Using Google Cloud Functions for Event-Driven Processing

Developing Applications with Google Cloud Platform

CLOUD FUNCTIONS, STACKDRIVER ERROR REPORTING, STACKDRIVER LOGGING

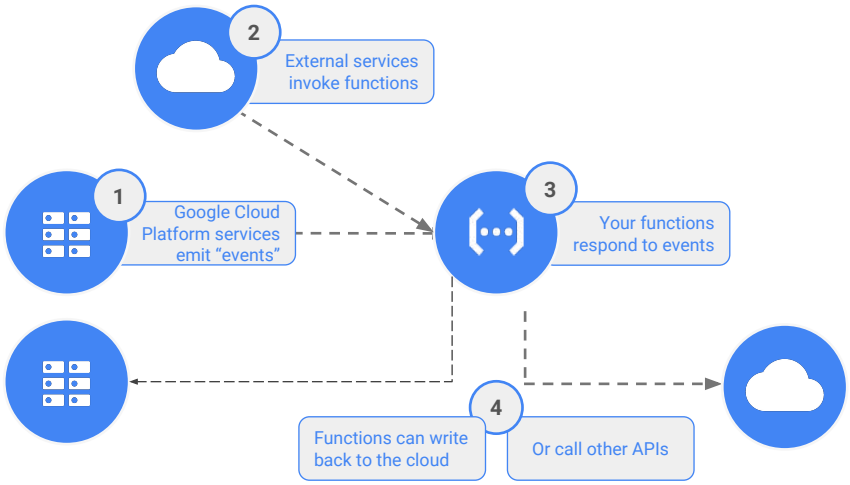
QWIKLABS PROCESSING CLOUD PUB/SUB DATA USING CLOUD FUNCTIONS

Google Cloud

Version 2.0
Last modified: 2017-09-25

For more information, see:
https://cloud.google.com/functions
https://cloud.google.com/functions/docs/tutorials/

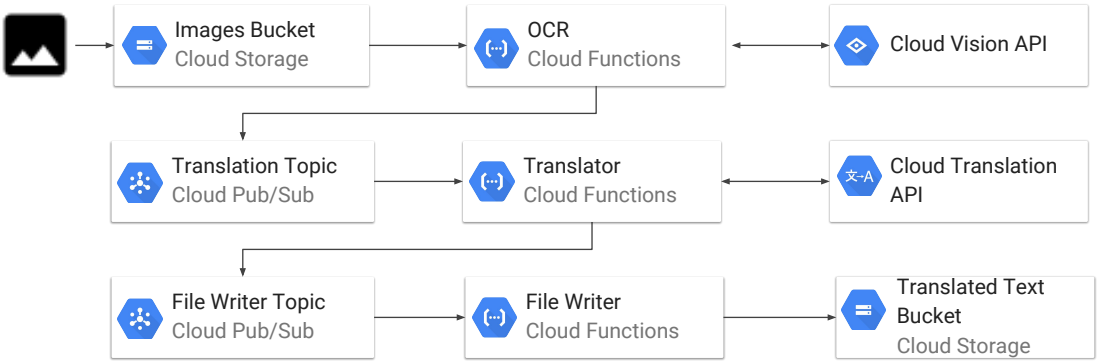
Overview



Google Cloud

Google Cloud Platform services emit events, such as when files are uploaded to a Google Cloud Storage bucket or messages are published to a Google Cloud Pub/Sub topic. External services can invoke functions in response to events in those systems, such as when a commit is made to a Github repository. Cloud Functions are triggered in response to these events. Cloud Functions can invoke other APIs or write data back to the cloud.

Cloud Functions enable event-driven, serverless, highly scalable microservices



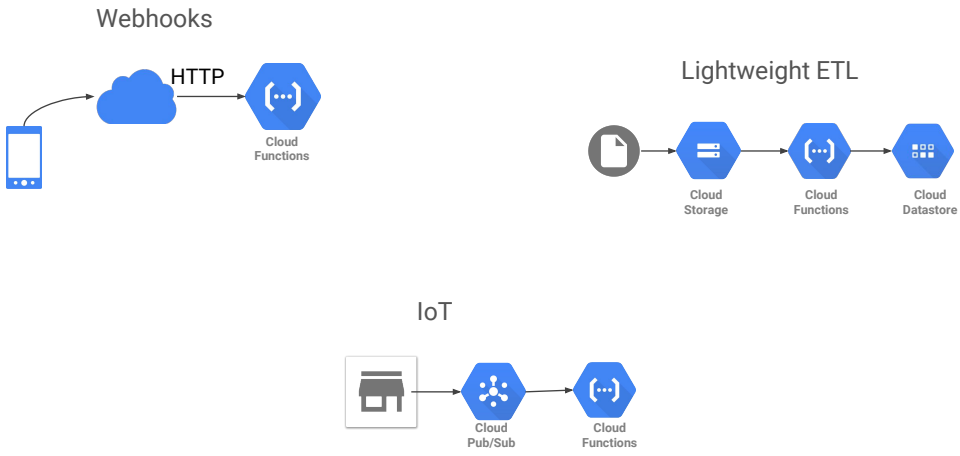
Google Cloud

With Cloud Functions, you can develop an application that is event-driven, serverless, and highly scalable. Each function is a lightweight microservice that enables you to integrate application components and data sources. Cloud Functions are ideal for microservices that require a small piece of code to quickly process data related to an event. Cloud Functions are priced according to how long your function runs, the number of times it is invoked, and the resources that you provision for the function.

Consider an application that allows users to upload images that contain text to a bucket in Google Cloud Storage. The OCR Cloud Function is triggered when a new image is uploaded to the Images bucket. This function uses the Google Cloud Vision API to extract text from images and then queues up the text in Cloud Pub/Sub for translation. The Translator Cloud Function, triggered by the Cloud Pub/Sub topic, invokes the Google Cloud Translation API to translate the text. It queues up the translated text in the File Writer topic in Cloud Pub/Sub. The File Writer Cloud Function writes the translated text for each image as separate files in Cloud Storage.

In this example, all components of the application use fully managed services and APIs such as Cloud Storage, Cloud Pub/Sub, Cloud Functions, Cloud Vision API, and Cloud Translation API. These services scale automatically depending on the volume of incoming data and required compute resources. This scalability and reliability enables you to focus on your application code.

Use cases



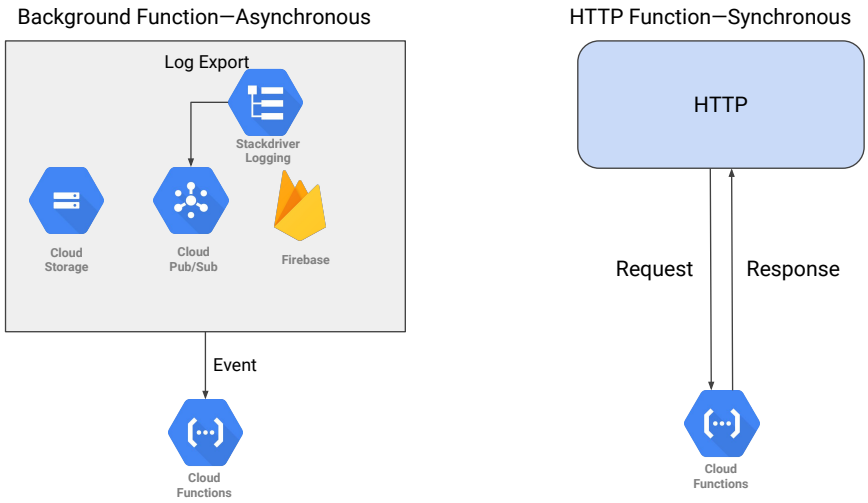
Google Cloud

Cloud Functions can be used in a variety of use cases that require lightweight microservices or event-driven processing. Cloud Functions can serve as webhooks. For example, you can set up a webhook that is invoked automatically after each commit to a Git repository. External and internal clients can make direct HTTP calls to invoke microservices that are deployed as Cloud Functions.

You can use Cloud Functions for lightweight extract-transform-load (ETL) operations. For example, when a file is uploaded to Cloud Storage, a Cloud Function can be triggered to transform and upload the contents to a database.

You can also use Cloud Functions to process IoT streaming data or other application messages that are published to a Cloud Pub/Sub topic.

Cloud Functions can have asynchronous and synchronous triggers



Cloud Functions can be triggered asynchronously in response to the following events:

- Cloud Storage events such as object creation, deletion, and updates
- Cloud Pub/Sub events including logs exported from Google Stackdriver Logging
- Firebase Realtime Database, Firebase Authentication, and Google Analytics for Firebase events

Asynchronously triggered functions are called *background* functions.

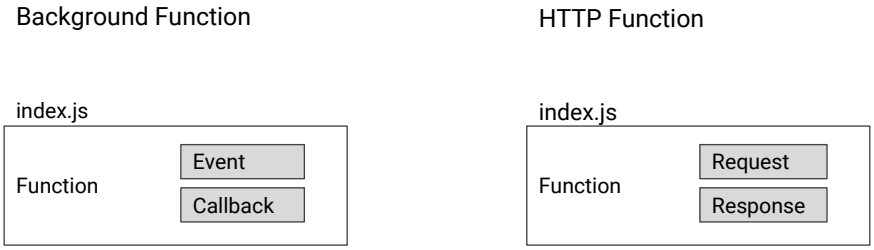
Cloud Functions can also be invoked synchronously by direct HTTP request-response. Functions invoked in this manner are called HTTP functions.

Cloud Functions have a default timeout value of 60 seconds. Ensure that the function will complete execution before timing out. For HTTP functions, it is important to keep the execution time to a minimum to avoid a negative user experience.

For more information, see:

- Overview of Logs Export: https://cloud.google.com/logging/docs/export/
- Cloud Functions for Firebase: https://firebase.google.com/docs/functions/

Write your Cloud Function



Specify dependencies in a package.json file

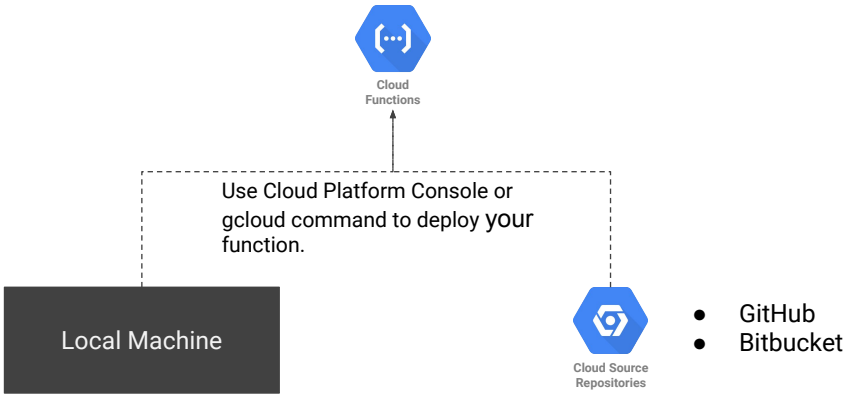
Google Cloud

You can write the code for Cloud Functions as Node.js functions in an index.js file. A background function takes an event and callback function as input parameters. An HTTP function takes request and response objects as input parameters.

You do not need to upload zip files with packaged dependencies. You can specify any dependencies in a package.json file. The Cloud Functions service automatically installs all dependencies before running your code.

Cloud Functions provides access to a local disk mount point (/tmp) that you can use for temporary file processing. Writing to the /tmp location uses the memory that you have allocated to your function. Make sure to clean up any temporary files.

Deploy your Cloud Function

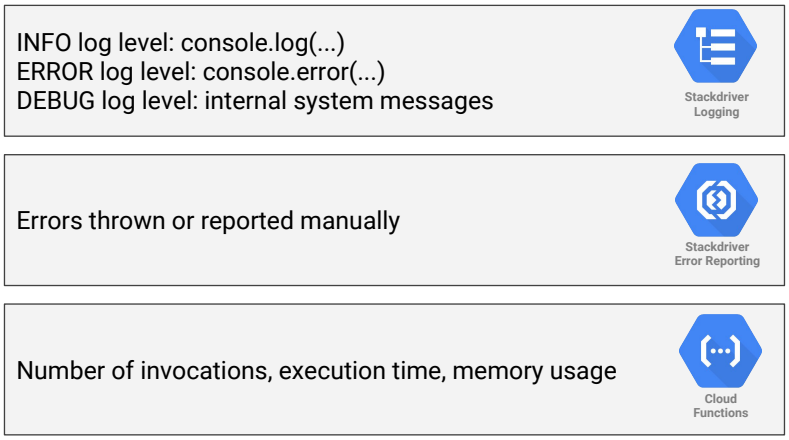


Google Cloud

You can deploy your function from your local machine by using the Cloud Platform Console or the gcloud command. The gcloud command automatically zips your code and uploads it to the Cloud Storage staging bucket that you specify. For more information about deploying from your local machine, see https://cloud.google.com/functions/docs/deploying/filesystem.

You can also deploy your code directly from an independent Google Cloud Source Repository or from a Cloud Source Repository that is automatically synchronized with a Github or Bitbucket repository. For more information about deploying from source control, see https://cloud.google.com/functions/docs/deploying/repo.

Cloud Functions supports logging, error reporting, and monitoring



Google Cloud

You can view the output from your console.log and console.error messages in Stackdriver Logging.

If your code throws errors or in case of other uncaught errors, Cloud Functions automatically captures these errors in Stackdriver Error Reporting. If you only want to report these errors to Stackdriver Error Reporting without throwing the error up to the caller, you can programmatically report these errors to Stackdriver Error Reporting. For more information about manual error reporting, see https://cloud.google.com/functions/docs/monitoring/error-reporting#manually_reporting_errors.

In the Cloud Platform Console, you can view metrics related to the number of invocations, execution time, and memory usage for your function.