository	Select **Web-Demo**.
Default Branch	Select **master**.

2. **Template**: Select **Blank Template** and click **OK**.

Step 3 Configure a workflow.

1. Click ✎ next to **Stage_1**. In the **Edit Stage** dialog box, enter the name **Build** and click **Confirm**.

Figure 3-5 Editing the stage name



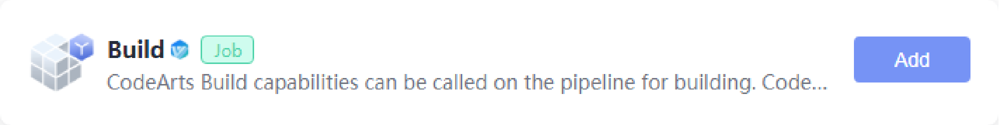
Stage_1
1 job(s)
NewJob
+ Parallel job
Pass Conditions

Edit Stage
*Stage Name: Build
Always Run: ○ Yes ● No
Confirm Cancel

2. Click **Job**.

Click **Add** next to **Build** in the **New Job** window.

Figure 3-6 Adding a job



Build Job
CodeArts Build capabilities can be called on the pipeline for building. Code...
Add

3. Configure task information by referring to the following table and click **OK**.

Table 3-9 Editing a build task

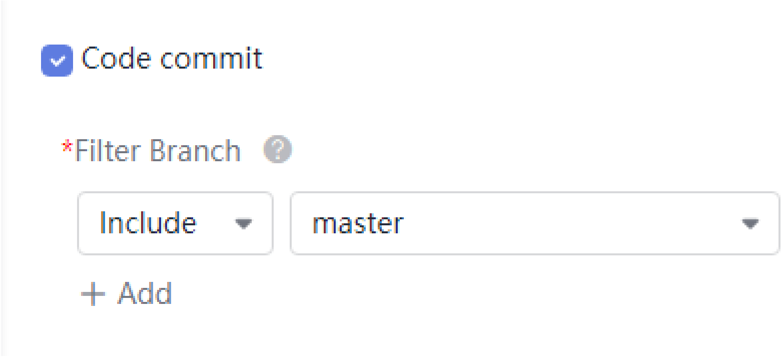
Parameter	Suggestion
Name	Retain the default value.
Select Task	Select **Web-Demo-docker**.

Parameter	Suggestion
Repository	Select **Web-Demo**.

4. Click **Stage** and change the stage name to **Deploy**.
5. Click **Job** and add the **Deploy** extension.
6. Select **web-demo-k8s** and select the job configured in **Step 3.3**.

Step 4 Click the **Execution Plan** tab, select **Code commit**, select **master** from the branch filter drop-down list.

Figure 3-7 Configuring the execution plan



☑ Code commit

*Filter Branch

Include ▾ master ▾

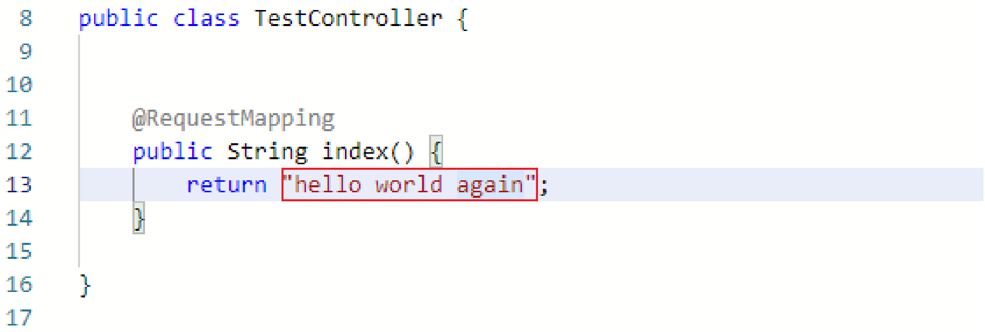
+ Add

Step 5 Click **Save**.

Step 6 Go to the code repository and search for and open the **TestController.java** file.

Click ✎, change **hello world** to **hello world again**, submit the information, and click **OK**.

Figure 3-8 Modifying code



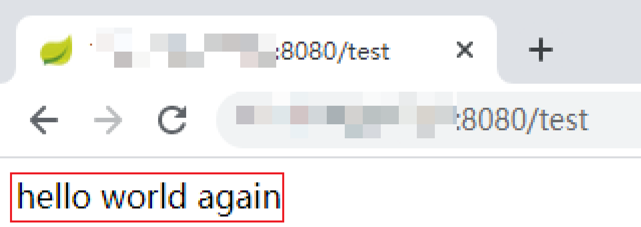
```
public class TestController {


    @RequestMapping
    public String index() {
        return "hello world again";
    }

}

```

Step 7 Return to the Pipeline page. You can see that the pipeline is running.

When ✓ is displayed, access **http://IP:8080/test** again. The following figure shows the access result.

If the task fails to be executed, check the failure cause. You can open the step details page to view the task logs and rectify the fault based on the logs.

Figure 3-9 Pipeline execution result



[redacted]:8080/test

[redacted]:8080/test

hello world again

----End

Follow-up Operations

To avoid incurring unnecessary fees, you can release any unused resources once you have completed this example.

The following resources can be released.

Table 3-10 Releasing resources

Resource	Releasing Method
CodeArts project	Choose **Settings** > **General** > **Basic Information**, click **Delete Project**, and follow the prompts to delete the project.
SWR organization and image	1. Log in to the SWR console. 2. On the **My Images** page, select the image created in this example, click **Delete**, and follow the prompts to delete the image. 3. On the **Organizations** page, click the name of the organization to be deleted. Click **Delete** and follow the prompts to delete the organization.
CCE cluster	Log in to the CCE console. Locate the target cluster in the list, click ···, select **Delete Cluster**, and follow the prompts to delete it.

NOTICE

Released resources cannot be restored. Exercise caution when performing this operation.