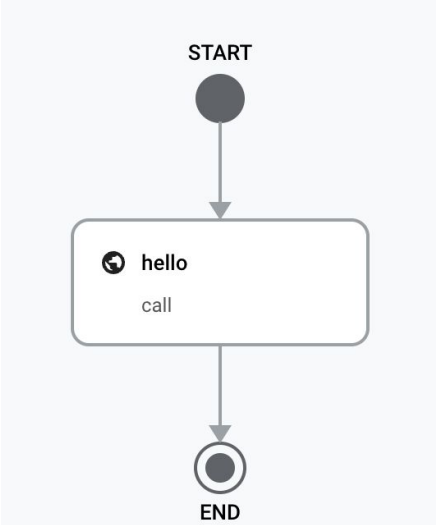


GCP Workflows

Integrate services across Google Cloud and external systems,
streamline data processing and transform your business operations.

WG

About me

William Gomez

Web Developer & Data Engineer at **Huge**

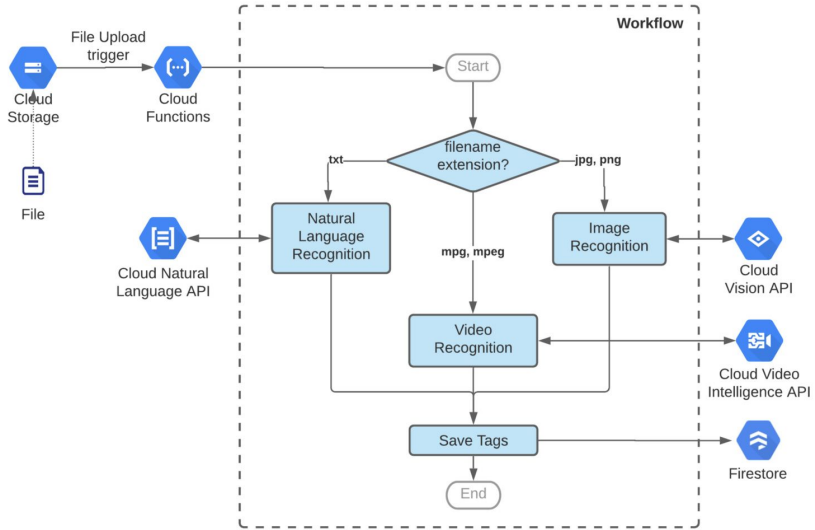
Full stack wannabe

Python Medellin meetups co-organizer

Volleyball and soccer lover

Learning about **fitness & cooking**

Introduction



Introduction

Google Cloud Workflows: An Overview

Google Cloud Workflows is a serverless workflow orchestration service that allows you to automate your business processes, data processing pipelines, and microservices orchestration across Google Cloud and third-party services.

In this presentation, we will cover the basics of Google Cloud Workflows, its philosophy, and some interesting use cases.

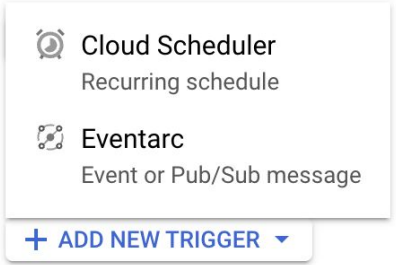
What is Google Cloud Workflows?

Defining Google Cloud Workflows

1 Workflows can be triggered by events or scheduled to run at a specific time or interval.

2 Workflows are composed of a set of tasks that can be executed sequentially or in parallel. Each task consists of a set of steps that can interact with Google Cloud services and third-party services.

3 YAML format provides a programmatic way to define workflows as code.




```yaml
main:
    params: [input]
    steps:
    - hello:
        call: sys.log
        args:
          text: "Hello world"
          severity: INFO
```

Workflow Hello World

Workflow Hello World + Cloud Scheduler

1 Enable API

Enable the Cloud Scheduler API

Triggering recurring, scheduled executions of your workflow requires Cloud Scheduler. Please enable the API to continue.

2 Define Cron Frequency

Define the schedule

Name *
each-day-6-pm

Must be unique across the jobs in the same region

Region *
us-central1 (Iowa)

Frequency *
0 18 * * *

Schedules are specified using unix-cron format. E.g. every minute: "* * * * *", every 3 hours: "0 */3 * * *", every Monday at 9:00: "0 9 * * 1". Learn more

- Minute and Hour:
  - At 6:00 PM

Timezone *
Colombia Standard Time (COT)

Jobs in set in timezones affected by Daylight Saving Time can run outside of cadence during DST change. Using a UTC timezone can avoid the problem. Learn more

CONTINUE

3 Add message and Service Account

Configure the execution

Workflow's argument

Press Alt+F1 for Accessibility Options.

```
1  {
2      "message": "hello"
3  }
```

You can pass an optional JSON object as input to your workflow. Learn more

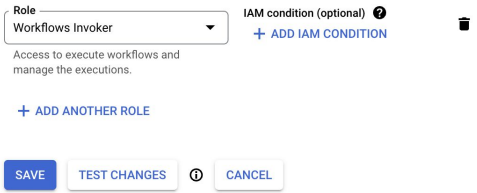
Execution's call log level *
Not specified

Service account *
sa-hello-world-workflow

This service account must have the Workflows Invoker role to schedule a workflow. Learn more

Workflow Hello World + Cloud Scheduler

4 Add SA permissions

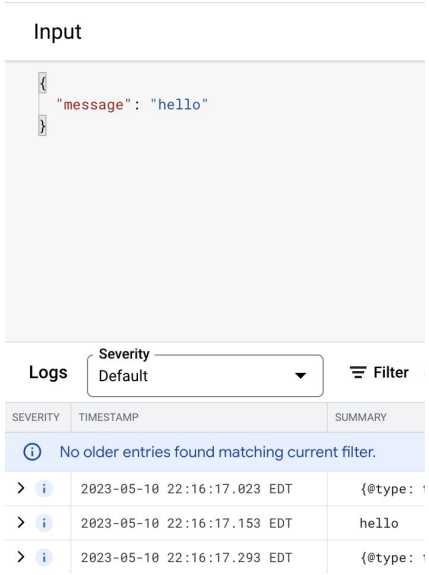


https://cloud.google.com/workflows/docs/access-control

5 Use the input

```
main:
    params: [input]
    steps:
    - hello:
        call: sys.log
        args:
          text: ${input.message}
          severity: INFO
```

6 Check the execution



Let’s review the Syntax

Workflow Arguments

RUNTIME ARGUMENTS

```
main:
  params: [args]
  steps:
    - read_runtime_args:
        assign:
          - arg1: ${args.arg1}
          - arg2: ${args.arg2}
```

https://cloud.google.com/workflows/docs/reference/syntax/syntax-cheat-sheet

Data types

VARIABLES AND DATA TYPES

```
- data_types:
    assign:
      - my_integer: 1      # 64 bit, signed
      - my_double: 4.1     # 64 bit, signed floating point number
      - my_string: "hello" # unicode <= 256 KB length
      - my_boolean: true   # true/false, True/False, TRUE/FALSE
      - my_null: null
      - my_list: ["zero","one","two"]
      - my_map:
          name: Lila
          last_name: Barton
          birthYear: 1990
- conversion_functions:
    assign:
      - to_double: double("2.7") # string, integer to double
      - to_int: int(2.7)         # string, double to integer
      - to_string: string(1.7)   # int, double, boolean to string
```

`get_type(arg)`

Returns a string indicating the data type of **`arg`**: one of **`boolean`**, **`bytes`**, **`double`**, **`integer`**, **`list`**, **`map`**, **`string`**, or **`null`**.

If the operand is a function or a subworkflow, a **`TypeError`** is raised.

https://cloud.google.com/workflows/docs/reference/syntax/syntax-cheat-sheet
https://cloud.google.com/workflows/docs/reference/stdlib/expression-helpers

It is not python!! But it could be close to it.

LISTS

```
- list_ops:
    assign:
      - my_list: ["zero","one","two"]
      - my_list_len: ${len(my_list)}
      - key_exists: ${"Key1" in my_list}
      - my_list[0]: 0
      - idx: 0
      - my_list[idx + 1]: 1
      - my_list[my_list_len - 1]: 2
      - my_list: ${list.concat(my_list, 3)}
      - my_multi_dimen_list: [[10, 11, 12], [20, 21, 22]]
      - my_multi_dimen_list[0][1]: "Value11"
```

MAPS

```
- map_ops:
    assign:
      - my_map: {"Key1": "hello"}
      - map_len: ${len(my_map)}
      - key_exists: ${"Key1" in my_map}
      - key_list: ${keys(my_map)}
      - key_is_null: ${default(map.get(my_map, "Key1"), "Couldn't find key!")}
      - key_with_special_char: '${"foo" + var.key["special!key"]}'
      - key_str: "Key"
      - my_map.Key1: "Value1"
      - my_map["Key2"]: "Value2"
      - my_map[key_str + "3"]: "Value3"
      - my_nested_map: {"NestedMapKey": {"Key1":"Value1"}}
      - my_nested_map.NestedMapKey.Key2: "Value2"
```

```
default(val, defaultVal)
```

https://cloud.google.com/workflows/docs/reference/syntax/syntax-cheat-sheet

Flow

CONTROLLING FLOW

```
- step_with_next:
    assign:
      - foo: "bar"
    next: step_with_nested_steps
- step_with_end:
    assign:
       - foo: "bar"
    next: end
- step_with_nested_steps:
```

ITERATION

```
- for-in-list:
    steps:
      - assignList:
          assign:
            - list: [1, 2, 3, 4, 5]
            - sum: 0
      - loopList:
          for:
            value: v
            in: ${list}
            steps:
              - sumList:
                  assign:
                      - sum: ${sum + v}
```

CONDITIONS

```
- switch_basic:
    switch:
      - condition: ${my_integer < 10}
        next: switch_embedded_steps
- switch_embedded_steps:
    switch:
      - condition: ${my_integer < 10}
        steps:
          - stepA:
              assign:
                - foo: "bar"
          - stepB:
              assign:
                - foo: "bar"
```

```
if(condition, ifTrue, ifFalse)
```

https://cloud.google.com/workflows/docs/reference/syntax/syntax-cheat-sheet

Parallel

PARALLEL ITERATION

```
- parallel_loop:
    parallel:
      shared: [total]
      for:
        value: postId
        in: ${args.posts}
        steps:
          - getPostCommentCount:
              call: http.get STANDARD LIBRARY
              args:
                url: ${"https://example.com/postComments/" + postId}
              result: numComments
          - add:
              assign:
                - total: ${total + numComments}
```

PARALLEL BRANCHES

```
- parallel_branches:
    parallel:
      shared: [user, notification]
      branches:
        - getUser:
            steps:
              - getUserCall:
                  call: http.get STANDARD LIBRARY
                  args:
                    url: ${"https://example.com/users/" + args.userId}
                  result: user
        - getNotification:
            steps:
              - getNotificationCall:
                  call: http.get
                  args:
                    url: ${"https://example.com/notification/" + args.notificationId}
                  result: notification
```

https://cloud.google.com/workflows/docs/reference/syntax/syntax-cheat-sheet

Subworkflows

SUBWORKFLOWS

```
- call_subworkflow:
    call: subworkflow_name_message
    args:
      first_name: "Ada"
      last_name: "Lovelace"
    result: output
- call_subworkflow2:
    assign:
      - output2: ${subworkflow_name_message("Sherlock","Holmes")}

subworkflow_name_message:
  params: [first_name, last_name, country: "England"]
  steps:
    - prepareMessage:
        return: ${"Hello " + first_name + " " + last_name + " from " + country + "."}
```

https://cloud.google.com/workflows/docs/reference/syntax/syntax-cheat-sheet

Raise and catch errors

RAISE ERRORS

```
- raise_custom_string_error:
    raise: "Something went wrong."
- raise_custom_map_error:
    raise:
      code: 55
      message: "Something went wrong."
```

CATCH ERRORS

```
- try_retry_except:
    try:
      steps: # steps is only needed if multiple steps
        - step_a:
            call: http.get
            args:
              url: https://host.com/api
            result: api_response1
    except:
      as: e
      steps:
        - known_errors:
            switch:
            - condition: ${not("HttpError" in e.tags)}
              return: "Connection problem."
            - condition: ${e.code == 404}
              return: "Sorry, URL wasn't found."
            - condition: ${e.code == 403}
              return: "Authentication error."
        - unhandled_exception:
            raise: ${e}
```

https://cloud.google.com/workflows/docs/reference/syntax/syntax-cheat-sheet

Workflows Pricing

Workflows pricing

INTERNAL STEPS	PRICE PER MONTH
First 5,000 steps	Free
Steps 5,000 to 100,000,000	$0.01 per increment of 1,000 steps

EXTERNAL HTTP CALLS	PRICE PER MONTH
First 2,000 calls	Free
Steps 2,000 to 100,000,000	$0.025 per increment of 1,000 calls

Google Cloud services use
`*.appspot.com`,
`*.cloud.goog`,
`*.cloudfunctions.net`, or
`*.run.app`

External http calls:
`http.get`,
`http.post`,
`http.request`

Optimize usage

- Combine assignments into one step.
- Avoid excessive use of sys.log steps. Consider using call logging instead.

- When using connectors that wait for long-running operations, set a custom polling policy that optimizes latency for cost. https://cloud.google.com/workflows/docs/reference/googleapis#invoke_a_connector_call
  If you expect an operation to take over an hour, you might want a policy that initially polls after one minute in case of an immediate failure, and then every 15 minutes after that.

https://cloud.google.com/workflows/pricing#optimize-usage

Don’t overuse Workflows

What business logic should NOT go inside a Workflow

- Iterate a list that can grow with the time over a API call. (This will reach sooner or later the 100k steps).
- Start saving the results of different processes in variables. (This will reach easily 512KB memory limit).
- Overuse APIs with complicated schemas, like BigQuery and Firestore.
- Play with dates or list transformations (List concatenations is not possible).
- Create always resources from connectors (First always try to create resources with terraform)

Workflows limits

1. Maximum steps: 100k
2. Memory limit for all variables in a single execution is 512KB
3. The maximum size of an HTTP response is 2MB (if saved to a variable, the memory limit for variables applies)

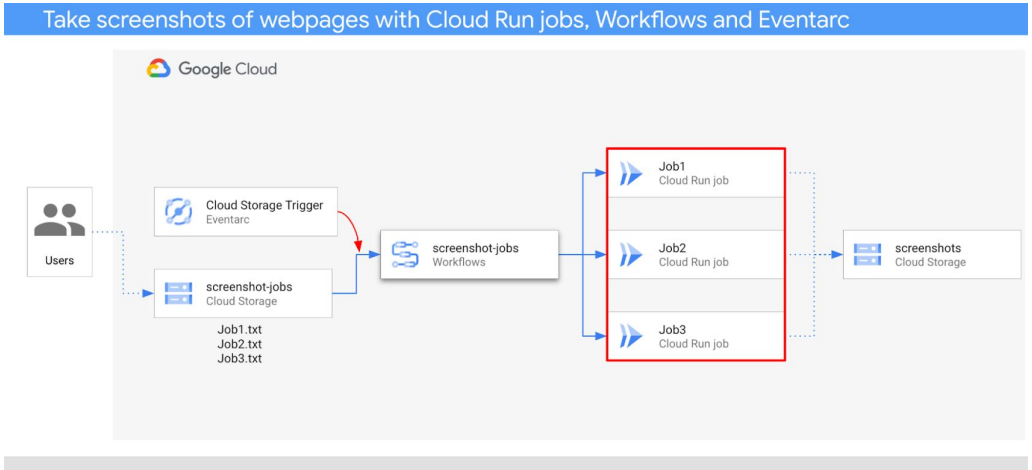
Iterate a list that can grow with the time over a API call. (This will reach sooner or later the 100k steps)

Solution:

- Use Cloud Storage and Cloud Run to process batch requests.

Eventarc is a serverless event ingestion service offered by Google Cloud Platform (GCP).

When an Eventarc-enabled Google Cloud service generates an event, it publishes the event to Eventarc, which then delivers the event to the appropriate target(s) based on the event's configuration.



Start saving the results of different processes in variables. (This will reach easily 512KB memory limit).

Solution:

- Everything should be saved in the storage, is cheap and all services can send info to it. For example, BigQuery can export the data with just EXPORT DATA:

```
EXPORT DATA
 OPTIONS(
 uri='s3://bucket/folder/*',
 format='JSON',
 overwrite=true
) AS
SELECT field1, field2 FROM mydataset.table1
```

Overuse APIs with complicated schemas, like BigQuery and Firestore.

```
response:
"{"cacheHit":false,"jobComplete":true,"jobReference":{"jobId":"job_-
ZfQpAbEpKXQjrFYr0GhN7I7Olmw","location":"US","projectId":"personalpage-
270523"},"kind":"bigquery#queryResponse","rows":[{"f":[{"v":"2023-05-11"}]}],"schema":{"fields":
[{"mode":"NULLABLE","name":"f0_","type":"STRING"}]},"totalBytesProcessed":"0","totalRows":"1"}"
```

Solution:

- Export data options as in previous slide.
- Firebase for example has: googleapis.firestore.v1.projects.databases.exportDocuments (It is to be honest a weird format BUT you can upload it to BigQuery easily, and then export it as JSON to the storage)

Overuse APIs with complicated schemas, like BigQuery and Firestore.

```
- exportDocuments:
    call: googleapis.firestore.v1.projects.databases.exportDocuments
    args:
      name: $${parent_resource}
      body:
        outputUriPrefix: $${export_path}
        collectionIds:
          - "queries"
    result: exportDocumentsResult
```

```
- bq_path: $${"gs://bucket/all_namespaces/kind_queries/all_namespaces_kind_queries.export_metadata"}
```

```
- load_files:
    call: googleapis.bigquery.v2.jobs.insert
    args:
      projectId: ${project_id}
      body:
        configuration:
          load:
            destinationTable:
              projectId: $${project_id}
              datasetId: $${datasetId}
              tableId: ${firestore_queries_table}
            sourceFormat: DATASTORE_BACKUP
            sourceUris:
              - $${bq_path}
            writeDisposition: WRITE_TRUNCATE # New table
```

https://stackoverflow.com/questions/74576263/get-a-firestore-document-from-a-workflow-as-json-value-only-without-the-metadat/76020136#76020136

Play with dates or list transformations (List concatenations is not possible).

```
DECLARE start_day DATE DEFAULT '{START_DATE}';
DECLARE end_day DATE DEFAULT '{END_DATE}';

WITH RECURSIVE dates as (
  SELECT
    start_date as start_date,
    DATE_ADD(start_date, INTERVAL {INTERVAL}) as end_date,
  FROM UNNEST(GENERATE_DATE_ARRAY(start_day, end_day, INTERVAL {INTERVAL})) AS start_date
  ORDER BY start_date
)
SELECT
    FORMAT_DATETIME('{FORMAT}', start_date) AS start_date,
    CASE
      WHEN end_date < end_day THEN FORMAT_DATETIME('{FORMAT}', end_date)
      ELSE FORMAT_DATETIME('{FORMAT}', end_day)
    END AS end_date
FROM dates
```

Solution:

- Use BigQuery date solutions.

Create always resources from connectors (First always try to create resources with terraform)

main.tf

```hcl
provider "google" {
    project = var.project_id
}

resource "google_project_service" "workflows" {
    service            = "workflows.googleapis.com"
    disable_on_destroy = false
}

resource "google_service_account" "workflows_service_account" {
    account_id   = "sa-hello-world-workflow"
    display_name = "Sample Workflows Service Account"
}

resource "google_workflows_workflow" "workflows_example" {
    name            = "sample-workflow"
    region          = "us-central1"
    description     = "A sample workflow"
    service_account = google_service_account.workflows_service_account.id
    source_contents = templatefile("workflow.yaml", {
        project_id  = var.project_id
    })
}
```

variables.tf

```hcl
variable "project_id" {
    type = string
    description = "Project ID"
    default = "personalpage-270523"
}
```

workflow.yaml 1

```yaml
main:
    params: [input]
    steps:
    - hello:
        call: sys.log
        args:
          text: $${input.message}
          severity: INFO
    - call_bigquery:
        call: googleapis.bigquery.v2.jobs.query
        args:
          projectId: ${project_id}
          body:
            query: "SELECT CURRENT_DATE()"
```

Create always resources from connectors (First always try to create resources with terraform)

```
google_service_account.workflows_service_account: Creating...
google_project_service.workflows: Creating...
```

```
Error: Error when reading or editing Project Service : Request `List Project Services personalpage-270523` returned error: Failed
 to list enabled services for project personalpage-270523: googleapi: Error 403: Permission denied to list services for consumer co
ntainer [projects/347315878361]
```

```
→  WorkflowExample gcloud auth application-default login
Your browser has been opened to visit:
```

```
google_project_service.workflows: Creating...
google_service_account.workflows_service_account: Creating...
google_project_service.workflows: Creation complete after 4s [id=personalpage-270523/workflows.googleapis.com]
```

```
Error: Error creating service account: googleapi: Error 409: Service account sa-hello-world-workflow already exists within projec
t projects/personalpage-270523.
```

```
→  WorkflowExample terraform import google_service_account.workflows_service_account projects/personalpage-270523/serviceA
ccounts/sa-hello-world-workflow@personalpage-270523.iam.gserviceaccount.com
google_service_account.workflows_service_account: Importing from ID "projects/personalpage-270523/serviceAccounts/sa-hello
-world-workflow@personalpage-270523.iam.gserviceaccount.com"...
google_service_account.workflows_service_account: Import prepared!
  Prepared google_service_account for import
google_service_account.workflows_service_account: Refreshing state... [id=projects/personalpage-270523/serviceAccounts/sa-
hello-world-workflow@personalpage-270523.iam.gserviceaccount.com]
```

Create always resources from connectors (First always try to create resources with terraform)

```
➜  WorkflowExample terraform import google_service_account.workflows_service_account projects/personalpage-270523/serviceAccounts/sa-hello-world-workflow@personalpage-270523.iam.gserviceaccount.com
google_service_account.workflows_service_account: Importing from ID "projects/personalpage-270523/serviceAccounts/sa-hello-world-workflow@personalpage-270523.iam.gserviceaccount.com"...
google_service_account.workflows_service_account: Import prepared!
  Prepared google_service_account for import
google_service_account.workflows_service_account: Refreshing state... [id=projects/personalpage-270523/serviceAccounts/sa-hello-world-workflow@personalpage-270523.iam.gserviceaccount.com]
```

```
Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes

google_service_account.workflows_service_account: Modifying... [id=projects/personalpage-270523/serviceAccounts/sa-hello-world-workflow@personalpage-270523.iam.gserviceaccount.com]
google_service_account.workflows_service_account: Modifications complete after 7s [id=projects/personalpage-270523/serviceAccounts/sa-hello-world-workflow@personalpage-270523.iam.gserviceaccount.com]
google_workflows_workflow.workflows_example: Creating...
google_workflows_workflow.workflows_example: Creation complete after 1s [id=projects/personalpage-270523/locations/us-central1/workflows/sample-workflow]

Apply complete! Resources: 1 added, 1 changed, 0 destroyed.
```

Q & A

More resources:

- Create a workflow by using Terraform
  https://cloud.google.com/workflows/docs/create-workflow-terraform

- Execute a Cloud Run job using Workflows
  https://cloud.google.com/workflows/docs/tutorials/execute-cloud-run-jobs