Artifact Registry: Qwik Start

Activate Cloud Shell

Cloud Shell is a virtual machine that is loaded with development tools. It offers a persistent 5GB home directory and runs on the Google Cloud. Cloud Shell provides command-line access to your Google Cloud resources.

1. Click **Activate Cloud Shell** at the top of the Google Cloud console.

When you are connected, you are already authenticated, and the project is set to your **Project_ID**, qwiklabs-gcp-02-475f3b3397de. The output contains a line that declares the **Project_ID** for this session:

```
Your Cloud Platform project in this session is set to qwiklabs-gcp-02-475f3b3397de
```

gcloud is the command-line tool for Google Cloud. It comes pre-installed on Cloud Shell and supports tab-completion.

2. (Optional) You can list the active account name with this command:

```
gcloud auth list
```

Copied!

content_copy

3. Click **Authorize**.

Output:

```
ACTIVE: *
ACCOUNT: student-01-dc3c1194924a@qwiklabs.net

To set the active account, run:
    $ gcloud config set account `ACCOUNT`
```

4. (Optional) You can list the project ID with this command:

```
gcloud config list project
```

Copied!

content_copy

Output:

```
[core]
project = qwiklabs-gcp-02-475f3b3397de
```

Note: For full documentation of gcloud, in Google Cloud, refer to the gcloud CLI overview guide.

Task 1. Create a Docker repository

In this section, you'll create a private Docker repository within Artifact Registry. This repository will serve as a central location to store and manage your Docker images. You'll use the gcloud command-line tool to create the repository and then verify its creation through the Google Cloud Console.

1. Open a new Cloud Shell window by clicking the icon (>_) in the top right corner of the console.

2. Run the following command to get your Project ID and save it as an environment variable:

```
export PROJECT_ID=$(gcloud config get-value project)
```
Copied!

content_copy

3. Run the following command to create a new Docker repository named example-docker-repo in the location us-east4 with the description "Docker repository".

```
gcloud artifacts repositories create example-docker-repo --repository-format=docker \
  --location=us-east4 --description="Docker repository" \
  --project=$PROJECT_ID
```
Copied!

content_copy

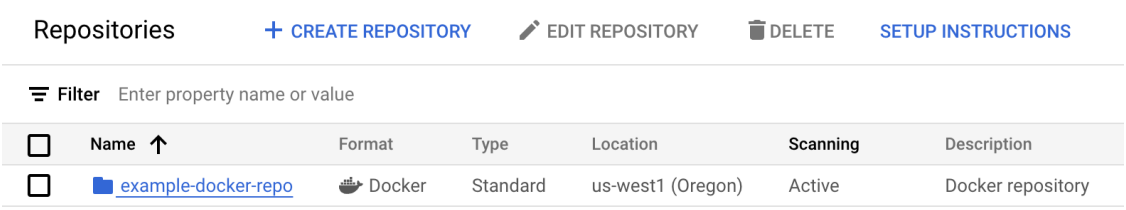
4. Run the following command to verify that your repository was created.

```
gcloud artifacts repositories list \
  --project=$PROJECT_ID
```
Copied!

content_copy

5. From the search bar at the top of the console, type **Artifact Registry** and select the first result.

6. On the **Artifact Registry** product page, verify you can see your repository. It should resemble the following:



7. Click the **example-docker-repo** repository. You should notice there are no files inside the repository. In the next sections, you will be adding files to the repository.

You can also create repositories from the **Artifact Registry** product page by clicking the **Create Repository** button and following the same steps as above.

Task 2. Configure authentication for Artifact Registry

To push and pull images from your newly created Docker repository, you need to configure Docker to authenticate with Artifact Registry. This involves setting up credentials that allow your Docker client to interact with the repository securely.

Before you can push or pull images, you will need to configure Docker to use the Google Cloud CLI to authenticate requests to Artifact Registry.

1. To set up authentication to Docker repositories in the region `us-east4`, run the following command:

```
gcloud auth configure-docker us-east4-docker.pkg.dev
```

Copied!

content_copy

The command updates your Docker configuration. You can now connect with Artifact Registry in your Google Cloud project to push and pull images.

For information about other authentication methods, see Authentication methods.

Task 3. Obtain an image to push

You'll need a Docker image to work with in this lab. Instead of building an image from scratch, you'll pull a pre-built sample image from a public repository. This will allow you to focus on interacting with Artifact Registry.

For this lab, you will push a sample image named hello-app.

1. Run the following command to pull version 1.0 of the image.

```
docker pull us-docker.pkg.dev/google-samples/containers/gke/hello-app:1.0
```

Copied!

content_copy

Image paths in Artifact Registry include multiple parts. For this sample image:

- us-docker.pkg.dev is the hostname for container images stored in Artifact Registry Docker repositories, which includes the location of the repository (us).
- google-samples is the project ID.
- containers is the repository ID.
- /gke/hello-app is the path to the image in the repository containers.

Task 4. Add the image to the repository

Now you'll add the sample image to your private repository. This involves tagging the image with the repository name to specify its destination and then pushing it to Artifact Registry.

Before you push the Docker image to Artifact Registry, you must tag it with the repository name.

Tag the image with a registry name

Tagging the image ensures it's pushed to the correct location, which for this lab is `us-east4-docker.pkg.dev`.

1. Run the following command to tag the image as sample-image:tag1:

```
docker tag us-docker.pkg.dev/google-samples/containers/gke/hello-app:1.0 \
us-east4-docker.pkg.dev/$PROJECT_ID/example-docker-repo/sample-image:tag1
```

Copied!

content_copy

Where:

- us-east4 is the repository location.
- us-east4-docker.pkg.dev is the hostname for the Docker repository you created.
- $PROJECT_ID is your Google Cloud Project ID.
- example-docker-repo is the ID of the repository you created.
- sample-image is the image name you want to use in the repository. The image name can be different than the local image name. For this lab you will store the image directly under the repository ID example-docker-repo.
- tag1 is a tag you're adding to the Docker image. If you didn't specify a tag, Docker will apply the default tag latest.

You are now ready to push the image to the repository you created.

Push the image to Artifact Registry

After you have configured authentication and tagged the local image, you can push the image to the repository that you created.

To push the Docker image, run the following command:

```
docker push us-east4-docker.pkg.dev/$PROJECT_ID/example-docker-repo/sample-image:tag1
```

Task 5. Pull the image from Artifact Registry

Finally, you'll pull the image that you just pushed to your private repository. This simulates how you would access and use images stored in Artifact Registry in a real-world scenario.

1. To pull the image from Artifact Registry onto your local machine, run the following command:

```
docker pull us-east4-docker.pkg.dev/$PROJECT_ID/example-docker-repo/sample-image:tag1
```

Copied!

content_copy

You should see output similar to the following:

```
latest: Pulling from [PROJECT-ID]/sample-image:tag1
Digest: sha256:70c42...
Status: Image is up to date for us-east4-docker.pkg.dev/$PROJECT_ID/example-docker-repo/sample-image:tag1
```