


Dataflow: Qwik Start - Python

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.
2. Click through the following windows:
 - Continue through the Cloud Shell information window.
 - Authorize Cloud Shell to use your credentials to make Google Cloud API calls.

When you are connected, you are already authenticated, and the project is set to your **Project_ID**, `qwiklabs-gcp-02-aa06b351945c`. The output contains a line that declares the **Project_ID** for this session:

```
Your Cloud Platform project in this session is set to quiklabs-gcp-02-aa06b351945c
```

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

3. (Optional) You can list the active account name with this command:
`gcloud auth list`

Copied!

content_copy

4. Click **Authorize**.

Output:

```
ACTIVE: *  
ACCOUNT: student-01-1475c7174b9c@qwiklabs.net
```

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

5. (Optional) You can list the project ID with this command:
`gcloud config list project`
Copied!

content_copy

Output:

```
[core]  
project = gwiklabs-gcp-02-aa06b351945c
```

Note: For full documentation of `gcloud`, in Google Cloud, refer to [the gcloud CLI overview guide](#).

Set the region

- In Cloud Shell, run the following command to set the project region for this lab:
`gcloud config set compute/region europe-west4`
Copied!

content_copy

Ensure that the Dataflow API is successfully enabled

To ensure access to the necessary API, restart the connection to the Dataflow API.

1. In the Cloud Console, enter "Dataflow API" in the top search bar. Click on the result for **Dataflow API**.
2. Click **Manage**.
3. Click **Disable API**.


If asked to confirm, click **Disable**.

4. Click **Enable**.

When the API has been enabled again, the page will show the option to disable.

Task 1. Create a Cloud Storage bucket

When you run a pipeline using Dataflow, your results are stored in a Cloud Storage bucket. In this task, you create a Cloud Storage bucket for the results of the pipeline that you run in a later task.

1. On the **Navigation menu** () , click **Cloud Storage > Buckets**.
2. Click **Create bucket**.
3. In the **Create bucket** dialog, specify the following attributes:
 - **Name**: To ensure a unique bucket name, use the following name: **qwiklabs-gcp-02-aa06b351945c-bucket**. Note that this name does not include sensitive information in the bucket name, as the bucket namespace is global and publicly visible.
 - **Location type**: Multi-region
 - **Location**: us
 - A location where bucket data will be stored.
4. Click **Create**.
5. If Prompted Public access will be prevented, click **Confirm**.

Task 2. Install the Apache Beam SDK for Python

1. To ensure that you use a supported Python version, begin by running the Python3.9 Docker Image:

```
docker run -it -e DEVSHELL_PROJECT_ID=$DEVSHELL_PROJECT_ID python:3.9 /bin/bash
```

Copied!

content_copy

This command pulls a Docker container with the latest stable version of Python 3.9 and then opens up a command shell for you to run the following commands inside your container.

2. After the container is running, install the latest version of the Apache Beam SDK for Python by running the following command from a virtual environment:

```
pip install 'apache-beam[gcp]'==2.42.0
```

Copied!

content_copy

You will see some warnings returned that are related to dependencies. It is safe to ignore them for this lab.

3. Run the `wordcount.py` example locally by running the following command:
`python -m apache_beam.examples.wordcount --output OUTPUT_FILE`
Copied!

content_copy

You may see a message similar to the following:

```
INFO:root:Missing pipeline option (runner). Executing pipeline using
the default runner: DirectRunner.
INFO:oauth2client.client:Attempting refresh to obtain initial
access token
```

This message can be ignored.

4. You can now list the files that are on your local cloud environment to get the name of the `OUTPUT_FILE`:

```
ls
```

Copied!

content_copy

5. Copy the name of the `OUTPUT_FILE` and `cat` into it:
`cat <file name>`

Copied!

content_copy

Your results show each word in the file and how many times it appears.

Task 3. Run an example Dataflow pipeline remotely

1. Set the `BUCKET` environment variable to the bucket you created earlier:

```
BUCKET=gs://<bucket name provided earlier>
```

Copied!

content_copy

2. Now you'll run the `wordcount.py` example remotely:
`python -m apache_beam.examples.wordcount --project $DEVSHHELL_PROJECT_ID \`
`--runner DataflowRunner \`
`--staging_location $BUCKET/staging \`
`--temp_location $BUCKET/temp \`

```
--output $BUCKET/results/output \  
--region europe-west4
```

Copied!

content_copy

In your output, wait until you see the message:

```
JOB MESSAGE DETAILED: Workers have started successfully.
```

Then continue with the lab.

Task 4. Check that your Dataflow job succeeded

1. Open the Navigation menu and click **Dataflow** from the list of services. You should see your **wordcount** job with a **status** of **Running** at first.
2. Click on the name to watch the process. When all the boxes are checked off, you can continue watching the logs in Cloud Shell. The process is complete when the status is **Succeeded**.
3. Click **Navigation menu** > **Cloud Storage** in the Cloud Console.
4. Click on the name of your bucket. In your bucket, you should see the **results** and **staging** directories.
5. Click on the **results** folder and you should see the output files that your job created:
6. Click on a file to see the word counts it contains.