Rate and review

After you create and stage your Dataflow template, run the template with the Google Cloud Console, REST API, or gcloud command-line tool. You can deploy Dataflow template jobs from many environments, including App Engine standard environment, Cloud Functions, an other constrained environments.

Note: In addition to the template file, running templated pipeline relies on files that we staged and referenced at the time of template creation. If you move or remove the sta files, your pipeline job will fail.

Using the Cloud Console

You can use the Cloud Console to run Google-provided and custom Dataflow templates.

Google-provided templates

To run a Google-provided template:

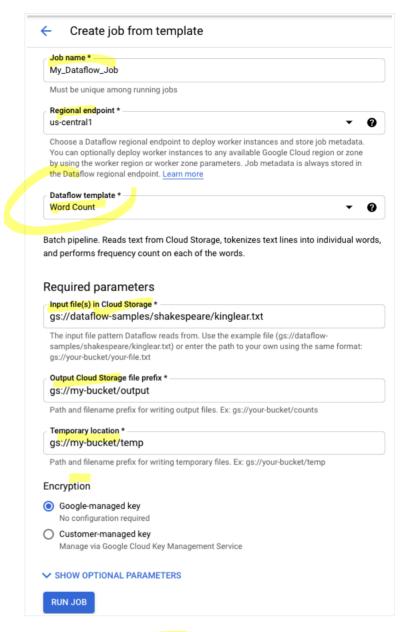
1. Go to the Dataflow page in the Cloud Console.

Go to the Dataflow page

2. Click + CREATE JOB FROM TEMPLATE.



3. Select the Google-provided template that you want to run from the **Dataflow template** drop-down menu.



- 4. Enter a job name in the **Job Name** field.
- 5. Enter your parameter values in the provided parameter fields. You should not need the **Additional Parameters** section when you use a Google-provided template.
- 6. Click Run Job.

Custom templates

To run a custom template:

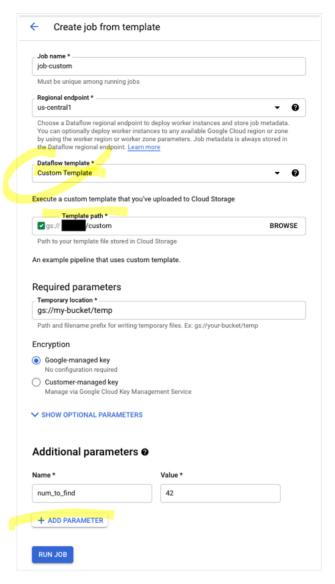
1. Go to the Dataflow page in the Cloud Console.

Go to the Dataflow page

2. Click CREATE JOB FROM TEMPLATE.



3. Select Custom Template from the Dataflow template drop-down menu.



- 4. Enter a job name in the Job Name field.
- Enter the Cloud Storage path to your template file in the template Cloud Storage path field.
- 6. If your template needs parameters, click on + ADD PARAMETER in the Additional parameters section. Enter in the Name and Value of the parameter. Repeat this step f each needed parameter.
- 7. Click Run Job.

Using the REST API

To run a template with a REST API request, send an HTTP POST request with your project II This request requires authorization.

See the REST API reference for projects templates launch to learn more about the available parameters.

Example 1: Creating a custom template batch job

This example projects.templates.launch request creates a batch job from a template that reads a text file and writes an output text file. If the request is successful, the response bod contains an instance of LaunchTemplateResponse.

You must modify the following values:

- Replace YOUR_PROJECT_ID with your project ID.
- Replace LOCATION with the Dataflow regional endpoint of your choice.
- Replace JOB_NAME with a job name of your choice.
- Replace YOUR_BUCKET_NAME with the name of your Cloud Storage bucket.
- Set gcsPath to the Cloud Storage location of the template file.
- Set parameters to your list of key/value pairs.
- Set tempLocation to a location where you have write permission. This value is requite to run Google-provided templates.

```
POST https://dataflow.googleapis.com/v1b3/projects/YOUR_PROJECT_ID/loc
{
    "jobName": "JOB_NAME",
    "parameters": {
        "inputFile": "gs://YOUR_BUCKET_NAME/input/my_input.txt",
        "outputFile": "gs://YOUR_BUCKET_NAME/output/my_output"
},
    "environment": {
        "tempLocation": "gs://YOUR_BUCKET_NAME/temp",
        "zone": "us-central1-f"
}
```

Example 2: Creating a custom template streaming job

This example projects.templates.launch request creates a streaming job from a template the reads from a Pub/Sub topic and writes to a BigQuery table. The BigQuery table must alread exist with the appropriate schema. If successful, the response body contains an instance of LaunchTemplateResponse.

You must modify the following values:

- Replace YOUR_PROJECT_ID with your project ID.
- Replace LOCATION with the Dataflow regional endpoint of your choice.
- Replace JOB_NAME with a job name of your choice.
- Replace YOUR_BUCKET_NAME with the name of your Cloud Storage bucket.
- Replace YOUR_TOPIC_NAME with your Pub/Sub topic name.
- Replace YOUR_DATASET with your BigQuery dataset, and replace YOUR_TABLE_NAME with your BigQuery table name.
- Set gcsPath to the Cloud Storage location of the template file.
- Set parameters to your list of key/value pairs.
- Set tempLocation to a location where you have write permission. This value is requite to run Google-provided templates.

```
POST https://dataflow.googleapis.com/v1b3/projects/YOUR_PROJECT_ID/loc
{
    "jobName": "JOB_NAME",
    "parameters": {
        "topic": "projects/YOUR_PROJECT_ID/topics/YOUR_TOPIC_NAME",
        "table": "YOUR_PROJECT_ID: YOUR_DATASET. YOUR_TABLE_NAME"
},
    "environment": {
        "tempLocation": "gs://YOUR_BUCKET_NAME/temp",
        "zone": "us-central1-f"
}
```

Example 3: Updating a custom template streaming job

This example projects.templates.launch request shows you how to update a template streaming job.

- 1. Run Example 2: Creating a custom template streaming job to start a streaming templation.
- 2. Send the following HTTP POST request, with the following modified values:
 - Replace YOUR_PROJECT_ID with your project ID.
 - Replace LOCATION with the Dataflow regional endpoint of your choice.
 - Replace JOB_NAME with a job name of your choice.
 - Replace YOUR_BUCKET_NAME with the name of your Cloud Storage bucket.
 - Replace YOUR_TOPIC_NAME with your Pub/Sub topic name.
 - Replace YOUR_DATASET with your BigQuery dataset, and replace YOUR_TABLE_NAME with your BigQuery table name.
 - Set gcsPath to the Cloud Storage location of the template file.
 - Set parameters to your list of key/value pairs.
 - Set tempLocation to a location where you have write permission. This value is required to run Google-provided templates.

```
POST https://dataflow.googleapis.com/v1b3/projects/YOUR_PROJECT_1
{
    "jobName": "JOB_NAME",
    "parameters": {
        "topic": "projects/YOUR_PROJECT_ID/topics/YOUR_TOPIC_NAME
        "table": "YOUR_PROJECT_ID:YOUR_DATASET.YOUR_TABLE_NAME"
    },
    "environment": {
        "tempLocation": "gs://YOUR_BUCKET_NAME/temp",
        "zone": "us-central1-f"
    }
    "update": true
}
```

3. Access the Dataflow monitoring interface and verify that a new job with the same nar was created. This job has the status **Updated**.

Using the Google API Client Libraries

Consider using the Google API Client Libraries to easily make calls to the Dataflow REST AF This sample script uses the Google API Client Library for Python.

In this example, you must set the following variables:

- project: Set to your project ID.
- job: Set to a unique job name of your choice.
- template: Set to the Cloud Storage location of the template file.
- parameters: Set to a dictionary with the template parameters.

dataflow/run_template/main.py

View on GitHub

Feedback

```
# project = 'your-gcp-project'
# job = 'unique-job-name
# template = 'gs://dataflow-templates/latest/Word_Count'
# parameters = {
     'inputFile': 'gs://dataflow-samples/shakespeare/kinglear.txt',
     'output': 'gs://<your-gcs-bucket>/wordcount/outputs',
# }
dataflow = build('dataflow', 'v1b3')
request = dataflow.projects().templates().launch(
    projectId=project,
    gcsPath=template,
    body={
        'jobName': job,
        'parameters': parameters,
    }
)
response = request.execute()
```

For more information on the available options, see the projects.templates.launch met in the Cloud Dataflow REST API reference.

Using gcloud

Note: To use the gcloud command-line tool to run templates, you must have Clouc SDK version 138.0.0 or higher.

The gcloud command-line tool can run either a custom or a Google-provided template usi the gcloud dataflow jobs run command. Examples of running Google-provided template are documented in the Google-provided templates page.

For the following custom template examples, set the following values:

- Replace JOB_NAME with a job name of your choice.
- Replace YOUR_BUCKET_NAME with the name of your Cloud Storage bucket.
- You must include the --gcs-location flag. Set --gcs-location to the Cloud Storal location of the template file.
- Set --parameters to the comma-separated list of parameters to pass to the job.
 Spaces between commas and values are not allowed.

Example 1: Custom template, batch job

This example creates a batch job from a template that reads a text file and writes an output text file.

```
gcloud dataflow jobs run JOB_NAME \
    --gcs-location gs://YOUR_BUCKET_NAME/templates/MyTemplate \
    --parameters inputFile=gs://YOUR_BUCKET_NAME/input/my_input.txt,ou
```

The request returns a response with the following format.

```
id: 2016-10-11_17_10_59-1234530157620696789
projectId: YOUR_PROJECT_ID
type: JOB_TYPE_BATCH
```

Example 2: Custom template, streaming job

This example creates a streaming job from a template that reads from a Pub/Sub topic and writes to a BigQuery table. The BigQuery table must already exist with the appropriate sche

```
gcloud dataflow jobs run \textit{JOB\_NAME} \setminus --gcs-location gs://\textit{YOUR\_BUCKET\_NAME}/templates/MyTemplate \ --parameters topic=projects/project-identifier/topics/resource-nam
```

The request returns a response with the following format.

```
id: 2016-10-11_17_10_59-1234530157620696789
projectId: YOUR_PROJECT_ID
type: JOB_TYPE_STREAMING
```

For a complete list of flags for the gcloud dataflow jobs run command, see the gcloud tool reference.

Monitoring and Troubleshooting

The Dataflow Monitoring Interface allows you to monitor your Dataflow jobs. If a job fails, y can find troubleshooting tips, debugging strategies, and a catalog of common errors in the Troubleshooting Your Pipeline guide.

