CodeArts

Getting Started

Issue 01

Date 2025-07-01

HUAWEI CLOUD COMPUTING TECHNOLOGIES CO., LTD.

Trademarks and Permissions


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.

Contents

1 Guidance

Table 1-1 Service guidance

Service	Guidance
Overall process	• **Setting Up an ECS-based Code Development Pipeline** • **Setting Up a CCE-based Code Development Pipeline**
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 Setting Up an ECS-based Code Development Pipeline

This section describes how to use the built-in code repository of CodeArts to develop, build, and deploy an application.

The deployment is based on Elastic Cloud Server (ECS). This approach is suitable for traditional software packages.

To use container-based deployment, see **Setting Up a CCE-based Code Development Pipeline**.

Procedure

Step	Description
Preparations	Sign up for a HUAWEI ID, enable Huawei Cloud services, and top up your account. Then buy an ECS.
Step 1: Enable CodeArts Free Edition	CodeArts is billed yearly or monthly. Buy a package first before using CodeArts. For the operations in this section, enable the Free Edition instead.
Step 2: Create a Project	Projects are foundational for using CodeArts services. Create a project first before proceeding with subsequent operations.
Step 3: Create a Code Repository	Use CodeArts Repo to create a repository from the built-in template **Java Web Demo**. CodeArts Repo helps you manage your project code by version.
Step 4: Check Code	Use CodeArts Check to examine your code statically to control quality.
Step 5: Build and Archive the Software Package	Use CodeArts Build to compile your source code into object files, package them together with configuration and resource files, and then archive them to the release repository.

Step	Description
Step 6: Deploy the Build Package	Use CodeArts Deploy to deploy software packages in the release repository to a VM and then run the application.
Step 7: Configure a Pipeline	Use CodeArts Pipeline to link code check, build, and deployment tasks for continuous delivery. It runs automatically when you update your code.
Releasing Resources	After completing this practice, delete unused pay-per-use resources to prevent extra charges.

Preparations

- Sign up for a HUAWEI ID and enable Huawei Cloud services.
- Purchase an ECS with the following configurations:
  - **Billing Mode**: Select **Pay-per-use**.
  - **CPU Architecture**: Select **x86**.
  - **Specifications**: 2 vCPUs | 4 GiB or above, system disk ≥ 80 GiB
  - **OS**: **Public image** > **CentOS 7.6**
  - **EIP**: **Auto assign**

  After completing the purchase, add two inbound rules by referring to Configuring Security Group Rules.
  - Protocol: **TCP**; port: **22**; source: **0.0.0.0/0**
  - Protocol: **TCP**; port: **8080**; source: **0.0.0.0/0**

Step 1: Enable CodeArts Free Edition

Step 1 Go to the Buy CodeArts Package page.

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

Check the enabling record on the **CodeArts** page.

----End

Step 2: Create a Project

Step 1 Click **go to Workspace** on the CodeArts console.

Step 2 On the CodeArts homepage, click **Create** > **Create Project**.

Step 3 Select the **Scrum** template.

Step 4 Enter a project name (for example, **Demo**), and click **OK**. Keep the name under 128 characters.

The project is created, and the **Work Items** page is displayed.

----End

Step 3: Create a Code Repository

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

Step 2 Click **Create Repository**.

Step 3 Select **Template** and click **Next**.

Step 4 Select the **Java Web Demo** template and click **Next**.

Step 5 Enter a repository name (for example, **Web-Demo**), and click **OK**. Start the name with a letter, digit, or underscore (_), and use letters, digits, hyphens (-), underscores (_), and periods (.). Do not end the name with **.git**, **.atom**, or a period.

The repository is created, and its code file list is displayed.

----End

Step 4: Check Code

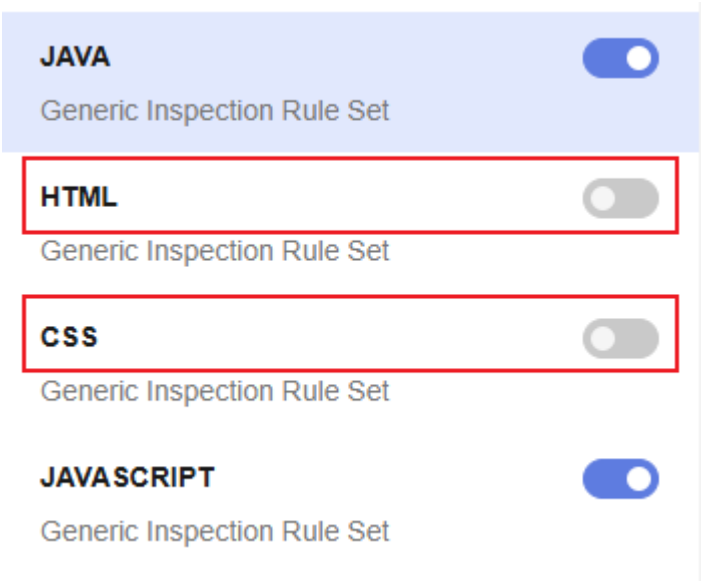
Step 1 In the navigation pane, choose **Code** > **Check**.

The automatically created task **Web-Demo-check** is displayed.

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

Step 3 Choose **Rule Sets**, and toggle off next to **CSS** and **HTML**.

Figure 2-1 Editing a rule set



Step 4 Click **Start Check** to start the task.

If Success is displayed, the task is successful.

If the task fails, rectify the fault by referring to **CodeArts Check FAQs**.

Step 5 Click the **Issues** tab to view the issue list.

Since there are no critical or major issues, you do not need to modify the code.

----End

Step 5: Build and Archive the Software Package

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

The automatically created build task **Web-Demo-build** is displayed.

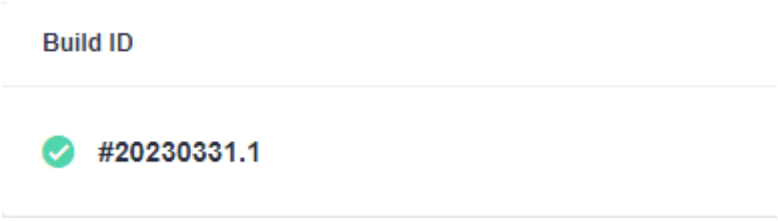
Step 2 Click ▷ in the row where the task is located to start the task. In the displayed dialog box, confirm the parameter settings and click **OK**.

If ✅ is displayed, the task is successful.

If the task fails, rectify the fault based on the failure step and the error message in logs. For details, see **CodeArts Build FAQs**.

Step 3 Click the task name to view its details. On the **Build History** tab, find the latest build ID and record it.

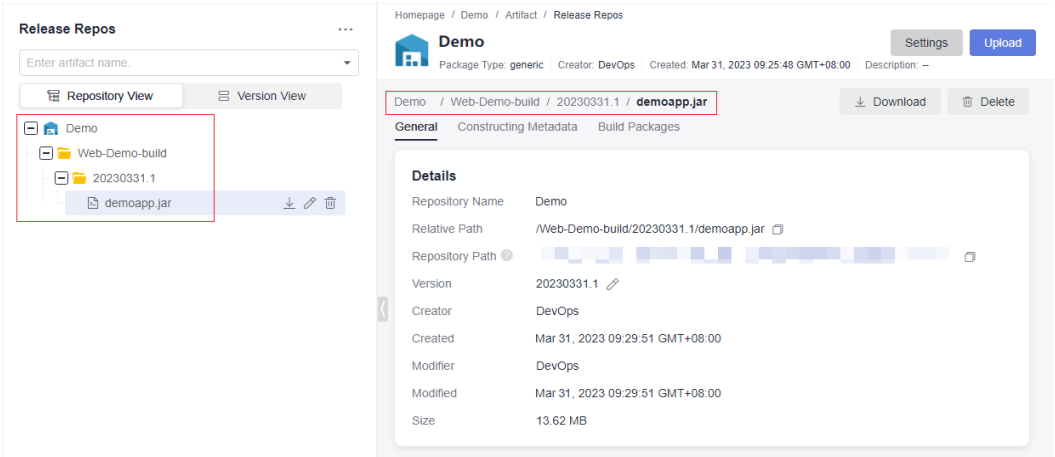
Figure 2-2 Build ID



Step 4 In the navigation pane, choose **Artifact** > **Release Repos**.

In the repository view, find the repository with the same name as your project, and go to the **Web-Demo-build** > *Build ID recorded in **Step 3*** folder to find the generated software package **demoapp.jar**.

Figure 2-3 Viewing the software package



----End

Step 6: Deploy the Build Package

Step 1 Configure the target host.

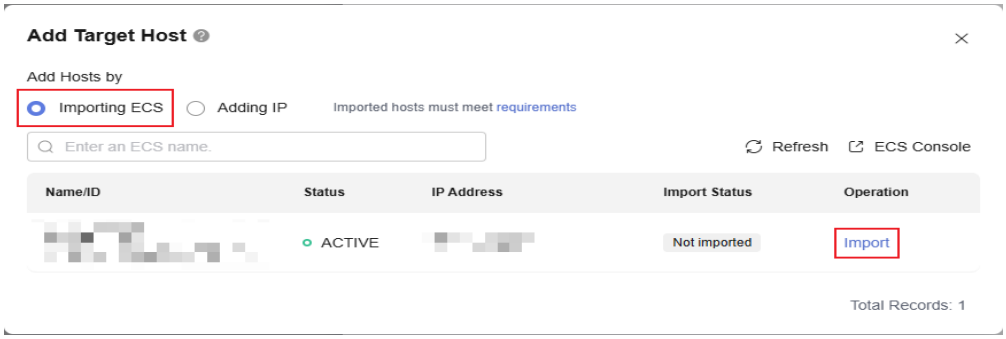
1. In the navigation pane, choose **Settings** > **General** > **Basic Resources**.
2. Click **Create Host Cluster**, configure the following information, and click **Save**.

Table 2-1 Creating a host cluster

Parameter	Example	Description
Cluster Name	host-group	The name of the host cluster to create. Enter 3 to 128 characters, including letters, digits, hyphens (-), underscores (_), and periods (.).
OS	Linux	The OS of the hosts to add to this cluster. Select **Linux** or **Windows**.
Host Connection Mode	Direct Connection	The way your target hosts will connect to CodeArts Deploy. Select **Direct Connection** or **Proxy**.
Execution Resource Pool	Official	A resource pool (or agent pool) is a collection of physical environments where commands are executed to deploy software packages. Choose the official agent pool or a self-hosted agent pool that contains your own servers.

3. After the system displays a message indicating that the host cluster is created, click **Add Host** on the **Target Hosts** tab, select **Importing ECS**, locate the ECS purchased in **Preparations**, and click **Import**.

Figure 2-4 Importing an ECS



4. Configure the following information and click **OK**.

Table 2-2 Adding a host

Parameter	Example	Description
Username	root	The username for logging in to the ECS. By default, it is **root** for a Linux ECS.
Password	Enter the password set when you purchase the ECS in **Preparations**.	The password for logging in to the ECS.
SSH Port	22	The default port is **22**. You can also use another one.

5. Check the host record. If the **Verification Result** column shows successful, the host is added.

   If the host fails to be added, rectify the fault based on the failure details. For details, see **Host Management FAQs**.

Step 2 Choose **CICD** > **Deploy** from the navigation pane.

The automatically created application **Web-Demo-deploy** is displayed.

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 2-3 Creating an environment

Parameter	Example	Description
Environment	demo-env	The name of the environment to create. Enter 3 to 128 characters, including letters, digits, hyphens (-), underscores (_), and periods (.).
Resource Type	Host	The resource type in the environment. The default value is **Host**.
OS	Linux	The OS of the hosts to add to this environment. Select **Linux** or **Windows**.

2. When the system displays a message indicating that the creation is successful, click **Import Host** on the **Resources** tab. In the displayed dialog box, select the host cluster and host configured in **Step 1** and click **Import**.
3. When the system shows a success message for the import, close the window.

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

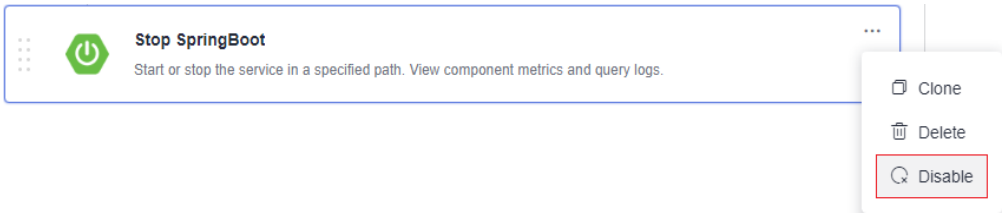
- **Install JDK**: Check that the JDK version is **openjdk-1.8.0**.
- **Select a Deployment Source**: Set the parameters based on the following table.

Table 2-4 Deployment source configuration

Parameter	Example	Description
Source	Build task	The source of the software package to deploy. Select **Artifact** or **Build task**.
Build Task	Web-Demo-build	Available only when **Source** is set to **Build task**.
Download Path	/usr/local/${package_name}/	The path on your target host for saving the software package.

- **Stop SpringBoot**: When you run deployments for the first time, this action will fail because Spring Boot has not yet run on the target host. Disable this action by clicking ··· on the action card and choosing **Disable**.

Figure 2-5 Disabling the "Stop SpringBoot" action



- **Start SpringBoot**: Retain the default settings.
- **Health Test Through URLs**: This action is optional. Disable it for this example.

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

Table 2-5 Configuration parameters

Name	Default Value
host_group	The environment **demo-env** added in **Step 4**
package_url	Not required for this example. Click in the same row to delete it.
service_port	**8080**
package_name	**demoapp**

Step 7 Click **Save & Deploy**. In the displayed dialog box, 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 failure step and the error message in logs. For details, see **CodeArts Deploy FAQs**.

Step 8 View the deployment result.

Open a new browser page and enter **http://*IP*:8080/test**. *IP* indicates the EIP of the ECS purchased in **Preparations**.

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

Figure 2-6 Deployment result

----End

Step 7: Configure a Pipeline

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

On the **Pipelines** tab, the automatically created pipeline **Web-Demo-pipeline** is displayed.

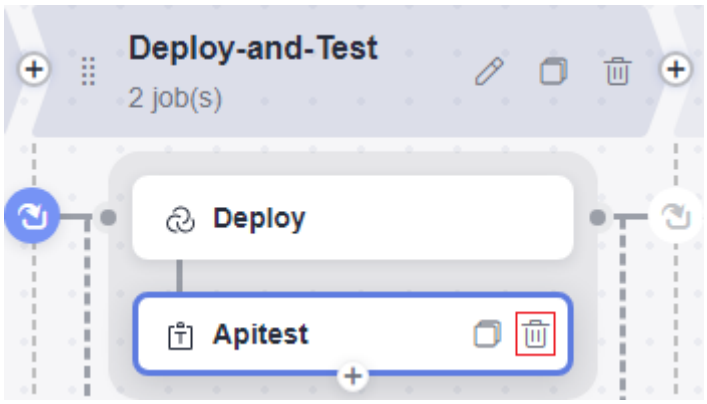
Step 2 Click ··· and choose **Edit**.

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

1. API testing is not involved in this example. So remove the API test task from the pipeline.

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

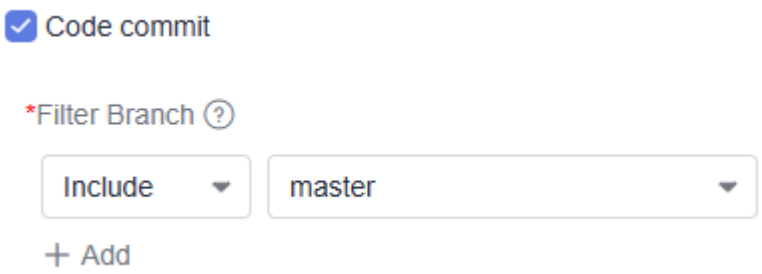
   Figure 2-7 Deleting a job

2. Click the **Deploy** job, select the build task **Build**, and keep other parameters the same as those set in **Step 6: Deploy the Build Package**.

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

Figure 2-8 Configuring the execution plan



Step 5 Click **Save**.

The updated execution plan is displayed.

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

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

Click , change **hello world** to **hello world again**, enter a commit message, and click **OK**.

Figure 2-9 Modifying code

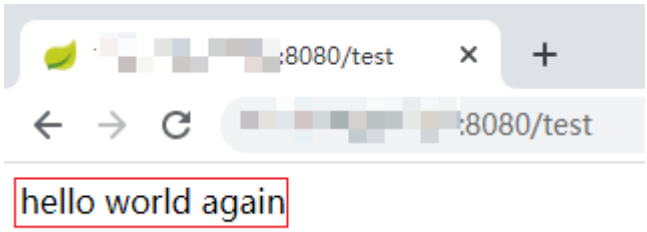
```
8   public class TestController {
9
10
11      @RequestMapping
12      public String index() {
13          return "hello world again";
14      }
15
16  }
17
```

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 pipeline fails, click the cause to view logs. Then rectify the fault by referring to **CodeArts Pipeline FAQs**.

Figure 2-10 Pipeline execution result



----End

Releasing Resources

WARNING

Released resources cannot be recovered. Exercise caution when performing these operations.

In this example, the pay-per-use resources involved are from ECS.

If you do not need the ECS after the trial, **delete the ECS** to release its resources.

Helpful Links

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 2-6 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 Setting Up a CCE-based Code Development Pipeline

This section describes how to use the built-in code repository of CodeArts to develop, build, and deploy an application.

This chapter uses Cloud Container Engine (CCE) for container-based deployment. To use traditional software package deployment, see **Setting Up an ECS-based Code Development Pipeline**.

Procedure

Step	Description
Preparations	Sign up for a HUAWEI ID and enable Huawei Cloud services. Top up your account. Buy a Cloud Container Engine (CCE) cluster and create a SoftWare Repository for Container (SWR) organization.
Step 1: Enable CodeArts Free Edition	CodeArts is billed yearly or monthly. Buy a package first before using CodeArts. For the operations in this section, enable the Free Edition instead.
Step 2: Create a Project	Projects are foundational for using CodeArts services. Create a project first before proceeding with subsequent operations.
Step 3: Create a Code Repository	Use CodeArts Repo to create a repository from the built-in template **Java Web Demo**. CodeArts Repo helps you manage your project code by version.
Step 4: Prepare a Dockerfile	Create a Dockerfile. It 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 5: Build an Image and Push It to SWR	Run a build task to compile the software source code into an image and archive the image to SWR.

Step	Description
Step 6: Create a Workload	Create a Deployment in CCE to load and run the demo image.
Step 7: Deploy the Image	Create an application in CodeArts Deploy to automate image deployment.
Step 8: Configure a Pipeline	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.
Releasing Resources	After completing this practice, delete unused pay-per-use resources to prevent extra charges.

Preparations

- **Sign up for a HUAWEI ID and enable Huawei Cloud services**.
- Buy a CCE cluster that meets the requirements listed in the following table.

  Table 3-1 Cluster configuration requirements

Category	Configuration	Reference
Cluster	**Pay-per-use** recommended • **Type**: **CCE Standard Cluster** • **Cluster Version**: Select the latest version. • **Controller Node Architecture**: **X86** • **Network Model**: **VPC network** • **Container CIDR Block**: **Auto select**	**Buying a CCE Standard/Turbo Cluster**
Node	**Pay-per-use** recommended • **Node Type**: **Elastic Cloud Server (VM)** • **Specifications**: **2 vCPUs | 8 GiB** or above • **Container Engine**: **Docker** • **OS**: **Public image** > **CentOS 7.6**	**Creating a Node**

- Create an organization named **web-demo** in SWR. If this name is already in use, enter another. For details, see **Creating an Organization**.

Step 1: Enable CodeArts Free Edition

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

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

Check the enabling record on the **CodeArts** page.

----End

Step 2: Create a Project

Step 1 Click **go to Workspace** on the CodeArts console.

Step 2 On the CodeArts homepage, click **Create** > **Create Project**.

Step 3 Select the **Scrum** template.

Step 4 Enter a project name (for example, **Demo**), and click **OK**. Keep the name under 128 characters.

The project is created, and the **Work Items** page is displayed.

----End

Step 3: Create a Code Repository

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

Step 2 Click **Create Repository**.

Step 3 Select **Template** and click **Next**.

Step 4 Select the **Java Web Demo** template and click **Next**.

Step 5 Enter a repository name (for example, **Web-Demo**), and click **OK**. Start the name with a letter, digit, or underscore (_), and use letters, digits, hyphens (-), underscores (_), and periods (.). Do not end the name with **.git**, **.atom**, or a period.

The repository is created, and its code file list is displayed.

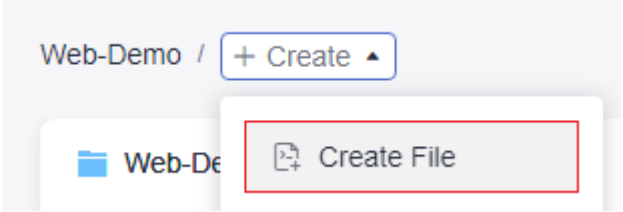
----End

Step 4: Prepare a Dockerfile

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

Step 5: Build an Image and Push It to SWR

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

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

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

   Table 3-2 Basic information

Parameter	Example	Description
Name	Web-Demo-docker	Build task name. Use a maximum of 115 characters, including letters, digits, underscores (_), and hyphens (-).
Code Source	Repo	Select **Repo**, **GitHub**, or other sources.
Repository	Web-Demo	The code repository to compile.
Default Branch	master	The repository branch to compile.

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**, and 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 3-3 Configuring image information

Parameter	Example	Description
Organization	Name of the organization created in **Preparations**	The organization to which the image will belong after being pushed to SWR.

Parameter	Example	Description
Image Tag	v1.0.0	Image version. Use a maximum of 128 characters, including letters, digits, periods (.), underscores (_), and hyphens (-). Do not start with a period or hyphen.

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

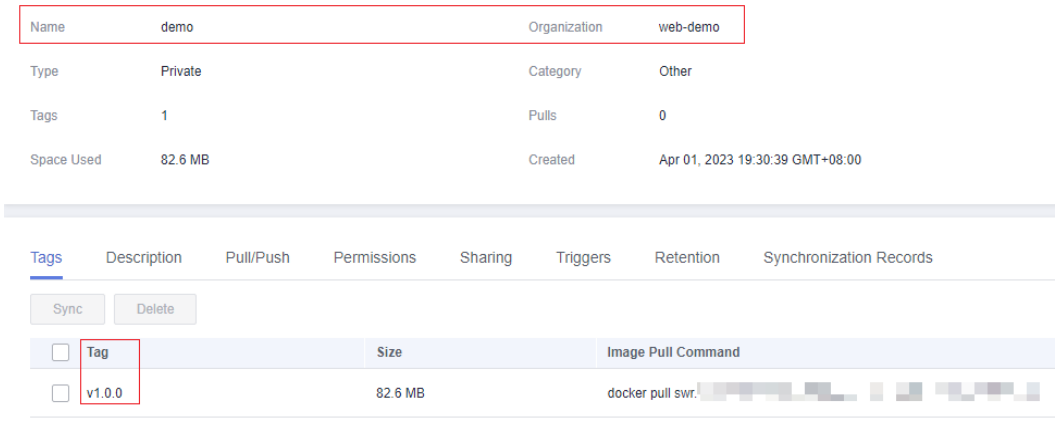
If is displayed, the task is successful. If the task fails, rectify the fault based on the failure step and the error message in logs. For details, see **CodeArts Build FAQs**.

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



----End

Step 6: Create a Workload

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**.

For details about the parameters, see **Creating a Deployment**.

Table 3-4 Creating workload

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

Step 4 When the creation is complete, click **View Workload Details** to go back to the details page. A record is displayed 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**.

For details about the parameters, see **Creating a LoadBalancer Service**.

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

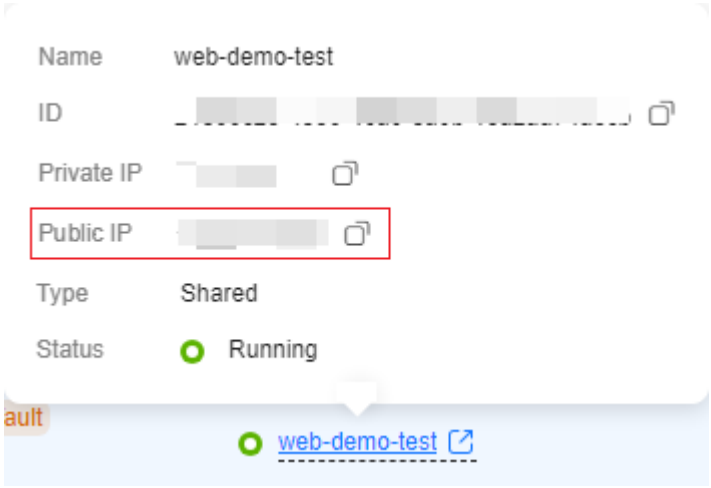
Table 3-5 Configuring access mode

Parameter	Example
Service Name	web-demo
Service Type	LoadBalancer
Service Affinity	Cluster level
Load Balancer	1. Choose **Shared** > **Auto create**. 2. Configure the following parameters: – **Instance Name**: **web-demo-test** – **EIP**: **Auto create**
Port	• **Protocol**: **TCP** • **Container Port**: **8080** • **Service Port**: **8080**

After the creation is successful, a new record is displayed in the list.

Step 5 Refresh the page, and check the access mode list. When web-demo-test is displayed, hover over the load balancer name in the **Service Type** column, and copy the public IP address in the pop-up window.

Figure 3-3 Copying the access address



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

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

Figure 3-4 Deployment result

----End

Step 7: Deploy the Image

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

1. Click **Create Application**, enter an application name (for example, **web-demo-k8s**), and click **Next**. The name can contain 3–128 characters, including letters, digits, hyphens (-), and underscores (_).
2. Select **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 3-6 Configuring deployment actions

Parameter	Example	Description
Region	The region where the cluster purchased in **Preparations** is located	The region of the target cluster.
Cluster Name	The name of the cluster purchased in **Preparations**.	The name of the target cluster.
Namespace	default	The namespace of the target cluster.
Workload	web-demo	The workload to deploy.
Container	The container name displayed in the **Container Settings** area involved in **Step 6: Create a Workload**	The name of the container to deploy the workload in.

Step 3 Click **Save & Deploy**.

If Successful is displayed, the test is successful. If the deployment fails, rectify the fault based on the failure step and the error message in logs. For details, see **CodeArts Deploy FAQs**.

----End

Step 8: Configure a Pipeline

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 3-7 Pipeline basic information

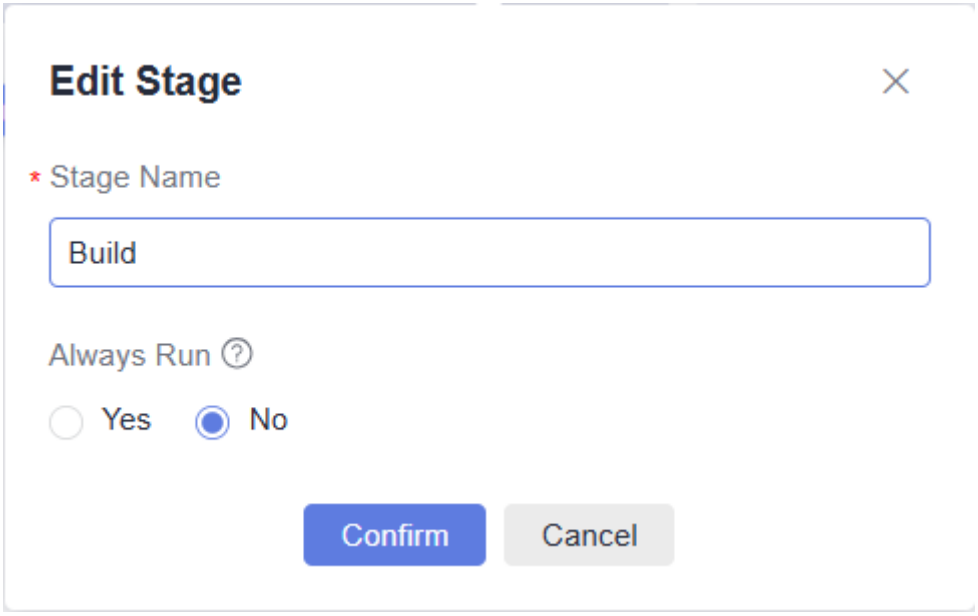
Parameter	Example	Description
Name	pipeline-web-demo	Pipeline name. Use a maximum of 128 characters, including letters, digits, hyphens (-), and underscores (_).
Pipeline Source	Repo	The code source of the pipeline. Select **Repo**, **Git**, or other sources.
Repository	Web-Demo	The code repository to be associated with the pipeline.
Default Branch	master	The repository branch to be associated with the pipeline.

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

Step 3 Configure the workflow.

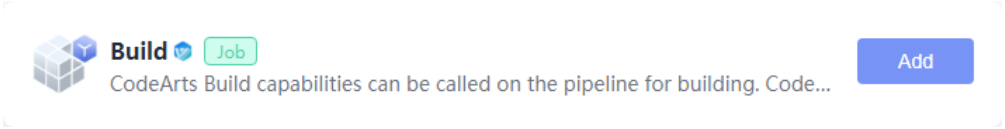
1. Click next to **Stage_1**. In the **Edit Stage** dialog box, enter a stage name and click **Confirm**. The name can contain 1–128 characters, including letters, digits, spaces, and special characters (-_,;:./()). Do not start or end with a space.

Figure 3-5 Editing the stage name



2. Click **Parallel Job**, and select **From empty**. The job creation window is displayed on the right.
3. Locate **Build** in the list and click **Add**.

Figure 3-6 Adding a job



4. Configure the job information by referring to the following table and click **OK**.

Table 3-8 Editing a build job

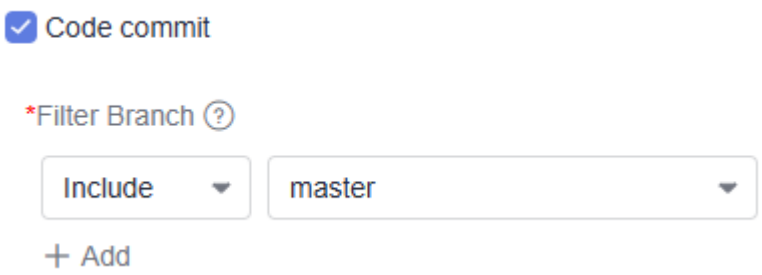
Parameter	Example	Description
Name	Retain the default value.	Job name. Use 1–128 characters, including letters, digits, spaces, and special characters (-_,;:./()). Do not start or end the name with a space.

Parameter	Example	Description
Select Task	Web-Demo-docker	Select a build task whose code source is pipeline or the same repository as the pipeline you are configuring.
Repository	Web-Demo	Select the code repository associated with the build task.

5. Click **Stage** and change the stage name to **Deploy**. The new stage is displayed.
6. Click **New Job** and add the **Deploy** extension.
7. Select the **web-demo-k8s** task (or application), associate the build task set in **Step 3.4**, and click **OK**.

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

Figure 3-7 Configuring the execution plan



Step 5 Click **Save**.

The updated execution plan is displayed.

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

Click , change **hello world** to **hello world again**, enter a commit message, and click **OK**.

Figure 3-8 Modifying code

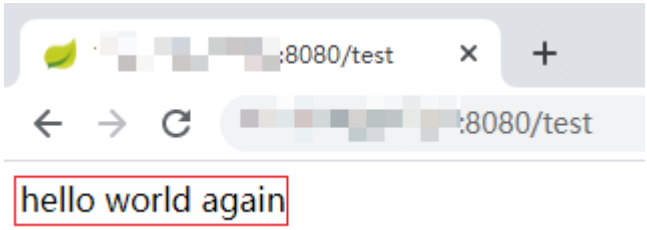
```
8   public class TestController {
9
10
11      @RequestMapping
12      public String index() {
13          return "hello world again";
14      }
15
16  }
17
```


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

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

If the pipeline fails, click the cause to view logs. Then rectify the fault by referring to **CodeArts Pipeline FAQs**.

Figure 3-9 Pipeline execution result



----End

Releasing Resources

WARNING

Released resources cannot be recovered. Exercise caution when performing these operations.

In this example, the pay-per-use resources involved are from CCE.

If you do not need CCE after the trial, **delete the pay-per-use cluster** to release its resources.

