

Stream Processing with Cloud Pub/Sub and Dataflow: Qwik Start | Google Cloud Skills Boost

Qwiklabs : 13-16 minutes

GSP903



Google Cloud Self-Paced Labs

Overview

Google Cloud Pub/Sub is a messaging service for exchanging event data among applications and services. A producer of data publishes messages to a Cloud Pub/Sub topic. A consumer creates a subscription to that topic. Subscribers either pull messages from a subscription or are configured as webhooks for push subscriptions. Every subscriber must acknowledge each message within a configurable window of time.

Dataflow is a fully-managed service for transforming and enriching data in stream (real-time) and batch modes with equal reliability and expressiveness. It provides a simplified pipeline development environment using the Apache Beam SDK, which has a rich set of windowing and session analysis primitives as well as an ecosystem of source and sink connectors.

Pub/Sub is a scalable, durable event ingestion and delivery system. Dataflow compliments Pub/Sub's scalable, at-least-once delivery model with message deduplication and exactly-once, in-order processing if you use windows and buffering.

What you'll do

1. Read messages published to a Pub/Sub topic
2. Window (or group) the messages by timestamp
3. Write the messages to Cloud Storage

Setup

Before you click the Start Lab button

Read these instructions. Labs are timed and you cannot pause them. The timer, which starts when you click **Start Lab**, shows how long Google Cloud resources will be made available to you.

This hands-on lab lets you do the lab activities yourself in a real cloud environment, not in a simulation or demo environment. It does so by giving you new, temporary credentials that you use to sign in and access Google Cloud for the duration of the lab.

To complete this lab, you need:

- Access to a standard internet browser (Chrome browser recommended).

Note: Use an Incognito or private browser window to run this lab. This prevents any conflicts between your personal account and the Student account, which may cause extra charges incurred to your personal account.

- Time to complete the lab---remember, once you start, you cannot pause a lab.

Note: If you already have your own personal Google Cloud account or project, do not use it for this lab to avoid extra charges to your account.

How to start your lab and sign in to the Google Cloud Console

1. Click the **Start Lab** button. If you need to pay for the lab, a pop-up opens for you to select your payment method. On the left is the **Lab Details** panel with the following:
 - The **Open Google Console** button
 - Time remaining
 - The temporary credentials that you must use for this lab
 - Other information, if needed, to step through this lab

2. Click **Open Google Console**. The lab spins up resources, and then opens another tab that shows the **Sign in** page.

Tip: Arrange the tabs in separate windows, side-by-side.

Note: If you see the **Choose an account** dialog, click **Use Another Account**.

3. If necessary, copy the **Username** from the **Lab Details** panel and paste it into the **Sign in** dialog. Click **Next**.

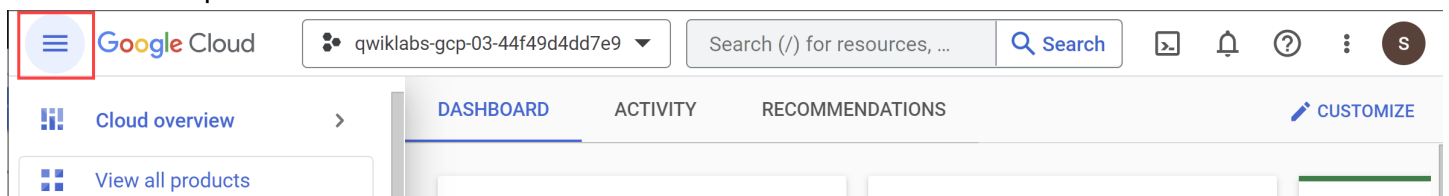
4. Copy the **Password** from the **Lab Details** panel and paste it into the **Welcome** dialog. Click **Next**.

Important: You must use the credentials from the left panel. Do not use your Google Cloud Skills Boost credentials. **Note:** Using your own Google Cloud account for this lab may incur extra charges.

5. Click through the subsequent pages:
 - Accept the terms and conditions.
 - Do not add recovery options or two-factor authentication (because this is a temporary account).
 - Do not sign up for free trials.


After a few moments, the Cloud Console opens in this tab.

Note: You can view the menu with a list of Google Cloud Products and Services by clicking the **Navigation menu** at the top-left.



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**. The output contains a line that declares the **PROJECT_ID** for this session:

Your Cloud Platform project in this session is set to YOUR_PROJECT_ID

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
```

3. Click **Authorize**.

4. Your output should now look like this:

Output:

```
ACTIVE: * ACCOUNT: student-01-xxxxxxxxxxxx@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
```

Output:

```
[core] project = <project_ID>
```

Example output:

```
[core] project = qwiklabs-gcp-44776a13dea667a6 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 {{{project_0.default_region | "REGION"}}
```

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 project resources

1. In **Cloud Shell**, create variables for your bucket, project, and region:

```
PROJECT_ID=$(gcloud config get-value project) BUCKET_NAME="${PROJECT_ID}-bucket"
TOPIC_ID=my-id REGION=us-central1 AE_REGION=us-central
```

Note: Cloud Storage bucket names must be globally unique. Your Qwiklabs Project ID is always unique, so that is used in your bucket name in this lab.

2. Create a Cloud Storage bucket owned by this project:

```
gsutil mb gs://$BUCKET_NAME
```

3. Create a Pub/Sub topic in this project:

```
gcloud pubsub topics create $TOPIC_ID
```

4. Create an App Engine app for your project:

```
gcloud app create --region=$AE_REGION
```

5. Create a Cloud Scheduler job in this project. The job publishes a message to a Pub/Sub topic at one-minute intervals:

```
gcloud scheduler jobs create pubsub publisher-job --schedule="* * * * *" --topic=$TOPIC_ID --message-body="Hello!"
```

6. If prompted to enable the Cloud Scheduler API, press **y** and enter.

Click *Check my progress* to verify the objective. Create Project Resources

7. Start the job:

```
gcloud scheduler jobs run publisher-job
```

8. Use the following commands to clone the quickstart repository and navigate to the sample code directory:

```
git clone https://github.com/GoogleCloudPlatform/java-docs-samples.git cd java-docs-samples/pubsub/streaming-analytics
```

Note: Execute the python commands individually.

Click *Check my progress* to verify the objective. Start the cloud scheduler job

Task 2. Review code to stream messages from Pub/Sub to Cloud Storage

Code sample

Review the following sample code, which uses Dataflow to:

- Read Pub/Sub messages.
- Window (or group) messages into fixed-size intervals by publish timestamps.
- Write the messages in each window to files in Cloud Storage.

```
import java.io.IOException; import org.apache.beam.examples.common.WriteOneFilePerWindow; import
org.apache.beam.sdk.Pipeline; import org.apache.beam.sdk.io.gcp.pubsub.PubsubIO; import
org.apache.beam.sdk.options.Default; import org.apache.beam.sdk.options.Description; import
org.apache.beam.sdk.options.PipelineOptions; import
org.apache.beam.sdk.options.PipelineOptionsFactory; import
org.apache.beam.sdk.options.StreamingOptions; import org.apache.beam.sdk.options.Validation.Required;
import org.apache.beam.sdk.transforms.windowing.FixedWindows; import
org.apache.beam.sdk.transforms.windowing.Window; import org.joda.time.Duration; public class
PubSubToGcs { * Define your own configuration options. Add your own arguments to be processed * by the
command-line parser, and specify default values for them. public interface PubSubToGcsOptions extends
PipelineOptions, StreamingOptions { @Description("The Cloud Pub/Sub topic to read from.") @Required
String getInputTopic(); void setInputTopic(String value); @Description("Output file's window size in number
of minutes.") @Default.Integer(1) Integer getWindowSize(); void setWindowSize(Integer value);
@Description("Path of the output file including its filename prefix.") @Required String getOutput(); void
setOutput(String value); } public static void main(String[] args) throws IOException { // The maximum
number of shards when writing output. int numShards = 1; PubSubToGcsOptions options =
PipelineOptionsFactory.fromArgs(args).withValidation().as(PubSubToGcsOptions.class);
options.setStreaming(true); Pipeline pipeline = Pipeline.create(options); pipeline // 1) Read string messages
from a Pub/Sub topic. .apply("Read PubSub Messages",
PubsubIO.readStrings().fromTopic(options.getInputTopic())) // 2) Group the messages into fixed-sized
minute intervals.
.apply(Window.into(FixedWindows.of(Duration.standardMinutes(options.getWindowSize())))) // 3) Write one
file to GCS for every window of messages. .apply("Write Files to GCS", new
WriteOneFilePerWindow(options.getOutput(), numShards)); // Execute the pipeline and wait until it finishes
running. pipeline.run().waitUntilFinish(); } } Note: To explore the sample code further, visit the respective
java-docs-samples and python-docs-samples GitHub pages.
```

Task 3. Start the pipeline

1. To start the pipeline, run the following command:

```
mvn compile exec:java \ -Dexec.mainClass=com.examples.pubsub.streaming.PubSubToGcs \ -
Dexec.cleanupDaemonThreads=false \ -Dexec.args=" \ --project=$PROJECT_ID \ --region=$REGION \ --
inputTopic=projects/$PROJECT_ID/topics/$TOPIC_ID \ --output=gs://$BUCKET_NAME/samples/output \ --
runner=DataflowRunner \ --windowSize=2" Note: When executing the python command, replace
project_id, bucket_name, and region with your project id, bucket name, and assigned lab region.
```

The preceding command runs locally and launches a Dataflow job that runs in the cloud.

2. When the command returns **Workers have started successfully**, exit the local program using Ctrl+C.

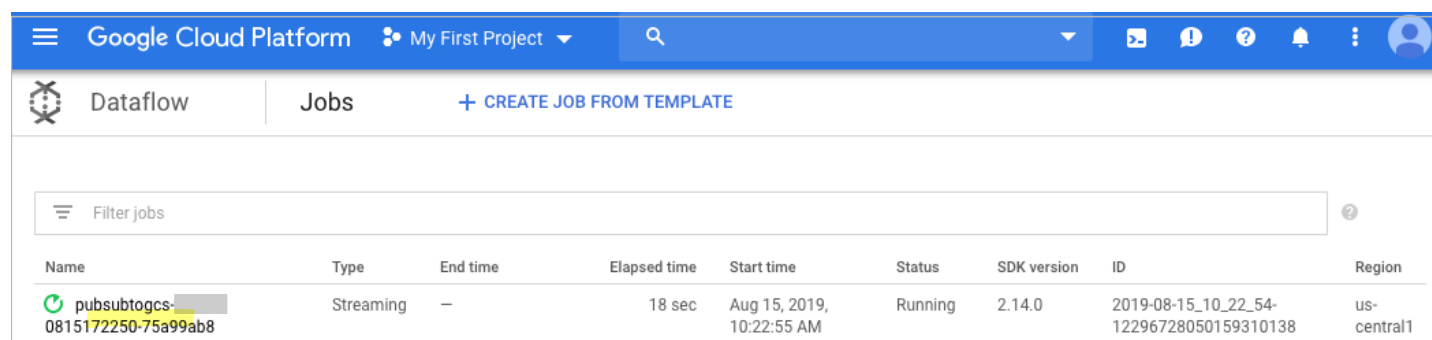
Note: To exit your Python development environment, type and enter exit.

Click *Check my progress* to verify the objective. Start the pipeline and launch dataflow job


Task 4. Observe job and pipeline progress

1. You can observe the job's progress in the Dataflow console.

[Go to the Dataflow console.](#)



The screenshot shows the Google Cloud Platform Dataflow console. The top navigation bar includes the Google Cloud Platform logo, the project name 'My First Project', a search bar, and various utility icons. Below the navigation bar, the 'Dataflow' section is active, showing a 'Jobs' tab and a '+ CREATE JOB FROM TEMPLATE' button. A 'Filter jobs' input field is present. The main table lists the job details:

Name	Type	End time	Elapsed time	Start time	Status	SDK version	ID	Region
 pubsubtogcs-0815172250-75a99ab8	Streaming	—	18 sec	Aug 15, 2019, 10:22:55 AM	Running	2.14.0	2019-08-15_10_22_54-12296728050159310138	us-central1

2. Open the job details view to see:

- Job structure
- Job logs
- Stage metrics

Google Cloud Platform My First Project

Job details LOGS

Job

Job summary

System latency (seconds) Aug 15, 2019 10:22 AM

Data freshness (seconds) Aug 15, 2019 10:22 AM

No data for this time interval

Create alerting

Read PubSub Messages Running

Window.Into() Running

Write Files to GCS Running

Job name pubsubtogcs-0815172250-75a99ab8

Job ID 2019-08-15-10-22-54-12296728050159310138

Region us-central1

Job status Running

Stop job

SDK version Apache Beam SDK for Java 2.14.0

Job type Streaming

Start time Aug 15, 2019, 10:22:55 AM

Elapsed time 1 min 25 sec

Encryption type Google-managed key

Autoscaling

Workers 1

Current state Worker pool started.

You may have to wait a few minutes to see the output files in Cloud Storage.

- You can see the output files by navigating to **Navigation menu > Cloud Storage**. Click on your bucket name and then click **Samples**:

Cloud Storage

Bucket details

qwiklabs-gcp-00-13966fee4d3b

OBJECTS CONFIGURATION PERMISSIONS RETENTION LIFECYCLE

Buckets > qwiklabs-gcp-00-13966fee4d3b > samples

UPLOAD FILES UPLOAD FOLDER CREATE FOLDER MANAGE HOLDS DOWNLOAD DELETE

Filter by name prefix only Filter Filter objects and folders

Name	Size	Type	Created time	Storage class	Last modified	Public access	Encryption	Retention
output-10-08-10-10-0	68 B	application/octet-stream	Sep 8, 2021, 3...	Standard	Sep 8, 202...	Not public	Google-managed key	—
output-10-10-10-10-2-0	34 B	application/octet-stream	Sep 8, 2021, 3...	Standard	Sep 8, 202...	Not public	Google-managed key	—
output-10-10-10-10-12-1	34 B	application/octet-stream	Sep 8, 2021, 3...	Standard	Sep 8, 202...	Not public	Google-managed key	—
output-10-12-10-10-14-0	68 B	application/octet-stream	Sep 8, 2021, 3...	Standard	Sep 8, 202...	Not public	Google-managed key	—
output-10-14-10-10-16-0	68 B	application/octet-stream	Sep 8, 2021, 3...	Standard	Sep 8, 202...	Not public	Google-managed key	—
output-10-16-10-10-18-0	34 B	application/octet-stream	Sep 8, 2021, 3...	Standard	Sep 8, 202...	Not public	Google-managed key	—
output-10-16-10-10-18-1	34 B	application/octet-stream	Sep 8, 2021, 3...	Standard	Sep 8, 202...	Not public	Google-managed key	—
output-10-18-10-20-1	68 B	application/octet-stream	Sep 8, 2021, 3...	Standard	Sep 8, 202...	Not public	Google-managed key	—

- Alternatively, use the command line below to check which files have been written out:

```
gsutil ls gs://{BUCKET_NAME}/samples/
```

The output should look like the following:

```
gs://{BUCKET_NAME}/samples/output-22:30-22:32-0-of-1 gs://{BUCKET_NAME}/samples/output-22:32-22:34-0-of-1 gs://{BUCKET_NAME}/samples/output-22:34-22:36-0-of-1
```

gs://{BUCKET_NAME}/samples/output-22:36-22:38-0-of-1

Task 5. Cleanup

1. Delete the Cloud Scheduler job:

```
gcloud scheduler jobs delete publisher-job
```

2. If prompted Do you want to continue press Y and enter.
3. In the Dataflow console, stop the job.
4. With your job selected from the Dataflow Console, press the **Stop** button.
5. Click **Stop Job > Cancel** to cancel the pipeline without draining.
6. Delete the topic:

```
gcloud pubsub topics delete $TOPIC_ID
```

7. Delete the files created by the pipeline:

```
gsutil -m rm -rf "gs://{BUCKET_NAME}/samples/output*" gsutil -m rm -rf "gs://{BUCKET_NAME}/temp/"
```

8. Remove the Cloud Storage bucket:

```
gsutil rb gs://{BUCKET_NAME}
```

Congratulations!

You created a Dataflow pipeline which read messages from your Pub/Sub topic, windowed them by timestamp, and wrote them to your cloud storage bucket.

Next step / learn more

- If you would like to window Pub/Sub messages by a custom timestamp, you can specify the timestamp as an attribute in the Pub/Sub message, and then use the custom timestamp with [PubsubIO's withTimestampAttribute](#).
- Learn about the [Beam programming model](#) used to write streaming and batch Dataflow jobs
- Check out these example Beam pipelines for streaming data into BigQuery: [Java](#), [Python](#).
- Take a look at Google's [open-source Dataflow templates designed for streaming](#).

Google Cloud training and certification

...helps you make the most of Google Cloud technologies. [Our classes](#) include technical skills and best practices to help you get up to speed quickly and continue your learning journey. We offer fundamental to advanced level training, with on-demand, live, and virtual options to suit your busy schedule. [Certifications](#) help you validate and prove your skill and expertise in Google Cloud technologies.

Manual Last Updated May 4, 2023

Lab Last Tested May 4, 2023

Copyright 2023 Google LLC All rights reserved. Google and the Google logo are trademarks of Google LLC. All other company and product names may be trademarks of the respective companies with which they are associated.