Installing Red Hat CodeReady Workspaces 2.1

Supriya Takkhi
Robert Kratky
rkratky@redhat.com

Michal Maléř
mmaler@redhat.com

Fabrice Flore-Thébault
ffloreth@redhat.com

Yana Hontyk
yhontyk@redhat.com
Abstract

Information for administrators installing Red Hat CodeReady Workspaces.
Table of Contents

CHAPTER 1. INSTALLING CODEREADY WORKSPACES ON OPENSIFT CONTAINER PLATFORM 4  ..... 4
1.1. INSTALLING CODEREADY WORKSPACES ON OPENSIFT 4 FROM OPERATORHUB  4
   1.1.1. Creating the CodeReady Workspaces project in OpenShift 4 web console  4
      Prerequisites  4
      Procedure  5
   1.1.2. Installing the CodeReady Workspaces Operator in OpenShift 4 web console  5
      Prerequisites  5
      Procedure  5
   1.1.3. Installing CodeReady Workspaces using the CodeReady Workspaces Operator in OpenShift 4 web console  6
      Prerequisites  6
      Procedure  6
   1.1.4. Viewing the state of the CodeReady Workspaces instance deployment in OpenShift 4 web console  7
   1.1.5. Finding CodeReady Workspaces instance URL in OpenShift 4 web console  8
   1.1.6. Viewing the state of the CodeReady Workspaces cluster deployment using OpenShift 4 CLI tools  8
   1.1.7. Finding CodeReady Workspaces cluster URL using the OpenShift 4 CLI  10
   1.1.8. Enabling SSL on OpenShift 4  10
      Prerequisites  10
      Procedure  10
   1.1.9. Logging in to CodeReady Workspaces on OpenShift for the first time using OAuth  11
      Prerequisites  11
      Procedure  11
   1.1.10. Logging in to CodeReady Workspaces on OpenShift for the first time registering as a new user  12
      Prerequisites  12
      Procedure  12
1.2. INSTALLING CODEREADY WORKSPACES USING CLI MANAGEMENT TOOL  12
   1.2.1. Installing the crwctl CLI management tool  12
   1.2.2. Installing CodeReady Workspaces using CodeReady Workspaces CLI management tool  13
      1.2.2.1. Installing with default settings  13
      1.2.2.2. Installing with custom settings  13

CHAPTER 2. INSTALLING CODEREADY WORKSPACES ON OPENSIFT 3 USING THE OPERATOR  ..... 15
2.1. PREPARING OPENSIFT 3 FOR INSTALLING CODEREADY WORKSPACES  15
2.2. INSTALLING CODEREADY WORKSPACES ON OPENSIFT 3 USING THE OPERATOR  16
2.3. INSTALLING CODEREADY WORKSPACES ON OPENSIFT 3 USING THE OPERATOR AND SSL  17

CHAPTER 3. INSTALLING CODEREADY WORKSPACES IN A RESTRICTED ENVIRONMENT  ............. 18
3.1. INSTALLING CODEREADY WORKSPACES IN A RESTRICTED ENVIRONMENT USING OPERATORHUB  18
3.2. INSTALLING CODEREADY WORKSPACES IN A RESTRICTED ENVIRONMENT USING CLI MANAGEMENT TOOL  19
   3.2.1. Preparing an image registry for installing CodeReady Workspaces in a restricted environment  19
   3.2.2. Preparing CodeReady Workspaces Custom Resource for restricted environment  21
      3.2.2.1. Downloading the default CheCluster Custom Resource  22
      3.2.2.2. Customizing the CheCluster Custom Resource for restricted environment  22
   3.2.3. Starting CodeReady Workspaces installation in a restricted environment using CodeReady Workspaces CLI management tool  23
3.3. PREPARING CODEREADY WORKSPACES CUSTOM RESOURCE FOR INSTALLING BEHIND A PROXY  23

CHAPTER 4. UPGRADING CODEREADY WORKSPACES  ............................................. 25
4.1. UPGRADING CODEREADY WORKSPACES USING OPERATORHUB  25
4.2. UPGRADING CODEREADY WORKSPACES USING CLI MANAGEMENT TOOL ON OPENSIFT 3  25
4.3. UPGRADING CODEREADY WORKSPACES FROM PREVIOUS MAJOR VERSION  26
CHAPTER 5. ADVANCED CONFIGURATION OPTIONS ........................................... 27
  5.1. CODEREADY WORKSPACES CONFIGMAPS AND THEIR BEHAVIOR 27
    5.1.1. CodeReady Workspaces installed using an Operator 27
  5.2. CONFIGURING NAMESPACE STRATEGIES 28
    5.2.1. One namespace per workspace strategy 28
    5.2.2. One namespace for all workspaces strategy 28
    5.2.3. One namespace per user strategy 29
    5.2.4. Allowing user-defined workspace namespaces 29
  5.3. DEPLOYING CODEREADY WORKSPACES WITH SUPPORT FOR GIT REPOSITORIES WITH SELF-SIGNED CERTIFICATES 30
  5.4. ADDING CUSTOM PUBLIC SSL CERTIFICATES TO CODEREADY WORKSPACES TRUST-STORE 30
  5.5. CODEREADY WORKSPACES CONFIGMAPS FIELDS REFERENCE 31
    5.5.1. server settings related to the CodeReady Workspaces server 31
    5.5.2. database configuration settings related to the database used by CodeReady Workspaces 34
    5.5.3. auth configuration settings related to authentication used by CodeReady Workspaces installation 35
    5.5.4. storage configuration settings related to persistent storage used by CodeReady Workspaces 37
    5.5.5. k8s configuration settings specific to CodeReady Workspaces installations on OpenShift 37
    5.5.6. installation defines the observed state of CodeReady Workspaces installation 38
    5.5.7. Limits for workspaces 39
    5.5.8. Limits for the workspaces of an user 39
    5.5.9. Limits for the workspaces of an organization 39

CHAPTER 6. UNINSTALLING CODEREADY WORKSPACES ................................ 41
  6.1. UNINSTALLING CODEREADY WORKSPACES AFTER OPERATORHUB INSTALLATION 41
    6.1.1. Uninstalling CodeReady Workspaces using the OpenShift web console 41
    6.1.2. Uninstalling CodeReady Workspaces using oc commands 42
  6.2. UNINSTALLING CODEREADY WORKSPACES AFTER CRWCTL INSTALLATION 43
1. INSTALLING CODEREADY WORKSPACES ON OPENSHIFT CONTAINER PLATFORM 4

1.1. INSTALLING CODEREADY WORKSPACES ON OPENSHIFT 4 FROM OPERATORHUB

Operators are a method of packaging, deploying, and managing a OpenShift application which also provide the following:

- Repeatability of installation and upgrade.
- Constant health checks of every system component.
- Over-the-air (OTA) updates for OpenShift components and ISV content.
- A place to encapsulate knowledge from field engineers and spread it to all users.

On OpenShift, Red Hat CodeReady Workspaces can be installed using the OperatorHub Catalog present in the OpenShift web console.

Following steps are described:

- Section 1.1.1, “Creating the CodeReady Workspaces project in OpenShift 4 web console”.
- Section 1.1.2, “Installing the CodeReady Workspaces Operator in OpenShift 4 web console”.
- Section 1.1.3, “Installing CodeReady Workspaces using the CodeReady Workspaces Operator in OpenShift 4 web console”.
- Section 1.1.4, “Viewing the state of the CodeReady Workspaces instance deployment in OpenShift 4 web console”.
- Section 1.1.6, “Viewing the state of the CodeReady Workspaces cluster deployment using OpenShift 4 CLI tools”.
- Section 1.1.5, “Finding CodeReady Workspaces instance URL in OpenShift 4 web console”.
- Section 1.1.7, “Finding CodeReady Workspaces cluster URL using the OpenShift 4 CLI”.
- Section 1.1.8, “Enabling SSL on OpenShift 4”.
- Section 1.1.9, “Logging in to CodeReady Workspaces on OpenShift for the first time using OAuth”.
- Section 1.1.10, “Logging in to CodeReady Workspaces on OpenShift for the first time registering as a new user”.

1.1.1. Creating the CodeReady Workspaces project in OpenShift 4 web console

This section describes how to create the CodeReady Workspaces project in OpenShift 4 web console.

**Prerequisites**

- An administrator account on a running instance of OpenShift 4.
1. Open the OpenShift web console.

2. In the left panel, navigate to Projects.

3. Click the Create Project button.

4. Enter the project details:
   - In the Name field, type codeready.
   - In the Display Name field, type Red Hat CodeReady Workspaces.

5. Click the Create button.

1.1.2. Installing the CodeReady Workspaces Operator in OpenShift 4 web console

This section describes how to install the CodeReady Workspaces Operator in OpenShift 4 web console.

**Prerequisites**

- An administrator account on a running instance of OpenShift 4.
- Administrative rights on an existing project named codeready on this instance of OpenShift 4. See Section 1.1.1, “Creating the CodeReady Workspaces project in OpenShift 4 web console”.
- The Red Hat CodeReady Workspaces 2.0 Operator is not installed.

**Procedure**

1. Open the OpenShift web console.

2. In the left panel, navigate to the Operators → OperatorHub section.

3. In the Search by keyword field, type Red Hat CodeReady Workspaces.

4. Click on the Red Hat CodeReady Workspaces tile.

5. Click the Install button in the Red Hat CodeReady Workspaces pop-up window.

6. In the A specific namespace on the cluster field, in the cluster drop-down list, select the namespace into which the previous version of the CodeReady Workspaces Operator was installed.

7. Click the Subscribe button.

8. In the left panel, navigate to the Operators → Installed Operators section.

9. Red Hat CodeReady Workspaces is displayed as an installed Operator having the InstallSucceeded status.

10. Click on the Red Hat CodeReady Workspaces name in the list of installed operators.

11. Navigate to the Overview tab.

12. In the Conditions sections at the bottom of the page, wait for this message: install strategy completed with no errors.
13. Navigate to the Events tab.

14. Wait for this message: **install strategy completed with no errors.**

### 1.1.3. Installing CodeReady Workspaces using the CodeReady Workspaces Operator in OpenShift 4 web console

This section describes how to install CodeReady Workspaces using the CodeReady Workspaces Operator in OpenShift 4 web console.

**Prerequisites**

- An administrator account on a running instance of OpenShift 4.
- At least one OAuth user provisioned on this instance of OpenShift 4.
- The CodeReady Workspaces Operator is installed on this instance of OpenShift 4. See Section 1.1.2, “Installing the CodeReady Workspaces Operator in OpenShift 4 web console”

**Procedure**

1. Open the OpenShift web console.

2. Navigate to the Operators → Installed Operators section.

3. Click **Red Hat CodeReady Workspaces** in the list of installed operators.

4. Click the **Create Instance** link in Provided APIs section.

5. The **Create CodeReady Workspaces Cluster** page is displayed.

6. Leave the default values as they are.

7. Click the **Create** button in the bottom-left corner of the window.

8. The **codeready** cluster is created.
1.1.4. Viewing the state of the CodeReady Workspaces instance deployment in OpenShift 4 web console

This section describes how to view the state of the CodeReady Workspaces instance deployment in OpenShift 4 web console.

Prerequisites

- An administrator account on a running instance of OpenShift 4.
- A CodeReady Workspaces is being deployed on this instance of OpenShift 4.

Procedure

1. Open the OpenShift web console.

2. Navigate to the Operators → Installed Operators section.

3. Click Red Hat CodeReady Workspaces in the list of installed operators.


5. Click codeready-workspaces CheCluster in the table. The Overview tab is displayed.

6. Watch the content of the Message field. The field contain error messages, if any. The expected content is None.

7. Navigate to the Resources tab. The screen displays the state of the resources assigned to the CodeReady Workspaces deployment.
Finding CodeReady Workspaces instance URL in OpenShift 4 web console

This section describes how to find the CodeReady Workspaces instance URL in OpenShift 4 web console.

Prerequisites

- A running Red Hat CodeReady Workspaces instance.
  - See Section 1.1.4, “Viewing the state of the CodeReady Workspaces instance deployment in OpenShift 4 web console”.
  - See Section 1.1.6, “Viewing the state of the CodeReady Workspaces cluster deployment using OpenShift 4 CLI tools”.

Procedure

1. Open the OpenShift web console.

2. In the left panel, navigate to the Operators → Installed Operators section.

3. Click the Red Hat CodeReady Workspaces Operator tile.

4. Click codeready-workspaces CheCluster in the table. The Overview tab is displayed.

5. Read the value of the CodeReady Workspaces URL field.

Viewing the state of the CodeReady Workspaces cluster deployment using OpenShift 4 CLI tools

This section describes how to view the state of the CodeReady Workspaces cluster deployment using Red Hat CodeReady Workspaces 2.1 Installation Guide.
This section describes how to view the state of the CodeReady Workspaces cluster deployment using OpenShift 4 CLI tools.

**Prerequisites**


**Procedure**

1. Run the following commands to select the `crw` project:

   ```bash
   $ oc project <project_name>
   ```

2. Run the following commands to get the name and status of the Pods running in the selected project:

   ```bash
   $ oc get pods
   ```

3. Check that the status of all the Pods is **Running**.

<table>
<thead>
<tr>
<th>NAME</th>
<th>READY</th>
<th>STATUS</th>
<th>RESTARTS</th>
<th>AGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>codeready-8495f4946b-jrzdc</td>
<td>0/1</td>
<td>Running</td>
<td>0</td>
<td>86s</td>
</tr>
<tr>
<td>codeready-operator-578765d954-99szc</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>42m</td>
</tr>
<tr>
<td>keycloak-74f6b9654-g9vp5</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>4m32s</td>
</tr>
<tr>
<td>postgres-5d579c6847-w6wx5</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>5m14s</td>
</tr>
</tbody>
</table>

4. To see the state of the CodeReady Workspaces cluster deployment, run:

   ```bash
   $ oc logs --tail=10 -f `oc get pods -o name | grep operator`
   ```

   **Example output of the command:**

   ```
   time="2020-02-03T12:37:24Z" level=info msg="Deployment 'codeready' successfully scaled to 1"
   time="2020-02-03T12:37:24Z" level=info msg="Updating codeready-workspaces CR with Keycloak URL status: http://keycloak-workspaces.apps-crc.testing"
   time="2020-02-03T12:37:24Z" level=info msg="Custom resource codeready-workspaces updated"
   time="2020-02-03T12:37:24Z" level=info msg="Updating codeready-workspaces CR with status: CodeReady Workspaces server: Available"
   time="2020-02-03T12:37:24Z" level=info msg="Custom resource codeready-workspaces updated"
   time="2020-02-03T12:37:24Z" level=info msg="Updating codeready-workspaces CR with CodeReady Workspaces server URL: http://codeready-workspaces.apps-crc.testing"
   time="2020-02-03T12:37:24Z" level=info msg="Custom resource codeready-workspaces updated"
   time="2020-02-03T12:37:24Z" level=info msg="CodeReady Workspaces is now available at: http://codeready-workspaces.apps-crc.testing""
   time="2020-02-03T12:37:24Z" level=info msg="Updating codeready-workspaces CR with version: 2.0"
   time="2020-02-03T12:37:24Z" level=info msg="Custom resource codeready-workspaces updated"
   ```
1.1.7. Finding CodeReady Workspaces cluster URL using the OpenShift 4 CLI

This section describes how to obtain the CodeReady Workspaces cluster URL using the OpenShift 4 CLI (command line interface). The URL can be retrieved from the OpenShift logs or from the `checluster` Custom Resource. **Prerequisites**

* CodeReady Workspaces is deployed in an OpenShift cluster. To verify the state of the deployment, see Section 1.1.6, “Viewing the state of the CodeReady Workspaces cluster deployment using OpenShift 4 CLI tools”. * User is located in a CodeReady Workspaces installation namespace.

**Procedure**

1. To retrieve a CodeReady Workspaces cluster URL from the `checluster` CR (Custom Resource), run:

   ```bash
   $ oc get checluster --output jsonpath='{.items[0].status.cheURL}'
   ```

   The CodeReady Workspaces cluster URL is displayed.

   Alternatively, wait for the deployment logs as mentioned in the Prerequisites section to display the message: **Red Hat CodeReady Workspaces is now available at:** followed by the CodeReady Workspaces cluster URL:

   ```text
   time="2020-02-04T08:47:13Z" level=info msg="Custom resource codeready-workspaces updated"
   time="2020-02-04T08:47:13Z" level=info msg="Updating codeready-workspaces CR with CodeReady Workspaces server URL: http://codeready-workspaces.apps-crc.testing"
   time="2020-02-04T08:47:13Z" level=info msg="Custom resource codeready-workspaces updated"
   time="2020-02-04T08:47:13Z" level=info msg="CodeReady Workspaces is now available at: http://codeready-workspaces.apps-crc.testing"
   ```

2. After the deployment is finished, in the log output, click the CodeReady Workspaces cluster URL to start the CodeReady Workspaces instance, or use the following command to isolate it:

   ```bash
   $ oc logs --tail=10 `oc get pods -o name | grep operator` | grep "available at" | awk -F'available at: ' '{print $2}' | sed 's/"//'
   ```

1.1.8. Enabling SSL on OpenShift 4

**Prerequisites**

* A running Red Hat CodeReady Workspaces cluster.

  - See Section 1.1.4, “Viewing the state of the CodeReady Workspaces instance deployment in OpenShift 4 web console”.

  - See Section 1.1.6, “Viewing the state of the CodeReady Workspaces cluster deployment using OpenShift 4 CLI tools”.

**Procedure**

1. Open the OpenShift web console.

2. In the left panel, navigate to the **Operators → Installed Operators** section.
3. Click on the Red Hat CodeReady Workspaces Operator tile.

4. Click on eclipse-che in the table.

5. Navigate to the Overview tab.

6. Toggle the TLS MODE switch to True.

7. Click Confirm change.

8. Navigate to the Resources tab.

9. Wait that the Pods are restarted.

10. Navigate to the Overview tab.

11. Click the Red Hat CodeReady Workspaces URL link.

12. Notice that the link is redirected to HTTPS.


1.1.9. Logging in to CodeReady Workspaces on OpenShift for the first time using OAuth

This section describes how to log in to CodeReady Workspaces on OpenShift for the first time using OAuth.

Prerequisites

- A running Red Hat CodeReady Workspaces cluster.
  - See Section 1.1.4, “Viewing the state of the CodeReady Workspaces instance deployment in OpenShift 4 web console”.
  - See Section 1.1.6, “Viewing the state of the CodeReady Workspaces cluster deployment using OpenShift 4 CLI tools”.

- Know the Red Hat CodeReady Workspaces URL.
  - See Section 1.1.5, “Finding CodeReady Workspaces instance URL in OpenShift 4 web console”.
  - See Section 1.1.7, “Finding CodeReady Workspaces cluster URL using the OpenShift 4 CLI”.

Procedure

1. Navigate to the Red Hat CodeReady Workspaces URL to display the Red Hat CodeReady Workspaces login page.

2. Choose the OpenShift OAuth option.
3. The **Authorize Access** page is displayed.

4. Click on the **Allow selected permissions** button.

5. Update the account information: fill in the **Username**, **Email**, **First name** and **Last name** field and click the **Submit** button.

6. The browser displays the Red Hat CodeReady Workspaces **Dashboard**.

### 1.1.10. Logging in to CodeReady Workspaces on OpenShift for the first time registering as a new user

This section describes how to log in to CodeReady Workspaces on OpenShift for the first time registering as a new user.

#### Prerequisites

- A running Red Hat CodeReady Workspaces cluster.
  - See Section 1.1.4, “Viewing the state of the CodeReady Workspaces instance deployment in OpenShift 4 web console”.
  - See Section 1.1.6, “Viewing the state of the CodeReady Workspaces cluster deployment using OpenShift 4 CLI tools”.

- Know the Red Hat CodeReady Workspaces URL.
  - See Section 1.1.5, “Finding CodeReady Workspaces instance URL in OpenShift 4 web console”.
  - See Section 1.1.7, “Finding CodeReady Workspaces cluster URL using the OpenShift 4 CLI”.

#### Procedure

1. Navigate to the **Red Hat CodeReady Workspaces URL** to display the Red Hat CodeReady Workspaces login page.

2. Choose the **Register as a new user** option.

3. Update the account information: fill in the **Username**, **Email**, **First name** and **Last name** field and click the **Submit** button.

4. The browser displays the Red Hat CodeReady Workspaces **Dashboard**.

### 1.2. INSTALLING CODEREADY WORKSPACES USING CLI MANAGEMENT TOOL

#### 1.2.1. Installing the crwctl CLI management tool

This section describes how to install crwctl, the CodeReady Workspaces CLI management tool.

#### Procedure


3. Extract the archive.
4. Place the extracted binary in your $PATH.

1.2.2. Installing CodeReady Workspaces using CodeReady Workspaces CLI management tool

This section describes how to install CodeReady Workspaces using the CodeReady Workspaces CLI management tool.

NOTE

Use CodeReady Workspaces CLI management tool to install CodeReady Workspaces only if OperatorHub is not available. This method is not officially supported for OpenShift Container Platform 4.1 or later.

Prerequisites

- CodeReady Workspaces CLI management tool is installed.
- OpenShift Container Platform 4 CLI is installed.
- Access to an OpenShift Container Platform instance

1.2.2.1. Installing with default settings

Procedure

1. Log in to OpenShift Container Platform 4:

   ```
   $ oc login ${OPENSHIFT_API_URL} -u ${OPENSHIFT_USERNAME} -p ${OPENSHIFT_PASSWORD}
   ```

2. Run this command to install red-hat-codeready-workspaces with defaults settings:

   ```
   $ crwctl server:start
   ```

1.2.2.2. Installing with custom settings

Procedure

To override specific settings of the red-hat-codeready-workspaces installation, provide a dedicated custom resource when running the above `crwctl` command:

1. Download the default custom resource YAML file.

2. Name the downloaded custom resource `org_v1_che_cr.yaml`, and copy it into the current directory.

3. Modify the `org_v1_che_cr.yaml` file to override or add any field.

4. Run the installation using the `org_v1_che_cr.yaml` file to override the CodeReady Workspaces CLI management tool defaults:
$ crwctl server:start --che-operator-cr-yaml=org_v1_che_cr.yaml

NOTE

Some basic installation settings can be overridden in a simpler way by using additional `crwctl` arguments. To display the list of available arguments:

$ crwctl server:start --help
Operators are a method of packaging, deploying, and managing a OpenShift application which also provide the following:

- Repeatability of installation and upgrade.
- Constant health checks of every system component.
- Over-the-air (OTA) updates for OpenShift components and ISV content.
- A place to encapsulate knowledge from field engineers and spread it to all users.

This chapter describes how to install CodeReady Workspaces on OpenShift 3, with the CLI management tool, using the Operator method.

### 2.1. PREPARING OPENSHIFT 3 FOR INSTALLING CODEREADY WORKSPACES

**Prerequisites**

- A running instance of OpenShift 3.11.
- Administrator rights on this OpenShift 3 instance.
- The `oc` OpenShift 3.11 CLI management tool is installed and configured. See [Installing the OpenShift 3.11 CLI](#).
- The `crwctl` CLI management tool is installed. See [the CodeReady Workspaces 2.1 Installation Guide](#).

**Procedure**

1. Log in to OpenShift. See [Basic Setup and Login](#).
   
   ```
   $ oc login
   ```

2. Run the following command to verify that the version of the `oc` OpenShift CLI management tool is 3.11:
   
   ```
   $ oc version
   oc v3.11.0+0cbc58b
   ```

3. Run the following commands to create a dummy project to find the URL that this OpenShift instance is using to deploy applications.

   ```
   $ oc new-project hello-world
   $ oc new-app centos/httpd-24-centos7~https://github.com/openshift/httpd-ex
   $ oc expose svc/httpd-ex
   $ oc get route httpd-ex
   ```

<table>
<thead>
<tr>
<th>NAME</th>
<th>HOST/PORT</th>
<th>PATH</th>
<th>SERVICES</th>
<th>PORT</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
TERMINATION WILDCARD
httpd-ex httpd-ex-hello-world.apps.rhpds311.openshift.opentlc.com httpd-ex 8080-tcp
None

4. Extract the domain from httpd-ex-hello-world.apps.rhpds311.openshift.opentlc.com. It is the part after the first name: apps.rhpds311.openshift.opentlc.com. Remember this URL as <OPENSHIFT_APPS_URL>.

5. Remove the dummy project:

```
$ oc delete project hello-world
```

2.2. INSTALLING CODEREADY WORKSPACES ON OPENSHIFT 3 USING THE OPERATOR

This section describes how to install CodeReady Workspaces on OpenShift 3 with the CLI management tool, using the Operator method.

Prerequisites

- A running instance of OpenShift 3.11.
- Administrator rights on this OpenShift 3 instance.
- The `oc` OpenShift 3.11 CLI management tool is installed and configured. See Installing the OpenShift 3.11 CLI.
- The `crwctl` CLI management tool is installed. See the CodeReady Workspaces 2.1 Installation Guide.
- The URL for applications on this OpenShift instance is known as `<OPENSHIFT_APPS_URL>`. See Section 2.1, “Preparing OpenShift 3 for installing CodeReady Workspaces”.

Procedure

1. Create a new project called codeready-workspaces:

```
$ oc new-project codeready-workspaces
```

2. Run the following command to create the CodeReady Workspaces instance:

```
$ crwctl server:start --platform=openshift --installer=operator \
  --domain=<OPENSHIFT_APPS_URL>
```

Verification steps

1. The output of the previous command ends with:

```
Command server:start has completed successfully.
```

2. Navigate to the CodeReady Workspaces cluster instance: `http(s)://codeready-<openshift_deployment_name>.<domain_name>`. 
2.3. INSTALLING CODEREADY WORKSPACES ON OPENSHIFT 3 USING THE OPERATOR AND SSL

This section describes how to install CodeReady Workspaces on OpenShift 3 with the CLI management tool, using the Operator method and the SSL option.

Prerequisites

- A running instance of OpenShift 3.11.
- Administrator rights on this OpenShift 3 instance.
- The `oc` OpenShift 3.11 CLI management tool is installed and configured. See Installing the OpenShift 3.11 CLI.
- The `crwctl` CLI management tool is installed. See the CodeReady Workspaces 2.1 Installation Guide.
- The URL for applications on this OpenShift instance is known as `<OPENSHIFT_APPS_URL>`. See Section 2.1, “Preparing OpenShift 3 for installing CodeReady Workspaces”.

Procedure

1. Run the following command to create the CodeReady Workspaces instance:

   ```bash
   $ crwctl server:start -n <project-name> --domain=<OPENSHIFT_APPS_URL> --tls
   ```

   **NOTE**

   To create the CodeReady Workspaces instance on demo OpenShift clusters that have not been setup with a valid certificate for the routes, run the command with the `--self-signed-cert` flag.

Verification steps

1. The output of the previous command ends with:

   ```bash
   Command server:start has completed successfully.
   ```

2. Navigate to the CodeReady Workspaces cluster instance. The domain is now prefixed with HTTPS and using Let’s Encrypt ACME certificates: http(s)://codeready-workspaces.<OPENSHIFT_APPS_URL>
CHAPTER 3. INSTALLING CODEREADY WORKSPACES IN A RESTRICTED ENVIRONMENT

By default, Red Hat CodeReady Workspaces uses various external resources, mainly container images available in public registries.

To deploy CodeReady Workspaces in an environment where these external resources are not available (for example, on a cluster that is not exposed to the public Internet):

1. Identify the image registry used by the OpenShift cluster, and ensure you can push to it.
2. Push all the images needed for running CodeReady Workspaces to this registry.
3. Configure CodeReady Workspaces to use the images that have been pushed to the registry.
4. Proceed to the CodeReady Workspaces installation.

The procedure for installing CodeReady Workspaces in restricted environments is different based on the installation method you use:

- Installation through OperatorHub on Openshift 4.3 and above
- Installation through the cli management tool on both OpenShift 3.11 or 4.x

Notes on network connectivity in restricted environments

Restricted network environments range from a private subnet in a cloud provider to a separate network owned by a company, disconnected from the public Internet. Regardless of the network configuration, CodeReady Workspaces works provided that the Routes that are created for CodeReady Workspaces components (codeready-workspaces-server, identity provider, devfile and plugin registries) are accessible from inside the OpenShift cluster.

Take into account the network topology of the environment to determine how best to accomplish this. For example, on a network owned by a company or an organization, the network administrators must ensure that traffic bound from the cluster can be routed to Route hostnames. In other cases, for example, on AWS, create a proxy configuration allowing the traffic to leave the node to reach an external-facing Load Balancer.

When the restricted network involves a proxy, follow the instructions provided in Section 3.3, “Preparing CodeReady Workspaces Custom Resource for installing behing a proxy”.

3.1. INSTALLING CODEREADY WORKSPACES IN A RESTRICTED ENVIRONMENT USING OPERATORHUB

Prerequisites

- A running OpenShift cluster. See the OpenShift Container Platform 4.3 documentation for instructions on how to install an OpenShift cluster on a restricted network.
- Access to the mirror registry used to installed the OpenShift disconnected cluster in restricted network. See the Related OpenShift Container Platform 4.3 documentation about creating a mirror registry for installation in a restricted network.
On disconnected OpenShift 4 clusters running on restricted networks, an Operator can be successfully installed from OperatorHub only if it meets the additional requirements defined in Enabling your Operator for restricted network environments.

The CodeReady Workspaces operator meets these requirements and is therefore compatible with the official documentation about OLM on a restricted network.

Procedure

To install CodeReady Workspaces from OperatorHub:

1. Build a `redhat-operators` catalog image. See Building an Operator catalog image.

2. Configure OperatorHub to use this catalog image for operator installations. See Configuring OperatorHub for restricted networks.

3. Proceed to the CodeReady Workspaces installation as usual as described in Section 1.1, “Installing CodeReady Workspaces on OpenShift 4 from OperatorHub”.

3.2. INSTALLING CODEREADY WORKSPACES IN A RESTRICTED ENVIRONMENT USING CLI MANAGEMENT TOOL

**NOTE**

Use CodeReady Workspaces CLI management tool to install CodeReady Workspaces on restricted networks only if installation through OperatorHub is not available. This method is not officially supported for OpenShift Container Platform 4.1 or later.

Prerequisites

- A running OpenShift cluster. See the OpenShift Container Platform 4.2 documentation for instructions on how to install an OpenShift cluster.

3.2.1. Preparing an image registry for installing CodeReady Workspaces in a restricted environment

Prerequisites

- The `oc` tool is installed.

- An image registry that is accessible from the OpenShift cluster. Ensure you can push to it from a location that has, at least temporarily, access to the Internet.

- The `podman` tool is installed.

**NOTE**

When pushing images to other registry than the OpenShift internal registry, and the `podman` tool fails to work, use the `docker` tool instead.

The following placeholders are used in this section.

Table 3.1. Placeholders used in examples
NOTE

For the OpenShift internal registry, the placeholder values are typically the following:

Table 3.2. Placeholders for the internal OpenShift registry

| <internal-registry> | image-registry.openshift-image-registry.svc:5000 |
| <organization>     | openshift                                         |

See [OpenShift documentation](https://openshift.redhat.com/docs/) for more details.

Procedure

1. Define the environment variable with the external endpoint of the image registry:
   For the OpenShift internal registry, use:
   ```bash
   $ REGISTRY_ENDPOINT=$(oc get route default-route --namespace openshift-image-registry \
   --template="{{ .spec.host }}")
   ```
   For other registries, use the host name and port of the image registry:
   ```bash
   $ REGISTRY_ENDPOINT=<internal-registry>
   ```

2. Log into the internal image registry:
   ```bash
   $ podman login --username <user> --password <password> <internal-registry>
   ```

   **NOTE**

   When using the OpenShift internal registry, follow the steps described in the related [OpenShift documentation](https://openshift.redhat.com/docs/) to first expose the internal registry through a route, and then log in to it.

3. Download, tag, and push the necessary images. Repeat the step for every image in the following lists:
   ```bash
   $ podman pull <image_name>:<image_tag>
   $ podman tag <image_name>:<image_tag>
   $REGISTRY_ENDPOINT/<organization>/<image_name>:<image_tag>
   $ podman push $REGISTRY_ENDPOINT/<organization>/<image_name>:<image_tag>
   ```

   Essential images
The following infrastructure images are included in every workspace launch:

- registry.redhat.io/codeready-workspaces/crw-2-rhel8-operator:2.1
- registry.redhat.io/codeready-workspaces/server-rhel8:2.1
- registry.redhat.io/codeready-workspaces/pluginregistry-rhel8:2.1
- registry.redhat.io/codeready-workspaces/devfileregistry-rhel8:2.1
- registry.redhat.io/codeready-workspaces/pluginbroker-artifacts-rhel8:2.1
- registry.redhat.io/codeready-workspaces/pluginbroker-metadata-rhel8:2.1
- registry.redhat.io/codeready-workspaces/jwtproxy-rhel8:2.1
- registry.redhat.io/codeready-workspaces/machineexec-rhel8:2.1
- registry.redhat.io/codeready-workspaces/theia-rhel8:2.1
- registry.redhat.io/codeready-workspaces/theia-dev-rhel8:2.1
- registry.redhat.io/codeready-workspaces/theia-endpoint-rhel8:2.1
- registry.redhat.io/rhscl/postgresql-96-rhel7:1-47
- registry.redhat.io/redhat-sso-7/sso73-openshift:1.0-15
- registry.redhat.io/ubi8-minimal:8.0-213

**Workspace-specific images**

These are images that are required for running a workspace. A workspace generally uses only a subset of the images below. It is only necessary to include the images related to required technology stacks.

- registry.redhat.io/codeready-workspaces/stacks-cpp-rhel8:2.1
- registry.redhat.io/codeready-workspaces/stacks-dotnet-rhel8:2.1
- registry.redhat.io/codeready-workspaces/stacks-golang-rhel8:2.1
- registry.redhat.io/codeready-workspaces/stacks-java-rhel8:2.1
- registry.redhat.io/codeready-workspaces/stacks-node-rhel8:2.1
- registry.redhat.io/codeready-workspaces/stacks-php-rhel8:2.1
- registry.redhat.io/codeready-workspaces/stacks-python-rhel8:2.1
- registry.redhat.io/codeready-workspaces/plugin-java11-rhel8:2.1
- registry.redhat.io/codeready-workspaces/plugin-openshift-rhel8:2.1
- registry.redhat.io/codeready-workspaces/plugin-kubernetes-rhel8:2.1

**3.2.2. Preparing CodeReady Workspaces Custom Resource for restricted environment**
When installing CodeReady Workspaces in a restricted environment using `crctl` or OperatorHub, provide a **CheCluster** custom resource with additional information.

### 3.2.2.1. Downloading the default **CheCluster** Custom Resource

**Procedure**

1. Download the default custom resource YAML file.

2. Name the downloaded custom resource **org_v1_che_cr.yaml**. Keep it for further modification and usage.

### 3.2.2.2. Customizing the **CheCluster** Custom Resource for restricted environment

**Prerequisites**

- All required images available in an image registry that is visible to the OpenShift cluster where CodeReady Workspaces is to be deployed. This is described in **Section 3.2.1, “Preparing an image registry for installing CodeReady Workspaces in a restricted environment”**, where the placeholders used in the following examples are also defined.

**Procedure**

1. In the **CheCluster** Custom Resource, which is managed by the CodeReady Workspaces Operator, add the fields used to facilitate deploying an instance of CodeReady Workspaces in a restricted environment:

   ```yaml
   # [...]
   spec:
     server:
       airGapContainerRegistryHostname: '<internal-registry>'
       airGapContainerRegistryOrganization: '<organization>'
   # [...]
   
   Setting these fields in the Custom Resource uses `<internal-registry>` and `<organization>` for all images. This means, for example, that the Operator expects the offline plug-in and devfile registries to be available at:

   ```shell
   <internal-registry>/<organization>/pluginregistry-rhel8:<ver>
   <internal-registry>/<organization>/pluginregistry-rhel8:<ver>
   
   For example, to use the OpenShift 4 internal registry as the image registry, define the following fields in the **CheCluster** Custom Resource:

   ```yaml
   # [...]
   spec:
     server:
       airGapContainerRegistryHostname: 'image-registry.openshift-image-registry.svc:5000'
       airGapContainerRegistryOrganization: 'openshift'
   # [...]
   ```

2. In the downloaded **CheCluster** Custom Resource, add the two fields described above with the proper values according to the container-image registry with all the mirrored container images.
3.2.3. Starting CodeReady Workspaces installation in a restricted environment using CodeReady Workspaces CLI management tool

This section describes how to start the CodeReady Workspaces installation in a restricted environment using the CodeReady Workspaces CLI management tool.

Prerequisites

- CodeReady Workspaces CLI management tool is installed.
- The `oc` tool is installed.
- Access to an OpenShift instance.

Procedure

1. Log in to OpenShift Container Platform:

   ```bash
   $ oc login ${OPENSHIFT_API_URL} --username ${OPENSHIFT_USERNAME} \
   --password ${OPENSHIFT_PASSWORD}
   ```

2. Install CodeReady Workspaces with the customized Custom Resource to add fields related to restricted environment:

   ```bash
   $ crwctl server:start \ 
   --che-operator-image=<image-registry>\<organization>/server-operator-rhel8:2.1 \ 
   --che-operator-cr-yaml=org_v1_che_cr.yaml
   ```

3.3. PREPARING CODEREADY WORKSPACES CUSTOM RESOURCE FOR INSTALLING BEHIND A PROXY

This procedure describes how to provide necessary additional information to the `CheCluster` custom resource when installing CodeReady Workspaces behind a proxy.

Procedure

1. In the `CheCluster` Custom Resource, which is managed by the CodeReady Workspaces Operator, add the fields used to facilitate deploying an instance of CodeReady Workspaces in a restricted environment:

   ```yaml
   # [...] 
   spec: 
   server: 
   proxyURL: '<URL of the proxy, with the http protocol, and without the port>' 
   proxyPort: '<Port of proxy, typically 3128>' 
   # [...] 
   ```

2. In addition to those basic settings, the proxy configuration usually requires adding the host of the external OpenShift cluster API URL in the list of the hosts to be accessed from CodeReady Workspaces without using the proxy.

   To retrieve this cluster API host, run the following command against the OpenShift cluster:

   ```bash
   $ oc whoami --show-server | sed 's#https://##' | sed 's#::.*##'
   ```
+ The corresponding field of the CheCluster Custom Resource is nonProxyHosts. If a host already exists in this field, use | as a delimiter to add the cluster API host:

```yaml
# [...]  
spec:
  server:
    nonProxyHosts: 'anotherExistingHost|<cluster api host>'
# [...]  
```
CHAPTER 4. UPGRADING CODEREADY WORKSPACES

This chapter describes how to upgrade a CodeReady Workspaces instance to CodeReady Workspaces 2.1.

4.1. UPGRADING CODEREADY WORKSPACES USING OPERATORHUB

This section describes how to upgrade from CodeReady Workspaces 2.0 to CodeReady Workspaces 2.1 on OpenShift 4 using the OpenShift web console. This method is using the Operator from OperatorHub.

Prerequisites

- An administrator account on an OpenShift 4 instance.
- An instance of CodeReady Workspaces 2.0, running on the same instance of OpenShift 4, installed using an Operator from OperatorHub.

Procedure

1. Open the OpenShift web console.
2. Navigate to the Operators → Installed Operators section.
3. Click Red Hat CodeReady Workspaces in the list of installed operators.
4. Navigate to the Subscription tab and enable the following options:
   - Channel: latest
   - Approval: Automatic

Verification steps

1. Log in to the CodeReady Workspaces instance.
2. The 2.1 version number is visible at the bottom of the page.

4.2. UPGRADING CODEREADY WORKSPACES USING CLI MANAGEMENT TOOL ON OPENSHIFT 3

This section describes how to upgrade from CodeReady Workspaces 2.0 to CodeReady Workspaces 2.1 on OpenShift 3 using the CLI management tool.

Prerequisites

- An administrative account on an OpenShift 3 instance.
- A running instance of Red Hat CodeReady Workspaces running on OpenShift 3, installed using the CLI management tool.
- The crwctl management tool installed.

Procedure
1. In all running workspaces in the CodeReady Workspaces 2.0 instance, save and push changes to Git repositories.

2. Run the following command:

   ```
   $ crwctl server:update
   ```

**Verification steps**

1. Log in to the CodeReady Workspaces instance.

2. The 2.1 version number is visible at the bottom of the page.

### 4.3. UPGRADING CODEREADY WORKSPACES FROM PREVIOUS MAJOR VERSION

This section describes how to perform an upgrade from the previous major version of Red Hat CodeReady Workspaces (1.2).

**Procedure**

- See Upgrading CodeReady Workspaces section in CodeReady Workspaces 2.1 Installation Guide
CHAPTER 5. ADVANCED CONFIGURATION OPTIONS

The following section describes advanced deployment and configuration methods for Red Hat CodeReady Workspaces.

5.1. CODEREADY WORKSPACES CONFIGMAPS AND THEIR BEHAVIOR

The following section describes CodeReady Workspaces configMaps and how they behave.

A configMap is provided as an editable file that lists options to customize the CodeReady Workspaces environment. Based on the CodeReady Workspaces installation method, configMaps can be used to customize the working environment. The type of configMaps available in your CodeReady Workspaces environment varies based on the method used for installing CodeReady Workspaces.

5.1.1. CodeReady Workspaces installed using an Operator

Operators are software extensions to OpenShift that use custom resources to manage applications and their components.

CodeReady Workspaces installed using the Operator provides the user with an automatically generated configMap called codeready.

The codeready configMap contains the main properties for the CodeReady Workspaces server, and is in sync with the information stored in the CheCluster Custom Resource file. User modifications of the codeready configMap after installing CodeReady Workspaces using the Operator are automatically overwritten by values that the Operator obtains from the CheCluster Custom Resource.

To edit the codeready configMap, edit the Custom Resource manually. The configMap derives values from the CheCluster field. User modifications of the CheCluster Custom Resource field cause the Operator to change the attributes of the codeready configMap accordingly. The configMap changes automatically trigger a restart of the CodeReady Workspaces Pod.

To add custom properties to the CodeReady Workspaces server, such as environment variables that are not automatically generated in the codeready configMap by the Operator, or to override automatically generated properties, the CheCluster Custom Resource has a customCheProperties field, which expects a map.

For example, to override the default memory limit for workspaces, add the CHE_WORKSPACE_DEFAULT_MEMORY_LIMIT_MB property to customCheProperties:

```yaml
apiVersion: org.eclipse.che/v1
kind: CheCluster
metadata:
  name: eclipse-che
namespace: che
spec:
  server:
    cheImageTag: 
    devfileRegistryImage: 
    pluginRegistryImage: 
    tlsSupport: true
    selfSignedCert: false
  customCheProperties:
```


Previous versions of the CodeReady Workspaces Operator had a configMap named `custom` to fulfill this role. If the CodeReady Workspaces Operator finds a `configMap` with the name `custom`, it adds the data it contains into the `customCheProperties` field, redeploys CodeReady Workspaces, and deletes the `custom` configMap.

### 5.2. CONFIGURING NAMESPACE STRATEGIES

#### NOTE

The term `namespace` (Kubernetes) is used interchangeably with `project` (OpenShift).

The namespace strategies are configured using the `CHE_INFRA_KUBERNETES_NAMESPACE_DEFAULT` environment variable.

#### WARNING

There are legacy variables `CHE_INFRA_KUBERNETES_NAMESPACE` and `CHE_INFRA_OPENSHIFT_PROJECT`. Keep these variables unset for a new installations. Changing these variables during an update can lead to data loss.

### 5.2.1. One namespace per workspace strategy

The strategy creates a new namespace for each new workspace.

To use the strategy, the `CHE_INFRA_KUBERNETES_NAMESPACE_DEFAULT` variable value must contain the `<workspaceID>` identifier. It can be used alone or combined with other identifiers or any string.

**Example 5.1. One namespace per workspace**

To assign namespace names composed of a `che-ws` prefix and workspace id, set:

```
CHE_INFRA_KUBERNETES_NAMESPACE_DEFAULT=che-ws-<workspaceID>
```

### 5.2.2. One namespace for all workspaces strategy

The strategy uses one predefined namespace for all workspaces.

To use the strategy, the `CHE_INFRA_KUBERNETES_NAMESPACE_DEFAULT` variable value must be the name of the desired namespace to use.

**Example 5.2. One namespace for all workspaces**

```
CHE_WORKSPACE_DEFAULT__MEMORY__LIMIT__MB: "2048"
auth:
...
To have all workspaces created in `che-workspaces` namespace, set:

```
CHE_INFRA_KUBERNETES_NAMESPACE_DEFAULT=che-workspaces
```

**IMPORTANT**

To run more than one workspace at a time when using this strategy together with the common PVC strategy, configure persistent volumes to use **ReadWriteMany** access mode.

### 5.2.3. One namespace per user strategy

The strategy isolates each user in their own namespace.

To use the strategy, the `CHE_INFRA_KUBERNETES_NAMESPACE_DEFAULT` variable value must contain one or more user identifiers. Currently supported identifiers are `<username>` and `<userId>`.

**Example 5.3. One namespace per user**

To assign namespace names composed of a `che-ws` prefix and individual usernames (`che-ws-user1`, `che-ws-user2`), set:

```
CHE_INFRA_KUBERNETES_NAMESPACE_DEFAULT=che-ws-<username>
```

**IMPORTANT**

To run more than one workspace at a time when using this strategy together with the common PVC strategy, configure persistent volumes to use **ReadWriteMany** access mode.

To limit the number of concurrently running workspaces per user to one, set the `CHE_LIMITS_USER_WORKSPACES_RUN_COUNT` environment variable to 1.

To limit the number of concurrently running workspaces per user to one (1):

- For Helm Chart deployments: set the `.global.workspace.number` parameter to 1.
- For Operator deployments: set the `spec.server.cheCustomProperties.CHE_LIMITS_USER_WORKSPACE_RUN_COUNT` variable of the CheCluster Custom Resource (CR) to 1.

### 5.2.4. Allowing user-defined workspace namespaces

CodeReady Workspaces server can be configured to honor the user selection of a namespace when a workspace is created. This feature is disabled by default. To allow user-defined workspace namespaces:

- For Operator deployments, set the following field in the CheCluster Custom Resource:

  ```yaml
  allowUserDefinedWorkspaceNamespaces
  ```

29
5.3. DEPLOYING CODEREADY WORKSPACES WITH SUPPORT FOR
GIT REPOSITORIES WITH SELF-SIGNED CERTIFICATES

This procedure describes how to configure CodeReady Workspaces for deployment with support for Git
operations on repositories that use self-signed certificates.

Prerequisites

- Git version 2 or later

Procedure

Configuring support for self-signed Git repositories.

1. Create a new configMap with details about the Git server:

   $ oc create configmap che-git-self-signed-cert --from-file=ca.crt \\
   --from-literal=githost=<host:port> -n {prod-namespace}

   In the command, substitute <host:port> for the host and port of the HTTPS connection on the
   Git server (optional).

   NOTE

   - When githost is not specified, the given certificate is used for all HTTPS
     repositories.
   - The certificate file must be named ca.crt.

2. Configure the workspace exposure strategy:
   Update the gitSelfSignedCert property. To do that, execute:

   $ oc patch checluster codeready-workspaces -n workspaces --type=json  \\
   -p '[{"op": "replace", "path": "/spec/server/gitSelfSignedCert", "value": true}]'

3. Create and start a new workspace. Every container used by the workspace mounts a special
   volume that contains a file with the self-signed certificate. The repository’s .git/config file
   contains information about the Git server host (its URL) and the path to the certificate in the
   http section (see Git documentation about .git/config). For example:

   [http "https://10.33.177.118:3000"]
   sslCAInfo = /etc/che/git/cert/ca.crt

5.4. ADDING CUSTOM PUBLIC SSL CERTIFICATES TO CODEREADY
WORKSPACES TRUST-STORE

This procedure describes how to configure CodeReady Workspaces to be able to perform HTTP
requests to unrecognized resources.

Procedure

1. Save the public certificate(s) that you want to apply.
2. Create a new `configMap` with the certificate(s):

   $ oc create configmap <config-map name> --from-file=<certificate file path> -n=che

   To apply more than one certificate, add `-from-file=<certificate file path>` key to the command.

3. Define the certificates config-map name.

   If CodeReady Workspaces is deployed using a Helm Chart

   1. Clone the `che` project

   2. Go to `deploy/kubernetes/helm/che` directory

   3. Set the `global.tls.serverTrustStoreConfigMapName` property to previously created config-map name. To do that, add the following option to the `helm upgrade` command:

   ```
   $ helm upgrade che -n che --set global.tls.serverTrustStoreConfigMapName=<config-map name> --set global.ingressDomain=<kubernetes-cluster-domain>.
   ```

   `<kubernetes-cluster-domain>` On Minikube, use `$(minikube ip).mycluster.mycompany.com`

   If CodeReady Workspaces is deployed using Operators

   Set the `serverTrustStoreConfigMapName` property to previously created config-map name. To do that, execute:

   ```
   $ oc patch checluster eclipse-che -n che --type=json -p 
   '{"op": "replace", "path": "/spec/server/serverTrustStoreConfigMapName", "value": "<config-map name>"}"
   ```

5.5. CODEREADY WORKSPACES CONFIGMAPS FIELDS REFERENCE

5.5.1. server settings related to the CodeReady Workspaces server

<table>
<thead>
<tr>
<th>Property</th>
<th>Default value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>airGapContainerRegistryHostname</code></td>
<td>omit</td>
<td>An optional host name or URL to an alternative container registry to pull images from. This value overrides the container registry host name defined in all default container images involved in a CodeReady Workspaces deployment. This is particularly useful to install CodeReady Workspaces in an air-gapped environment.</td>
</tr>
<tr>
<td><code>airGapContainerRegistryOrganization</code></td>
<td>omit</td>
<td>Optional repository name of an alternative container registry to pull images from. This value overrides the container registry organization defined in all default container images involved in a CodeReady Workspaces deployment. This is particularly useful to install CodeReady Workspaces in an air-gapped environment.</td>
</tr>
<tr>
<td><code>cheDebug</code></td>
<td>false</td>
<td>Enables the debug mode for CodeReady Workspaces server.</td>
</tr>
<tr>
<td>Property</td>
<td>Default value</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>---------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>cheFlavor</td>
<td>codeready-workspaces</td>
<td>Flavor of the installation.</td>
</tr>
<tr>
<td>cheHost</td>
<td>The Operator</td>
<td>A public host name of the installed CodeReady Workspaces server.</td>
</tr>
<tr>
<td></td>
<td>automatically sets</td>
<td></td>
</tr>
<tr>
<td></td>
<td>the value.</td>
<td></td>
</tr>
<tr>
<td>cheImagePullPolicy</td>
<td>Always for nightly</td>
<td>Overrides the image pull policy used in CodeReady Workspaces deployment.</td>
</tr>
<tr>
<td></td>
<td>or latest images,</td>
<td></td>
</tr>
<tr>
<td></td>
<td>IfNotPresent in</td>
<td></td>
</tr>
<tr>
<td></td>
<td>other cases</td>
<td></td>
</tr>
<tr>
<td>cheImageTag</td>
<td>omit</td>
<td>Overrides the tag of the container image used in CodeReady Workspaces</td>
</tr>
<tr>
<td></td>
<td></td>
<td>deployment. Omit it or leave it empty to use the default image tag provided</td>
</tr>
<tr>
<td></td>
<td></td>
<td>by the Operator.</td>
</tr>
<tr>
<td>cheImage</td>
<td>omit</td>
<td>Overrides the container image used in CodeReady Workspaces deployment.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This does not include the container image tag. Omit it or leave it empty to</td>
</tr>
<tr>
<td></td>
<td></td>
<td>use the default container image provided by the Operator.</td>
</tr>
<tr>
<td>cheLogLevel</td>
<td>INFO</td>
<td>Log level for the CodeReady Workspaces server: INFO or DEBUG.</td>
</tr>
<tr>
<td>cheWorkspaceClusterRole</td>
<td>omit</td>
<td>Custom cluster role bound to the user for the workspaces. Omit or leave</td>
</tr>
<tr>
<td></td>
<td></td>
<td>empty to use the default roles.</td>
</tr>
<tr>
<td>customCheProperties</td>
<td>omit</td>
<td>Map of additional environment variables that will be applied in the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>generated codeready-workspaces config map to be used by the CodeReady</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Workspaces server, in addition to the values already generated from other</td>
</tr>
<tr>
<td></td>
<td></td>
<td>fields of the CheCluster custom resource (CR). If customCheProperties</td>
</tr>
<tr>
<td></td>
<td></td>
<td>contains a property that would be normally generated in codeready-</td>
</tr>
<tr>
<td></td>
<td></td>
<td>workspaces config map from other CR fields, then the value defined in the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>customCheProperties will be used instead.</td>
</tr>
<tr>
<td>devfileRegistryImage</td>
<td>omit</td>
<td>Overrides the container image used in the Devfile registry deployment.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This includes the image tag. Omit it or leave it empty to use the default</td>
</tr>
<tr>
<td></td>
<td></td>
<td>container image provided by the Operator.</td>
</tr>
<tr>
<td>devfileRegistryMemoryLimit</td>
<td>256Mi</td>
<td>Overrides the memory limit used in the Devfile registry deployment.</td>
</tr>
<tr>
<td>devfileRegistryMemoryRequest</td>
<td>16Mi</td>
<td>Overrides the memory request used in the Devfile registry deployment.</td>
</tr>
<tr>
<td>Property</td>
<td>Default value</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------</td>
<td>------------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>devfileRegistry PullPolicy</td>
<td>Always for nightly or latest images, and IfNotPresent in other cases</td>
<td>Overrides the image pull policy used in the Devfile registry deployment.</td>
</tr>
<tr>
<td>devfileRegistry Url</td>
<td>The Operator automatically sets the value.</td>
<td>Public URL of the Devfile registry that serves sample, ready-to-use devfiles. Set it if you use an external devfile registry (see the externalDevfileRegistry field).</td>
</tr>
<tr>
<td>externalDevfile Registry</td>
<td>false</td>
<td>Instructs the Operator to deploy a dedicated Devfile registry server. By default a dedicated devfile registry server is started. If externalDevfileRegistry set to true, the Operator does not start a dedicated registry server automatically and you need to set the devfileRegistryUrl field manually.</td>
</tr>
<tr>
<td>externalPluginRegistry</td>
<td>false</td>
<td>Instructs the Operator to deploy a dedicated Plugin registry server. By default, a dedicated plug-in registry server is started. If externalPluginRegistry set to true, the Operator does not deploy a dedicated server automatically and you need to set the pluginRegistryUrl field manually.</td>
</tr>
<tr>
<td>nonProxyHosts</td>
<td>omit</td>
<td>List of hosts that will not use the configured proxy. Use `</td>
</tr>
<tr>
<td>pluginRegistryImage</td>
<td>omit</td>
<td>Overrides the container image used in the Plugin registry deployment. This includes the image tag. Omit it or leave it empty to use the default container image provided by the Operator.</td>
</tr>
<tr>
<td>pluginRegistryMemoryLimit</td>
<td>256Mi</td>
<td>Overrides the memory limit used in the Plugin registry deployment.</td>
</tr>
<tr>
<td>pluginRegistryMemoryRequest</td>
<td>16Mi</td>
<td>Overrides the memory request used in the Plugin registry deployment.</td>
</tr>
<tr>
<td>pluginRegistryPullPolicy</td>
<td>Always for nightly or latest images, and IfNotPresent in other cases</td>
<td>Overrides the image pull policy used in the Plugin registry deployment.</td>
</tr>
<tr>
<td>pluginRegistryUrl</td>
<td>the Operator sets the value automatically</td>
<td>Public URL of the Plugin registry that serves sample ready-to-use devfiles. Set it only when using an external devfile registry (see the externalPluginRegistry field).</td>
</tr>
</tbody>
</table>
### Property | Default value | Description
---|---|---
proxyPassword | omit | Password of the proxy server. Only use when proxy configuration is required.
proxyPort | omit | Port of the proxy server. Only use when configuring a proxy is required (see also the **proxyURL** field).
proxyURL | omit | URL (protocol+host name) of the proxy server. This drives the appropriate changes in the **JAVA_OPTS** and **https(s)_proxy** variables in the CodeReady Workspaces server and workspaces containers. Only use when configuring a proxy is required.
proxyUser | omit | User name of the proxy server. Only use when configuring a proxy is required (see also the **proxyURL** field).
selfSignedCert | false | Enables the support of OpenShift clusters with routers that use self-signed certificates. When enabled, the Operator retrieves the default self-signed certificate of OpenShift routes and adds it to the Java trust store of the CodeReady Workspaces server. Required when activating the **tlsSupport** field on demo OpenShift clusters that have not been setup with a valid certificate for the routes.
serverMemoryLimit | 1Gi | Overrides the memory limit used in the CodeReady Workspaces server deployment.
serverMemoryRequest | 512Mi | Overrides the memory request used in the CodeReady Workspaces server deployment.
tlsSupport | false | Instructs the Operator to deploy CodeReady Workspaces in TLS mode. Enabling TLS requires enabling the **selfSignedCert** field.

#### 5.5.2. database configuration settings related to the database used by CodeReady Workspaces

<table>
<thead>
<tr>
<th>Property</th>
<th>Default value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>chePostgresDb</td>
<td>dbche</td>
<td>PostgreSQL database name that the CodeReady Workspaces server uses to connect to the database.</td>
</tr>
<tr>
<td>chePostgresHostName</td>
<td>the Operator sets the value automatically</td>
<td>PostgreSQL Database host name that the CodeReady Workspaces server uses to connect to. Defaults to <strong>postgres</strong>. Override this value only when using an external database. (See the field <strong>externalDb</strong>.)</td>
</tr>
</tbody>
</table>
### CHAPTER 5. ADVANCED CONFIGURATION OPTIONS

<table>
<thead>
<tr>
<th>Property</th>
<th>Default value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>chePostgresPassword</td>
<td>auto-generated value</td>
<td>PostgreSQL password that the CodeReady Workspaces server uses to connect to the database.</td>
</tr>
<tr>
<td>chePostgresPort</td>
<td>5432</td>
<td>PostgreSQL Database port that the CodeReady Workspaces server uses to connect to. Override this value only when using an external database (see field <code>externalDb</code>).</td>
</tr>
<tr>
<td>chePostgresUser</td>
<td>pgche</td>
<td>PostgreSQL user that the CodeReady Workspaces server uses to connect to the database.</td>
</tr>
<tr>
<td>externalDb</td>
<td>false</td>
<td>Instructs the Operator to deploy a dedicated database. By default, a dedicated PostgreSQL database is deployed as part of the CodeReady Workspaces installation. If set to <code>true</code>, the Operator does not deploy a dedicated database automatically, you need to provide connection details to an external database. See all the fields starting with: <code>chePostgres</code>.</td>
</tr>
<tr>
<td>postgresImagePullPolicy</td>
<td>Always <code>nightly</code> or <code>latest</code> images, and <code>IfNotPresent</code> in other cases</td>
<td>Overrides the image pull policy used in the PostgreSQL database deployment.</td>
</tr>
<tr>
<td>postgresImage</td>
<td>omit</td>
<td>Overrides the container image used in the PostgreSQL database deployment. This includes the image tag. Omit it or leave it empty to use the default container image provided by the Operator.</td>
</tr>
</tbody>
</table>

#### 5.5.3. auth configuration settings related to authentication used by CodeReady Workspaces installation

<table>
<thead>
<tr>
<th>Property</th>
<th>Default value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>externalIdentityProvider</td>
<td>false</td>
<td>By default, a dedicated Identity Provider server is deployed as part of the CodeReady Workspaces installation. But if <code>externalIdentityProvider</code> is <code>true</code>, then no dedicated identity provider will be deployed by the Operator and you might need to provide details about the external identity provider you want to use. See also all the other fields starting with: <code>identityProvider</code>.</td>
</tr>
<tr>
<td>identityProviderAdminUserName</td>
<td>admin</td>
<td>Overrides the name of the Identity Provider admin user.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Property</th>
<th>Default value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>identityProviderClientId</td>
<td>omit</td>
<td>Name of an Identity provider (Keycloak / RH SSO) client-id that must be used for CodeReady Workspaces. This is useful to override it ONLY if you use an external Identity Provider (see the <code>externalIdentityProvider</code> field). If omitted or left blank, it will be set to the value of the <code>flavor</code> field suffixed with <code>-public</code>.</td>
</tr>
<tr>
<td>identityProviderImagePullPolicy</td>
<td><code>Always</code> for nightly or latest images, and <code>IfNotPresent</code> in other cases</td>
<td>Overrides the image pull policy used in the Identity Provider (Keycloak / RH SSO) deployment.</td>
</tr>
<tr>
<td>identityProviderImage</td>
<td>omit</td>
<td>Overrides the container image used in the Identity Provider (Keycloak / RH SSO) deployment. This includes the image tag. Omit it or leave it empty to use the default container image provided by the Operator.</td>
</tr>
<tr>
<td>identityProviderPassword</td>
<td>omit</td>
<td>Overrides the password of Keycloak admin user. Override it only when using an external Identity Provider (see the <code>externalIdentityProvider</code> field). Omit or leave empty to set an auto-generated password.</td>
</tr>
<tr>
<td>identityProviderPostgresPassword</td>
<td>the Operator sets the value automatically</td>
<td>Password for The Identity Provider (Keycloak / RH SSO) to connect to the database. This is useful to override it ONLY if you use an external Identity Provider (see the <code>externalIdentityProvider</code> field).</td>
</tr>
<tr>
<td>identityProviderRealm</td>
<td>omit</td>
<td>Name of an Identity provider (Keycloak / RH SSO) realm. Override it only when using an external Identity Provider (see the <code>externalIdentityProvider</code> field). Omit or leave empty blank to set it to the value of the <code>flavor</code> field.</td>
</tr>
<tr>
<td>identityProviderURL</td>
<td>the Operator sets the value automatically</td>
<td>Instructs the Operator to deploy a dedicated Identity Provider (Keycloak or RH SSO instance). Public URL of the Identity Provider server (Keycloak / RH SSO server). Set it only when using an external Identity Provider (see the <code>externalIdentityProvider</code> field).</td>
</tr>
<tr>
<td>oAuthClientName</td>
<td>the Operator sets the value automatically</td>
<td>Name of the OpenShift OAuthClient resource used to setup identity federation on the OpenShift side. See also the <code>OpenShiftOAuth</code> field.</td>
</tr>
<tr>
<td>oAuthSecret</td>
<td>the Operator sets the value automatically</td>
<td>Name of the secret set in the OpenShift OAuthClient resource used to setup identity federation on the OpenShift side. See also the <code>OAuthClientName</code> field.</td>
</tr>
</tbody>
</table>
CHAPTER 5. ADVANCED CONFIGURATION OPTIONS

### 5.5.4. Storage configuration settings related to persistent storage used by CodeReady Workspaces

<table>
<thead>
<tr>
<th>Property</th>
<th>Default value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>openShiftOAuth</code></td>
<td>true on OpenShift</td>
<td>Enables the integration of the identity provider (Keycloak / RHSSO) with OpenShift OAuth. This allows users to login with their OpenShift login and have their workspaces created under personal OpenShift namespaces. The <code>kubeadmin</code> user is not supported, and logging through does not allow access to the CodeReady Workspaces Dashboard.</td>
</tr>
<tr>
<td><code>updateAdminPassword</code></td>
<td>false</td>
<td>Forces the default <code>admin</code> CodeReady Workspaces user to update password on first login.</td>
</tr>
</tbody>
</table>

### 5.5.5. k8s configuration settings specific to CodeReady Workspaces installations on OpenShift

<table>
<thead>
<tr>
<th>Property</th>
<th>Default value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ingressClass</code></td>
<td>nginx</td>
<td>Ingress class that defines which controller manages ingresses.</td>
</tr>
</tbody>
</table>
## Property

<table>
<thead>
<tr>
<th>Property</th>
<th>Default value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ingressDomain</strong></td>
<td>omit</td>
<td>Global ingress domain for a K8S cluster. This field must be explicitly specified. This drives the <code>kubernetes.io/ingress.class</code> annotation on CodeReady Workspaces-related ingresses.</td>
</tr>
<tr>
<td><strong>ingressStrategy</strong></td>
<td>multi-host</td>
<td>Strategy for ingress creation. This can be <code>multi-host</code> (host is explicitly provided in ingress), <code>single-host</code> (host is provided, path-based rules) and <code>default-host</code> <em>(no host is provided, path-based rules)</em>.</td>
</tr>
<tr>
<td><strong>securityContext FsGroup</strong></td>
<td>1724</td>
<td>FSGroup the CodeReady Workspaces Pod and Workspace Pods containers will run in.</td>
</tr>
<tr>
<td><strong>securityContext RunAsUser</strong></td>
<td>1724</td>
<td>ID of the user the CodeReady Workspaces Pod and Workspace Pods containers will run as.</td>
</tr>
<tr>
<td><strong>tlsSecretName</strong></td>
<td>omit</td>
<td>Name of a secret that is used to set ingress TLS termination if TLS is enabled. See also the <code>tlsSupport</code> field.</td>
</tr>
</tbody>
</table>

### 5.5.6. installation defines the observed state of CodeReady Workspaces installation

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>cheClusterRunning</strong></td>
<td>Status of a CodeReady Workspaces installation. Can be Available, Unavailable, or Available, Rolling Update in Progress.</td>
</tr>
<tr>
<td><strong>cheURL</strong></td>
<td>Public URL to the CodeReady Workspaces server.</td>
</tr>
<tr>
<td><strong>cheVersion</strong></td>
<td>Currently installed CodeReady Workspaces version.</td>
</tr>
<tr>
<td><strong>dbProvisioned</strong></td>
<td>Indicates whether a PostgreSQL instance has been correctly provisioned.</td>
</tr>
<tr>
<td><strong>devfileRegistryURL</strong></td>
<td>Public URL to the Devfile registry.</td>
</tr>
<tr>
<td><strong>helpLink</strong></td>
<td>A URL to where to find help related to the current Operator status.</td>
</tr>
<tr>
<td><strong>keycloakProvisioned</strong></td>
<td>Indicates whether an Identity Provider instance (Keycloak / RH SSO) has been provisioned with realm, client and user.</td>
</tr>
<tr>
<td><strong>keycloakURL</strong></td>
<td>Public URL to the Identity Provider server (Keycloak / RH SSO).</td>
</tr>
<tr>
<td><strong>message</strong></td>
<td>A human-readable message with details about why the Pod is in this state.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>openShiftOAuthProvisioned</td>
<td>Indicates whether an Identity Provider instance (Keycloak / RH SSO) has been configured to integrate with the OpenShift OAuth.</td>
</tr>
<tr>
<td>pluginRegistryURL</td>
<td>Public URL to the Plugin registry.</td>
</tr>
<tr>
<td>reason</td>
<td>A brief CamelCase message with details about why the Pod is in this state.</td>
</tr>
</tbody>
</table>

### 5.5.7. Limits for workspaces

<table>
<thead>
<tr>
<th>Property</th>
<th>Default value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>che.limits.workspace.env.ram</td>
<td>16gb</td>
<td>The maximum amount of RAM that a user can allocate to a workspace when they create a new workspace. The RAM slider is adjusted to this maximum value.</td>
</tr>
<tr>
<td>che.limits.workspace.idle.timeout</td>
<td>1800000</td>
<td>The length of time that a user is idle with their workspace when the system will suspend the workspace and then stopping it. Idleness is the length of time that the user has not interacted with the workspace, meaning that one of our agents has not received interaction. Leaving a browser window open counts toward idleness.</td>
</tr>
</tbody>
</table>

### 5.5.8. Limits for the workspaces of an user

<table>
<thead>
<tr>
<th>Property</th>
<th>Default value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>che.limits.user.workspaces.ram</td>
<td>16gb</td>
<td>The total amount of RAM that a single user is allowed to allocate to running workspaces. A user can allocate this RAM to a single workspace or spread it across multiple workspaces.</td>
</tr>
<tr>
<td>che.limits.user.workspaces.count</td>
<td>1800000</td>
<td>The maximum number of workspaces that a user is allowed to create. The user will be presented with an error message if they try to create additional workspaces. This applies to the total number of both running and stopped workspaces.</td>
</tr>
<tr>
<td>che.limits.user.workspaces.run.count</td>
<td>1</td>
<td>The maximum number of running workspaces that a single user is allowed to have. If the user has reached this threshold and they try to start an additional workspace, they will be prompted with an error message. The user will need to stop a running workspace to activate another.</td>
</tr>
</tbody>
</table>

### 5.5.9. Limits for the workspaces of an organization
<table>
<thead>
<tr>
<th>Property</th>
<th>Default value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>che.limits.organization.workspaces.ram</td>
<td>-1</td>
<td>The total amount of RAM that a single organization (team) is allowed to allocate to running workspaces. An organization owner can allocate this RAM however they see fit across the team's workspaces.</td>
</tr>
<tr>
<td>che.limits.organization.workspaces.count</td>
<td>-1</td>
<td>The maximum number of workspaces that a organization is allowed to own. The organization will be presented an error message if they try to create additional workspaces. This applies to the total number of both running and stopped workspaces.</td>
</tr>
<tr>
<td>che.limits.organization.workspaces.run.count</td>
<td>-1</td>
<td>The maximum number of running workspaces that a single organization is allowed. If the organization has reached this threshold and they try to start an additional workspace, they will be prompted with an error message. The organization will need to stop a running workspace to activate another.</td>
</tr>
</tbody>
</table>
CHAPTER 6. UNINSTALLING CODEREADY WORKSPACES

This section describes uninstallation procedures for Red Hat CodeReady Workspaces installed on OpenShift. The uninstallation process leads to a complete removal of CodeReady Workspaces-related user data. The appropriate uninstallation method depends on what method was used to install the CodeReady Workspaces instance.

- For CodeReady Workspaces installed using OperatorHub, see Section 6.1, “Uninstalling CodeReady Workspaces after OperatorHub installation”.
- For CodeReady Workspaces installed using crwctl, see Section 6.2, “Uninstalling CodeReady Workspaces after crwctl installation”

6.1. UNINSTALLING CODEREADY WORKSPACES AFTER OPERATORHUB INSTALLATION

Users have two options for uninstalling CodeReady Workspaces from an OpenShift cluster. The following sections describe both of these methods:

- Using the OpenShift Administrator Perspective web UI
- Using `oc` commands from the terminal

6.1.1. Uninstalling CodeReady Workspaces using the OpenShift web console

This section describes how to uninstall CodeReady Workspaces from a cluster using the OpenShift Administrator Perspective main menu.

Prerequisites

- CodeReady Workspaces was installed on an OpenShift cluster using OperatorHub.

Procedure: deleting the CodeReady Workspaces deployment

1. Open the OpenShift web console.
2. Navigate to the Operators > Installed Operators section.
3. Click Red Hat CodeReady Workspaces in the list of installed operators.
5. In the row that displays information about the specific CodeReady Workspaces cluster, delete the CodeReady Workspaces Cluster deployment using the drop-down menu illustrated as three horizontal dots situated on the right side of the screen.
6. Alternatively, delete the CodeReady Workspaces deployment by clicking the displayed Red Hat CodeReady Workspaces Cluster, `red-hat-codeready-workspaces`, and select the Delete cluster option in the Actions drop-down menu on the top right.

Procedure: deleting the CodeReady Workspaces Operator

1. Open the OpenShift web console.
2. Navigate to the Operators > Installed Operators section in OpenShift Developer Perspective.
3. In the row that displays information about the specific Red Hat CodeReady Workspaces Operator, uninstall the CodeReady Workspaces Operator using the drop-down menu illustrated as three horizontal dots situated on the right side of the screen.

4. Accept the selected option, Also completely remove the Operator from the selected namespace.

5. Alternatively, uninstall the Red Hat CodeReady Workspaces Operator by clicking the displayed Red Hat CodeReady Workspaces Operator, Red Hat CodeReady Workspaces, followed by selecting the Uninstall Operator option in the Actions drop-down menu on the top right.

6.1.2. Uninstalling CodeReady Workspaces using oc commands

This section provides instructions on how to uninstall a CodeReady Workspaces instance using oc commands.

Prerequisites

- CodeReady Workspaces was installed on an OpenShift cluster using OperatorHub.
- OpenShift command-line tools (oc) are installed on the local workstation.

Procedure

The following procedure provides command-line outputs as examples. Note that output in the user terminal may differ.

To uninstall a CodeReady Workspaces instance from a cluster:

1. Sign in to the cluster:

```
$ oc login -u <username> -p <password> <cluster_URL>
```

2. Switch to the project where the CodeReady Workspaces instance is deployed:

```
$ oc project <codeready-workspaces_project>
```

3. Obtain the CodeReady Workspaces cluster name. The following shows a cluster named red-hat-codeready-workspaces:

```
$ oc get checluster
NAME          AGE
red-hat-codeready-workspaces   27m
```

4. Delete the CodeReady Workspaces cluster:

```
$ oc delete checluster red-hat-codeready-workspaces
checluster.org.eclipse.che "red-hat-codeready-workspaces" deleted
```

5. Obtain the name of the CodeReady Workspaces cluster service version (CSV) module. The following detects a CSV module named red-hat-codeready-workspaces.v2.1:

```
$ oc get csv
NAME DISPLAY VERSION REPLACES PHASE
```
6. Delete the CodeReady Workspaces CSV:

$ oc delete csv red-hat-codeready-workspaces.v2.1
clusterserviceversion.operators.coreos.com "red-hat-codeready-workspaces.v2.1" deleted

6.2. UNINSTALLING CODEREADY WORKSPACES AFTER CRWCTL INSTALLATION

This section describes how to uninstall an instance of Red Hat CodeReady Workspaces that was installed using the `crwctl` tool.

**IMPORTANT**

- For CodeReady Workspaces installed using the `crwctl server:start` command and the `-n` argument (custom namespace specified), use the `-n` argument also to uninstall the CodeReady Workspaces instance.
- For installations that did not use the `-n` argument, the created namespace is named workspaces by default.

**Prerequisites**

- CodeReady Workspaces was installed on an OpenShift cluster using `crwctl`.
- OpenShift command-line tools (`oc`) and `crwctl` are installed on the local workstation.
- The user is logged in a CodeReady Workspaces cluster using `oc`.

**Procedure**

1. Stop the Red Hat CodeReady Workspaces Server:

   $ crwctl server:stop

2. Obtain the name of the CodeReady Workspaces namespace:

   $ oc get checluster --all-namespaces -o=jsonpath="{.items[0].metadata.namespace}"

3. Remove CodeReady Workspaces from the cluster:

   $ crwctl server:delete -n <namespace>

   This removes all CodeReady Workspaces installations from the cluster.