Oracle® Cloud

Migrating Oracle Developer Cloud Service (Traditional) to Oracle Cloud Infrastructure

F20559-01
July 2019

Oracle Cloud Migrating Oracle Developer Cloud Service (Traditional) to Oracle Cloud Infrastructure,

F20559-01

Primary Author: Himanshu Marathe

Contributing Authors: Eric Jendrock

Contents

1 Learn About Migrating to Oracle Cloud Infrastructure

Why Migrate to Oracle Cloud Infrastructure 1-1
About the Migration Scope 1-1
Compare Oracle Cloud Infrastructure to Traditional 1-2
  About the Build System in DevCS on OCI 1-2
About the Migration Task Flow 1-4

2 Prepare to Migrate DevCS to Oracle Cloud Infrastructure

Migrate Users and Roles 2-1
Required Identity Domain Roles 2-1
Create the Target DevCS Instance in the OCI Region 2-2
Set Up the OCI Connections on the Target Oracle Cloud Account 2-2
  Set Up the OCI Account 2-2
  Get the Required OCI Input Values 2-7
  Set Up the OCI Connection in the Target DevCS Instance 2-10
Set Up an OCI Object Storage Bucket 2-12
  Set Up the OCI Object Storage Bucket 2-12

3 Migrate an Oracle Developer Cloud Service Instance to Oracle Cloud Infrastructure

Open the Source and the Target DevCS Instances 3-1
Migrate the Organization's Properties 3-3
  Gather Information About the Organization From the Source DevCS Instance 3-3
  Configure Organization Properties in the Target DevCS Instance 3-4
Migrate Projects 3-4
  Gather Information About a Project From the Source DevCS Instance 3-4
  Export Project Data From the Source DevCS Instance 3-6
    Export a Project's Data to an OCI Object Storage Bucket 3-8
  Create a Project With an Exported Project's Data in the Target DevCS Instance 3-9
Migrate Other Artifacts and Data 3-11
  Project Users 3-11

Gather Information About Users From the Source Project 3-11
Add Users in the Target Project 3-12
Mirrored Private Git Repositories 3-13
Gather Information About Mirrored Private Git Repositories From the Source Project 3-13
Mirror a Private Git Repository in the Target Project 3-14
Build Jobs 3-15
Builds Page User Interface 3-15
Gather Information About the Job's Software Packages From the Source Project 3-21
Create a VM Template and its Build VMs in the Target DevCS Instance 3-21
Create and Configure a Job in the Target Project 3-24
Create and Configure Pipelines 3-36
Run a Build of a Job or a Pipeline 3-41
Docker Registries 3-41
Gather Information About a Linked Docker Registry From the Source Project 3-42
Link Docker Registries in the Target Project 3-43
Maven Artifacts 3-43
Gather Information About a Maven Artifact From the Source Project 3-44
Upload Maven Artifacts in the Target Project 3-44
Gather Information About Maven Administration Settings From the Source Project 3-45
Configure Maven Administration Settings in the Target Project 3-46
Environments 3-47
Gather Information About an Environment and Service Instances From the Source Project 3-48
Create an Environment and Add Service Instances in the Target Project 3-49
Deployment Configurations 3-50
Gather Information About a Deployment Configuration From the Source Project 3-50
Create a Deployment Configuration in the Target Project 3-51
Webhooks 3-52
Gather Information About a Webhook From the Source Project 3-53
Create a Webhook in the Target Project 3-53
Template Definition 3-54
Gather Information About the Source Project's Template Definition 3-54
Define the Target Project as a Template 3-57
Announcements 3-59
Gather Information About an Announcement From the Source Project 3-59
Create an Announcement in the Target Project 3-60
RSS/ATOM Feeds 3-61
Gather Information About a RSS/ATOM Feed From the Source Project 3-61

Add a RSS/ATOM Feed in the Target Project 3-62
Custom Link Rules 3-62
Gather Information About a Link Rule From the Source Project 3-63
Create a Link Rule in the Target Project 3-63
Remove Your Membership or Ownership 3-64
Migrate User Preferences 3-65
User Preferences 3-65
Gather Information About Your User Preferences From the Source DevCS Instance 3-65
Set User Preferences in the Target DevCS Instance 3-66
Project Preferences 3-67
Git Repositories and Branches 3-67
Jobs 3-67
Agile Boards 3-68
4 Complete the Post-Migration Tasks
Test the Target DevCS Instance 4-1
Clean Up Resources in Traditional 4-1
Delete Projects in the Source DevCS Instance 4-1
Delete OCI Object Storage Buckets 4-2
Delete an OCI Object Storage Bucket 4-2

1

Learn About Migrating to Oracle Cloud Infrastructure

The next few topics explain the benefits of migrating your existing Oracle Developer Cloud Service (DevCS) instance from Traditional to Oracle Cloud Infrastructure (OCI) and provide an overview of the migration process.

Why Migrate to Oracle Cloud Infrastructure

Oracle encourages you to migrate your existing cloud resources to Oracle Cloud Infrastructure regions. You can gain several advantages by doing so.

In Oracle Cloud, you provision resources in specific regions, which are localized to geographic locations. Certain regions support the Oracle Cloud Infrastructure platform.

Oracle Cloud Infrastructure is Oracle's modern cloud platform that's based on the latest cloud technologies and standards. It provides more consistent performance and better features at lower costs. Oracle continues to invest in Oracle Cloud Infrastructure, including the addition of new regions, services, and features. See Data Regions for Platform and Infrastructure Services.

You can benefit from these additional administrative features when you migrate your cloud resources to Oracle Cloud Infrastructure:

- Organize cloud resources into a hierarchy of logical compartments.
- Create fine-grained access policies for each compartment.

About the Migration Scope

In this documentation, DevCS on Traditional is referred as the source DevCS instance and DevCS on OCI is referred as the target DevCS instance.

Before you migrate your source DevCS instance from Traditional to OCI, ensure that the target DevCS instance meets the prerequisites for the migration.

You need a DevCS instance in the OCI region set up with a dedicated OCI compartment and a bucket. This guide includes procedures that describe how to create the target DevCS instance, and how to set up a compartment and a bucket, but doesn't include detailed procedures on the configuration of basic OCI security, network, and storage resources that might be required to support your instance. Instead, this guide provides references to the OCI documentation, as appropriate.

Using the Export/Import Data feature of a project, you can automate the migration of some of the project's artifacts of the source DevCS instance to the project of the target DevCS instance. The artifacts that aren't migrated must be manually re-created in the target DevCS instance project.

This guide includes detailed procedures that explain how to export and import projects as well as migrate artifacts. This guide doesn't describe the new features available in the target DevCS instance. To find more about DevCS on OCI and its new features, see What is Oracle Developer Cloud Service? in *Using Oracle Developer Cloud Service*. To learn about OCI key concepts and terminology, see Key Concepts and Terminology in the Oracle Cloud Infrastructure documentation.

Compare Oracle Cloud Infrastructure to Traditional

Oracle Cloud services run in an Oracle Cloud account and use Oracle Identity Cloud Service, Oracle Identity and Access Management, or their own identity management systems to manage users and control access to cloud services.

For more information about both types of accounts, see About Oracle Cloud Accounts in *Getting Started with Oracle Cloud*.

Traditional Cloud Accounts do not use the Oracle Identity Cloud Service but use the traditional Identity and Access Management software to manage users and roles. When you migrate to OCI, you must migrate your users from Oracle Identity and Access Management to Oracle Identity Cloud Service. See Migrating from Traditional Cloud Accounts to Cloud Accounts with Identity Cloud Service in *Administering Oracle Identity Cloud Service*.

There aren't many differences between DevCS on Traditional and DevCS on OCI. DevCS on OCI employs a new build system that run builds on your OCI Compute VMs, and then stores the build and Maven artifacts on your OCI Object Storage buckets. To use DevCS on OCI, you must configure DevCS to use OCI compartments and buckets.

About the Build System in DevCS on OCI

In DevCS, you run builds of jobs on a Virtual Machine (VM), called a Build VM. In DevCS on Traditional, you run builds in an internal shared pool of Build VMs, which are owned by Oracle and are also shared between DevCS customers. You don't control the software installed on the Build VMs, not its Operating System (OS). The internal shared Build VMs run on Oracle Linux 6. The build artifacts and Maven artifacts are stored on the internal storage space of DevCS.

Builds in DevCS on OCI run on OCI Compute VMs that you own and don't share with other DevCS customers. The builds run faster compared to builds running in the internal shared pool and are more secure. You can also choose the platform of the Build VM and the software installed on it. The build and Maven artifacts are stored in your OCI Object Storage buckets.

This table summarizes the difference between the build system of DevCS on Traditional and DevCS on OCI.

DevCS on Traditional	DevCS on OCI
Builds run in an internal pool of VMs	Builds run on OCI Compute VMs
Oracle owns the Build VMs	You own the Build VMs
Build VMs are shared between DevCS customers	Build VMs are not shared between DevCS customers
The number of Build VMs are limited	You define the number of Build VMs as per your usage
You can't choose the region and shape of VMs	You can choose the region and shape of VMs

DevCS on Traditional	DevCS on OCI
Build VMs run on Oracle Linux 6	You can choose to run Build VMs on Oracle Linux 6 or Oracle Linux 7
No control over software available on Build VMs	You can choose the software available on Build VMs
Build artifacts are stored in the internal storage space of DevCS	Build artifacts are stored in OCI Object Storage buckets

A Build VM Template defines the operating system and the software installed on Build VMs. After creating the templates, you allocate some Build VMs to each Build VM Template. When your organization's members create jobs, they associate a Build VM template with a job. When a build of the job runs, it runs on Build VMs allocated to the job's Build VM Template.

One Build VM runs one build at a time. When you allocate Build VMs to a Build VM Template, set the number of VMs to the number of jobs that you expect to run in parallel using that template. For example, let's assume you have 10 jobs (say Job 1, Job 2, through Job 10) that use the same Build VM Template with only one Build VM allocated to it. If all jobs run their builds at the same time, only one job's build runs (say Job 1) and other nine (Job 2 through Job 10) wait until the build of the job is complete. If each build takes two minutes to complete, Job 10 has to wait for 18 minutes before its build starts.

If you increase the number of Build VMs allocated to the Build VM Template to five, the wait time of jobs reduces. Assuming builds of Job 1 through Job 10 run at the same time, Job 1 to Job 5 run immediately on the five Build VMs and Job 6 to Job 10 wait until a Build VM is available. If each build runs for two minutes, then their wait time is two minutes. If you allocate 10 Build VMs, then there is no waiting time. All builds run immediately.

Note that the more VMs you have running at a specific time, the higher the cost. So, allocate the number of Build VMs to the Build VM Templates to match your usage. You can always add or remove VMs, based on your actual usage.

These steps describe what happens when a build of a job runs in DevCS on OCI:

1. The build executor checks the Build VM template of the job and then searches for a Build VM that's allocated to the template.

   - If an allocated Build VM is available, the build executor immediately runs the build on it.
   - If all Build VMs allocated to the Build VM template are busy running builds of other jobs using the same Build VM template, the build executor waits until a Build VM becomes available and then runs the current job's build on it.
   - If no Build VM is allocated, the build fails.

2. The Build VM installs the software defined in the Build VM Template if any of these events happen.

   - A build runs on a Build VM for the first time
   - A Build VM wakes up after its sleep timeout period
   - The software of a Build VM template is updated

   Note that software installation takes some time.

3. After installing the software, the build executor clones the job’s Git repositories (if configured) to the Build VM, sets up the environment, runs the job's defined build steps, creates artifacts (if configured), and performs post-build steps (if configured).
4. After a build is complete, the build executor copies the artifacts (if generated) to the OCI Object Storage bucket.
5. The Build VM waits for some time for any queued builds. If no builds run in the wait time period, the Build VM uninstalls its software and stops.

To learn more about the software you can install on Build VMs and the new build system in DevCS on OCI, see Software Installed on the Build Executor and Configure and Run Project Jobs and Builds in *Using Oracle Developer Cloud Service*.

About the Migration Task Flow

At a high level, the migration process is composed of these tasks:

Task	Time Required	More Information
Create the target DevCS instance	30 minutes	Create the Target DevCS Instance in the OCI Region
Set up the OCI connections in the target DevCS instance	15 minutes	Set Up the OCI Connections on the Target Oracle Cloud Account
Set up OCI Object Storage bucket	10 minutes	Set Up an OCI Object Storage Bucket
Export a project's data in the source DevCS instance to an OCI bucket	15 minutes	Export Project Data From the Source DevCS Instance
Create a project with exported project's data in the target DevCS instance.	10 minutes	Create a Project With an Exported Project's Data in the Target DevCS Instance
Migrate the artifacts that weren't exported from the source DevCS project to the target DevCS project.	1.5 - 2 hours for all artifacts of a project	Migrate Other Artifacts and Data
Clean up the source DevCS instance	20 minutes	Clean Up Resources in Traditional

2 Prepare to Migrate DevCS to Oracle Cloud Infrastructure

Before you migrate your service instance from Traditional to Oracle Cloud Infrastructure (OCI), you should create and configure required resources, and learn about the required identity domain roles for the migration.

In this documentation, DevCS on Traditional is referred as the source DevCS instance and DevCS on OCI is referred as the target DevCS instance. A project in the source or target DevCS instance is referred as the source project or the target project.

To prepare for migration, you'll do the following:

1. Migrate users and roles from Traditional to Identity Cloud Service
2. Assign the required identity domain roles to yourself in the source and the target Oracle Cloud account
3. Create the target DevCS instance in the OCI region
4. Set up the required OCI connections in the target Oracle Cloud account
5. Set up an OCI Object Storage bucket in the target Oracle Cloud account.
   You'll use the bucket to export the data of the source DevCS projects.

Migrate Users and Roles

You can migrate traditional cloud accounts to cloud accounts with Identity Cloud Service by migrating users and their roles. Make sure that you have administrative privileges to export users and role memberships from the traditional cloud account and to import users and role memberships in Oracle Identity Cloud Service.

To migrate users and roles, see Migrating Users, Migrating Role Memberships, and Migrating Identity Domain Administrator Roles in *Administering Oracle Identity Cloud Service*. To learn about how DevCS roles are mapped between a traditional cloud account and Identity Cloud Service, see Mapping Between Traditional Cloud Roles and Application Roles in Oracle Identity Cloud Service.

Required Identity Domain Roles

Make sure you're assigned the identity domain roles required for the migration.

You need this role:	To:
`DEVCS_APP_ENTITLEMENT_ADMINISTRATOR` (Administrator Role for Developer Cloud Service Provisioning)	Create the target DevCS instance You need this role in the target Oracle Cloud account.

You need this role:	To:
`DEVELOPER_ADMINISTRATOR` (Developer Service Administrator)	Get VMs and VM templates details from the source DevCS instance and create them in the target DevCS instance You need this role in the source as well as the target Oracle Cloud account.
`OCI_Administrator` (OCI Administrator)	Set up the OCI account You need this role in the target Oracle Cloud account.

To find out how to grant an identity domain role, see Add Users, Assign Policies and Roles in *Getting Started with Oracle Cloud*.

Create the Target DevCS Instance in the OCI Region

If your source and target instances are located in the same Oracle Cloud account, remember that they can't have identical instance names.

1. Sign in to Oracle Cloud and open the My Services Dashboard page.
2. In the **Developer** tile, click **Action** ☰ and select **Open Service Console**.

   If the Developer tile isn’t available on the page, click **Customize Dashboard**. Under Platform, find the Developer service, click **Show**, and then close the Customize Dashboard window.
3. In the **Instances** tab, click **Create Instance**.
4. On the Create New Instance page, enter a unique name in **Instance**. In **Description**, enter a description.

   The name helps you to identify the service instance in the tenant domain.
5. Click **Next**.
6. On the Service Details page, click **Next**.
7. On the Confirmation page, click **Create**.

Set Up the OCI Connections on the Target Oracle Cloud Account

After creating the target DevCS instance, set up your target Oracle Cloud account to host DevCS resources and connect to OCI Compute and OCI Object Storage. DevCS runs builds on OCI Compute VMs, and stores build and Maven artifacts on the OCI Object Storage buckets.

Set Up the OCI Account

To set up the account, sign in as the OCI administrator and follow these steps:

1. Open the OCI dashboard.

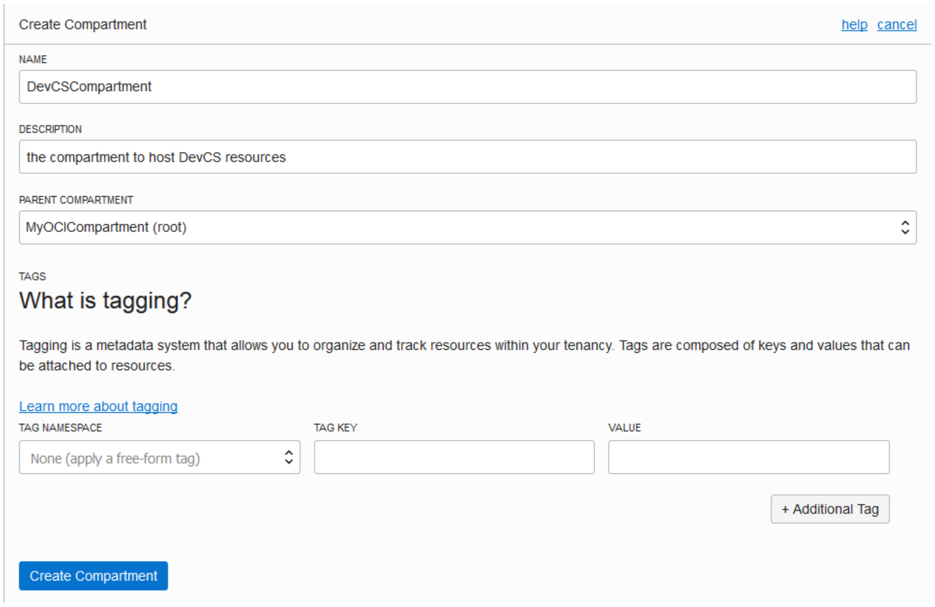
   See Access Oracle Cloud Infrastructure Services.
2. On the Compartments page, create a compartment to host DevCS resources.

a. In the left navigation bar, under **Governance and Administration**, go to **Identity** and click **Compartments**.

b. To create the compartment in the tenancy (root compartment), click **Create Compartment**.

c. In the Create Compartment dialog box, fill in the fields, and click **Create Compartment**.

Here's an example:



Create Compartment

NAME: DevCSCompartment

DESCRIPTION: the compartment to host DevCS resources

PARENT COMPARTMENT: MyOCICompartment (root)

TAGS

What is tagging?

Tagging is a metadata system that allows you to organize and track resources within your tenancy. Tags are composed of keys and values that can be attached to resources.

TAG NAMESPACE: None (apply a free-form tag)

To learn more about compartments, see Working with Compartments.

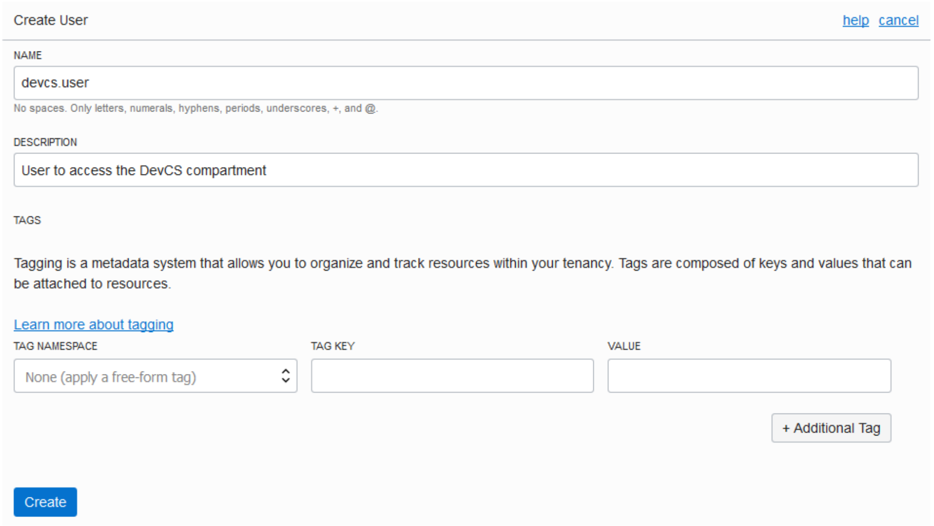
3. Create a user to access the DevCS compartment.

a. In the left navigation bar, under **Governance and Administration**, go to **Identity** and click **Users**.

b. Click **Create User**.

c. In the Create User dialog box, fill in the fields, and click **Create**.

Here's an example:



NAME

devcs.user

No spaces. Only letters, numerals, hyphens, periods, underscores, +, and @.

DESCRIPTION

User to access the DevCS compartment

TAGS

Tagging is a metadata system that allows you to organize and track resources within your tenancy. Tags are composed of keys and values that can be attached to resources.

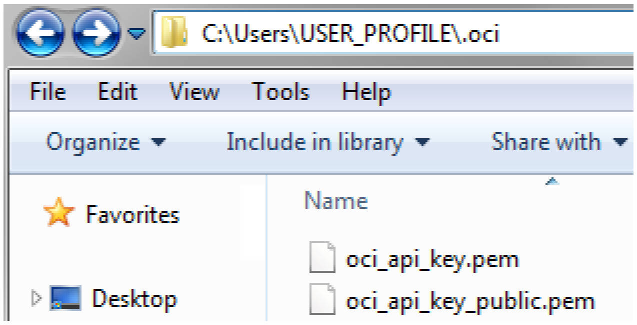
Learn more about tagging

TAG NAMESPACE	TAG KEY	VALUE
None (apply a free-form tag)		

To learn more about OCI users, see Working with Users.

4. On your computer, generate a private-public key pair in the PEM format.

   To find out how to generate a private-public key pair in the PEM format, see How to Generate an API Signing Key.
   Here's an example of private-public key files on a Windows computer:

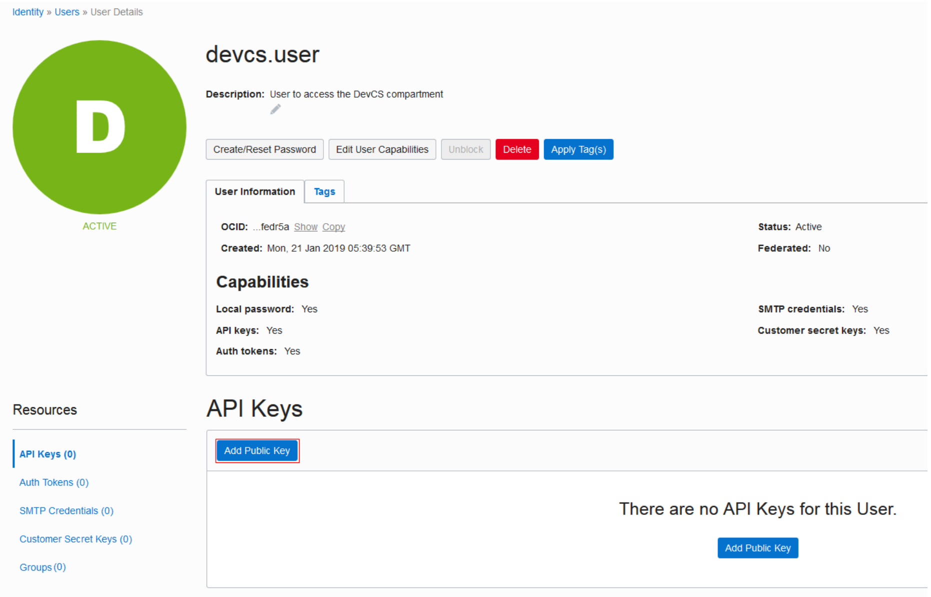


C:\Users\USER_PROFILE\.oci

oci_api_key.pem
oci_api_key_public.pem

5. Upload the public key to the user's details page.
   a. Open the public key file in a text editor and copy its contents.
   b. In the left navigation bar of the OCI dashboard, click under **Governance and Administration**, go to **Identity** and click **Users**.
   c. Click the user's name created in Step 3.
   d. In the User Details page, click **Add Public Key**.

      Here's an example:



devcs.user

Description: User to access the DevCS compartment

OCID: ...fedr5a Show Copy

Created: Mon, 21 Jan 2019 05:39:53 GMT

Status: Active

Federated: No

Capabilities

Local password: Yes

API keys: Yes

Auth tokens: Yes

SMTP credentials: Yes

Customer secret keys: Yes

API Keys

There are no API Keys for this User.

e. In the Add Public Key dialog box, paste the contents of the public key file, then click **Add**.

To learn more about uploading keys, see How to Upload the Public Key.

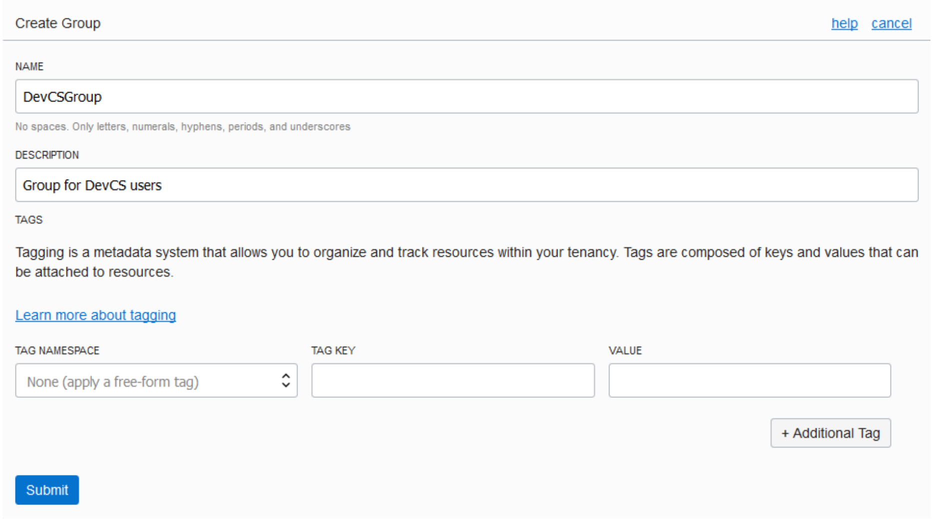
6. On the Groups page, create a group for the user who can access the DevCS compartment and add the user to the group.

a. In the left navigation bar, under **Governance and Administration**, go to **Identity** and click **Groups**.

b. Click **Create Group**.

c. In the Create Group dialog, fill in the fields and click **Submit**.

Here's an example:



Create Group

NAME

DevCSGroup

No spaces. Only letters, numerals, hyphens, periods, and underscores

DESCRIPTION

Group for DevCS users

TAGS

Tagging is a metadata system that allows you to organize and track resources within your tenancy. Tags are composed of keys and values that can be attached to resources.

Learn more about tagging

TAG NAMESPACE: None (apply a free-form tag)

TAG KEY:

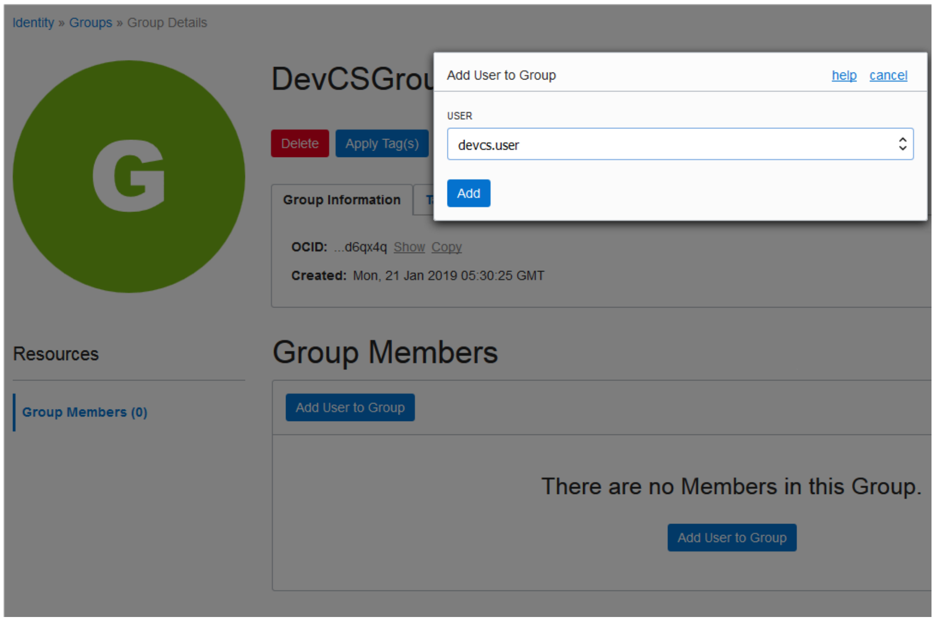
VALUE:

d. On the Groups page, click the group's name.

e. On the Group Details page, click **Add User to Group**.

f. In the Add User to Group dialog box, select the user created in Step 3, and click **Add**.

   Here's an example:



To learn more about groups, see Working with Groups.

7. In the root compartment, not the DevCS compartment, create a policy to allow the group created in step 6 to access the DevCS compartment.

   a. In the left navigation bar, under **Governance and Administration**, go to **Identity** and click **Policies**.

   b. On the left side of the Policies page, from the **Compartment** list, select the root compartment.

   c. Click **Create Policy**.

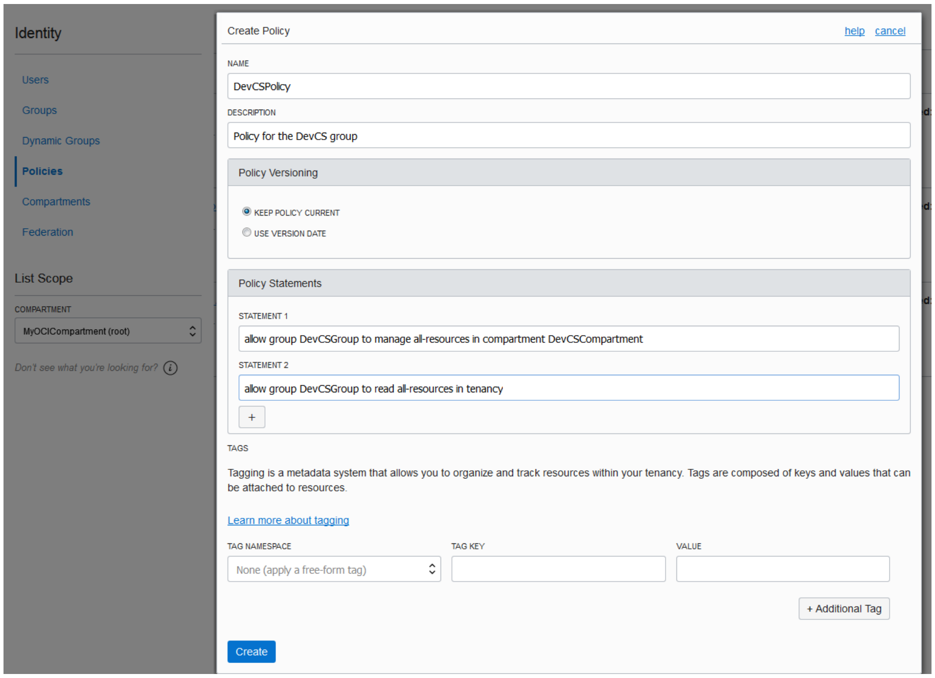
   d. In **Name** and **Description**, enter a unique name and a description.

   e. In **Policy Statements**, add these statements.

      - `allow group <group-name> to manage all-resources in compartment <compartment-name>`
        This grants all permissions to the DevCS group users to manage all resources within the DevCS compartment.

      - `allow group <group-name> to read all-resources in tenancy`
        This grants read permissions to the DevCS group so that its users can read—but not use, create or modify—all resources inside and outside the DevCS compartment. The group users can't use, create, or modify the resources. This statement is optional.

      Here's an example:



Create Policy

NAME: DevCSPolicy

DESCRIPTION: Policy for the DevCS group

Policy Versioning

● KEEP POLICY CURRENT

○ USE VERSION DATE

Policy Statements

STATEMENT 1: allow group DevCSGroup to manage all-resources in compartment DevCSCompartment

STATEMENT 2: allow group DevCSGroup to read all-resources in tenancy

TAGS

Tagging is a metadata system that allows you to organize and track resources within your tenancy. Tags are composed of keys and values that can be attached to resources.

Learn more about tagging

TAG NAMESPACE: None (apply a free-form tag)

TAG KEY:

VALUE:

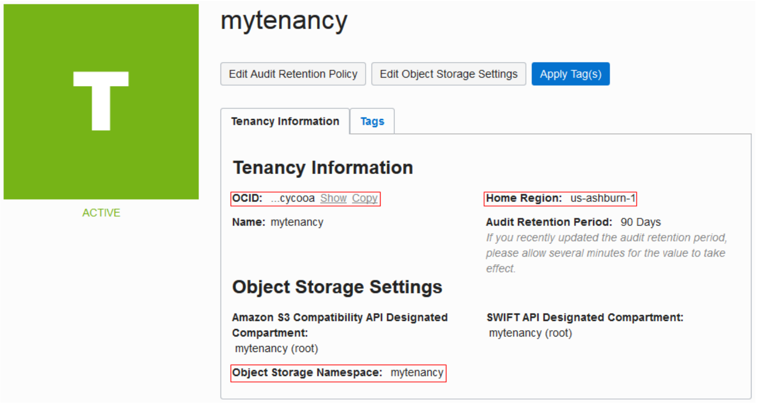
f. Click **Create**.

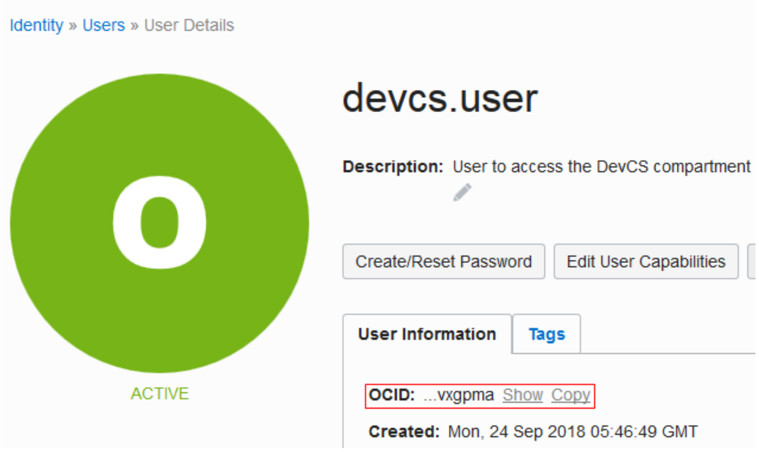
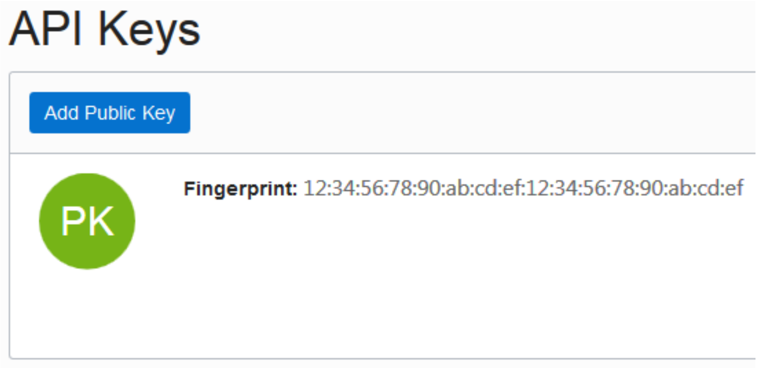
To learn more about policies, see Working with Policies.

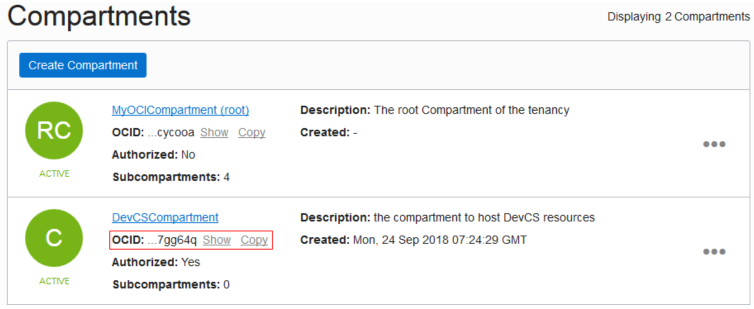
Get the Required OCI Input Values

Every Oracle Cloud Infrastructure resource has an Oracle-assigned unique ID called an Oracle Cloud Identifier (OCID). To connect to OCI, you need the account's tenancy OCID, home region, the compartment's OCID that hosts DevCS resources, and the OCID and fingerprint of the user who can access the DevCS compartment. To connect to OCI Object Storage, you need the Storage namespace. You can get these values from the OCI Console pages.

This table describes how to get the OCI input values required for the connection.

To get these values ...	Do this:
Tenancy OCID, Home Region, and Storage Namespace	On the OCI console, from the left navigation bar, select **Administration** > **Tenancy Details**. The **Tenancy Information** tab displays the tenancy OCID in **OCID**, home region in **Home Region**, and the storage namespace in **Object Storage Namespace**. Here's an example:  T ACTIVE Edit Audit Retention Policy Edit Object Storage Settings Apply Tag(s) Tenancy Information Tags mytenancy Tenancy Information **OCID:** ...cycooa Show Copy **Home Region:** us-ashburn-1 **Name:** mytenancy **Audit Retention Period:** 90 Days *If you recently updated the audit retention period, please allow several minutes for the value to take effect.* Object Storage Settings **Amazon S3 Compatibility API Designated Compartment:** mytenancy (root) **SWIFT API Designated Compartment:** mytenancy (root) **Object Storage Namespace:** mytenancy

To get these values ...	Do this:
User OCID and Fingerprint	On the OCI console, from the left navigation bar, under **Governance and Administration**, select **Identity** > **Users**. The **User Information** tab displays the user OCID in **OCID**. Click the **Copy** link to copy it to the clipboard. Here's an example of `devcs.user`:  Identity » Users » User Details O ACTIVE Create/Reset Password Edit User Capabilities User Information Tags Show Copy devcs.user **Description:** User to access the DevCS compartment **OCID:** ...vxgpma **Created:** Mon, 24 Sep 2018 05:46:49 GMT To get the fingerprint of the public key associated with your OCI account, scroll down to the **API Keys** section and copy the fingerprint value.  **Fingerprint:** 12:34:56:78:90:ab:cd:ef:12:34:56:78:90:ab:cd:ef

To get these values ...	Do this:
Compartment OCID	On the OCI console, from the left navigation bar, select **Identity** > **Compartments**. The Compartments list displays the compartments with the compartment OCID in the **OCID** field. Click the **Copy** link to copy it to the clipboard. Here's an example:  Compartments Displaying 2 Compartments Create Compartment RC ACTIVE MyOCICompartment (root) **OCID:** ...cycooa Show Copy **Authorized:** No **Subcompartments:** 4 **Description:** The root Compartment of the tenancy **Created:** - C ACTIVE DevCSCompartment **OCID:** ...7gg64q Show Copy **Authorized:** Yes **Subcompartments:** 0 **Description:** the compartment to host DevCS resources **Created:** Mon, 24 Sep 2018 07:24:29 GMT

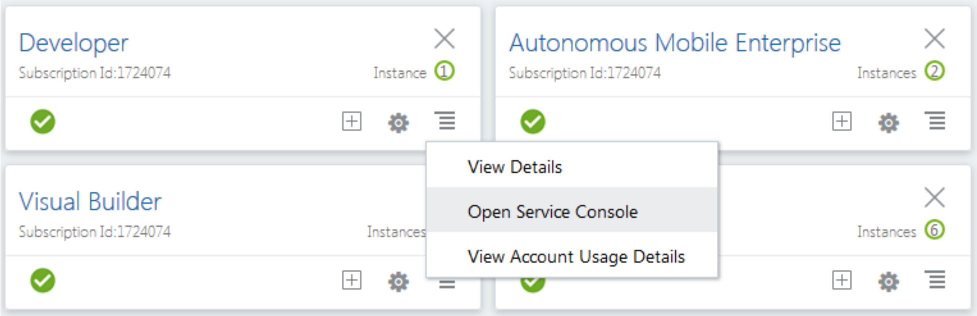
Set Up the OCI Connection in the Target DevCS Instance

To connect to OCI, get the DevCS compartment, user details, and the required OCID values. Then, create an OCI connection from DevCS. If you're not the OCI administrator, get the details from the OCI administrator.

You must be the **Organization Administrator** to create the connection.

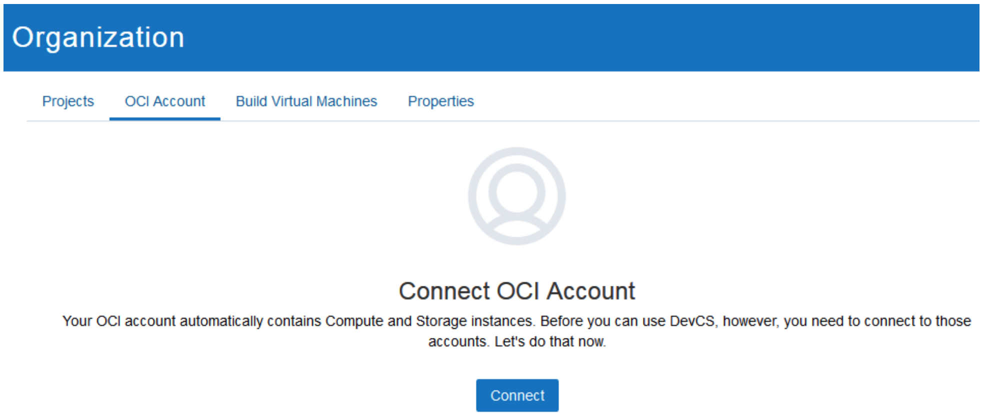
1. Sign in to the target DevCS instance.

   On the Oracle Cloud Dashboard, in the **Developer** tile, click **Action** ≣ and select **Open Service Console**.

   If the Developer tile isn't visible, click **Customize Dashboard**. Under Platform, find the Developer service, click **Show**, and then close the Customize Dashboard window.

2. In the **Instances** tab, for your DevCS instance, click **Manage this instance** and select **Access Service Instance**.
3. In the navigation bar, click **Organization** .
4. Click the **OCI Account** tab.
5. To create a connection, click **Connect**.

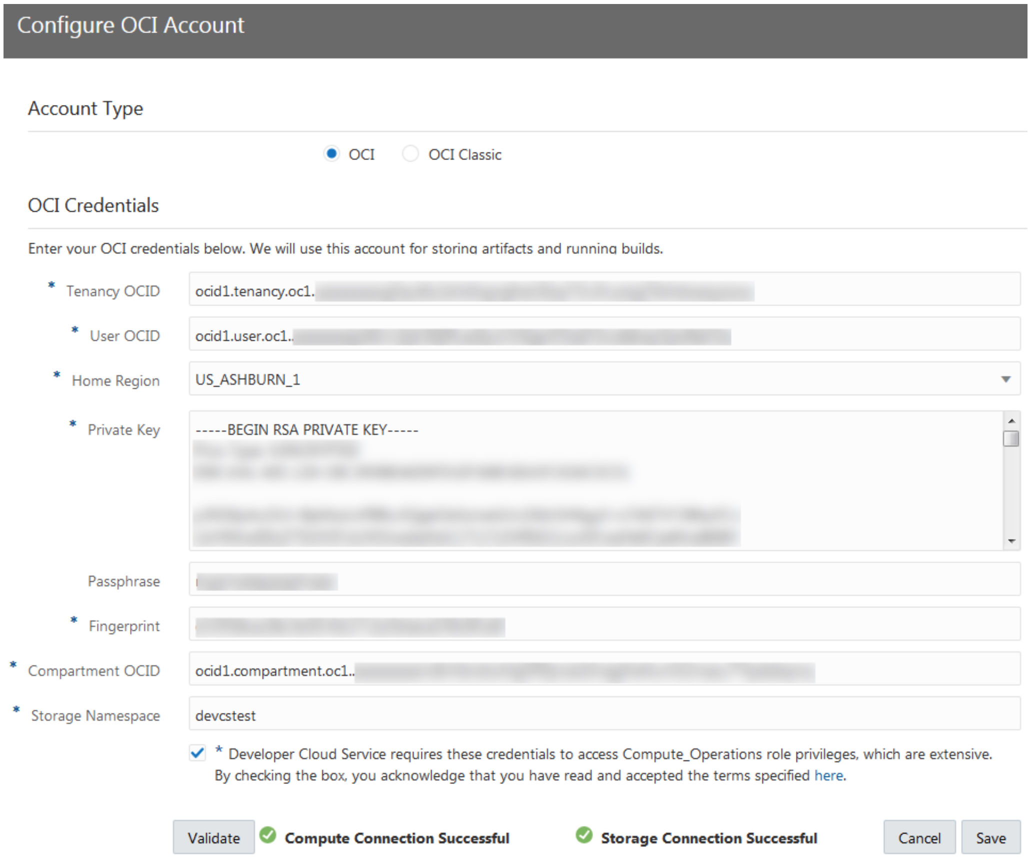


Connect OCI Account

Your OCI account automatically contains Compute and Storage instances. Before you can use DevCS, however, you need to connect to those accounts. Let's do that now.

6. In **Account Type**, select **OCI**.
7. In **Tenancy OCID**, enter the tenancy's OCID copied from the Tenancy Details page.
8. In **User OCID**, enter the OCID of the user who can access the DevCS compartment.
9. In **Home Region**, select the home region of the OCI account.
10. In **Private Key**, enter the private key of the user who can access the DevCS compartment.
11. In **Passphrase**, enter the passphrase used to encrypt the private key. If no passphrase was used, leave the field empty.
12. In **Fingerprint**, enter the fingerprint value of the private-public key pair.
13. In **Compartment OCID**, enter the compartment's OCID copied from the Compartments page.
14. In **Storage Namespace**, enter the storage namespace copied from the Tenancy Details page.
15. To agree to terms and conditions, select the terms and conditions check box.
16. To validate the connection details, click **Validate**.
17. After validating the connection details, click **Save**.

Here's an example of an **OCI Account** tab filed with required OCI details.



Configure OCI Account

Account Type

● OCI ○ OCI Classic

OCI Credentials

Enter your OCI credentials below. We will use this account for storing artifacts and running builds.

* Tenancy OCID	ocid1.tenancy.oc1.[redacted]
* User OCID	ocid1.user.oc1.[redacted]
* Home Region	US_ASHBURN_1
* Private Key	-----BEGIN RSA PRIVATE KEY----- [redacted]
Passphrase	[redacted]
* Fingerprint	[redacted]
* Compartment OCID	ocid1.compartment.oc1..[redacted]
* Storage Namespace	devcstest

☑ * Developer Cloud Service requires these credentials to access Compute_Operations role privileges, which are extensive. By checking the box, you acknowledge that you have read and accepted the terms specified here.

Validate ✓ Compute Connection Successful ✓ Storage Connection Successful Cancel Save

Set Up an OCI Object Storage Bucket

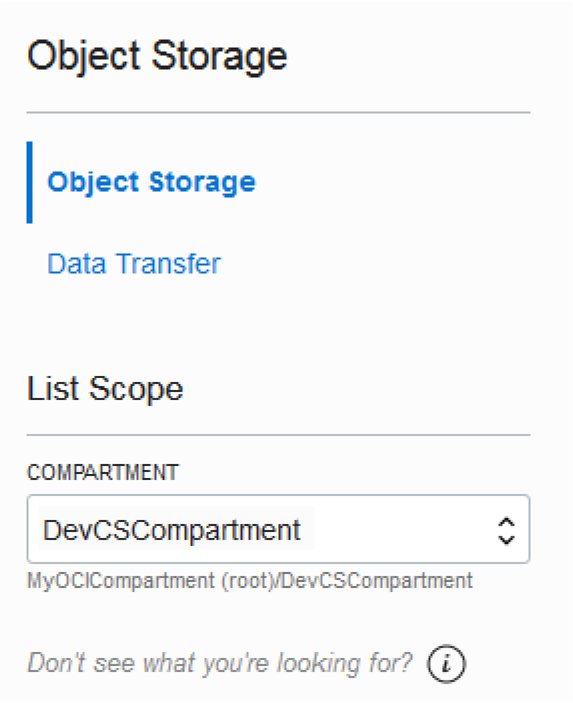
To export a project's data, you need an OCI Object Storage bucket to host the data.

Set Up the OCI Object Storage Bucket

To set up the OCI Object Storage bucket, sign in as the OCI administrator and follow these steps:

1. In the compartment that hosts DevCS resources, create a bucket for the project.
   a. In the left navigation bar, under **Core Infrastructure**, go to **Object Storage** and click **Object Storage**.
   b. On the left side of the Object Storage page, from the **Compartment** list, select the DevCS compartment.

      Example:



COMPARTMENT

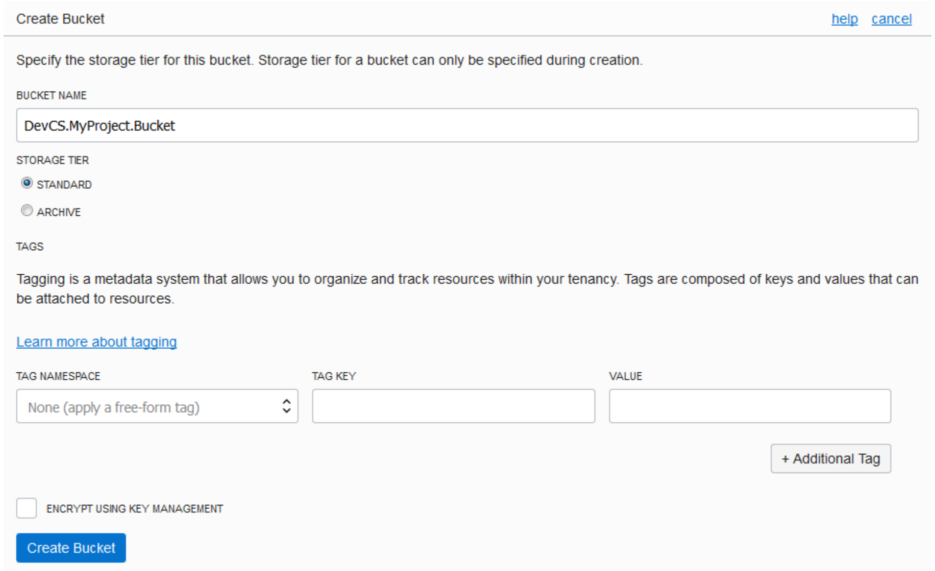
DevCSCompartment

MyOCICompartment (root)/DevCSCompartment

c. Click **Create Bucket**.

d. In the Create Bucket dialog box, fill in the details, and click **Create Bucket**.

   Example:



Create Bucket

Specify the storage tier for this bucket. Storage tier for a bucket can only be specified during creation.

BUCKET NAME

DevCS.MyProject.Bucket

STORAGE TIER

● STANDARD

○ ARCHIVE

TAGS

Tagging is a metadata system that allows you to organize and track resources within your tenancy. Tags are composed of keys and values that can be attached to resources.

TAG NAMESPACE	TAG KEY	VALUE
None (apply a free-form tag)		

□ ENCRYPT USING KEY MANAGEMENT

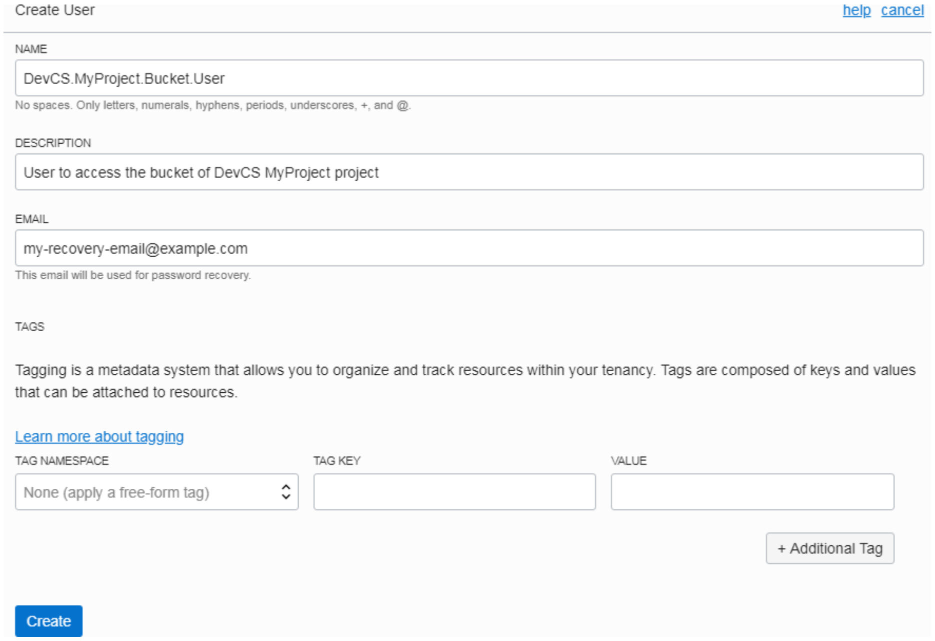
2. Create a user to access the bucket.

   a. In the left navigation bar, under **Governance and Administration**, go to **Identity** and click **Users**.

   b. Click **Create User**.

   c. In the Create User dialog box, fill in the fields, and click **Create**.

      Example:



Create User

NAME: DevCS.MyProject.Bucket.User

No spaces. Only letters, numerals, hyphens, periods, underscores, +, and @.

DESCRIPTION: User to access the bucket of DevCS MyProject project

EMAIL: my-recovery-email@example.com

This email will be used for password recovery.

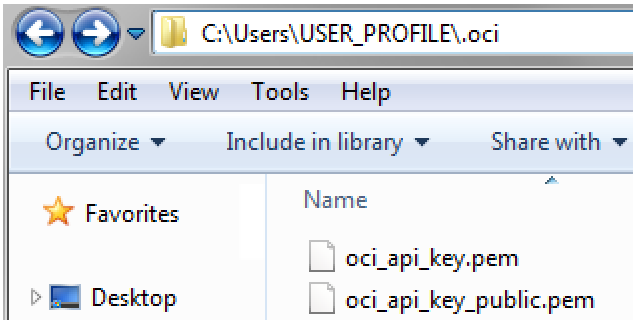
TAGS

Tagging is a metadata system that allows you to organize and track resources within your tenancy. Tags are composed of keys and values that can be attached to resources.

TAG NAMESPACE	TAG KEY	VALUE
None (apply a free-form tag)		

3. On your computer, generate a private-public key pair in the PEM format.

   To learn more, see How to Generate an API Signing Key.
   Example of private-public key files on a Windows computer:

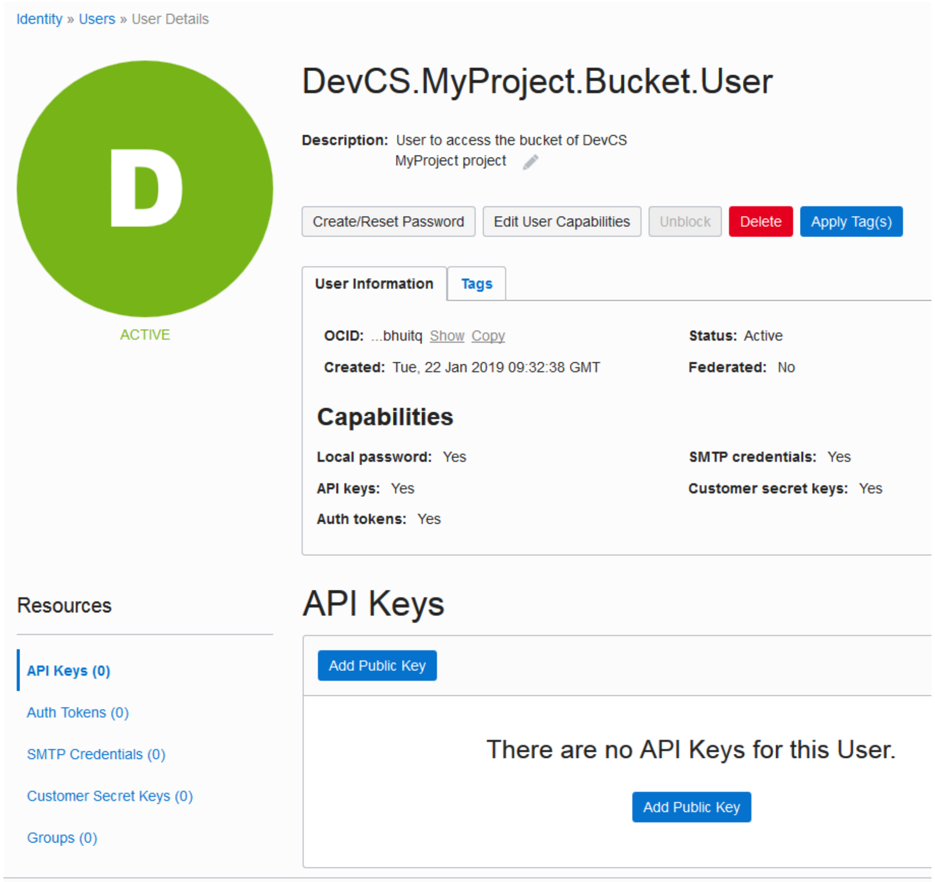
   

   C:\Users\USER_PROFILE\.oci

   oci_api_key.pem
   oci_api_key_public.pem

4. Upload the public key to the user's details page.
   a. Open the public key file in a text editor and copy its contents.
   b. In the left navigation bar of the OCI dashboard, under **Governance and Administration**, go to **Identity** and click **Users**.
   c. Click the user's name created in Step 2.
   d. In the User Details page, click **Add Public Key**.

      Example:



DevCS.MyProject.Bucket.User

Description: User to access the bucket of DevCS MyProject project

OCID: ...bhuitq

Status: Active

Created: Tue, 22 Jan 2019 09:32:38 GMT

Federated: No

Capabilities

Local password: Yes

SMTP credentials: Yes

API keys: Yes

Customer secret keys: Yes

Auth tokens: Yes

API Keys

There are no API Keys for this User.

e. In the Add Public Key dialog box, paste the contents of the public key file, and click **Add**.

To learn more, see How to Upload the Public Key.

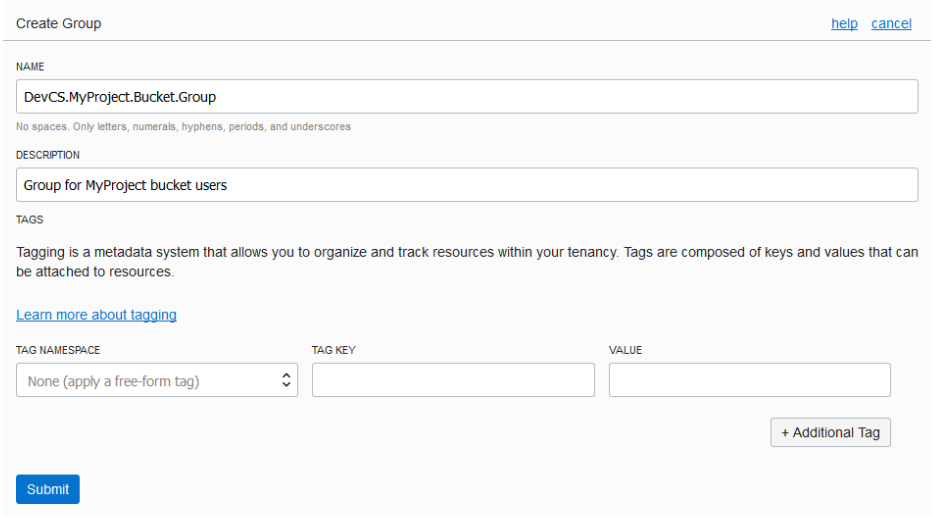
5. On the Groups page, create a group for the user who can access the bucket and add the user to the group.

   a. In the left navigation bar, under **Governance and Administration**, go to **Identity** and click **Groups**.

   b. Click **Create Group**.

   c. In the Create Group dialog, fill in the fields and click **Submit**.

   Example:



Create Group

NAME

DevCS.MyProject.Bucket.Group

No spaces. Only letters, numerals, hyphens, periods, and underscores

DESCRIPTION

Group for MyProject bucket users

TAGS

Tagging is a metadata system that allows you to organize and track resources within your tenancy. Tags are composed of keys and values that can be attached to resources.

Learn more about tagging

TAG NAMESPACE

None (apply a free-form tag)

TAG KEY

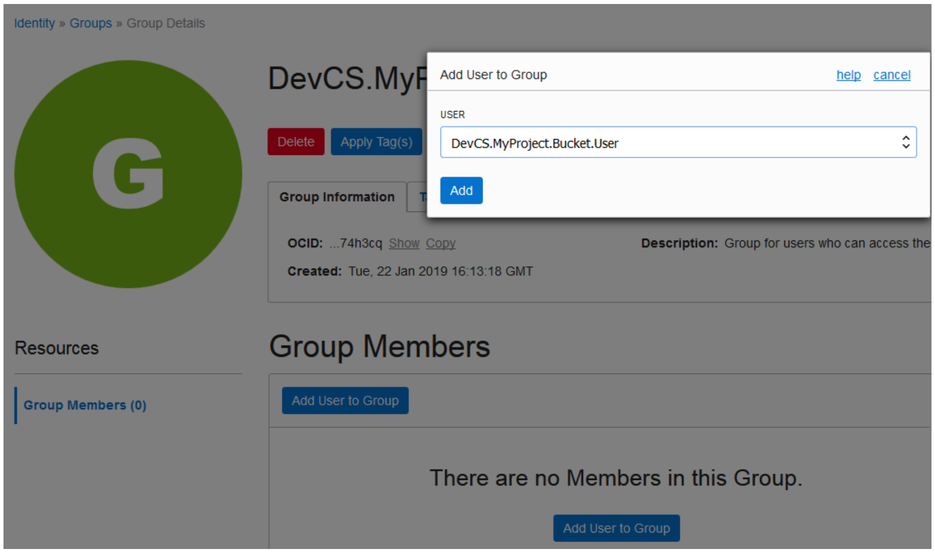
VALUE

d. On the Groups page, click the group's name.

e. On the Group Details page, click **Add User to Group**.

f. In the Add User to Group dialog box, select the user created in Step 2, and click **Add**.

Example:



Identity » Groups » Group Details

DevCS.MyP

Add User to Group

USER

DevCS.MyProject.Bucket.User

Group Information

OCID: ...74h3cq Show Copy

Created: Tue, 22 Jan 2019 16:13:18 GMT

Description: Group for users who can access the

Resources

Group Members (0)

Group Members

There are no Members in this Group.

To learn more, see Working with Groups.

6. In the DevCS compartment, create a policy with read and write access to the bucket.

   You can give read and write access to the same user, or create different users.

   a. In the left navigation bar, under **Governance and Administration**, go to **Identity** and click **Policies**.

   b. On the left side of the Policies page, from the **Compartment** list, select the DevCS compartment.

   c. Click **Create Policy**.

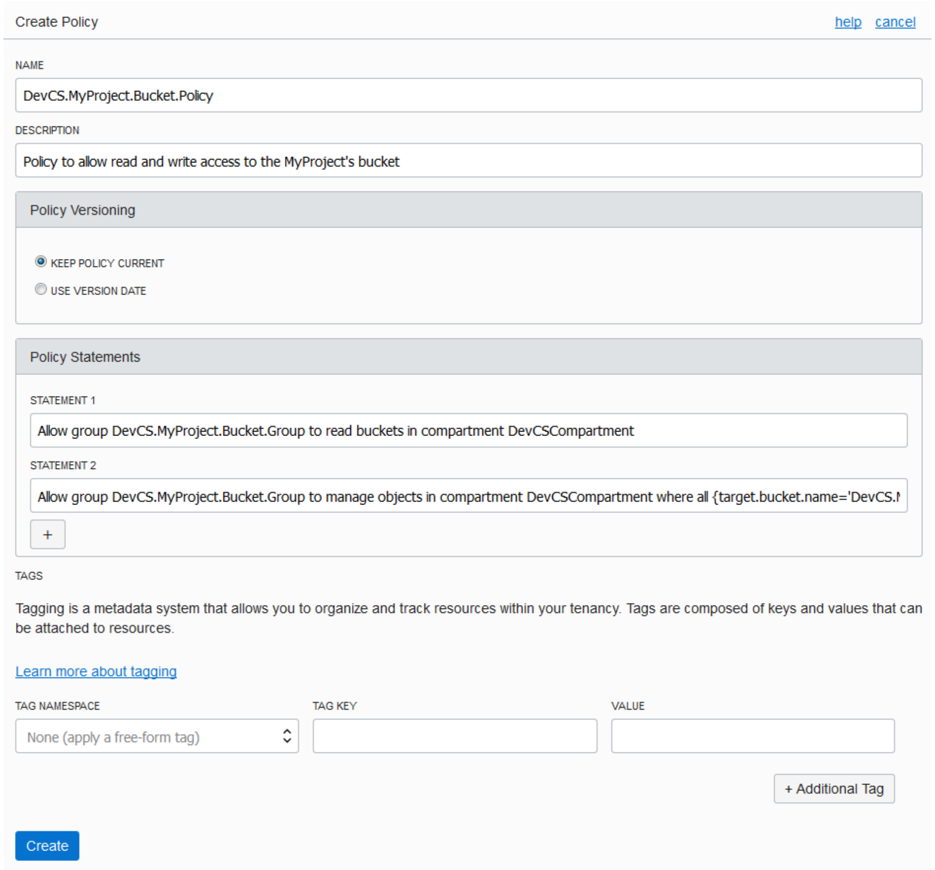
d. In **Name** and **Description**, enter a unique name and a description.

e. In **Policy Statements**, add statements to restrict read and write access to the bucket.

   To allow different user groups to read objects from and write objects to the bucket, create separate policies. Here are some statement examples:

To:	Add these statements:
Allow a group to read from and write objects to a bucket (required to import and export a project's data)	allow group <*group-name*> to read buckets in compartment <*compartment-name*> allow group <*group-name*> to manage objects in compartment <*compartment-name*> where all {target.bucket.name='<bucket-name>', any {request.permission='OBJECT_CREATE', request.permission='OBJECT_INSPECT'}}
Allow a group to download objects from a bucket (required to import a project's data)	allow group <*group-name*> to read buckets in compartment <*compartment-name*> allow group <*group-name*> to read objects in compartment <*compartment-name*> where target.bucket.name='<bucket-name>'

   Example:



Create Policy

NAME

DevCS.MyProject.Bucket.Policy

DESCRIPTION

Policy to allow read and write access to the MyProject's bucket

Policy Versioning

● KEEP POLICY CURRENT

○ USE VERSION DATE

Policy Statements

STATEMENT 1

Allow group DevCS.MyProject.Bucket.Group to read buckets in compartment DevCSCompartment

STATEMENT 2

Allow group DevCS.MyProject.Bucket.Group to manage objects in compartment DevCSCompartment where all {target.bucket.name='DevCS.M

TAGS

Tagging is a metadata system that allows you to organize and track resources within your tenancy. Tags are composed of keys and values that can be attached to resources.

Learn more about tagging

TAG NAMESPACE	TAG KEY	VALUE
None (apply a free-form tag)		

f. Click **Create**.

To learn more, see Working with Policies.

3 Migrate an Oracle Developer Cloud Service Instance to Oracle Cloud Infrastructure

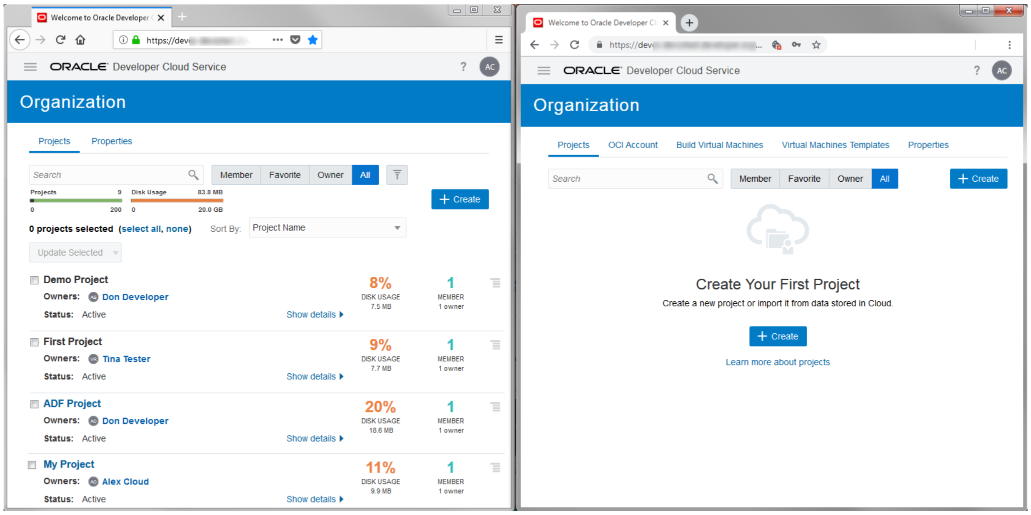
Migrating your projects from a source DevCS instance to a target involves migrating the Build VM and VM templates of the source instance, as well as each project's data.

In this documentation, the icon indicates either the source DevCS instance or the source project. The icon indicates either the target DevCS instance or the target project.

Open the Source and the Target DevCS Instances

Open the source DevCS instance in a browser and the target DevCS instance in another browser. You can't open both instances in the same session of a browser; however, you may open an instance in a tab and another instance in an incognito or a private tab of the same browser.

For example, in this image the source DevCS instance is open in Mozilla Firefox and the target DevCS instance is open in Google Chrome.

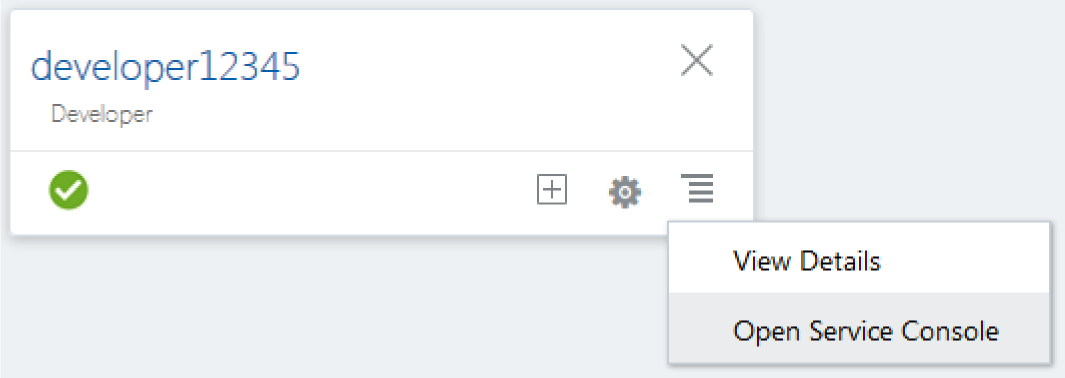


To view the list of supported browsers, see `https://www.oracle.com/technetwork/indexes/products/browser-policy-2859268.html`.

To open the source DevCS instance:

1. In a web browser, go to `https://cloud.oracle.com/home`, and click **Sign In**.
2. On the Sign In page, click **Sign In using Traditional Cloud Account**.
3. From the **Data Center** drop-down list, select your data center.

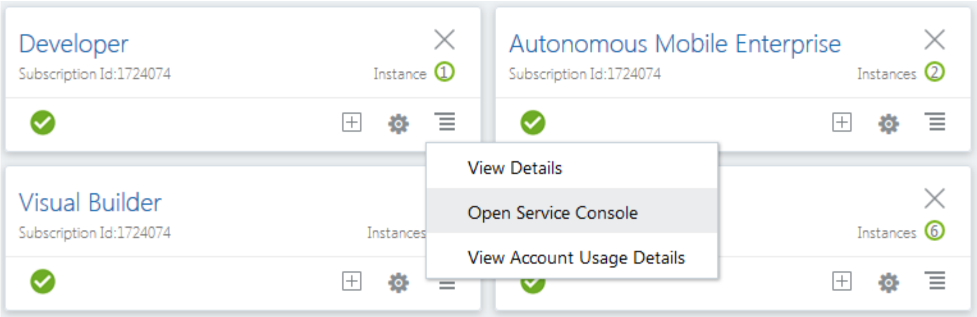
4. In **Traditional Cloud Account Name**, enter your identity domain name and click **Go**.
5. Enter your Oracle Cloud account credentials, and click **Sign In**.
6. On the Oracle Cloud Dashboard, in the **developerNNNNN** tile (where NNNNN is a number), click **Action** ≣ and select **Open Service Console**.



If the **developerNNNNN** tile isn't visible, click **Customize Dashboard**. Under **Platform**, find the Developer service, click **Show**, and then close the Customize Dashboard window.

To open the target DevCS instance:

1. In another web browser, go to `https://cloud.oracle.com/home`, and click **Sign In**.
2. From the **Cloud Account** drop-down list, select **Cloud Account with Identity Cloud Service.** In **Cloud Account Name,** enter your tenant name or the identity domain name.
3. Click **Next**.
4. On the sign-in page, enter your Oracle Cloud account credentials, and click **Sign In**.
5. Click **Dashboard**.
6. On the Oracle Cloud Dashboard, in the **Developer** tile, click **Action** ≣ and select **Open Service Console**.



If the Developer tile isn't visible, click **Customize Dashboard**. Under Platform, find the Developer service, click **Show**, and then close the Customize Dashboard window.

7. On the **Instances** tab, click **Manage this instance** ☰ and select **Access service instance**.

Migrate the Organization's Properties

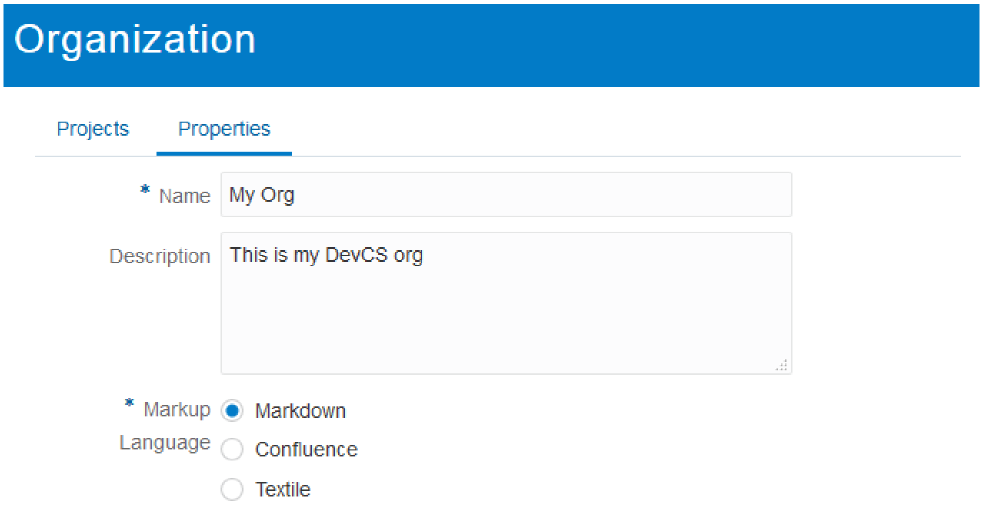
To migrate the source DevCS instance's organization properties, note its details and then configure the target DevCS instance's Organization with the same properties.

Gather Information About the Organization From the Source DevCS Instance

Perform these steps in the source DevCS instance.

1. Switch to the browser with the source DevCS instance.
2. In the navigation bar, click **Organization** .
3. Click the **Properties** tab.
4. Note the name, description, and the default wiki markup language of the organization.

   Example:



Organization

* Name	My Org
Description	This is my DevCS org
* Markup Language	● Markdown ○ Confluence ○ Textile

This table shows the name, description, and the default wiki markup language values from the above image.

Field	Value
Name	My Org
Description	This is my DevCS org

Field	Value
Markup Language	Markdown

Configure Organization Properties in the Target DevCS Instance

Perform these steps in the target DevCS instance.

1. Get the details of the Organization's properties from the source DevCS instance. You'll use the information to configure the target DevCS instance's Organization properties.

   Example:

Field	Value
Name	My Org
Description	This is my DevCS org
Markup Language	Markdown

2. Switch to the browser with the target DevCS instance.
3. In the navigation bar, click **Organization** .
4. Click the **Properties** tab.
5. In **Name** and **Description**, update the organization name and description.
6. In **Markup Language**, select the wiki markup language.

   Remember that the organization's wiki markup language defines the default wiki markup language of new projects, which you can change while creating a project.

When you're done, verify that the Organization's properties set in the target DevCS instance match the source DevCS instance.

Migrate Projects

To migrate projects from the source DevCS instance, you will export each project's data to an OCI Object Storage bucket.

You'll then create projects in the target DevCS instance using the exported data. Finally, you will copy any data or artifacts that were not part of the export from the source DevCS instance to the target DevCS instance.

Gather Information About a Project From the Source DevCS Instance

Note the name, description, security, and the wiki markup language of the project in the source DevCS instance.

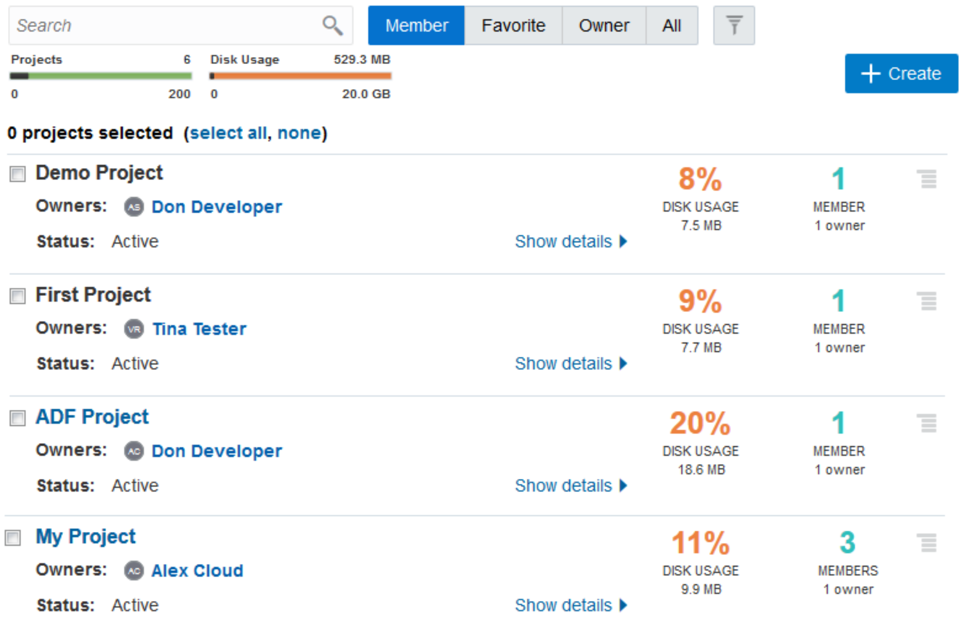
To get the details, you must be assigned the project's **Owner** role. If the role is not assigned to you, sign in as the **Organization Administrator** and assign the **Owner** role to yourself.

Perform these steps in the source DevCS instance.

1. Switch to the browser with the source DevCS instance.

2. In the navigation bar, click **Organization** .
3. In the **Projects** tab, click the **All** toggle button.
4. Find the project to migrate. If you can't find it in the list, use the Search box.
5. After finding the project, note whether the project's name is a link and whether you are an owner. Projects where you're not a member don't appear as links. If you're a member, but not an owner, your name wouldn't appear in the **Owners** list.

   In this example, you're signed in as Alex Cloud. Note that some projects aren't links.



0 projects selected (select all, none)

□ Demo Project
Owners: Don Developer
Status: Active
Show details
8% DISK USAGE 7.5 MB
1 MEMBER 1 owner

□ First Project
Owners: Tina Tester
Status: Active
Show details
9% DISK USAGE 7.7 MB
1 MEMBER 1 owner

□ ADF Project
Owners: Don Developer
Status: Active
Show details
20% DISK USAGE 18.6 MB
1 MEMBER 1 owner

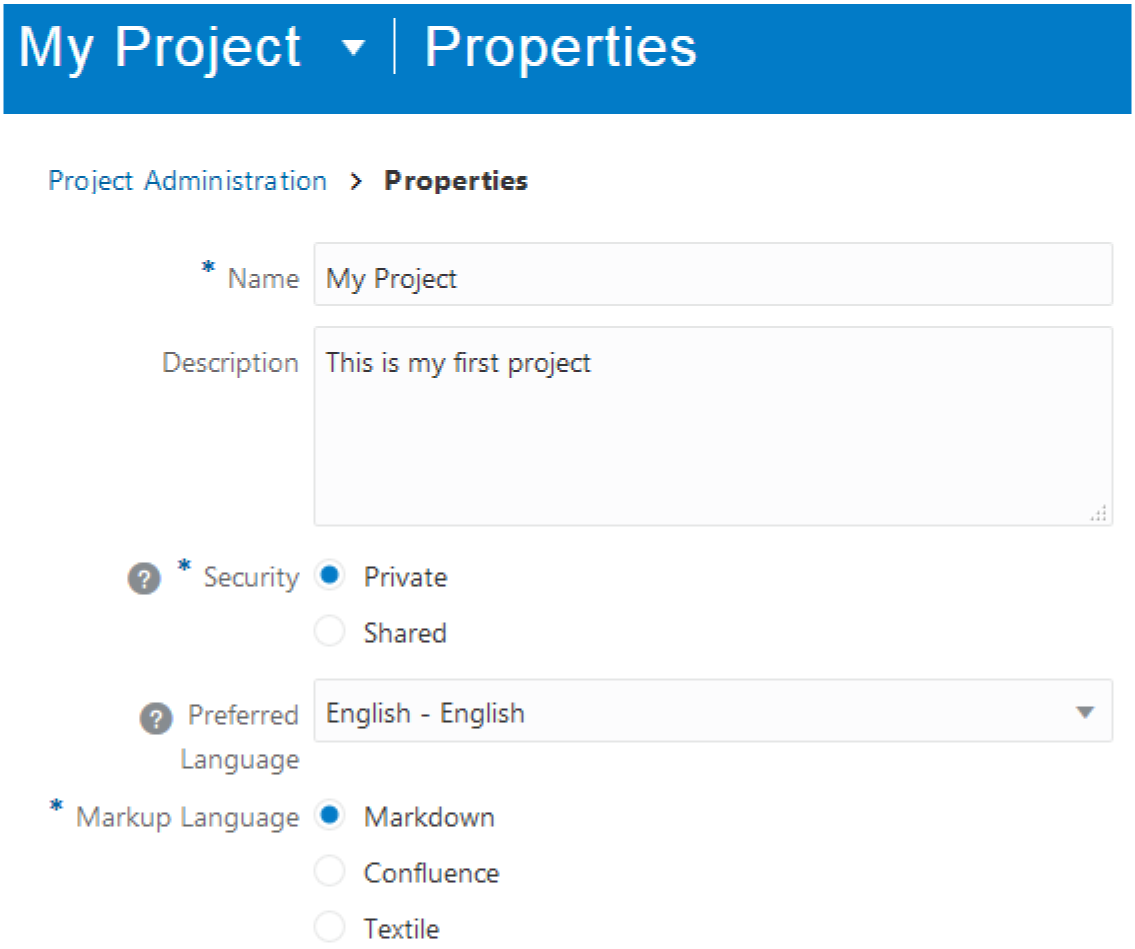
□ My Project
Owners: Alex Cloud
Status: Active
Show details
11% DISK USAGE 9.9 MB
3 MEMBERS 1 owner

   This table shows the project's name and information whether you're a member of the project.

Field	Value
Project Name	My Project
Are you a Member?	Yes
Are you an Owner?	Yes

6. If you're not an owner or the project doesn't appear as a link, select the project's check box. From **Update Selected**, select **Assign Me as Owner**.
7. Click the project name.
8. In the navigation bar, click **Project Administration** .
9. Click **Properties**.
10. Note the values of **Name**, **Description**, **Security**, and **Markup Language**.

    Example:



My Project ▾ | Properties

Project Administration > **Properties**

* Name: My Project

Description: This is my first project

* Security: ● Private ○ Shared

Preferred Language: English - English

* Markup Language: ● Markdown ○ Confluence ○ Textile

This table shows the Project Name, Description, Security, and Wiki values from the above image.

Field	Value
Project Name	My Project
Description	This is my first project
Security	Private
Wiki	Markdown

Export Project Data From the Source DevCS Instance

You can export a project’s data to an OCI Object Storage bucket.

When you export project data, not all the artifacts are included. You'll have to manually export the remaining artifacts and data manually.

This table shows you which artifacts are exported and which aren't:

Artifact	Exported?	Notes
Project users	No	When you export a project's data, its users are not exported, but all data associated to usernames (such as issue ownership and reviewers of a merge request) is preserved. After you import the project's data to another project, when you add a user to the project with the same username, the data associated to the username is automatically restored.
User's favorite settings or personal preferences	No	
Hosted Git repositories	Yes	
Mirrored public external Git repositories	Yes	
Mirrored private external Git repositories	No	Password protected external Git repositories aren't exported. After you import the project's data to another project, you must add each external private Git repository.
Branch restrictions	Yes	
Issues	Yes	
Agile boards	Yes	
Wiki pages	Yes	
Merge Requests	Yes	
Default reviewers of a branch	Yes	After you import the project's data to another project, default reviewers are added automatically after the same users are added to the target project.
Build jobs	No	
Deployment configurations	No	
Maven artifacts	No	
Environments	No	
Releases	Yes	
Snippets	Yes	
Linked Docker registries	No	
Project template definition	No	
Announcements	No	
Webhooks	No	
RSS/ATOM feeds	No	
Link rules	No	
Project tags	Yes	
Issue products and components	Yes	

Artifact	Exported?	Notes
Default owners of issue components	Yes	After you import the project's data to another project, owners are activated automatically after the same users are added to the target project.
Issue custom fields	Yes	
Named passwords	Yes	

Export a Project's Data to an OCI Object Storage Bucket

To export a project's data to an OCI Object Storage bucket, you need the bucket's name, private key and fingerprint of a user who can write objects to the bucket, and details of the compartment that hosts the bucket.

To get the input values, see Get the Required OCI Input Values.

Perform these steps for each project in the source DevCS instance as the project's **Owner**.

1. Open the source project.
2. In the navigation bar, click **Project Administration** ⚙.
3. Click **Data Export/Import**.
4. Click the **Job** tab.
5. In **Account Type**, select **OCI**.
6. In **Tenancy OCID**, enter the tenancy's OCID copied from the Tenancy Details page.
7. In **User OCID**, enter the user's OCID value who can access the bucket.
8. In **Home Region**, select the home region of the OCI account.
9. In **Private Key**, enter the private key of the user who can access the bucket.
10. In **Passphrase**, enter the passphrase used to encrypt the private key. If no passphrase was used, leave the field empty.
11. In **Fingerprint**, enter the fingerprint value of the private-public key pair.
12. In **Compartment OCID**, enter the compartment's OCID copied from the Compartments page.
13. In **Storage Namespace**, enter the storage namespace copied from the Tenancy Details page.
14. Click **Connect**.
15. In the Create Job section, in **Type**, select **Export**.
16. In **Name**, enter a name for the export job.
17. In **Description**, enter the job's description.
18. In **Storage Container**, select the bucket to export the project data.
19. In **Storage Object**, if required, update the default `.zip` file name.
20. Click **Export**.

21. In the Confirm Project Export dialog box, select the **Export project data** check box, and click **Yes**.
22. In the Exporting Project page, expand **Steps** to see the status of each module.
23. After the project export is successful, make a note of the bucket name and the archive file along with the project's other details.

    Example:

Field	Value
Project Name	My Project
Description	This is my first project
Security	Private
Wiki	Markdown
Bucket/Container Name	DevCS.MyProjects.Bucket
Archive File	Export-my-domain_My Project_2019-03-12T05-34-20.627Z-by_alex.admin.zip

Create a Project With an Exported Project's Data in the Target DevCS Instance

While creating a project, you can import data from an exported project's archive file stored in an OCI Object Storage bucket.

To import project data, you need these details:

- Name of the target bucket
- Name of the exported archive file
- Private key and fingerprint of a user who has the `BUCKET_INSPECT` (or `BUCKET_READ`) and `OBJECT_READ` permissions of the bucket
- Details of the compartment that hosts the bucket

After you have the required input values, you can import the project.

Perform these steps in the target DevCS instance.

1. Get the details of the project from the source DevCS instance.

    Example:

Field	Value
Project Name	My Project
Description	This is my first project
Security	Private
Wiki	Markdown
Bucket/Container Name	DevCS.MyProjects.Bucket
Archive File	Export-my-domain_My Project_2019-03-12T05-34-20.627Z-by_alex.admin.zip

2. Switch to the browser with the target DevCS instance.

3. In the navigation bar, click **Organization**, if required.
4. On the Organization page, click **+ Create Project**.
5. On the Project Details page of the New Project wizard, in **Name** and **Description**, enter the project name and a description.
6. In **Security**, select the project's privacy.
7. Click **Next**.
8. On the Template page, select **Import Project**, and click **Next**.
9. In the Storage Connection section of the Project Properties page, to import the project's data from an OCI Object Storage bucket, in **Account Type**, select **OCI**.
10. If you selected **OCI** as **Account Type**, enter the required connection details.
    a. In **Tenancy OCID**, enter the tenancy's OCID copied from the Tenancy Details page.
    b. In **User OCID**, enter the user's OCID value who can access the bucket.
    c. In **Home Region**, select the home region of the OCI account.
    d. In **Private Key**, enter the private key of the user who can access the bucket.
    e. In **Passphrase**, enter the passphrase used to encrypt the private key. If no passphrase was used, leave the field empty.
    f. In **Fingerprint**, enter the fingerprint value of the private-public key pair.
    g. In **Compartment OCID**, enter the compartment's OCID copied from the Compartments page.
    h. In **Storage Namespace**, enter the storage namespace copied from the Tenancy Details page.
11. Click **Next**.
12. On the Project Properties page, from **Wiki Markup**, select the project’s wiki markup language.
13. In **Container**, select the storage bucket or the container where the data was exported.
14. In **File**, select the exported file.
15. Click **Finish**.
16. After the import is complete, check the log to learn about the imported artifacts.
    a. In the navigation bar, click **Project Administration**.
    b. Click **Data Export/Import**.
    c. Click the **History** tab.
    d. Select the import entry.
    e. In the **Job Details** section, expand **Steps**.

In the created project, verify artifacts listed in the table of Export Project Data From the Source DevCS Instance with **Yes** value in the **Exported?** column have been imported.

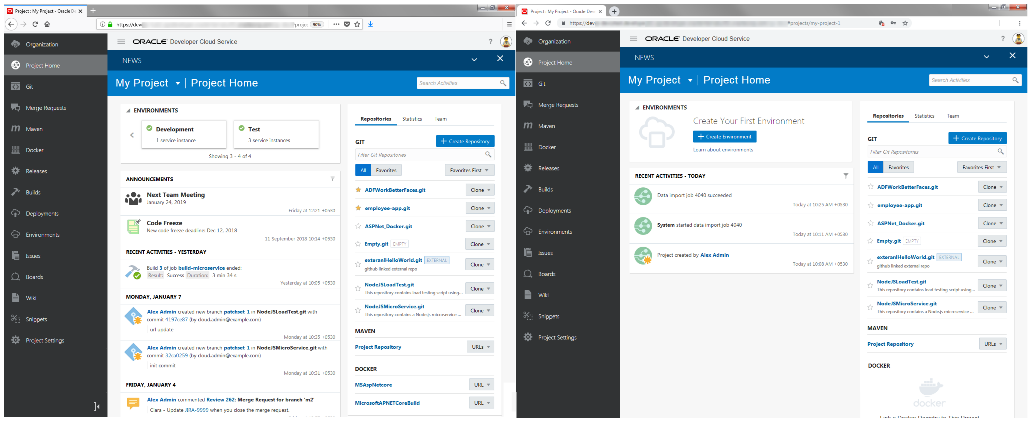
If the import fails, you can import the data again without creating the project. See Export Project Data to and Import Project Data from Oracle Cloud in *Using Oracle Developer Cloud Service*.

Migrate Other Artifacts and Data

After exporting a project from the source DevCS instance and creating the same name project with the exported project's data in the target DevCS instance, migrate the artifacts that weren't exported.

Before you begin, open the source project in a browser and the target project in another browser. Then, to migrate an artifact, note and copy its details from the source project and create or add it manually to the new project.

For example, in this image the source project is open in Mozilla Firefox and the target project is open in Google Chrome.



Project Users

To migrate users, you'll need to add them manually to the new project.

If you're switching to another identity domain, make sure that DevCS users of the source identity domain are added to the target identity domain with the same usernames and roles. See Create Users and Assign Roles in *Getting Started with Oracle Cloud*.

Follow these steps to migrate project users:

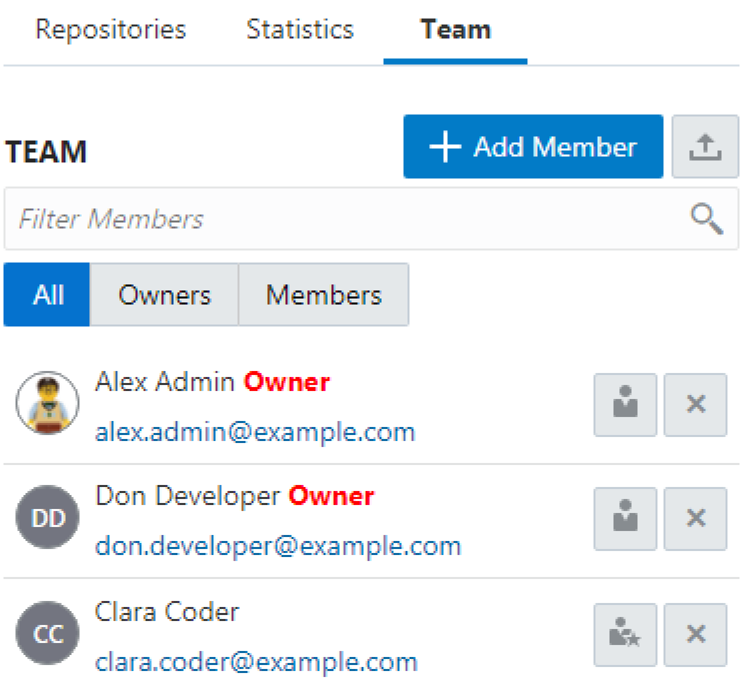
1. In the source and target projects, follow these steps:
   a. In the navigation bar, click **Project Home** .
   b. Click the **Team** tab.
2. In the source project, gather information about the usernames and roles of project users.
3. In the target project, add the users and assign them the same roles.
4. After adding users, with the **Team** tab open in both source and target projects, verify that users and roles in the target project match the source project.

Gather Information About Users From the Source Project

Perform these steps in the source project.

1. Switch to the browser with the source project.
2. In the **Team** tab, for each user, note the usernames and project roles.

   Example:

   Alex Admin **Owner**
   alex.admin@example.com

   DD Don Developer **Owner**
   don.developer@example.com

   CC Clara Coder
   clara.coder@example.com

   This table shows the User Name and Owner values from the above image.

User Name	Project Role
Alex Admin	Owner
Don Developer	Owner
Clara Coder	Member

3. Click **Export** ⭱.
4. In the Members List Export dialog box, copy the names of project members.
5. Click **OK** or **Close** ✕ to close the dialog box.

Add Users in the Target Project

Perform these steps in the target project.

1. Get the details of users from the source project. You'll use the information to add users and assign them the same roles in the target project.

   Example:

User Name	Project Role
Alex Admin	Owner
Don Developer	Owner
Clara Coder	Member

2. Switch to the browser with the target project.
3. In the **Team** tab, click **+ Create Member**.
4. In the New Member dialog box, select the **Multiple Users** check box.
5. In **Username List** text box, enter the comma delimited list of usernames exported from the source project.
6. Click **Add**.
7. For each user who was assigned the **Owner** role, click **Promote to Owner** .

When you're done, verify that the users and roles in the target project match the source project.

Before you exported the source project's data, you noted whether you're a member or an owner in the source project. Don't delete that data. You'll use it after migrating all source project's artifacts to the target project.

Mirrored Private Git Repositories

If you’ve been using private Git repositories on another platform, such as GitHub or Bitbucket, and have mirrored them to your project, note that they aren't exported to the archive file when you export project data.

Follow these steps to migrate mirrored private Git repositories:

1. In the source and target projects, follow these steps:
   a. In the navigation bar, click **Project Administration** ⚙.
   b. Click **Repositories**.
   c. Expand **External Repositories**.
2. Repeat these steps to migrate each mirrored private Git repository:
   a. In the source project, gather information about a mirrored private Git repository.
   b. In the target project, add a mirror to the private Git repository.
3. After migrating all private Git repositories, with the **Repositories** page open in both source and target projects, verify that repositories in the target project match the source project.

Gather Information About Mirrored Private Git Repositories From the Source Project

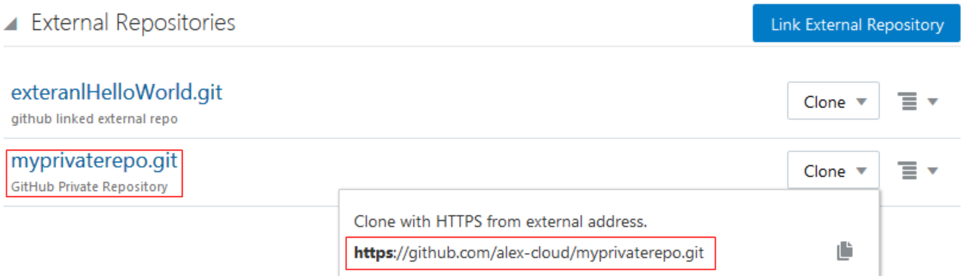
Perform these steps in the source project.

1. Switch to the browser with the source project.

2. In **External Repositories**, note the name, description, and URL of the mirrored Git repository.

   You can get the URL from the **Clone** drop-down menu.

   Example:



Clone with HTTPS from external address.

https://github.com/alex-cloud/myprivaterepo.git

   This table shows the Name, Description, and URL values from the above image.

Field	Value
Repository Name	myprivaterepo
Description	GitHub Private Repository
URL	https://github.com/alex-cloud/myprivaterepo.git

   Get the credentials of the repository too. They aren't displayed in the drop-down menu.

Mirror a Private Git Repository in the Target Project

Perform these steps in the target project.

1. Get the details of the mirrored private Git repository from the source project. You'll use the information to add the external Git repository in the target project.

   Example:

Field	Value
Repository Name	myprivaterepo
Description	GitHub Private Repository
URL	https://github.com/alex-cloud/myprivaterepo.git

2. Switch to the browser with the target project.
3. In **External Repositories**, click **+ Link External Repository**
4. In the New Repository dialog box, enter the URL of the external Git repository in **URL** and enter the repository's description in **Description**.
5. Expand the **Credentials for non-public repos** section and enter the credentials to access the external Git repository.
6. Click **Create**.
7. In the navigation bar, click **Git** .

8. From the **Repository** list, select the external repository and verify that you can access its files.

Build Jobs

When you create a project with the exported project's data, the build jobs of the source project are not migrated to the target project.

Follow these steps to migrate build jobs:

1. In the source and target projects, in the navigation bar, click **Builds** .
2. Repeat these steps to migrate each job:
   a. In the source project, gather information about the job's software packages.
   b. In the target instance, create a Build VM template with the job's software packages and allocate Build VMs to it.
   c. Open two browser windows, one that shows the configuration page of the source job and the other that shows the configuration page of the target job.
   d. Note the configuration settings of the source job and make the same settings in the target job.
   e. Save the target job.
3. If the source job has any dependencies, create a pipeline in the target project.
4. After creating the job and pipelines, run a build. If the build fails, verify the configuration.

Before you migrate jobs, you might want to learn about the differences between the **Builds** page's user interface on the source and target DevCS instances.

Builds Page User Interface

The **Builds** page and its sub-pages in the target DevCS instance have been redesigned and differ from the **Builds** pages in the source DevCS instance. In the **Builds** page of the target project, you'll notice some new fields and options were added, the names of some fields and options have changed, and some fields and options have been removed.

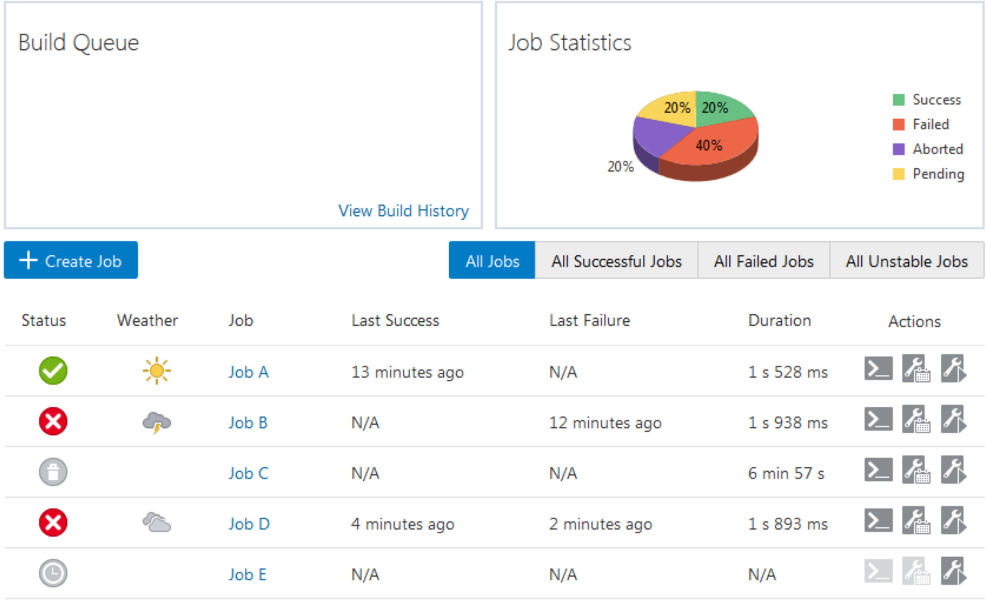
To see the differences, open the source project in a browser and the target project in another browser. Then, in both source and target projects, click **Builds** in the navigation bar.

Builds Page

On the target project's **Builds** page, you'll notice these differences:

- A new tab, **Pipelines**, is available for creating and configuring build pipelines. More on that later.
- Icons in the **Actions** column of the jobs table have changed.
- In the toggle buttons, **All Unstable jobs** is now **Test Failed Jobs**.
- Job Statistics doesn't show pending jobs anymore.

Example:



Build Queue

View Build History

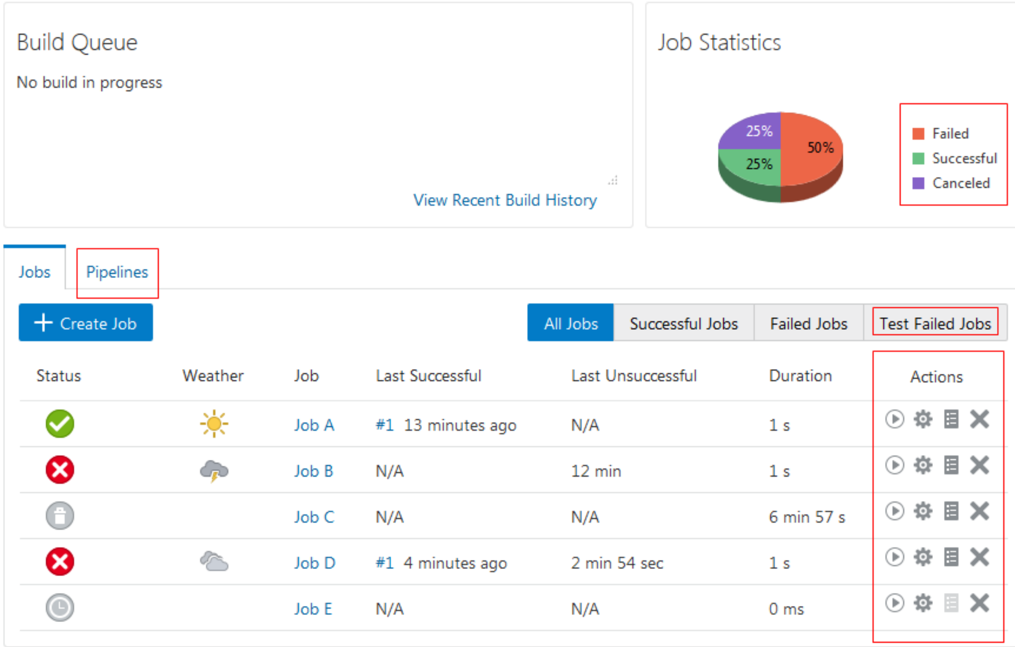
Job Statistics

+ Create Job

All Jobs | All Successful Jobs | All Failed Jobs | All Unstable Jobs

Status	Weather	Job	Last Success	Last Failure	Duration	Actions
✓	☀	Job A	13 minutes ago	N/A	1 s 528 ms	
✗	⛈	Job B	N/A	12 minutes ago	1 s 938 ms	
🗑		Job C	N/A	N/A	6 min 57 s	
✗	☁	Job D	4 minutes ago	2 minutes ago	1 s 893 ms	
🕒		Job E	N/A	N/A	N/A	

Builds Page on Traditional DevCS



Build Queue

No build in progress

View Recent Build History

Job Statistics

Jobs | Pipelines

+ Create Job

All Jobs | Successful Jobs | Failed Jobs | Test Failed Jobs

Status	Weather	Job	Last Successful	Last Unsuccessful	Duration	Actions
✓	☀	Job A	#1 13 minutes ago	N/A	1 s	▷ ⚙ ▤ ✕
✗	⛈	Job B	N/A	12 min	1 s	▷ ⚙ ▤ ✕
🗑		Job C	N/A	N/A	6 min 57 s	▷ ⚙ ▤ ✕
✗	☁	Job D	#1 4 minutes ago	2 min 54 sec	1 s	▷ ⚙ ▤ ✕
🕒		Job E	N/A	N/A	0 ms	▷ ⚙ ▤ ✕

Builds Page on OCI DevCS

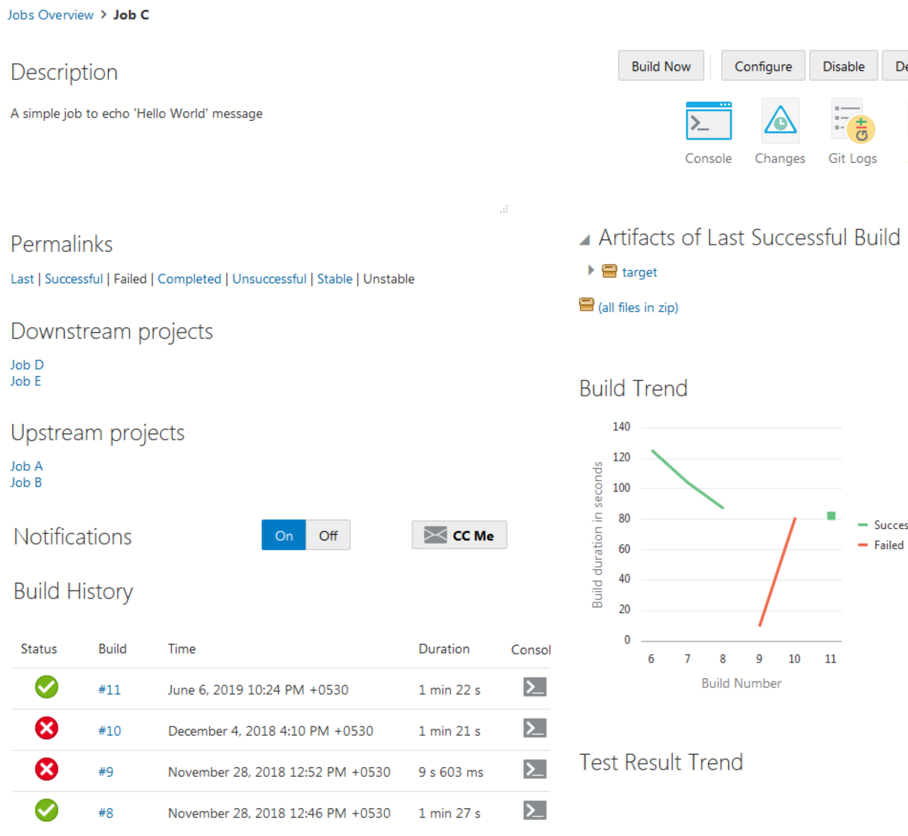
Job Details Page

Click a job's name to open the job's details page. On the target project, you'll notice these differences:

- **Descriptions** is now **Job Details**.

- There are some new report icons on the right. The icons of other reports have been updated too.
- Build artifacts under **Artifacts of Last Successful Build** are now available in the **Artifacts** report.
- Permalinks have moved above the build history
- A new **By** column in the build history indicates who ran the build.
- Icons in the **Actions** column of build history have changed.
- The graph type of **Build Trend** has changed.

Example:



Jobs Overview > **Job C**

Description

A simple job to echo 'Hello World' message

Permalinks

Last | Successful | Failed | Completed | Unsuccessful | Stable | Unstable

Artifacts of Last Successful Build

target

(all files in zip)

Downstream projects

Job D
Job E

Upstream projects

Job A
Job B

Notifications On Off CC Me

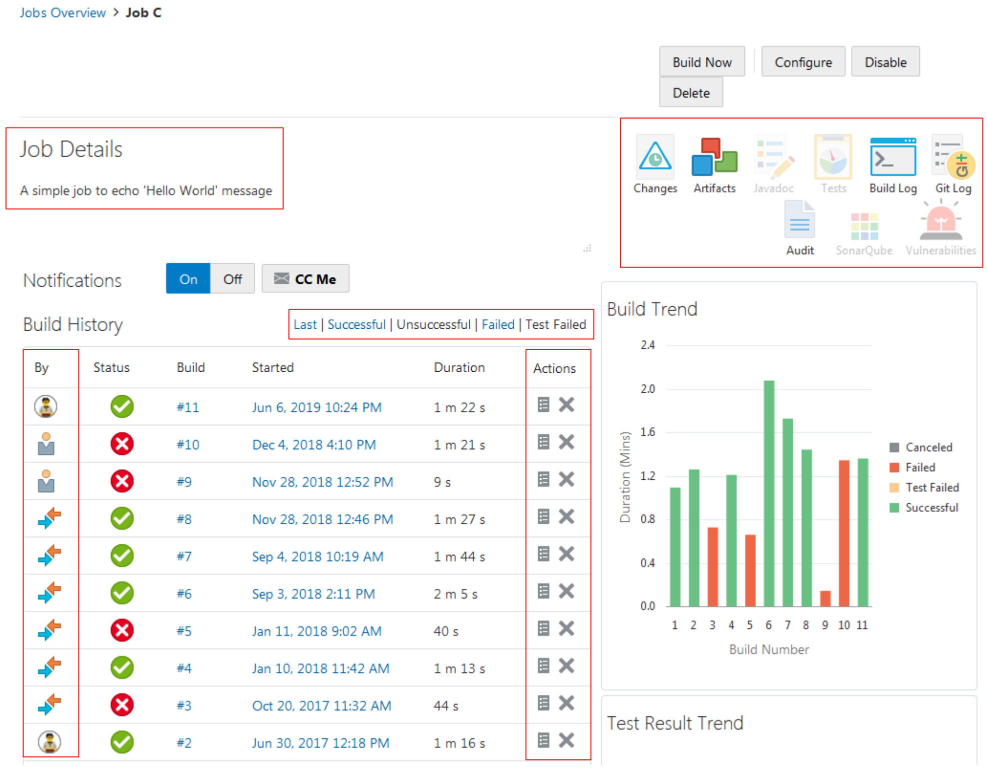
Build History

Status	Build	Time	Duration	Consol
✓	#11	June 6, 2019 10:24 PM +0530	1 min 22 s	
✗	#10	December 4, 2018 4:10 PM +0530	1 min 21 s	
✗	#9	November 28, 2018 12:52 PM +0530	9 s 603 ms	
✓	#8	November 28, 2018 12:46 PM +0530	1 min 27 s	

Build Trend

Test Result Trend

Job's Details Page on Traditional DevCS



Jobs Overview > **Job C**

Job Details

A simple job to echo 'Hello World' message

Notifications On Off CC Me

Build History Last | Successful | Unsuccessful | Failed | Test Failed

By	Status	Build	Started	Duration	Actions
	✓	#11	Jun 6, 2019 10:24 PM	1 m 22 s	✕
	✗	#10	Dec 4, 2018 4:10 PM	1 m 21 s	✕
	✗	#9	Nov 28, 2018 12:52 PM	9 s	✕
	✓	#8	Nov 28, 2018 12:46 PM	1 m 27 s	✕
	✓	#7	Sep 4, 2018 10:19 AM	1 m 44 s	✕
	✓	#6	Sep 3, 2018 2:11 PM	2 m 5 s	✕
	✗	#5	Jan 11, 2018 9:02 AM	40 s	✕
	✓	#4	Jan 10, 2018 11:42 AM	1 m 13 s	✕
	✗	#3	Oct 20, 2017 11:32 AM	44 s	✕
	✓	#2	Jun 30, 2017 12:18 PM	1 m 16 s	✕

Build Trend

Test Result Trend

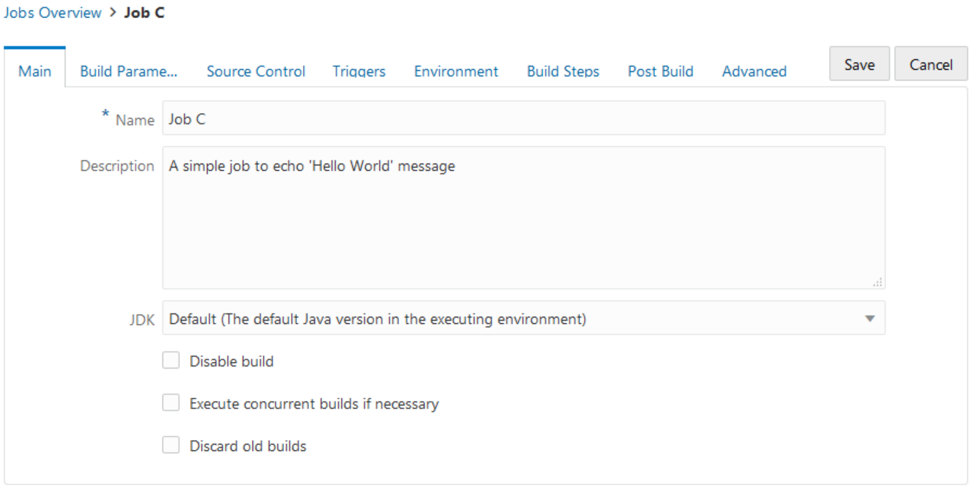
Job's Details Page on OCI DevCS

Job Configuration Page

Click the **Configure** button to open the job's configure page. On the target project, notice these differences:

- The order and labels of horizontal tabs have changed.
- The tabs are categorized into two vertical tabs: **Settings** and **Configure** .

Example:



Jobs Overview > **Job C**

* Name: Job C

Description: A simple job to echo 'Hello World' message

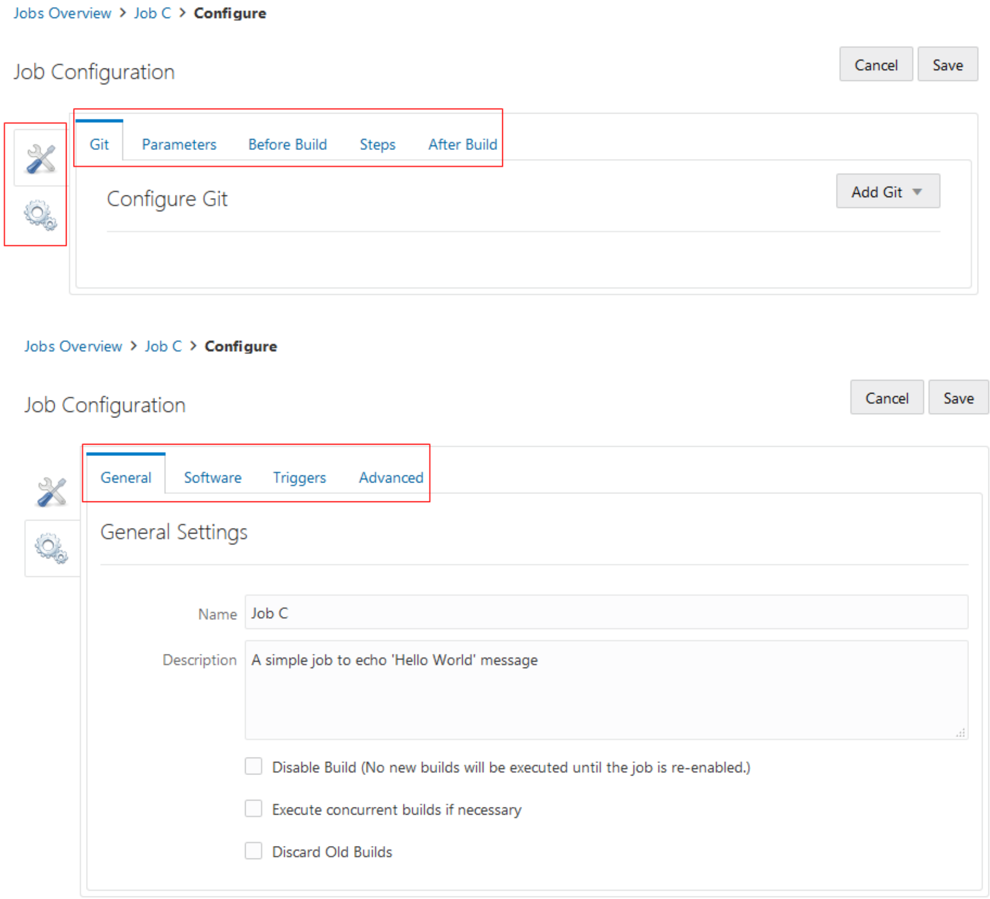
JDK: Default (The default Java version in the executing environment)

□ Disable build

□ Execute concurrent builds if necessary

□ Discard old builds

Job's Configuration Page on Traditional DevCS



Jobs Overview > Job C > **Configure**

Job Configuration

Configure Git

Jobs Overview > Job C > **Configure**

Job Configuration

General Settings

Name: Job C

Description: A simple job to echo 'Hello World' message

□ Disable Build (No new builds will be executed until the job is re-enabled.)

□ Execute concurrent builds if necessary

□ Discard Old Builds

Job's Configuration Page on OCI DevCS

Gather Information About the Job's Software Packages From the Source Project

Perform these steps in the source project.

1. Switch to the browser with the source project.
2. In the source project, click the job's name and then click **Configure** to open the job's configuration page.
3. From the **Build Steps** tab, note whether the job uses any of these build steps.
   - Gradle
   - Node.js
   - SQLcl
   - PSMcli

   If the source job uses any of the above software packages in the build steps, note their version (if specified explicitly) and add them to the job's software packages list. If the source job uses Node.js, note its version from the **Environment** tab.

   For example, this table shows a job's software packages list.

Software	Version
Gradle	3.5
Node.js	10.15.3

4. From the **Environment** tab, note whether the **Start Xvfb before the build, and shut it down after** check box is selected. If it is, add Xvfb to the job's software packages list.

Create a VM Template and its Build VMs in the Target DevCS Instance

Perform these steps in the target DevCS instance.

Before you create Build VMs Templates and Build VMs, note these points:

- Learn about the OCI regions and shapes where you'll create the VMs. A region is a localized geographic area, and an availability domain is one or more data centers located within a region. A shape is a template that determines the number of CPUs, amount of memory, and other resources allocated to a newly created instance. For more details, see Regions and Availability Domains and VM Shapes.
- When you create a Build VM Template, some software packages (such as Maven, Ant, and Java) are available by default in the template's required Build VM components. They're are not available in the Software Catalog.
- Some software packages are available on a particular platform only. For example, Xvfb is available on Oracle Linux 7. To find about the software packages available by default and their compatible platform, see Software Installed on the Build Executor in *Using Oracle Developer Cloud Service*.

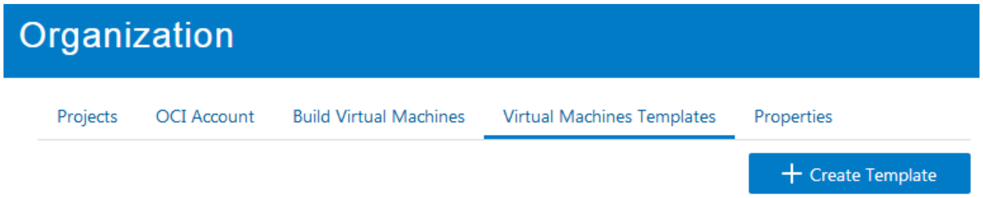
Let's get started.

1. Get the details of the software packages used by the source job.

Example:

Software	Version
Gradle	3.5
Node.js	10.15.3

2. Switch to the browser with the target DevCS instance.
3. In the navigation bar, click **Organization** .
4. Click the **Virtual Machines Templates** tab.

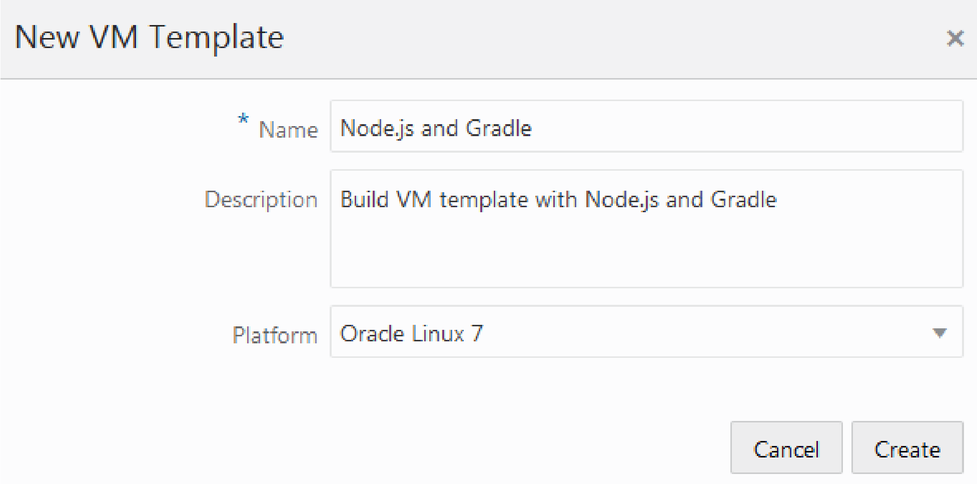


5. Check whether a Build VM Template exists with the same software packages and versions you noted in Step 1. If a template is available, note its name and jump to Step 12.

   If no such template is available, click **+ Create Template**.

6. In the New VM Template dialog box, enter the name and description of the VM template.

   Example:



* Name: Node.js and Gradle

Description: Build VM template with Node.js and Gradle

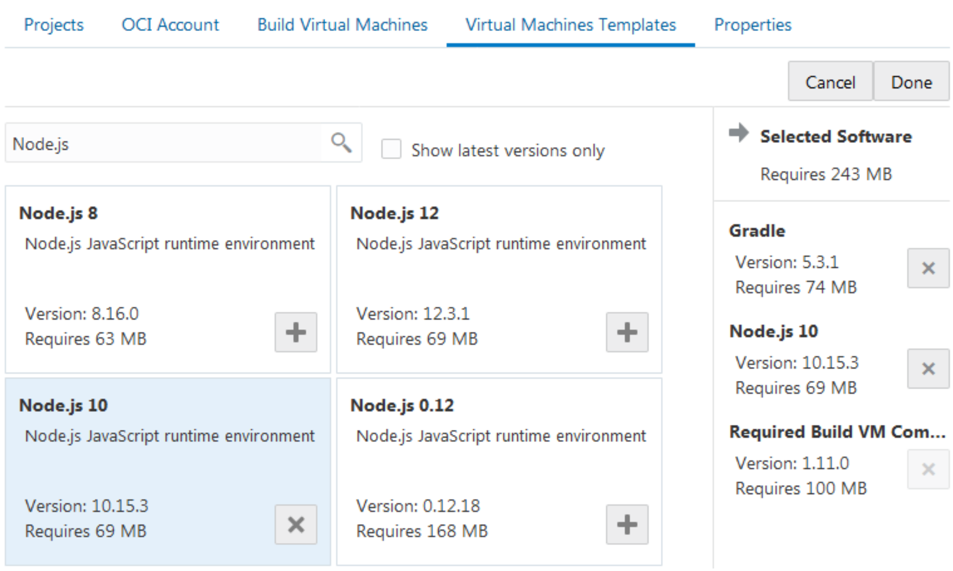
Platform: Oracle Linux 7

7. In **Platform**, select the operating system.

   The build system of the source DevCS instance uses Oracle Linux 6. You can select the same platform or switch to Oracle Linux 7. If you switch to Oracle Linux 7, note that you might not find the same version of the software package used by the source job in Oracle Linux 7 and your application code might become incompatible. In such a case, remember to update your code to make it compatible with the platform and its software packages.

8. Click **Create**.
9. Select the template and click **Configure Software**.
10. For each software package of the VM template, in the Software Catalog, click **Add** +. If a software package is dependent on another software package, a message displays.

    If you can't find a software package, in **Filter Software Packages**, enter the search term and click **Search** 🔍. To see the latest version of the software package, select the **Show latest versions only** check box. This is helpful if multiple versions of the same software package are available in the catalog.
    Example:



    For some software packages, you may not find the same version as you noted in the source job. In such a case, select the next available higher version or the latest version. If the software package version changes, note that your application code might become incompatible. In such a case, update your code to make it compatible with the software packages.

    To use Gradle, note that Gradle 5 is available in the target DevCS job. If you want to use an older version of Gradle, add **Gradle** to the Build VM template. When you configure the build job, use **Gradle Wrapper** and specify the older version of Gradle.

11. Click **Done**.
12. Click the **Build Virtual Machines** tab.
13. In the **Build VMs** tab, click **+ Create VM**.

    If sufficient number of VMs are already allocated to the template, then don't add Build VMs and skip the remaining steps.

14. In the Add Build VM dialog box, in **Quantity**, enter the number of VMs. In **VM Template**, select the Build VM Template. In **Region** and **Shape**, specify the VM's region and shape.

To minimize build execution delays, set the number of VMs to the number of jobs that you expect to run in parallel using that template. If the VM quota is available, that number of Build VMs will be added to the **Build Virtual Machines** tab. Note that the more VMs you have running at a specific time, the higher the cost. To minimize the higher cost, use the **Sleep Timeout** setting on the **Virtual Machines** tab to automatically shut down inactive VMs. You can always return to the **Build Virtual Machines** tab to add or remove VMs, based on your actual usage.

15. Click **Add**.

Create and Configure a Job in the Target Project

To migrate a job, note its configuration from the source project. In the target project, create a job with the same name and then configure it accordingly.

1. Switch to the browser with the source project.
2. In the source project, click the job's name and then click **Configure** to open the job's configuration page.
3. From the **Main** tab, note the job's name and description.

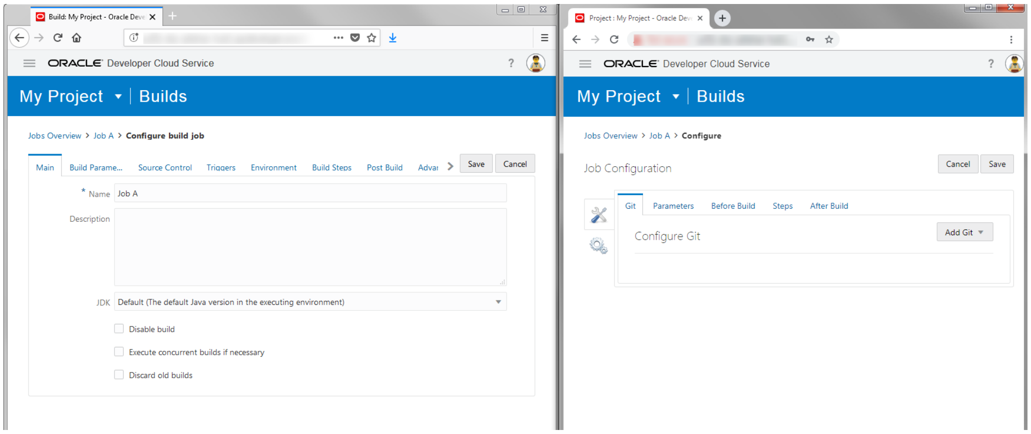
   Example:

Field	Value/Status
Job Name	build-microservice
Description	Job to build a Node.js microservice

4. Switch to the browser with the target project.
5. In the **Jobs** tab of the **Builds** page, click **+ Create Job**.
6. In the New Job dialog box, enter the job's name and description.

   In **Template**, select the Build VM template you created for the job and click **Create**.

Open the source job's configuration page in a browser and the target job's configuration page in another browser.



* Name: Job A

Description:

JDK: Default (The default Java version in the executing environment)

□ Disable build

□ Execute concurrent builds if necessary

□ Discard old builds

The following topics help you migrate each tab's configuration from the source job to the target job. While migrating a job, you would notice some new fields and configuration options in the target job. This guide doesn't describe them, but if you want to learn about them, see Configure and Run Project Jobs and Builds in *Using Oracle Developer Cloud Service*.

Main Tab

1. Switch to the browser with the source job.
2. Click the **Main** tab.
3. In the source job, note the **JDK** version. Then, follow these steps:
   a. Switch to the browser with the target job.
   b. Click **Settings** ⚙.
   c. Click the **Software** tab.
   d. In **Java**, select the Java version.

      JDK 6, 9, and 10 are not available in the target job. If the source job uses JDK 6, then use JDK 7 (or higher) in the target job. If the source job uses JDK 9 or 10, then use JDK 11 (or higher) in the target job.
4. In the source job, note the state and values of the **Disable Build**, **Execute concurrent builds if necessary**, and **Discard old builds** check boxes and its fields. If any of them is selected, follow these steps:
   a. Switch to the browser with the target job.
   b. Click **Settings** ⚙.
   c. Click the **General** tab.
   d. From the selected check boxes, copy the values from the fields of the source job and paste them in the target job's fields.

   In the target job, some field names have changed. To help you migrate, this table maps the old field names in the source job to the new field names in the target job. Fields not listed in the table have the same names.

Field Name in the Source Job	Field Name in the Target Job
Disable build	**Disable Build (No new build will be executed until the job is re-enabled)**
Discard old builds	**Discard Old Builds**
Days to keep builds	**Discard other builds after how many days**
Number of builds to keep	**Number of builds to keep at all times**
Days to keep artifacts	**Discard other artifacts after how many days**
Keep artifacts for this number of builds	**Minimum number of builds to retain artifacts for**

Build Parameters Tab

1. Switch to the browser with the source job.
2. Click the **Build Parameters** tab.
3. If the **This build is parameterized** check box is selected, note the parameter types and copy their values.

Example:

Parameter	Values
Boolean	• **Name**: myBoolParam • **Default Value**: Selected • **Description**: <empty>
String	• **Name**: myStringParam • **Default Value**: Hello World • **Description**: Enter the Welcome message

File and **Run** parameters aren't available in the target project. If the source job uses them, don't copy their values.

4. Switch to the browser with the target job.
5. Click **Configure** 🛠.
6. Click the **Parameters** tab.
7. For each parameter noted in Step 3, from the **Add Parameter** drop-down list, select the parameter.
8. Enter the same values as noted in the parameter's fields.

   If the source job uses merge request parameters, enter their default values in the target job. You can't change the merge request parameter names in the target job.

Source Control Tab

1. Switch to the browser with the source job.
2. Click the **Source Control** tab.
3. If the **Git** option is not selected, ignore the remaining steps. If it is selected, note the repository's name or URL from **Repository**. If a branch other than `master` is specified, note the branch's name too.

   For example, this table shows a job's Git repository details.

Git Repository	Value
Name/URL	MyNodeJSApp.git
Branch	Release_1.1

4. Switch to the browser with the target job.
5. Click **Configure** 🛠.
6. Click the **Git** tab.
7. From **Add Git**, select **Git**.
8. In **Repository**, select the Git repository's name or enter the URL. In **Branch**, specify the branch's name.
9. In the source job, if you've configured some advanced Git options, copy their values and paste them in the target job.

In the target job, some field names have changed. To help you migrate, this table maps the old field names in the source job to the new field names in the target job. Fields not listed in the table have the same names.

Field Name in the Source Job	Field Name in the Target Job
Advanced Repository Settings section	**Advanced Repository Options** section
Reference Spec	**Refspec**
Local Subdirectory	**Local Checkout Directory**
Advanced Git Settings section	**Advanced Git Settings** section
Included Regions	**Included Region**
Excluded Regions	**Excluded Region**
Excluded Users	**Excluded User**
Checkout/merge to local branch	In **Merge another branch**, specify the branch name to merge to. In **Checkout Revision**, specify the branch to checkout.
Config user.name Value	**Config user.name**
Config user.email Value	**Config user.email**
Merge before build • **Name of repository** • **Branch to merge to**	**Merge from another repository** • **Repository** • **Branch**
Wipe out workspace before build	**Wipe out workspace before checkout**

Triggers Tab

1. Switch to the browser with the source job.
2. Click the **Triggers** tab.
3. If the **When these jobs are built** check box is selected, you'll need to create a pipeline in the target project, after creating all dependent jobs that need to be in that pipeline. See Create and Configure Pipelines.
4. In the source job, if the **Based on this schedule** check box is selected, follow these steps:

   If a schedule is specified, do this:

   a. Copy the schedule.
   b. Switch to the browser with the target job.
   c. Click **Settings** ⚙.
   d. Click the **Triggers** tab.
   e. From the **Add Trigger** drop-down list, select **Periodic Build Trigger**.
   f. Select the **Expert mode** check box and paste the schedule in the text box.

   If no schedule is specified, do this:

   a. Switch to the browser with the target job.
   b. Click **Configure** 🔧.
   c. Click the **Git** tab.

   d. In **Git**, select the **Automatically perform build on SCM commit** check box.

5. In the source job, if the **Based on SCM polling schedule** check box is selected, follow these steps:

   a. Copy the schedule.

   b. Switch to the browser with the target job.

   c. Click **Settings** .

   d. Click the **Triggers** tab.

   e. From the **Add Trigger** drop-down list, select **SCM Polling Trigger**.

   f. Select the **Expert mode** check box and paste the schedule in the text box.

6. In the source job, if the **When Maven dependencies have been updated by Maven 3 integration** check box is selected, ignore it because the option isn't available in the target job.

Environment Tab

1. Switch to the browser with the source job.

2. Click the **Environment** tab.

3. If the **Start Xvfb before the build, and shut it down after** check box is selected, follow these steps:

   a. Switch to the browser with the target job.

   b. In the **Software** tab of **Settings** , ensure that the target job uses an Oracle Linux 7 Build VM template. Xvfb is available on Oracle Linux 7.

   c. Click **Configure** .

   d. Click the **Before Build** tab.

   e. From the **Add Before Build Action** drop-down list, select **Xvfb Wrapper**.

      In the target job, some field names have changed. To help you migrate, this table maps the old field names in the source job to the new field names in the target job. Fields not listed in the table have the same names.

Field Name in the Source Job	Field Name in the Target Job
Xvfb Specific Displayname	**Display Number**
Xvfb Screen	**Screen Size**
Xvfb Display Name Offset	**Screen Offset**
Xvfb Additional Options	**Additional Options**

4. In the source job, if the **Abort the build if it's stuck** check box is selected, note the value of **Timeout Minutes** and whether the **Fail the build** check box is selected. Then, follow these steps:

   a. Switch to the browser with the target job.

   b. Click **Settings** .

   c. Click the **Advanced** tab.

   d. Select the **Abort the build if it is stuck** check box.

      In **Hours** and **Minutes**, specify the timeout duration.

   e. If the **Fail the build** check box is selected in the source job, select the **Fail the build on abort** check box in the target job.

5. In the source job, if the **Add Timestamps to the Console Output** check box is selected, follow these steps:

   a. Switch to the browser with the target job.

   b. Click **Settings** ⚙.

   c. Click the **Advanced** tab.

   d. Select the **Add Timestamps to the Console Output** check box.

6. In the source job, if the **Connect Oracle Maven Repository** check box is selected, then follow these steps:

   a. Switch to the browser with the target job.

   b. Click **Configure** 🛠.

   c. Click the **Before Build** tab.

   d. From the **Add Before Build Action** drop-down list, select **Oracle Maven Repository Connection**.

   e. Click the **Use existing connection** toggle button to disable it.

   f. Switch to the browser with the source job.

   g. Copy the values of **OTN Login**, **OTN Password**, **Server Id**, and **Custom settings.xml**.

   h. Switch to the browser with the target job.

   i. Paste the copied values into the fields with the same names.

7. In the source job, if the **Use NodeJS version** check box is selected, then configure the target job to use a Build VM Template with the same (or higher) version of Node.js. You probably already did this when you created the Build VM Template.

Build Steps Tab

1. Switch to the browser with the source job.

2. Click the **Build Steps** tab.

3. If the source job has an **Execute Shell** build step, follow these steps:

   a. Switch to the browser with the target job.

   b. Click **Configure** 🛠.

   c. Click the **Steps** tab.

   d. From the **Add Step** drop-down list, select **Unix Shell**.

   e. Switch to the browser with the source job.

   f. Copy the script/commands from **Commands**.

   g. Switch to the browser with the target job.

   h. Paste the copied script/commands to **Script**.

4. If the source job has an **Invoke Ant** build step, follow these steps:

   a. Switch to the browser with the target job.

   b. Click **Configure** 🛠.

   c. Click the **Steps** tab.

   d. From the **Add Step** drop-down list, select **Ant**.

   e. Switch to the browser with the source job.

   f. Copy the values from **Targets**, **Build File**, **Properties**, and **Java Options**.

   g. Switch to the browser with the target job.

   h. Paste the copied values to the fields with the same names.

5. If the source job has an **Invoke Maven 2 (Legacy)** build step, note that Maven 2 isn't available on the target job. Only Maven 3 is.

   First, configure the source job to use Maven 3 and run a build to verify its stability. After it is stable, migrate the job to the target project. as described in the next step.

6. If the source job has an **Invoke Maven 3** build step, follow these steps:

   a. Switch to the browser with the target job.

   b. Click **Configure** 🛠.

   c. Click the **Steps** tab.

   d. From the **Add Step** drop-down list, select **Maven**.

   e. Switch to the browser with the source job.

   f. Copy the values from **Goals** and **POM File**.

   g. Switch to the browser with the target job.

   h. Paste the copied values to the fields with the same names.

   i. Switch to the browser with the source job.

   j. If you've configured the Maven build step with additional options, copy values from its fields and paste them in the target job.

      First, in the target job, expand **Advanced Maven Settings** to see the Maven advanced options.

      In the target job, some field names have changed. To help you migrate, this table maps the old field names in the source job to the new field names in the target job. Fields not listed in the table have the same names.

Field Name in the Source Job	Field Name in the Target Job
Private temporary directory	Use Private Temp Directory
Checksum Mode	Checksum
Snapshot Updates	Snapshot

7. If the source job has an **Invoke Gradle** build step, follow these steps:

   a. Switch to the browser with the target job.

b. In the **Software** tab of **Settings** , ensure that the target job uses a Build VM template with the Gradle software package.

c. Click **Configure** .

d. Click the **Steps** tab.

e. From the **Add Step** drop-down list, select **Gradle**.

f. In the source job, if the **Use Gradle Wrapper** option is selected, note the state of its check boxes and select them in the target job.

   If the version of Gradle used in your source job is different from the Gradle's version available in the Build VM template, select the **Use 'gradlew' wrapper** check box and specify the version in **Gradle version**.

   In the target job, some field names have changed. To help you migrate, this table maps the old field names in the source job to the new field names in the target job. Fields not listed in the table have the same names.

Field Name in the Source Job	Field Name in the Target Job
Use Gradle Wrapper	Use 'gradlew' wrapper
Make gradlew executable	Create 'gradlew' wrapper
From Root Build Script Dir	In root build script directory

g. In the source job, copy the values of other fields and paste them in the target job.

   In the target job, some field names have changed. To help you migrate, this table maps the old field names in the source job to the new field names in the target job. Fields not listed in the table have the same names.

Field Name in the Source Job	Field Name in the Target Job
Build step description	Description
Root Build script	Root build script directory
Snapshot Updates	Snapshot

8. If the source job has an **Invoke NodeJS** build step, follow these steps:

   a. Switch to the browser with the target job.

   b. In the **Software** tab of **Settings** , ensure that the target job uses a Build VM template with the same (or higher) version of Node.js that you used in the source job.

   c. Click **Configure** .

   d. Click the **Steps** tab.

   e. From the **Add Step** drop-down list, select **Node.js**.

   f. Switch to the browser with the source job.

   g. Copy the values from **Script**.

   h. Switch to the browser with the target job.

i. Select **Script** in **Source**, and then paste the copied contents in **NodeJS Script**.

9. If the source job has an **Copy artifacts from another job** build step, follow these steps:

   a. Switch to the browser with the target job.

   b. Click **Configure** 🛠.

   c. Click the **Before Build** tab.

   d. From the **Add Before Build Action** drop-down list, select **Copy Artifacts**.

   e. In the source job, copy the field values and paste them in the target job.

   In the target job, some field names have changed. To help you migrate, this table maps the old field names in the source job to the new field names in the target job. Fields not listed in the table have the same names.

Field Name in the Source Job	Field Name in the Target Job
Job Name	**From Job**
Stable build only	The option isn't available
Optional	**Optional (Do not fail build if artifacts copy failed)**

10. If the source job has an **Invoke SQLcl** build step, follow these steps:

    a. Switch to the browser with the target job.

    b. In the **Software** tab of **Settings** ⚙, ensure that the target job uses a Build VM template with SQLcl. You might have done this when you created the Build VM template.

    c. Click **Configure** 🛠.

    d. Click the **Steps** tab.

    e. From the **Add Before Build Action** drop-down list, select **SQLcl**.

    f. In the source job, copy the values from fields and enter those values into the fields with the same names in the target job.

11. If the source job has an **Invoke PSMcli** build step, follow these steps:

    a. Switch to the browser with the target job.

    b. In the **Software** tab of **Settings** ⚙, ensure that the target job uses a Build VM template with PSMcli. You might have done this when you created the Build VM template.

    c. Click **Configure** 🛠.

    d. Click the **Steps** tab.

    e. From the **Add Before Build Action** drop-down list, select **PSMcli**.

    f. In the source job, copy the values from fields and enter those values into the fields with the same names in the target job.

Post Build Tab

1. Switch to the browser with the source job.
2. Click the **Post Build** tab.
3. If the **Aggregate all downstream test results** check box is selected, ignore it because the option isn't available in the target job.
4. If the **Build other jobs** check box is selected, you'll need to create a pipeline in the target project, after first creating all dependent jobs. See Create and Configure Pipelines.
5. In the source job, if the **Archive the artifacts** check box is selected, follow these steps:
    1. Switch to the browser with the target job.
    2. Click **Configure** .
    3. Click the **After Build** tab.
    4. From the **Add After Build Action** drop-down list, select **Artifact Archiver**.
    5. Switch to the browser with the source job.
    6. In **Archive the artifacts**, note the state of check boxes or copy their values. In the target job, select the check boxes or paste the copied values.

       In the target job, some field names have changed. To help you migrate, this table maps the old field names in the source job to the new field names in the target job. Fields not listed in the table have the same names.

Field Name in the Source Job	Field Name in the Target Job
Files To Archive	**Files to archive**
Enable auto validation for file masks	The option isn't available
Excludes	**Files to exclude**
Discard all but the last successful/ stable artifact to save disk space	The option isn't available
Compression Type	The option isn't available

6. In the source job, if the **Record fingerprints of files to track usage** check box is selected, ignore it because the option isn't available in the target job.
7. In the source job, if the **Publish Javadoc** check box is selected, follow these steps:
    1. Switch to the browser with the target job.
    2. Click **Configure** .
    3. Click the **After Build** tab.
    4. From the **Add After Build Action** drop-down list, select **Javadoc Publisher**.
    5. Switch to the browser with the source job.
    6. In **Javadoc Publisher**, note the state of check boxes or copy their values. In the target job, select the check boxes or paste the copied values.

In the target job, some field names have changed. To help you migrate, this table maps the old field names in the source job to the new field names in the target job. Fields not listed in the table have the same names.

Field Name in the Source Job	Field Name in the Target Job
Retain Javadoc for each successful build	**Retain Javadoc for each build**

8. In the source job, if the **Publish JUnit test result report** check box is selected, follow these steps:

   a. Switch to the browser with the target job.

   b. Click **Configure** 🛠.

   c. Click the **After Build** tab.

   d. From the **Add After Build Action** drop-down list, select **JUnit Publisher**.

   e. Switch to the browser with the source job.

   f. In **Publish JUnit test result report**, note the state of check boxes or copy their values. In the target job, select the check boxes or paste the copied values.

      In the target job, some field names have changed. To help you migrate, this table maps the old field names in the source job to the new field names in the target job. Fields not listed in the table have the same names.

Field Name in the Source Job	Field Name in the Target Job
Test Report XMLs	**Include JUnit XMLs**

9. In the source job, if the **Archive Maven 3 artifacts** check box is selected, follow these steps:

   a. Switch to the browser with the target job.

   b. Click **Configure** 🛠.

   c. Click the **After Build** tab.

   d. From the **Add After Build Action** drop-down list, select **Artifact Archiver**.

      You can't add another **Artifact Archiver** post-build step if you've added it before.

   e. In **Artifact Archiver**, select the **Archive Maven artifacts** check box.

   f. Switch to the browser with the source job.

   g. In **Archive Maven 3 artifacts**, note the state of check boxes or copy their values. In the target job, select the check boxes or paste the copied values.

      In the target job, some field names have changed. To help you migrate, this table maps the old field names in the source job to the new field names in the target job. Fields not listed in the table have the same names.

Field Name in the Source Job	Field Name in the Target Job
Include generated POMs	**Include pom.xml**

Field Name in the Source Job	Field Name in the Target Job
Discard old artifacts	The option isn't available

10. In the source job, if the **Record fingerprints of Maven artifacts** check box is selected, ignore it because the option isn't available in the target job.
11. In the source job, if the **Git Publisher** check box is selected, follow these steps:
    a. Switch to the browser with the target job.

    b. Click **Configure**.

    c. Click the **After Build** tab.

    d. From the **Add After Build Action** drop-down list, select **Git Publisher**.

    In the source job, copy the values from fields and enter those values into the fields with the same names in the target job.
12. In the source job, if the **Notify that Maven dependencies have been updated by Maven 3 integration** check box is selected, ignore it because the option isn't available in the target job.
13. In the source job, if the **Oracle Cloud Service Deployment** check box is selected, follow these steps:
    a. Switch to the browser with the target job.

    b. Click **Configure**.

    c. Click the **After Build** tab.

    d. From **Add After Build Action**, select **Oracle Cloud Service Deployment**.

    In the source job, copy the values from fields and enter those values into the fields with the same names in the target job.

Advanced Tab

1. Switch to the browser with the source job.
2. Click the **Advanced Tab** tab.
3. If the **Quiet period** check box is selected, note its period and follow these steps:
    a. Switch to the browser with the target job.

    b. Click **Settings**.

    c. Click the **Advanced** tab.

    d. Select the **Quiet Period** check box and specify its period.
4. In the source job, if the **Retry count** check box is selected, note the **SCM Checkout** period and follow these steps:
    a. Switch to the browser with the target job.

    b. Click **Settings**.

    c. Click the **Advanced** tab.

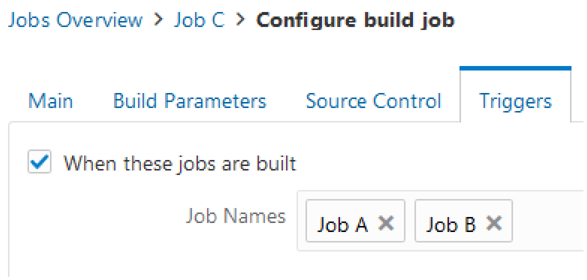
    d. Select the **Retry Count** check box and specify the period in **SCM Retries**.

5. In the source job, if the **Block build when upstream job is building** check box is selected, ignore it because the option isn't available in the target job.
6. In the source job, if the **Block build when downstream job is building** check box is selected, ignore it because the option isn't available in the target job.

Create and Configure Pipelines

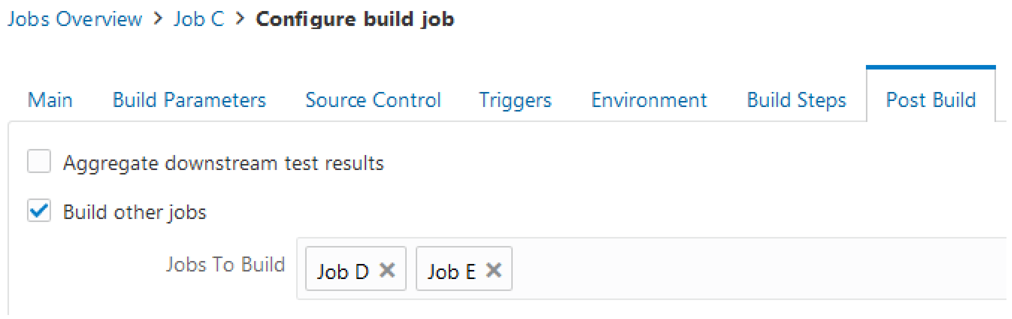
Some jobs of the source project might be configured to run when the builds of other jobs are complete, or to trigger builds of other jobs when their builds are complete. These dependencies are defined in the **Triggers** tab or the **Post Build** tab of the source job.

For example, in this image, Job C has an upstream dependency (or a many-to-one dependency) on Job A and Job B. When a build of any of these jobs complete, they trigger a build of Job C. Job C also has a downstream dependency (or a one-to-many dependency) on Job D and Job E. When a build of Job C is complete, it triggers builds of Job D and Job E.



☑ When these jobs are built

Job Names: Job A ✕ Job B ✕



□ Aggregate downstream test results

☑ Build other jobs

Jobs To Build: Job D ✕ Job E ✕

While migrating source jobs to the target project, you might have noticed that **When these jobs are built** and **Build other jobs** options aren't available in the target jobs. To set up a dependency in the target project, you'll create a pipeline. You must create the pipeline after creating and configuring all dependent jobs.

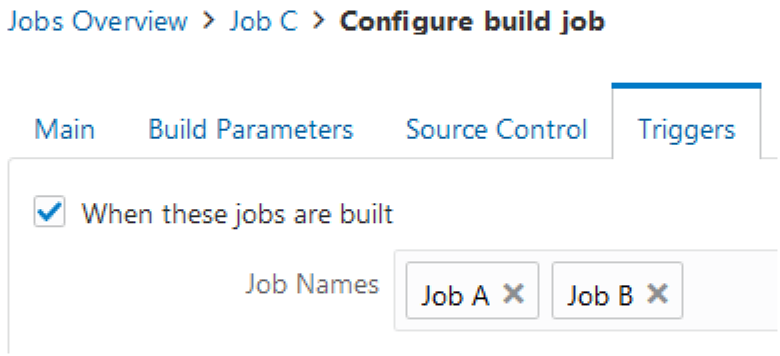
Before you create a pipeline, see What Is a Pipeline? and Use the Pipeline Designer in *Using Oracle Developer Cloud Service* to learn about pipelines and how to design pipeline diagrams.

Set Up a Pipeline

1. Switch to the browser with the source project.

2. Open the job with upstream or downstream dependencies.
3. Click the **Triggers** tab.
4. If the **When these jobs are built** check box is selected, note the name of the upstream jobs, and follow these steps.

   For example, you have a Job C that is triggered when Job A and Job B are built:

   Jobs Overview > Job C > **Configure build job**

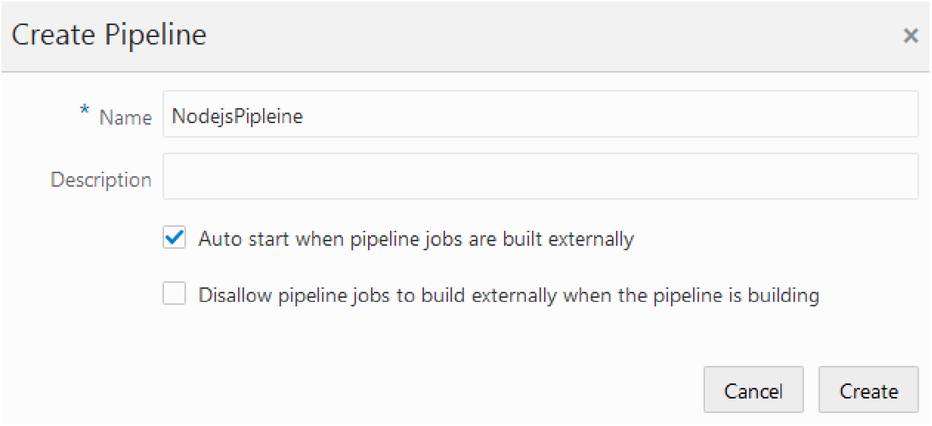
   Main | Build Parameters | Source Control | Triggers

   ☑ When these jobs are built

   Job Names: Job A ✕ Job B ✕

   a. Switch to the browser with the target project.
   b. On the **Builds** page, click the **Pipelines** tab.
   c. Click **+ Create Pipeline**.
   d. In the Create Pipeline dialog box, enter a name and a description.

      Example:

      Create Pipeline ×

      * Name: NodejsPipleine

      Description:

      ☑ Auto start when pipeline jobs are built externally

      □ Disallow pipeline jobs to build externally when the pipeline is building

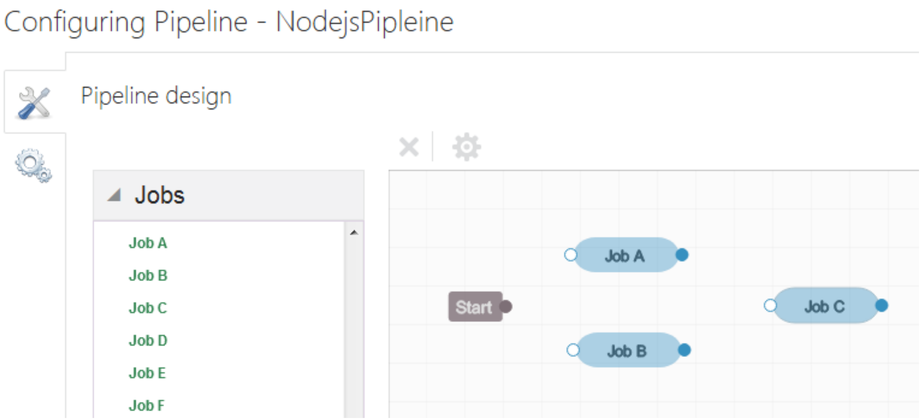
      Cancel | Create

      If you want trigger the pipeline's build if a job of the pipeline is triggered externally (outside the pipeline), leave the select the **Auto start when pipeline jobs are build externally** check box selected. To disable manual or automatic builds of the jobs that are part of the pipeline when the pipeline is running, select the **Disallow pipeline jobs to build externally when the pipeline is building** check box.

e. Click **Create**.

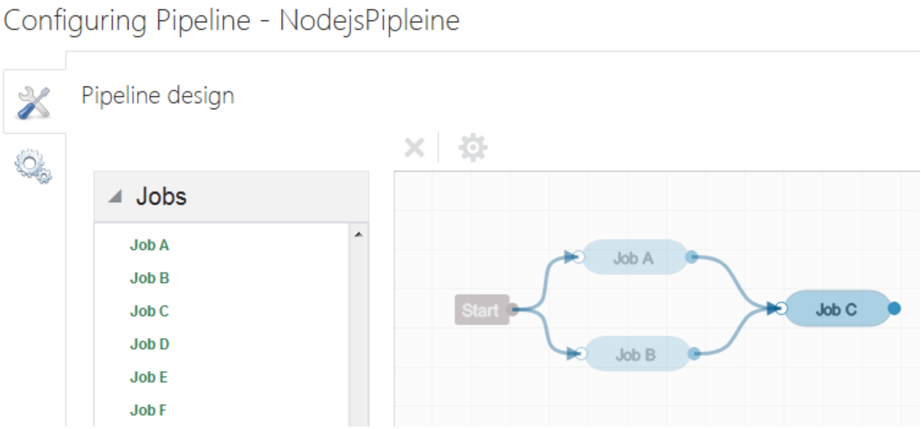
f. In the Pipeline Design tab, from the left list, drag-and-drop the upstream jobs to the designer area. Then, drag-and-drop the job you opened in Step 2 to the designer area.

   Example:



g. Design the dependencies.

   Example:



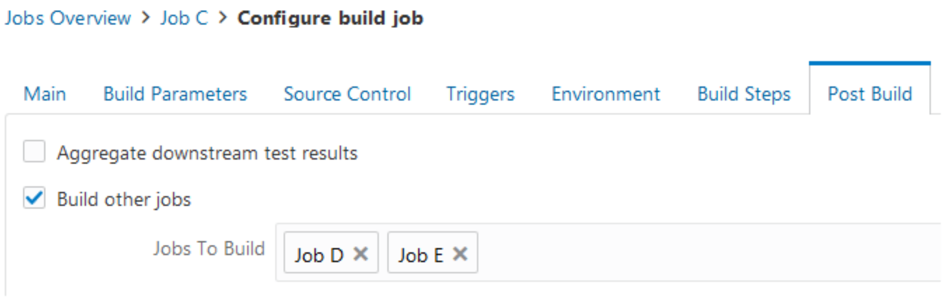
h. Click **Save**.

5. Switch to the browser with the source job.

6. Click the **Post Build** tab.

7. If the **Build other jobs** check box is selected, note the names of the downstream jobs, and follow these steps.

   For example, you have a Job C that triggers Job D and Job E:



Jobs Overview > Job C > **Configure build job**

□ Aggregate downstream test results

☑ Build other jobs

Jobs To Build: Job D ✕ Job E ✕

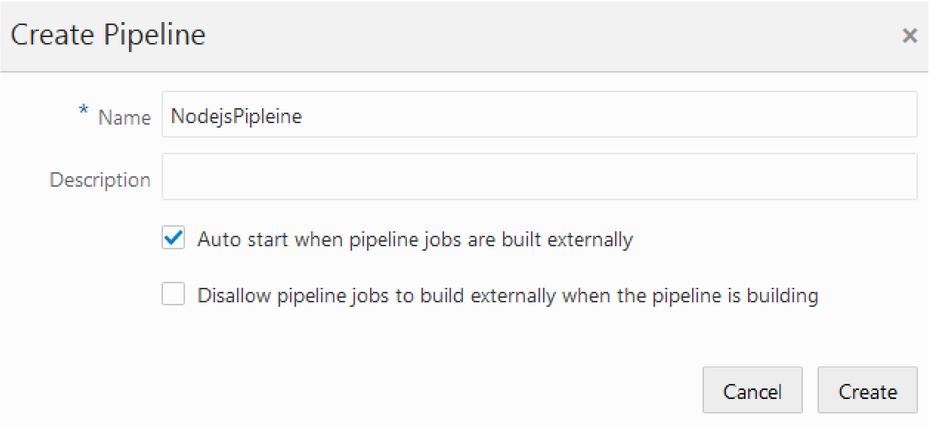
a. Switch to the browser with the target project.

b. On the **Builds** page, click the **Pipelines** tab.

c. Click **+ Create Pipeline**.

d. In the Create Pipeline dialog box, enter a name and a description.

   Example:



Create Pipeline

* Name: NodejsPipleine

Description:

☑ Auto start when pipeline jobs are built externally

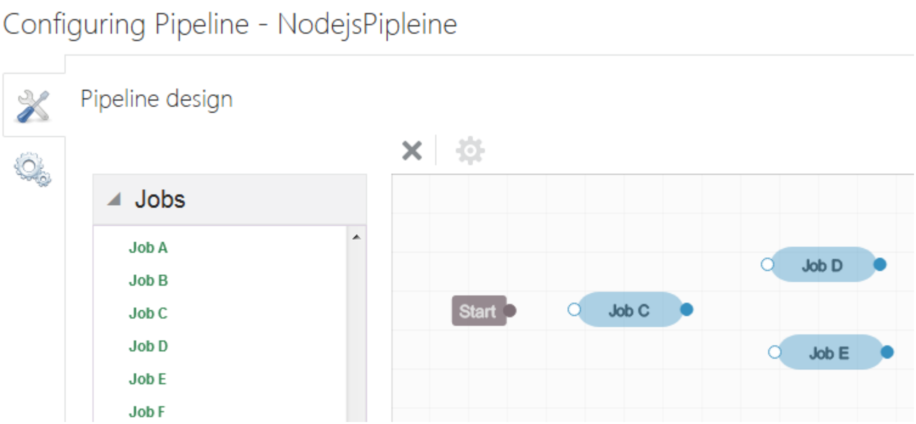
□ Disallow pipeline jobs to build externally when the pipeline is building

   If you want trigger the pipeline's build if a job of the pipeline is triggered externally (outside the pipeline), leave the select the **Auto start when pipeline jobs are build externally** check box selected. To disable manual or automatic builds of the jobs that are part of the pipeline when the pipeline is running, select the **Disallow pipeline jobs to build externally when the pipeline is building** check box.

e. Click **Create**.

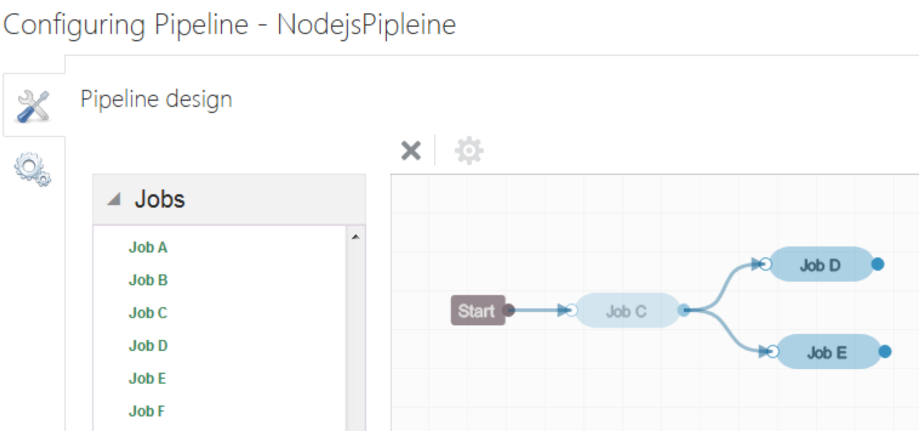
f. In the Pipeline Design tab, from the left list, drag-and-drop the job you opened in Step 2 to the designer area. Then, drag-and-drop the downstream jobs to the designer area.

   Example:



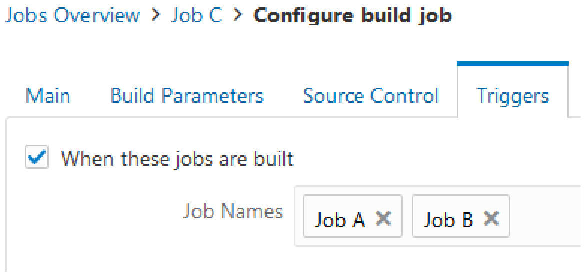
g. Design the dependencies.

Example:



h. Click **Save**.

If a job has both upstream and downstream jobs, then instead of creating another pipeline for the upstream jobs, you can use the pipeline you created for downstream jobs and extend it. In the **Pipelines** tab, for the pipeline you created for downstream jobs, click **Configure** ⚙ and design the diagram.
For example, if you've a job (Job C) with upstream jobs (Job A and Job B) and downstream jobs (Job D and Job E):

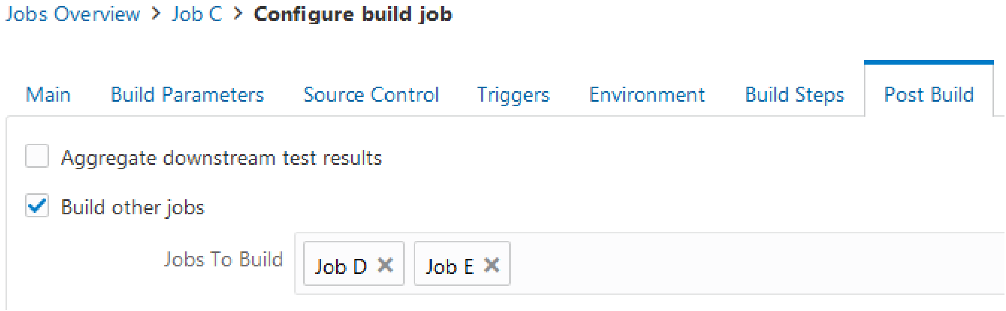


Jobs Overview > Job C > **Configure build job**

Main | Build Parameters | Source Control | Triggers

☑ When these jobs are built

Job Names: Job A ✕ Job B ✕



Jobs Overview > Job C > **Configure build job**

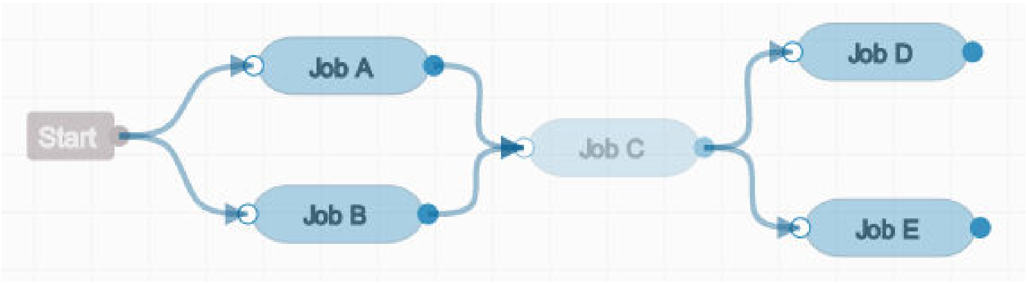
Main | Build Parameters | Source Control | Triggers | Environment | Build Steps | Post Build

□ Aggregate downstream test results

☑ Build other jobs

Jobs To Build: Job D ✕ Job E ✕

Then, design a pipeline diagram like this:



Run a Build of a Job or a Pipeline

To run a build of the target job, open the job's details page and click **Build Now**.

To run a build of a pipeline, in the **Pipelines** tab, click **Build** ▶.

After a build runs, open the job's details page and click the report icons to view the build reports. See View a Job’s Builds and Reports in *Using Oracle Developer Cloud Service*.

Docker Registries

Follow these steps to migrate Docker registries:

1. In the source and target projects, follow these steps:
   a. In the navigation bar, click **Project Administration** ⚙.
   b. Click **Repositories**.

   c. Expand **Docker Repositories**.

2. Repeat these steps to migrate each Docker registry:
   a. In the source project, gather information about a Docker registry.
   b. In the target project, add the Docker registry.
3. After migrating all Docker registries, with the **Repositories** page open in both source and target projects, verify that registries in the target project match the source project.

Gather Information About a Linked Docker Registry From the Source Project

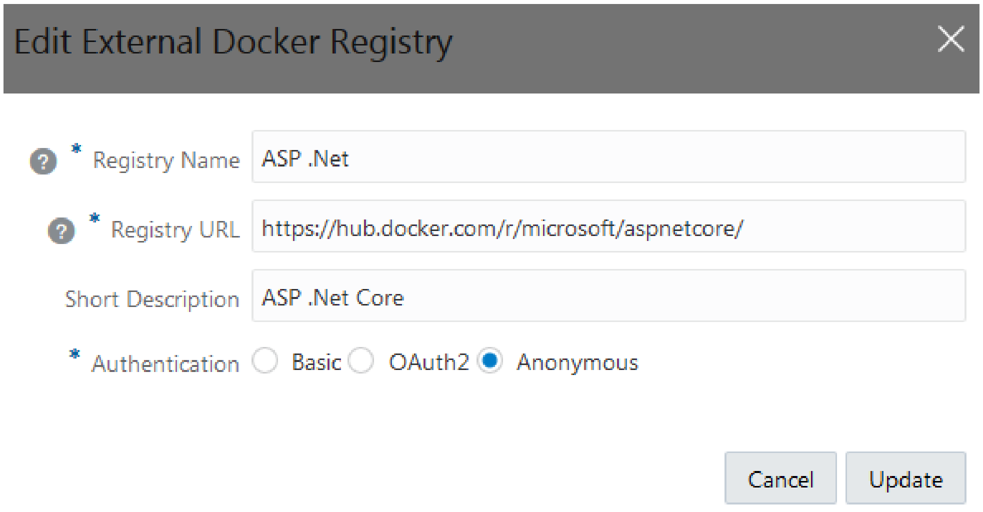
Perform these steps in the source project.

1. Switch to the browser with the source project.
2. In **Docker Repositories**, note the linked Docker registry's name, URL, description, and authentication.

   Click **Menu** ☰ of the registry and select **Edit**. Copy the required details from the Edit Registry dialog box.

   If a registry is private, get its access credentials too. The password (or the auth token) is not visible in the dialog box.

   Example:



Edit External Docker Registry

* Registry Name: ASP .Net
* Registry URL: https://hub.docker.com/r/microsoft/aspnetcore/
Short Description: ASP .Net Core
* Authentication: ○ Basic ○ OAuth2 ● Anonymous

   This table shows the Name, Description, URL, and Authentication values from the above image.

Field	Value
Registry Name	ASP.Net
Registry URL	https://hub.docker.com/r/microsoft/aspnetcore/
Short Description	ASP .Net Core
Authentication	Anonymous

Link Docker Registries in the Target Project

Perform these steps in the target project.

1. Get the details of the Docker registry from the source project. You'll use the information to add the Docker registry in the target project.

   Example:

Field	Value
Registry Name	ASP.Net
Registry URL	https://hub.docker.com/r/microsoft/aspnetcore/
Short Description	ASP .Net Core
Authentication	Anonymous

2. Switch to the browser with the target project.
3. In **Docker Registries**, click **+ Link External Registry**.
4. In **Registry Name** and **Short Description**, enter the Docker registry name and a description.
5. In **Registry URL**, enter the URL of the Docker registry.
6. In **Authentication**, select the authentication type.
7. Click **Create**.
8. In the navigation bar, click **Docker**
9. Browse and verify that you can access the repositories and images of the linked Docker registry.

Maven Artifacts

You might have several artifacts in the source project's Maven repository, such as dependencies and build artifacts. To migrate them, you'll need to note details of each artifact, download it, and then upload it again in the target project. You may choose not to migrate build artifacts stored in the Maven repository of the source project as you can generate and archive them by running a build in the target project.

Follow these steps to migrate Maven artifacts:

1. In the source and target projects,, in the navigation bar, click **Maven**.
2. Repeat these steps to migrate each Maven artifact:
   a. In the source project, gather information about a Maven artifact and download it to your computer.
   b. In the target project, upload the Maven artifact.
3. After migrating all Maven artifacts, with the **Maven** page open in both source and target projects, verify that artifacts in the target project match the source project.
4. In the source and target projects, follow these steps:
   a. In the navigation bar, click **Project Administration**.

   b. Click **Repositories**.

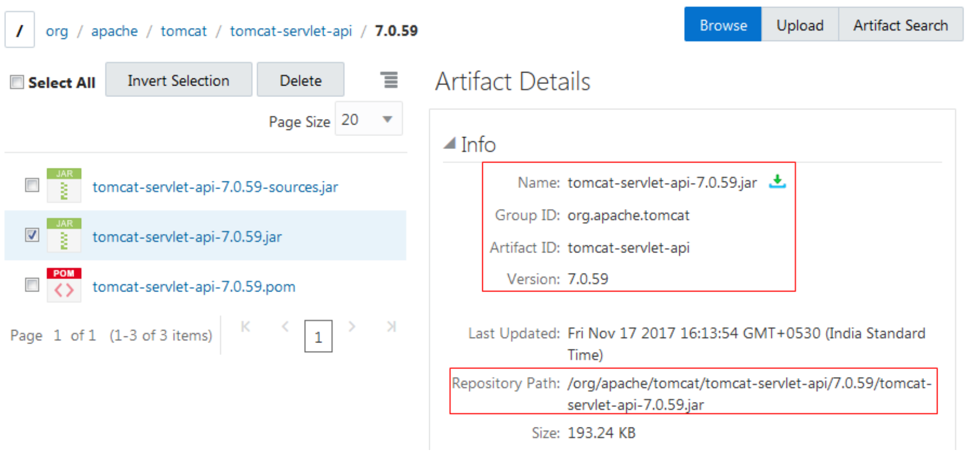
5. Note the Maven auto-cleanup and overwrite settings of the source project and make the same settings in the target project.
6. After migrating all Maven settings, with the **Repositories** page open in both source and target projects, verify that Maven settings in the target project match the source project.

Gather Information About a Maven Artifact From the Source Project

Perform these steps in the source project.

1. Switch to the browser with the source project.
2. On the right side of the page, click **Browse**.
3. Browse and select the artifact.
4. From the **Artifact Details** section on the right, note the artifact's name, path, group ID, artifact ID, and version number.

   Example:


/ org / apache / tomcat / tomcat-servlet-api / 7.0.59
Browse
Upload
Artifact Search
□ Select All
Invert Selection
Delete
Page Size 20
JAR
POM
Page 1 of 1 (1-3 of 3 items)
1

□ tomcat-servlet-api-7.0.59-sources.jar
☑ tomcat-servlet-api-7.0.59.jar
□ tomcat-servlet-api-7.0.59.pom

Artifact Details

Info

Name: tomcat-servlet-api-7.0.59.jar
Group ID: org.apache.tomcat
Artifact ID: tomcat-servlet-api
Version: 7.0.59

Last Updated: Fri Nov 17 2017 16:13:54 GMT+0530 (India Standard Time)
Repository Path: /org/apache/tomcat/tomcat-servlet-api/7.0.59/tomcat-servlet-api-7.0.59.jar
Size: 193.24 KB

   This table shows the Artifact Name, Path, Group ID, Artifact ID, and Version Number values from the above image.

Field	Value
Name	tomcat-servlet-api-7.0.59.jar
Path	/org/apache/tomcat/tomcat-servlet-api/7.0.59
Group ID	org.apache.tomcat
Artifact ID	tomcat-servlet-api
Version Number	7.0.59

5. Click **Download** ⤓.

Upload Maven Artifacts in the Target Project

Perform these steps in the target project.

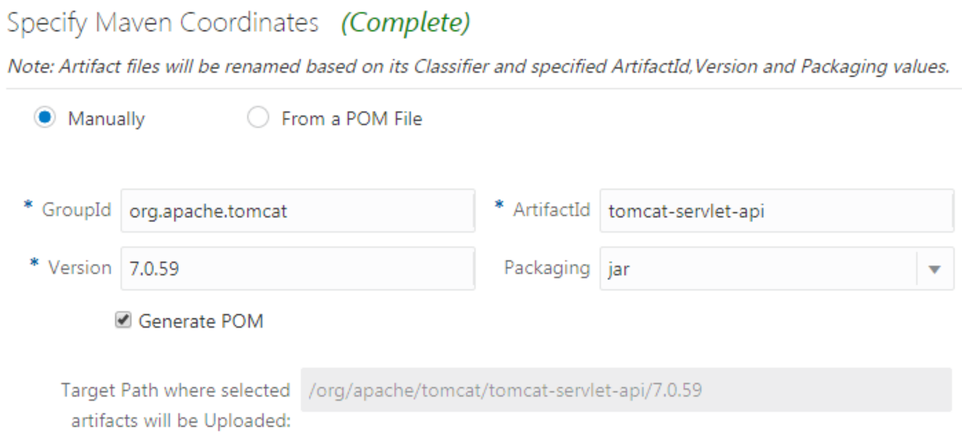
1. Get the details of the Maven artifact from the source project. You'll use the information to upload the artifact in the target project.

   Example:

Field	Value
Name	tomcat-servlet-api-7.0.59.jar
Path	/org/apache/tomcat/tomcat-servlet-api/7.0.59
Group ID	org.apache.tomcat
Artifact ID	tomcat-servlet-api
Version Number	7.0.59

2. Switch to the browser with the target project.
3. On the right side of the **Maven** page, click **Upload**.
4. In the **Upload Artifacts** section, add the artifact to upload, and specify its Maven coordinates (Group ID, Artifact ID, and version number) that you noted.

   Some fields might get populated automatically when you add the artifact in the **Upload** section. Make sure the path of the uploaded artifact is the same as you noted.
   Example:



Specify Maven Coordinates *(Complete)*

Note: Artifact files will be renamed based on its Classifier and specified ArtifactId,Version and Packaging values.

● Manually ○ From a POM File

* GroupId: org.apache.tomcat

* ArtifactId: tomcat-servlet-api

* Version: 7.0.59

Packaging: jar

☑ Generate POM

Target Path where selected artifacts will be Uploaded: /org/apache/tomcat/tomcat-servlet-api/7.0.59

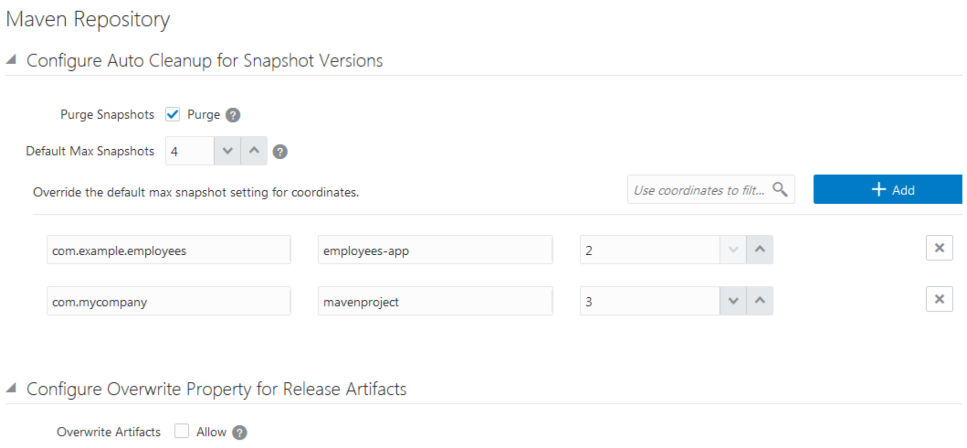
5. Click **Start Upload**.
6. After uploading, verify the path of the uploaded artifact is same as you noted in the source project.

Gather Information About Maven Administration Settings From the Source Project

Perform these steps in the source project.

1. Switch to the browser with the source project.
2. In **Maven Repository**, expand **Configure auto cleanup for Snapshot versions** and **Configure Overwrite Property for Release Artifacts**, and note their field values.

   Example:



Maven Repository

Configure Auto Cleanup for Snapshot Versions

Purge Snapshots ☑ Purge

Default Max Snapshots 4

Override the default max snapshot setting for coordinates.

com.example.employees	employees-app	2
com.mycompany	mavenproject	3

Configure Overwrite Property for Release Artifacts

Overwrite Artifacts □ Allow

This table shows the Field and Value values from the above image.

Field	Value
Purge Snapshots	Selected
Default Max Snapshots	4
Customized Snapshot Counts	Snapshot 1: • GroupID: com.example.employees • ArtifactID: employees-app • Snapshot Count: 2 Snapshot 2: • GroupID: com.mycompany • ArtifactID: mavenproject • Snapshot Count: 3
Overwrite Artifacts	Not selected

Configure Maven Administration Settings in the Target Project

Perform these steps in the target project.

1. Get the details of Maven administration settings from the source project. You'll use the information to configure Maven admin settings in the target project.

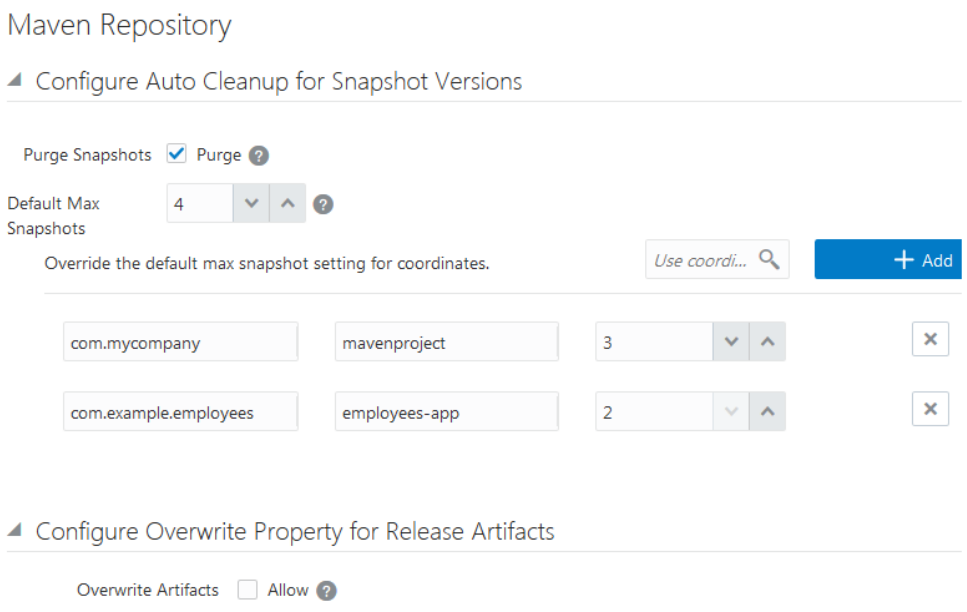
   Example:

Field	Value
Purge Snapshots	Selected
Default Max Snapshots	4

Field	Value
Customized Snapshot Counts	Snapshot 1: • GroupID: com.example.employees • ArtifactID: employees-app • Snapshot Count: 2 Snapshot 2: • GroupID: com.mycompany • ArtifactID: mavenproject • Snapshot Count: 3
Overwrite Artifacts	Not selected

2. Switch to the browser with the target project.
3. In **Maven Repository**, expand **Configure auto cleanup for Snapshot versions** and **Configure Overwrite Property for Release Artifacts**.
4. Set the fields of both sections as noted in the table.

   Example:



Maven Repository

Configure Auto Cleanup for Snapshot Versions

Purge Snapshots ☑ Purge

Default Max Snapshots: 4

Override the default max snapshot setting for coordinates.

com.mycompany	mavenproject	3
com.example.employees	employees-app	2

Configure Overwrite Property for Release Artifacts

Overwrite Artifacts □ Allow

Environments

Follow these steps to migrate environments and its service instances:

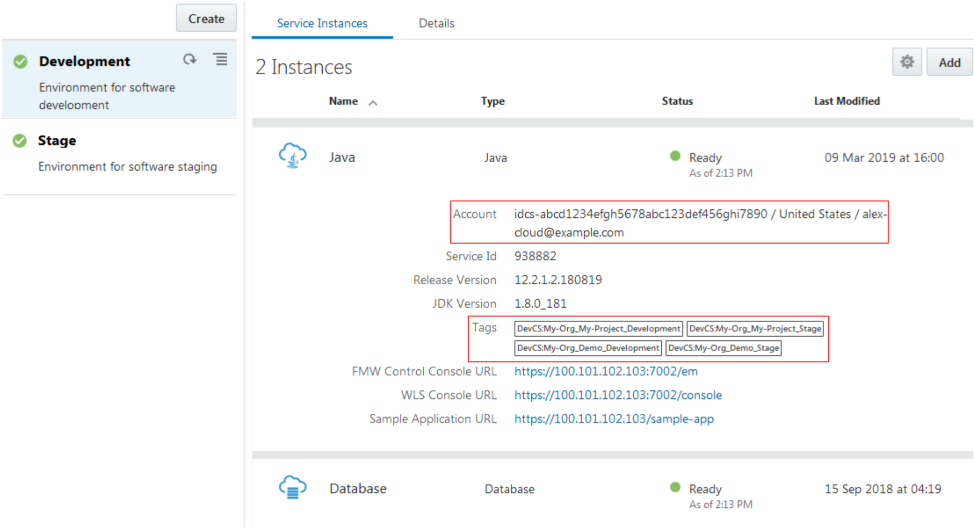
1. In the source and target projects, in the navigation bar, click **Environments** .
2. Repeat these steps to migrate each environment:
   a. In the source project, gather information about an environment and its service instances.
   b. In the target project, create the environment and add its service instances.
3. After migrating all environments, with the **Environments** page open in both source and target projects, verify that environments in the target project match the source project.

Gather Information About an Environment and Service Instances From the Source Project

Perform these steps in the source project.

1. Switch to the browser with the source project.
2. On the **Environments** page, from the left list, select the environment.
3. Note the account and tags of each service instance of the environment.

   Example:



Development
Environment for software development

Stage
Environment for software staging

2 Instances

Name	Type	Status	Last Modified
Java	Java	Ready As of 2:13 PM	09 Mar 2019 at 16:00
Database	Database	Ready As of 2:13 PM	15 Sep 2018 at 04:19

Account: idcs-abcd1234efgh5678abc123def456ghi7890 / United States / alex-cloud@example.com
Service Id: 938882
Release Version: 12.2.1.2.180819
JDK Version: 1.8.0_181
Tags: DevCS:My-Org_My-Project_Development DevCS:My-Org_My-Project_Stage DevCS:My-Org_Demo_Development DevCS:My-Org_Demo_Stage
FMW Control Console URL: https://100.101.102.103:7002/em
WLS Console URL: https://100.101.102.103:7002/console
Sample Application URL: https://100.101.102.103/sample-app

This table shows the Environment Name, Description, Service Instance, Account, and Tags values from the above image.

Environment Name	Description	Service Instance	Account	Tags
Development	Environment for software development	Java	idcs-abcd1234efgh5678abc123def456ghi7890 / United States / alex-cloud@example.com	DevCS:My-Org_My-Project_Development, DevCS:My-Org_My-Project_Stage, DevCS:My-Org_Demo_Development, DevCS:My-Org_Demo_Stag e

Environment Name	Description	Service Instance	Account	Tags
		Database	idcs-abcd1234efgh5678abc123def456ghi7890 / United States / alex-cloud@example.com	DevCS:My-Org_My-Project_Development, DevCS:My-Org_My-Project_Stage, DevCS:My-Org_Demo_Development, DevCS:My-Org_Demo_Stage

Create an Environment and Add Service Instances in the Target Project

Perform these steps in the target project.

1. Get the details of the environment from the source project.. You'll use the information to create the environment and add its service instances to them in the target project.

   Example:

Environment Name	Description	Service Instance	Account	Tags
Development	Environment for software development	Java	idcs-abcd1234efgh5678abc123def456ghi7890 / United States / alex-cloud@example.com	DevCS:My-Org_My-Project_Development, DevCS:My-Org_My-Project_Stage, DevCS:My-Org_Demo_Development, DevCS:My-Org_Demo_Stage
		Database	idcs-abcd1234efgh5678abc123def456ghi7890 / United States / alex-cloud@example.com	DevCS:My-Org_My-Project_Development, DevCS:My-Org_My-Project_Stage, DevCS:My-Org_Demo_Development, DevCS:My-Org_Demo_Stage

2. Switch to the browser with the target project.

3. On the **Environments** page, click **Create** (or **Create Environment** if the page is empty).
4. In **Environment Name**, enter the name of the environment. In **Description**, enter the description.
5. Click **Create**.
6. In the **Service Instances** tab, click **Add**.
7. Add the relevant service instances listed in the table from the appropriate identity domain. Make sure you specify the same tag names as applied to the service instance in the source DevCS project.

   To add instances from an identity domain other than the one you are currently connected to, click **Edit** at the top of the Add Service Instances dialog box.

Deployment Configurations

Follow these steps to migrate deployment configurations:

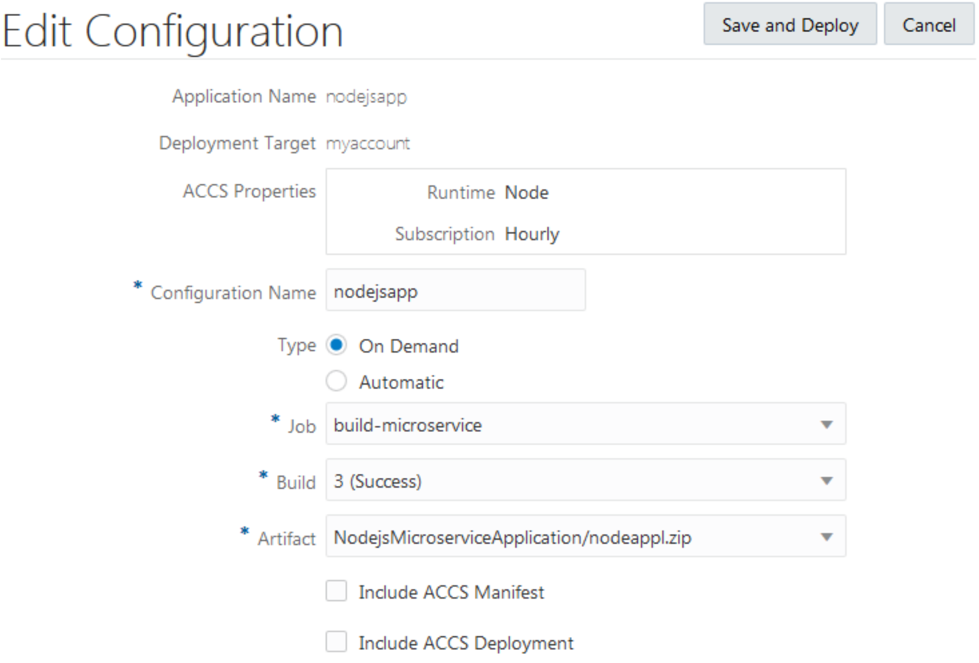
1. In the source and target projects, in the navigation bar, click **Deployments** .
2. Repeat these steps to migrate each deployment configuration:
   a. In the source project, gather information about a deployment configuration and its target.
   b. In the target project, create and configure the deployment configuration.
3. After migrating all deployment configurations, with the **Deployments** page open in both source and target projects, verify that deployment configurations in the target project match the source project.

Gather Information About a Deployment Configuration From the Source Project

Perform these steps in the source project.

1. Switch to the browser with the source project.
2. Select the deployment configuration's tile.
3. In the deployment configuration's tile, mouse over **Settings** ⚙ and select **Edit Configuration**.
4. Note the application name, configuration name, status, deployment target, type, job, and artifact.

   Example:



Edit Configuration

Application Name	nodejsapp
Deployment Target	myaccount
ACCS Properties	Runtime Node Subscription Hourly
* Configuration Name	nodejsapp
Type	● On Demand ○ Automatic
* Job	build-microservice
* Build	3 (Success)
* Artifact	NodejsMicroserviceApplication/nodeappl.zip
	□ Include ACCS Manifest
	□ Include ACCS Deployment

This table shows the Application Name, Deployment Target, Configuration Name, Status, Deployment Target, Type, Job, Build, and Artifact values from the above image.

Field	Value
Application Name	nodejsapp
Configuration Name	nodejsapp
Deployment Target	myaccount
Type	On Demand
Job	build-microservice
Build	3
Artifact	NodejsMicroserviceApplication/nodeappl.zip
Additional Info	ACCS Properties Runtime: Node Subscription: Hourly

Create a Deployment Configuration in the Target Project

Perform these steps in the target project.

1. Get the details of the deployment configuration from the source project. You'll use the information to create a deployment configuration in the target project.

   Example:

Field	Value
Application Name	nodejsapp
Configuration Name	nodejsapp

Field	Value
Deployment Target	myaccount
Type	On Demand
Job	build-microservice
Build	3
Artifact	NodejsMicroserviceApplication/nodeappl.zip
Additional Info	ACCS Properties Runtime: Node Subscription: Hourly

2. Switch to the browser with the target project.
3. On the **Build** page, ensure that build job and build artifact required by deployment configuration exist in the project.

   If it doesn't exist, then run a build of the deployment configuration's job to generate the build artifact.
4. On the **Deployments** page, click **+ Create Configuration**.
5. In **Configuration Name**, enter the configuration name you noted. By default, the same name is used by **Application Name**, however you can modify it.
6. In **Deployment Target**, click **New**, and create the deployment target.

   You'll need credentials of a user who can access the target service instance.
7. In **Type**, select the deployment type.
8. In **Job**, **Build**, and **Artifact**, specify the job, build, and artifact to deploy to the target service.
9. Click **Save**.
10. Return to the **Deployments** page. In the deployment configuration's tile, mouse over **Settings** ⚙ and select **Deploy** or **Redeploy**.

    If the deployment fails, check the configuration again.

Webhooks

Follow these steps to migrate webhooks:

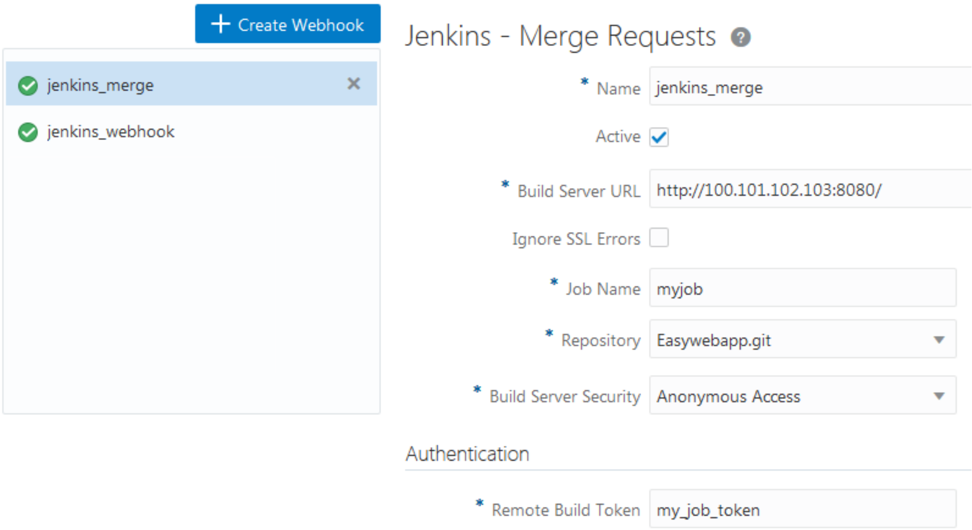
1. In the source and target projects, follow these steps:
   a. In the navigation bar, click **Project Administration** ⚙.
   b. Click **Webhooks**.
2. Repeat these steps to migrate each webhook:
   a. In the source project, gather information about a webhook.
   b. In the target project, create and configure the webhook.
3. After migrating all webhooks, with the **Webhooks** page open in both source and target projects, verify that webhooks in the target project match the source project.

Gather Information About a Webhook From the Source Project

Perform these steps in the source project.

1. Switch to the browser with the source project.
2. On the **Webhooks** page, in the left list, select the webhook and note the type and values of its fields. The webhook type is displayed as the title.

   Example:



Jenkins - Merge Requests

* Name: jenkins_merge

Active: ☑

* Build Server URL: http://100.101.102.103:8080/

Ignore SSL Errors: □

* Job Name: myjob

* Repository: Easywebapp.git

* Build Server Security: Anonymous Access

Authentication

* Remote Build Token: my_job_token

This table shows the webhook's fields and its values from the above image.

Field	Value
Webhook Type	Jenkins - Merge Requests
Name	jenkins_merge
Active	Selected
Build Server URL	http://100.101.102.103:8080/
Ignore SSL Errors	Not selected
Job Name	myjob
Repository	Easywebapp.git
Build Server Security	Anonymous Access
Remote Build Token	my_job_token

Create a Webhook in the Target Project

Perform these steps in the target project.

1. Get the details of the webhook from the source project. You'll use the information to create the webhook in the target project.

   Example:

Field	Value
Webhook Type	Jenkins - Merge Requests
Name	jenkins_merge
Active	Selected
Build Server URL	http://100.101.102.103:8080/
Ignore SSL Errors	Not selected
Job Name	myjob
Repository	Easywebapp.git
Build Server Security	Anonymous Access
Remote Build Token	my_job_token

2. Switch to the browser with the target project.
3. On the **Webhooks** page, click **Webhooks**.
4. Click **+ Create Webhook**.
5. In **Type**, select the webhook type.
6. In other fields, enter the values you noted.
7. When you're done, click **Create**.
8. On the Webhooks page, select the webhook, and click **Test**.

   If the test fails, reconfigure the webhook.

Template Definition

If the source project is a template, then to migrate its template variables and rules, you'll need to add them manually to the new project.

Follow these steps to migrate a project's template definition:

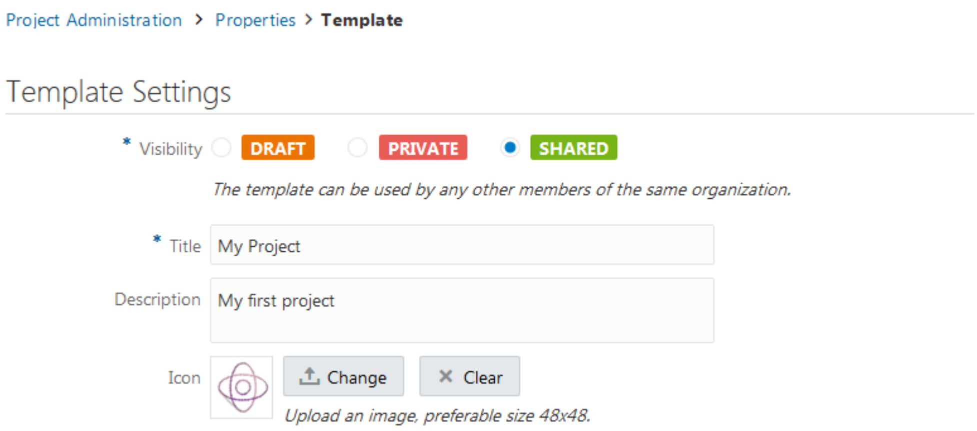
1. In the source and target projects, follow these steps:
   a. In the navigation bar, click **Project Administration** ⚙.
   b. Click **Properties**.
2. In the source project, gather information about the project's template definition.
3. In the target project, configure the template definition.

Gather Information About the Source Project's Template Definition

Perform these steps in the source project.

1. Switch to the browser with the source project.
2. In the **Template** section of the **Properties** page, click **Edit**.
3. Note the project's title, description, and visibility. If there's an icon, right-click the icon, save it to your computer, and note the path of the icon on your computer.

   Example:



Project Administration > Properties > **Template**

Template Settings

* Visibility ○ DRAFT ○ PRIVATE ● SHARED

The template can be used by any other members of the same organization.

* Title: My Project

Description: My first project

Icon

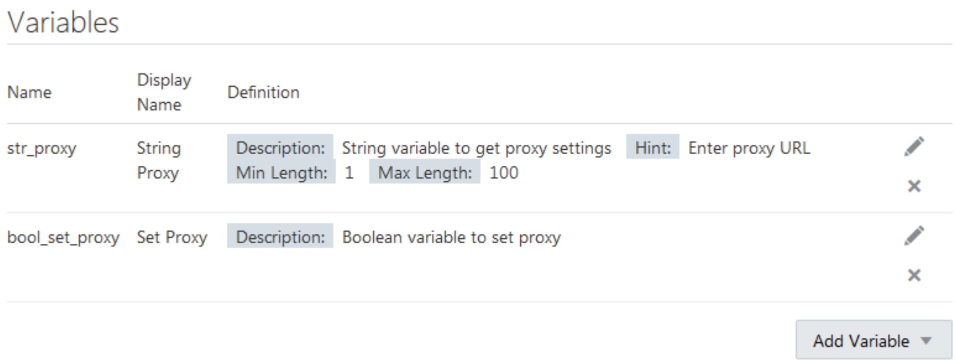
Upload an image, preferable size 48x48.

This table shows the Title and Visibility values from the above image. Also, note the icon's path.

Field	Value
Visibility	Shared
Title	My Project
Description	My first project
Icon path on your computer	D:\Images\Demo\demo_icon.png

4. For each variable in the **Variables** section, click **Edit** and note the name, display name, and values of the variable.

   Example:



Variables

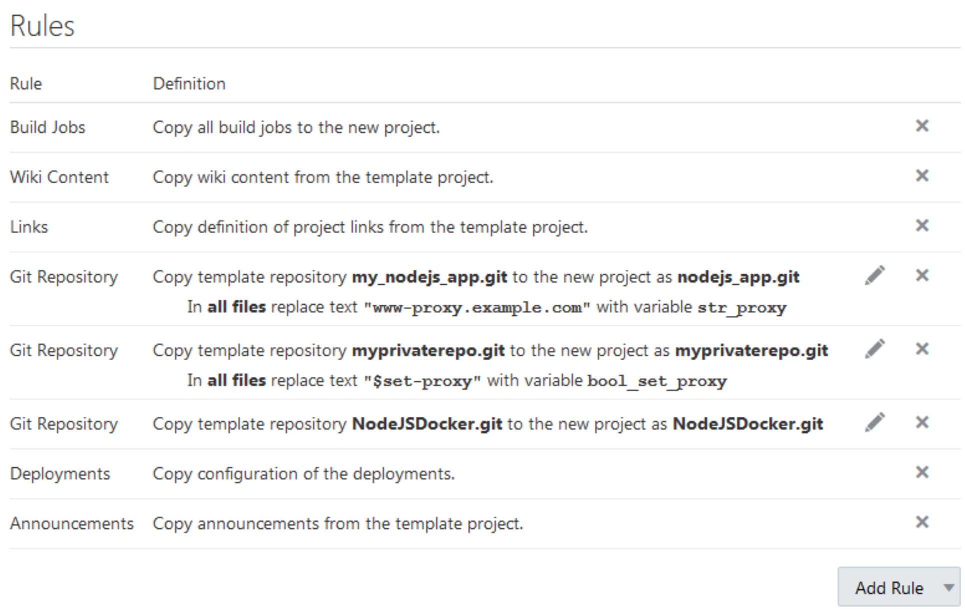
Name	Display Name	Definition
str_proxy	String Proxy	Description: String variable to get proxy settings Hint: Enter proxy URL Min Length: 1 Max Length: 100
bool_set_proxy	Set Proxy	Description: Boolean variable to set proxy

This table shows the Name, Display Name, Description, and Definition values from the above image.

Name	Display Name	Description	Definition
str_proxy	String Proxy	String variable to get proxy settings	Hint: Enter proxy URL Min Length: 1 Max Length: 100
bool_set_proxy	Set Proxy	Boolean variable to set proxy	True Constant: true False Constant: false

5. For each rule in the **Rules** section, note its name and definition. For each Git repository, click **Edit** ✎ and note the source repository name, the target repository name, and the replacement rules (if any).

   Example:



Rules

Rule	Definition
Build Jobs	Copy all build jobs to the new project.
Wiki Content	Copy wiki content from the template project.
Links	Copy definition of project links from the template project.
Git Repository	Copy template repository **my_nodejs_app.git** to the new project as **nodejs_app.git** In **all files** replace text `"www-proxy.example.com"` with variable `str_proxy`
Git Repository	Copy template repository **myprivaterepo.git** to the new project as **myprivaterepo.git** In **all files** replace text `"$set-proxy"` with variable `bool_set_proxy`
Git Repository	Copy template repository **NodeJSDocker.git** to the new project as **NodeJSDocker.git**
Deployments	Copy configuration of the deployments.
Announcements	Copy announcements from the template project.

This table shows the Rule, Definition, and Additional Notes values from the above image.

Rule	Definition	Additional Notes
Build Jobs	Copy all build jobs to the new project.	
Wiki Content	Copy wiki content from the template project.	
Links	Copy definition of project links from the template project.	

Rule	Definition	Additional Notes
Git Repository	Copy template repository **my_nodejs_app.git** to the new project as **nodejs_app.git**	Source Repository: my_nodejs_app.git Repository Name: nodejs_app.git Use target project name check box: Not selected Replacements: • In: All Files • Replace: www-proxy.example.com • With: Variable • variable value: str_proxy
Git Repository	Copy template repository **myprivaterepo.git** to the new project as **myprivaterepo.git**	Source Repository: myprivaterepo.git Repository Name: myprivaterepo.git Use target project name check box: Not selected Replacements: • In: All Files • Replace: $set-proxy • With: Variable • variable value: bool_set_proxy
Git Repository	Copy template repository **NodeJSDocker.git** to the new project as **NodeJSDocker.git**	Source Repository: NodeJSDocker.git Repository Name:NodeJSDocker.git Use target project name check box: Not selected
Deployments	Copy configuration of the deployments.	
Announcements	Copy announcements from the template project.	

Define the Target Project as a Template

☁ Perform these steps in the target project.

1. Get the details of the template from the source project. You'll use the information to define the project as a template.

   Example:

Field	Value
Visibility	Shared
Title	My Project
Description	My first project
Icon path on your computer	D:\Images\Demo\demo_icon.png

Get the details of variables from the source project.

Name	Display Name	Description	Definition
str_proxy	String Proxy	String variable to get proxy settings	Hint: Enter proxy URL Min Length: 1 Max Length: 100
bool_set_proxy	Set Proxy	Boolean variable to set proxy	True Constant: true False Constant: false

Get the details of rules from the source project.

Rule	Definition	Additional Notes
Build Jobs	Copy all build jobs to the new project.	
Wiki Content	Copy wiki content from the template project.	
Links	Copy definition of project links from the template project.	
Git Repository	Copy template repository **my_nodejs_app.git** to the new project as **nodejs_app.git**	Source Repository: my_nodejs_app.git Repository Name: nodejs_app.git Use target project name check box: Not selected Replacements: • In: All Files • Replace: www-proxy.example.com • With: Variable • variable value: str_proxy
Git Repository	Copy template repository **myprivaterepo.git** to the new project as **myprivaterepo.git**	Source Repository: myprivaterepo.git Repository Name: myprivaterepo.git Use target project name check box: Not selected Replacements: • In: All Files • Replace: $set-proxy • With: Variable • variable value: bool_set_proxy
Git Repository	Copy template repository **NodeJSDocker.git** to the new project as **NodeJSDocker.git**	Source Repository: NodeJSDocker.git Repository Name:NodeJSDocker.git Use target project name check box: Not selected

Rule	Definition	Additional Notes
Deployments	Copy configuration of the deployments.	
Announcements	Copy announcements from the template project.	

2. Switch to the browser with the target project.
3. In the **Template** section of the **Properties** page, click **Define Template**.
4. Click **Edit**.
5. In **Visibility**, set the template's visibility.
6. In **Variables**, define the variables.

   For each variable to add, from the **Add Variable** menu, select the variable type. Fill in the fields of the variable and click **Save** ✓.
7. In the **Rules** section, define the rules.
8. Click **Save**.

Announcements

Follow these steps to migrate announcements:

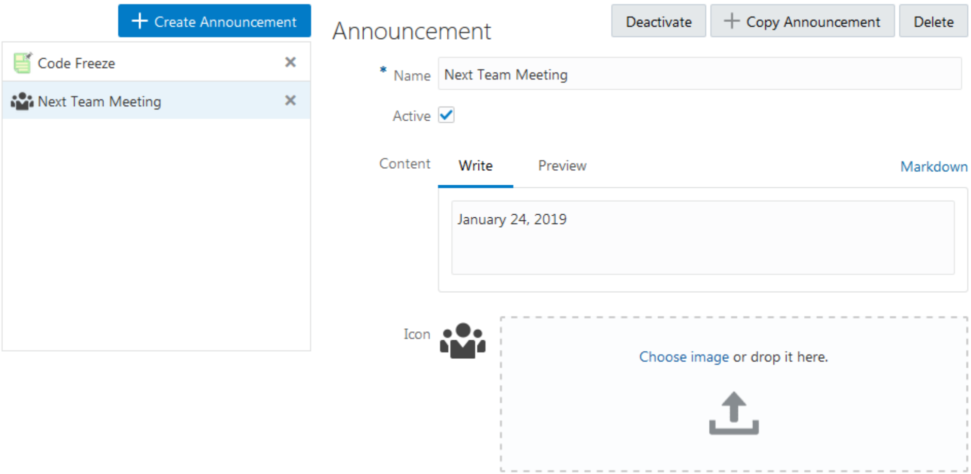
1. In the source and target projects, follow these steps:
   a. In the navigation bar, click **Project Administration** ⚙.
   b. Click **Announcements**.
2. Repeat these steps to migrate each announcement:
   a. In the source project, gather information about an announcement.
   b. In the target project, create and configure the announcement.
3. After migrating all announcements, with the **Announcements** page open in both source and target projects, verify that announcements in the target project match the source project.

Gather Information About an Announcement From the Source Project

Perform these steps in the source project.

1. Switch to the browser with the source project.
2. On the **Announcements** page, in the left list, select the announcement and note its name, active status, and content. If there's an icon, right-click the icon, save it to your computer, and note the path of the icon on your computer.

   Example:



Name: Next Team Meeting

Active: ☑

Content: January 24, 2019

This table shows the Announcement Name, Active, Content, and Icon values from the above image.

Field	Value/Status
Announcement Name	Next Team Meeting
Active	Selected
Content	January 24, 2019
Icon's path on your computer	D:\Announcements\team_icon.png

Create an Announcement in the Target Project

Perform these steps in the target project.

1. Get the details of the announcement from the source project. You'll use the information to create the announcement in the target project

   Example:

Field	Value/Status
Announcement Name	Next Team Meeting
Active	Selected
Content	January 24, 2019
Icon's path on your computer	D:\Announcements\team_icon.png

2. Switch to the browser with the target project.
3. On the **Announcements** page, click **+ New Announcement**.
4. In **Name** and **Contents**, enter the name and the announcement’s text.
5. Upload the icon's image, if there was an icon associated with the announcement in the source project.

6. Click **Create**.
7. In the navigation bar, click **Project Home** .
8. Above the activities feed, verify the announcement.

RSS/ATOM Feeds

Follow these steps to migrate RSS/ATOM feeds:

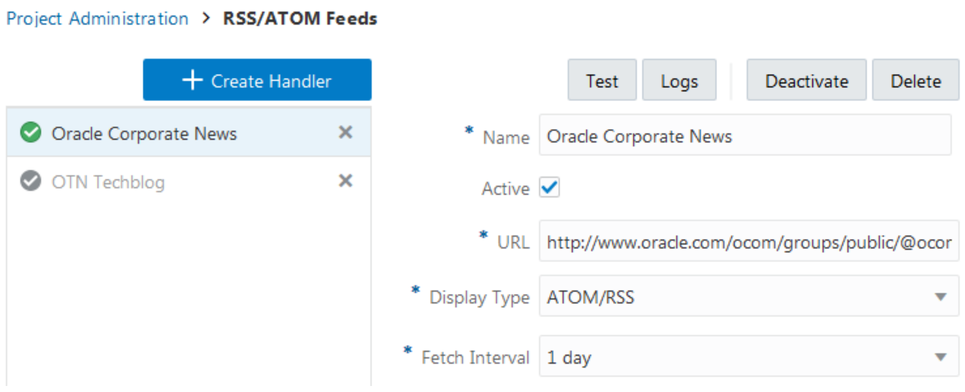
1. In the source and target projects, follow these steps:
   a. In the navigation bar, click **Project Administration** .
   b. Click **RSS/ATOM Feeds**.
2. Repeat these steps to migrate each RSS/ATOM feed:
   a. In the source project, gather information about a feed.
   b. In the target project, create and configure the feed.
3. After migrating all feeds, with the **RSS/ATOM Feeds** page open in both source and target projects, verify that feeds in the target project match the source project.

Gather Information About a RSS/ATOM Feed From the Source Project

Perform these steps in the source project.

1. Switch to the browser with the source project.
2. On the **RSS/ATOM Feeds** page, in the left list, select a feed and note its name, active status, URL, display type, and fetch interval.

   Example:



* Name Oracle Corporate News
Active ☑
* URL http://www.oracle.com/ocom/groups/public/@ocor
* Display Type ATOM/RSS
* Fetch Interval 1 day

This table shows the Feed Name, Active, URL, Display Type, and Fetch Interval values from the above image.

Field	Value/Status
Feed Name	Oracle Corporate News
Active	Yes

Field	Value/Status
URL	http://www.oracle.com/ocom/groups/public/@ocom/documents/webcontent/196280.xml
Display Type	ATOM/RSS
Fetch Interval	1 day

Add a RSS/ATOM Feed in the Target Project

Perform these steps in the target project.

1. Get the details of the RSS/ATOM feeds from the source project. You'll use the information to create RSS/ATOM feeds in the target project.

   Example:

Field	Value/Status
Feed Name	Oracle Corporate News
Active	Yes
URL	http://www.oracle.com/ocom/groups/public/@ocom/documents/webcontent/196280.xml
Display Type	ATOM/RSS
Fetch Interval	1 day

2. Switch to the browser with the target project.
3. On the **RSS/ATOM Feeds** page, click **+ New Handler**.
4. In **Name**, enter the name of the handler.
5. In **URL**, enter the URL of the feed.
6. In **Display Type**, select the feed’s display type.
7. In **Fetch Interval**, enter the feed’s fetch interval.
8. Click **Create**.
9. On the **RSS/ATOM Feeds** page, select the feed, and click **Test**.

   If the test fails, reconfigure the RSS/ATOM feed.
10. In the navigation bar, click **Project Home** .
11. In the activities feed, verify the feed's results.

Custom Link Rules

You don't need to migrate the default link rules as they are available in the target project.

Follow these steps to migrate link rules:

1. In the source and target projects, follow these steps:
   a. In the navigation bar, click **Project Administration** .
   b. Click **Links**.
2. Repeat these steps to migrate each link rule:

a. In the source project, gather information about a rule.

b. In the target project, create and configure the link rule.

3. After migrating all link rules, with the **Links** page open in both source and target projects, verify that link rules in the target project match the source project.

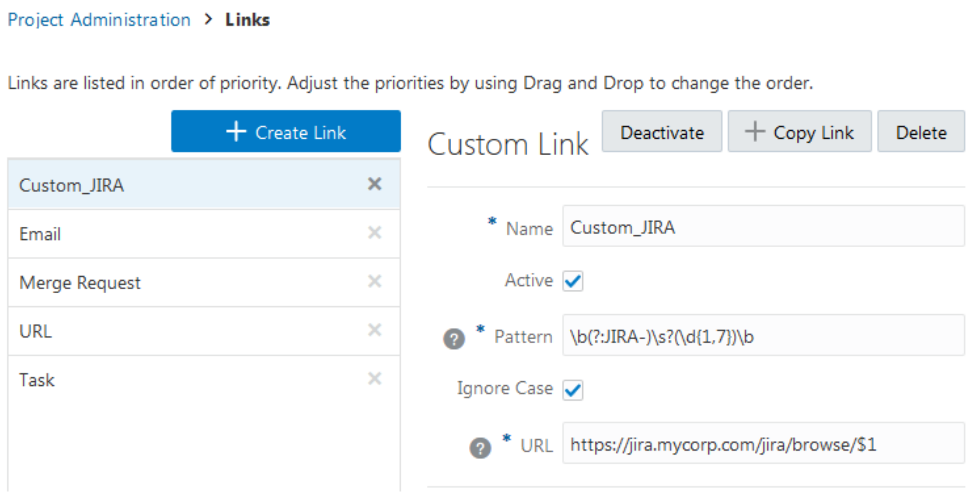
Gather Information About a Link Rule From the Source Project

Perform these steps in the source project.

1. Switch to the browser with the source project.

2. On the **Links** page, in the left list, select a custom rule and note its name, active status, pattern, ignore case status, and URL. Ignore the default link rules.

   Example:



Project Administration > **Links**

Links are listed in order of priority. Adjust the priorities by using Drag and Drop to change the order.

* Name Custom_JIRA

Active ☑

* Pattern \b(?:JIRA-)\s?(\d{1,7})\b

Ignore Case ☑

* URL https://jira.mycorp.com/jira/browse/$1

This table shows the Rule Name, Active Status, Pattern, Ignore Case, and URL values from the above image.

Field	Value/Status
Rule Name	Custom_JIRA
Active Status	Yes
Pattern	\b(?:JIRA-)\s?(\d{1,7})\b
Ignore Case	Selected
URL	https://jira.mycorp.com/jira/browse/$1

Create a Link Rule in the Target Project

Perform these steps in the target project.

1. Get the details of a link rule from the source project. You'll use the information to create a link rule in the target project.

   Example:

Field	Value/Status
Rule Name	Custom_JIRA
Active Status	Yes
Pattern	\b(?:JIRA-)\s?(\d{1,7})\b
Ignore Case	Selected
URL	https://jira.mycorp.com/jira/browse/$1

2. Switch to the browser with the target project.
3. On the **Links** page, click **+ Create Link**.
4. In **Name**, enter a name.
5. Deselect the **Active** check box, if it wasn't selected in the source project.
6. In **Pattern**, enter the RegExp link rule pattern.
7. In **URL**, enter the link URL.
8. Deselect the **Ignore Case** check box, if it was in the source project.
9. Expand **Test** and in **Test Value**, enter a test value. Verify the result link in **Test Result**.
10. Click **Create**.

Remove Your Membership or Ownership

Before you exported the source project's data, you noted whether you're a member or an owner of the project. If you were not a member or owner, you assigned the project's ownership to yourself. See Gather Information About a Project From the Source DevCS Instance.
After migrating all artifacts of the source project to the target project, if you were not a member or an owner of the source project, you should remove your membership or ownership of the target project.

Perform these steps in the target project.

1. Get the details of of the source project's ownership and membership.

   Example:

Field	Value
Project Name	Demo Project
Are you a Member?	No
Are you an Owner?	No

2. Open the target project.
3. In the navigation bar, click **Project Home** .
4. Click the **Team** tab.
5. If you were a member but not an owner of the source project, then for your username, click **Demote to Member** .
6. If you were not a member or an owner of the source project, then for your username, click **Remove** ✕.

Migrate User Preferences

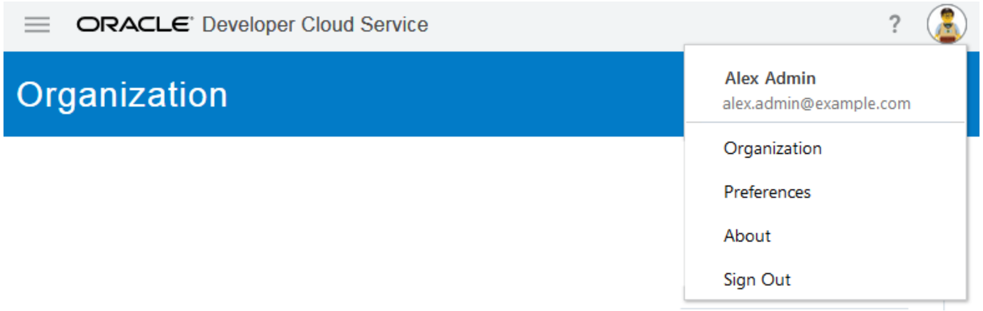
After migrating a project's artifacts and adding team members, ask each user to sign in to DevCS and set their user preferences and project preferences.

User Preferences

To migrate your DevCS user preferences, note them in the source DevCS instance and set them again in the target DevCS instance.

User preferences include their display name, email address, and email notification preferences.

1. In the source and target DevCS instances, click the user avatar, and select **Preferences**.



2. In the source DevCS instance, gather information about your preferences.
3. In the target project, set them again.
4. After setting preferences, with the **User Preferences** tab open in both source and target DevCS instances, verify that your preferences in the target project match the source project.

Gather Information About Your User Preferences From the Source DevCS Instance

Perform these steps in the source DevCS instance.

1. Switch to the browser with the source DevCS instance.
2. On the **User Preferences**, click the **Profile** tab.
3. Note the values of **First Name**, **Last Name**, and **Email Address**.

   Example:

Field	Value/Status
First Name	Alex
Last Name	Admin
Email Address	alex.admin@example.com

4. Click the **Notifications** tab.
5. Note the selected check boxes only.

   Example:

Field	Value/Status
First Name	Alex
Last Name	Admin
Email Address	alex.admin@example.com
Selected check boxes in **Notifications**	• **Issue updates, attachments, and comments** • **Merge Request updates and comments** • **Build activities** • **SCM/Push activities**

Set User Preferences in the Target DevCS Instance

☁ Perform these steps in the target DevCS instance.

1. Get the details of the user preferences from the source DevCS instance.

   Example:

Field	Value/Status
First Name	Alex
Last Name	Admin
Email Address	alex.admin@example.com
Selected check boxes in **Notifications**	• **Issue updates, attachments, and comments** • **Merge Request updates and comments** • **Build activities** • **SCM/Push activities**

2. Switch to the browser with the target DevCS instance.
3. On the **User Preferences**, click the **Profile** tab.
4. Set the name and email address. Changing the name and email address doesn't change your account's name and email, but the display name and display email address on DevCS pages.
5. Click the **Notifications** tab.
6. Select the check boxes that were selected in the source DevCS instance.
7. Click the **Authentication** tab.
8. Add the public key of the SSH private-public key pair. You must upload the public key to access a Git repository via SSH.

   If you don't have a key pair, generate it. See Generate an SSH Key in *Using Oracle Developer Cloud Service*.
   If you don't plan to use SSH to access a Git repository, ignore this step.

Project Preferences

To migrate your project preferences, note them in the source project and set them again in the target project.

Project preferences include favorite Git repositories and Agile boards, and the watch settings of branches and jobs.

Git Repositories and Branches

1. In the source and target projects, follow these steps:
   a. In the navigation bar, click **Project Home** .
   b. Click the **Repositories** tab.
2. Switch to the browser with the source project.
3. Note the repositories marked as your favorite. You can identify them with the ★ icon.
4. Switch to the browser with the target project.
5. In the **Repositories** tab, click the **Favorite** ☆ icon of the repository to mark it as your favorite.
6. In the source and target projects, follow these steps:
   a. In the navigation bar, click **Git** .
   b. Click the **Refs** tab.
7. Switch to the browser with the source project.
8. For each repository in the **Repositories** list, note the branches you watch. They are marked with the **Subscribed** ✓ icon.
9. Switch to the browser with the target project.
10. In the **Refs** tab, select the repository and click **cc** in the branch to watch it.

Jobs

In the source and target projects, open the **Builds** page and repeat these steps for each job in the source project:

1. Switch to the browser with the source project.
2. Click the job's name to open it.
3. If **CCme** is **CCed**, click it and note the states of check boxes in the CC Me dialog box. Click **Cancel**. Then, follow these steps in the target project:
   a. Switch to the browser with the target project.
   b. Open the same job.
   c. Click **CCme**. In the CC Me dialog box, select the check boxes as noted, and click **OK**.
   d. Switch to the browser with the source project.

Agile Boards

1. In the source and target projects, in the navigation bar, click **Boards**.
2. Switch to the browser with the source project.
3. Note the boards marked as your favorite. You can identify them with the ★ icon.
4. Switch to the browser with the target project.
5. For the boards marked as favorite in the source project, click the **Favorite** ☆ icon to mark it as your favorite.

4

Complete the Post-Migration Tasks

After successfully migrating the projects from DevCS on Traditional to DevCS on OCI, test the projects and its artifacts thoroughly, and then perform cleanup tasks.

Test the Target DevCS Instance

After all projects and their artifacts are migrated, thoroughly test all artifacts of each project.

You may open the source project in a web browser window and the target project in another browser window, and match each artifact in both projects. You have some time before you're asked to delete your source DevCS instance and its projects.

Clean Up Resources in Traditional

After thoroughly testing all projects and their data, clean up the resources in the source DevCS instance.

You can't delete DevCS on Traditional service instance. The service instance was made available to you when you created a PaaS instance, such as a Java Cloud Service instance. As long as your PaaS instance is active, the DevCS instance remains active. When it phases out, DevCS phases out too.

Delete Projects in the Source DevCS Instance

Perform these steps in the source DevCS instance as the **Organization Administrator**.

1. Open the source DevCS instance.
2. In the navigation bar, click **Organization** .
3. In the **Projects** tab, click **select all** to select all projects.
4. Click **Update Selected** and select **Delete**.
5. In the Delete Project dialog box, click **Delete**.
6. With all projects selected, click **Update Selected** and select **Remove Forever**.
7. In the Remove Forever dialog box, select the **I understand that projects will be permanently deleted** check box and click **Remove Forever**.

All projects of the source DevCS instance are now deleted.

Delete OCI Object Storage Buckets

After deleting projects of the source DevCS instance, delete the OCI Object Storage buckets you used to export project archive files, unless you want to use them for future storage.

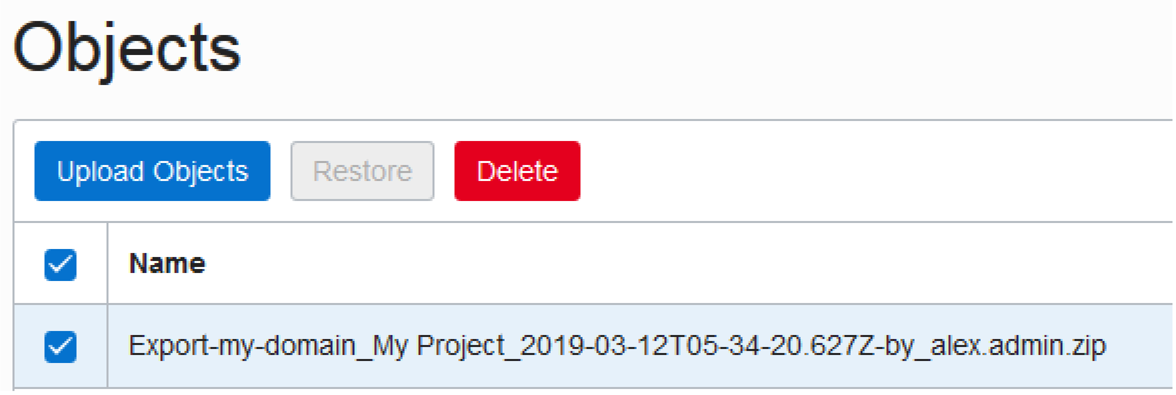
Delete an OCI Object Storage Bucket

To delete an OCI Object Storage bucket, delete its objects first.

To delete a bucket and its objects, sign in to OCI dashboard as the OCI administrator:

1. Open the OCI dashboard.
2. In the left navigation bar, under **Core Infrastructure**, go to **Object Storage** and click **Object Storage**.
3. Click the bucket name.
4. In the **Objects** section, select the check box of the first column heading to select all archive files.

   Example:



☑	Name
☑	Export-my-domain_My Project_2019-03-12T05-34-20.627Z-by_alex.admin.zip

5. Click **Delete**.
6. In the Delete Objects dialog box, click **Delete**.
7. Scroll up to the top of the page and click the **Delete** button below the bucket name.
8. In the Confirm Delete dialog box, click **Delete**.