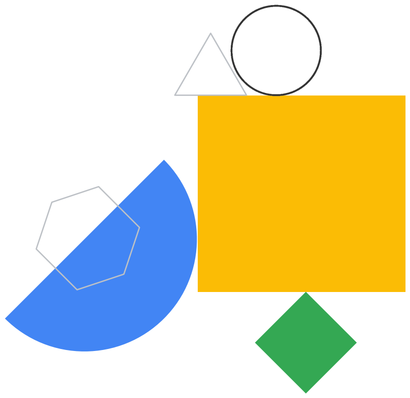
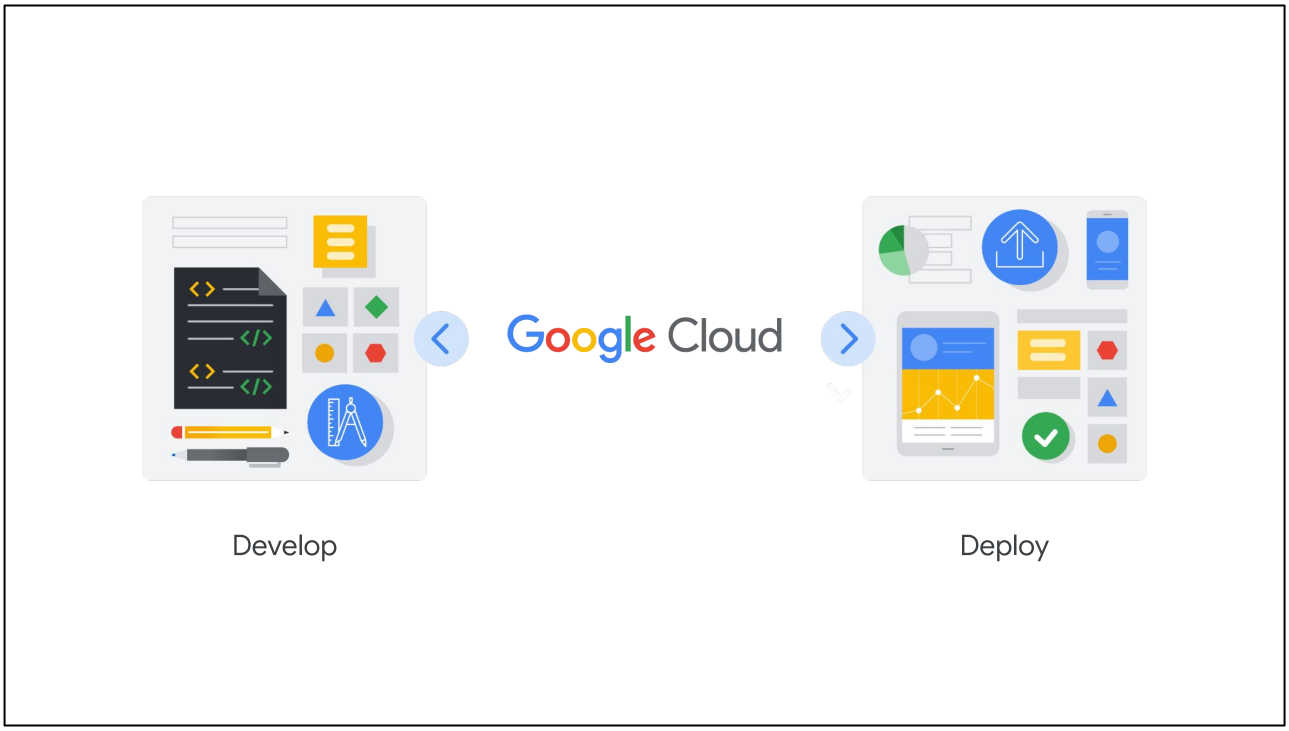
Google Cloud



# Google Cloud Core Infrastructure Module 7

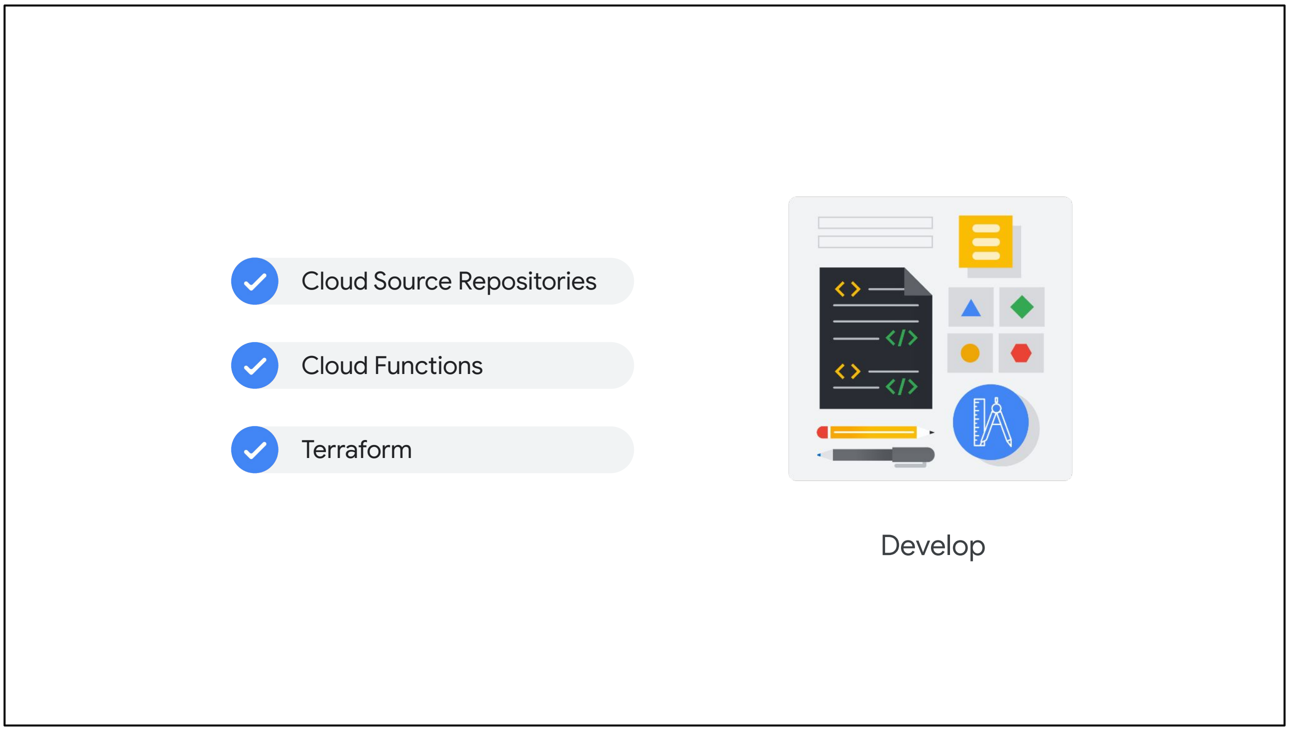
On-demand course

March 2022



Many users develop impressive applications using Google Cloud’s products and services. And when an app is ready, Google Cloud can also be used to deploy it.

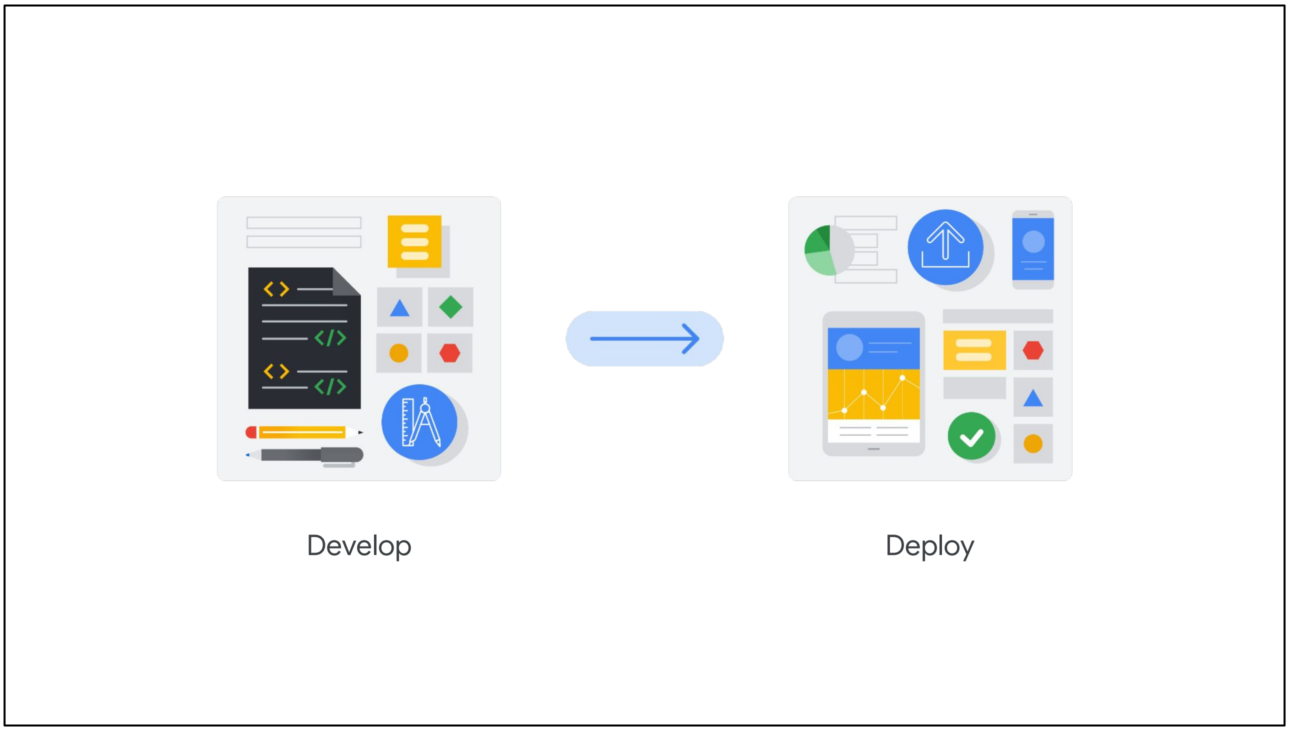
In this section of the course, we’ll explore Google Cloud methods for development in the cloud,



- ✓ Cloud Source Repositories
- ✓ Cloud Functions
- ✓ Terraform

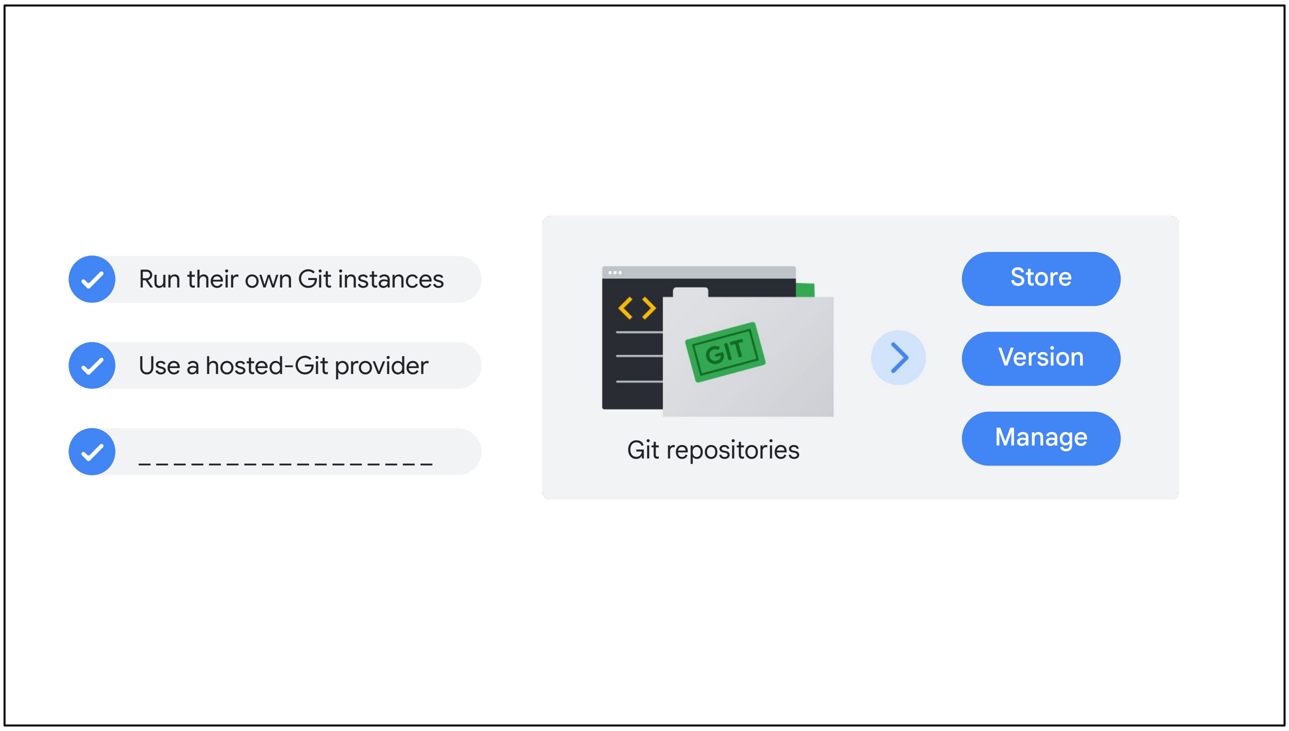
Develop

which includes Cloud Source Repositories, Cloud Functions, and Terraform.



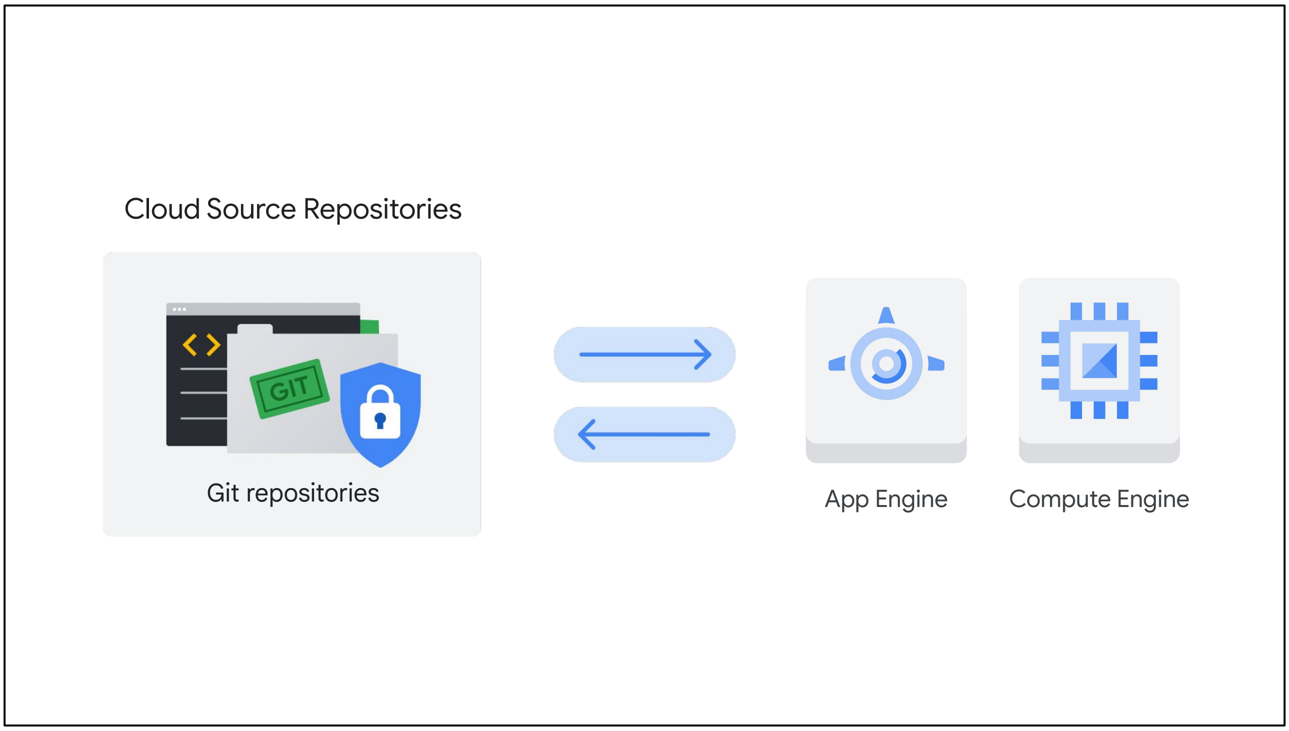
After that, we'll look at deployment with infrastructure as code.

Let's begin by looking at development in the cloud.



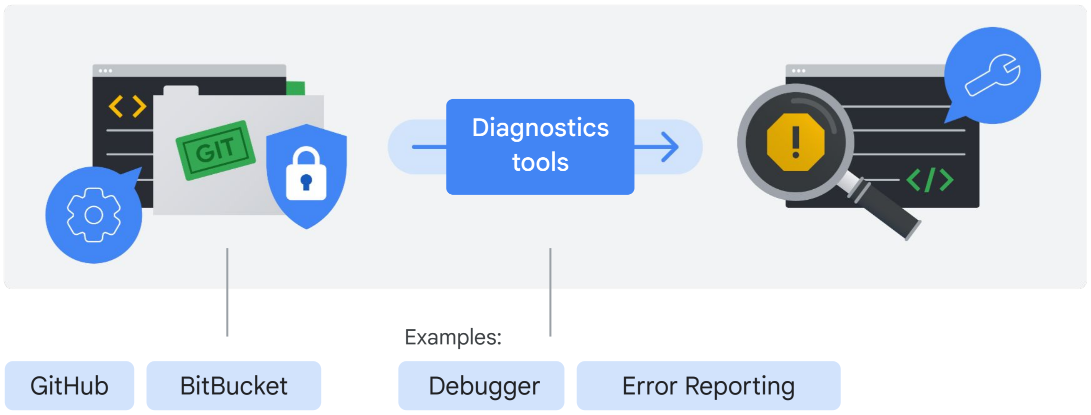
Many Google Cloud customers use Git repositories to store, version, and manage their source code trees. That means they either run their own Git instances, which is a great option if total control is required, or they use a hosted-Git provider, which means less work if total control isn't required.

But what if there were a third option, where you could keep code private to a Google Cloud project and use IAM permissions to protect it, but not have to maintain the Git instance yourself?



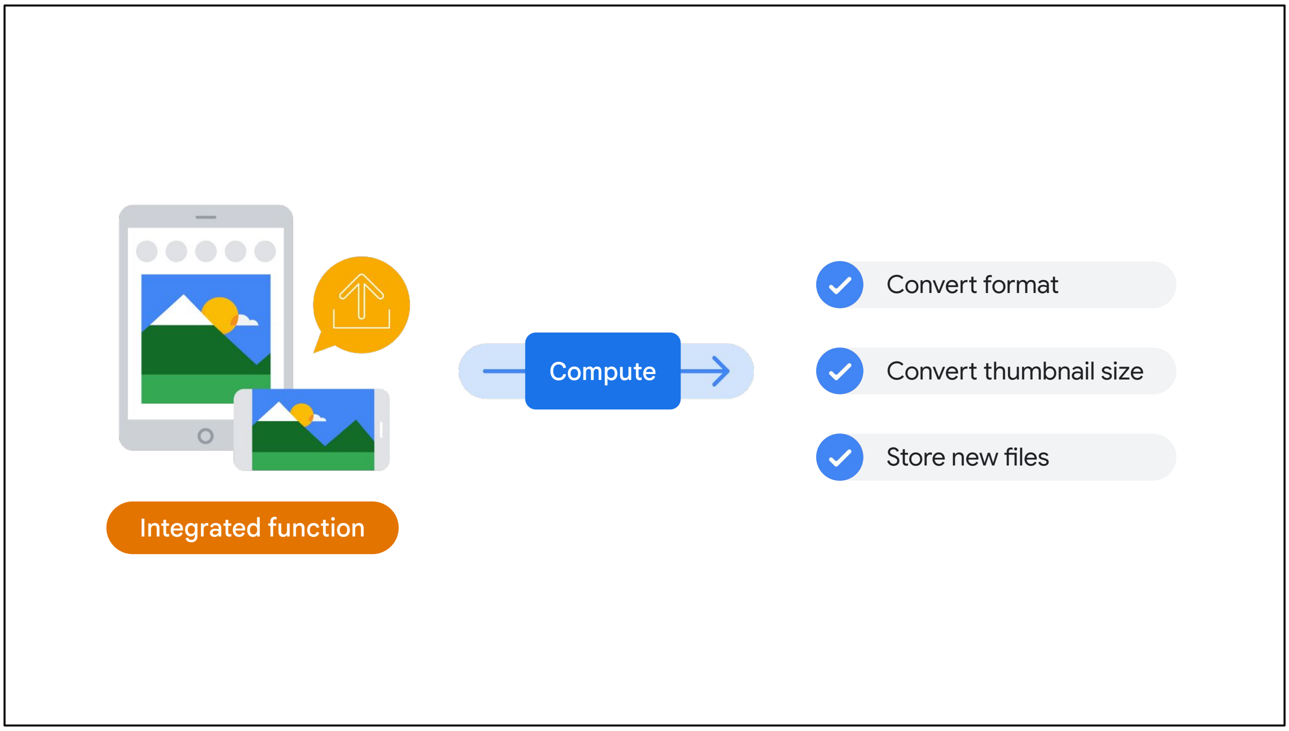
Well, that third option *is* available with **Cloud Source Repositories**.

Cloud Source Repositories provides full-featured Git repositories hosted on Google Cloud that support the collaborative development of any application or service, including those that run on App Engine and Compute Engine.

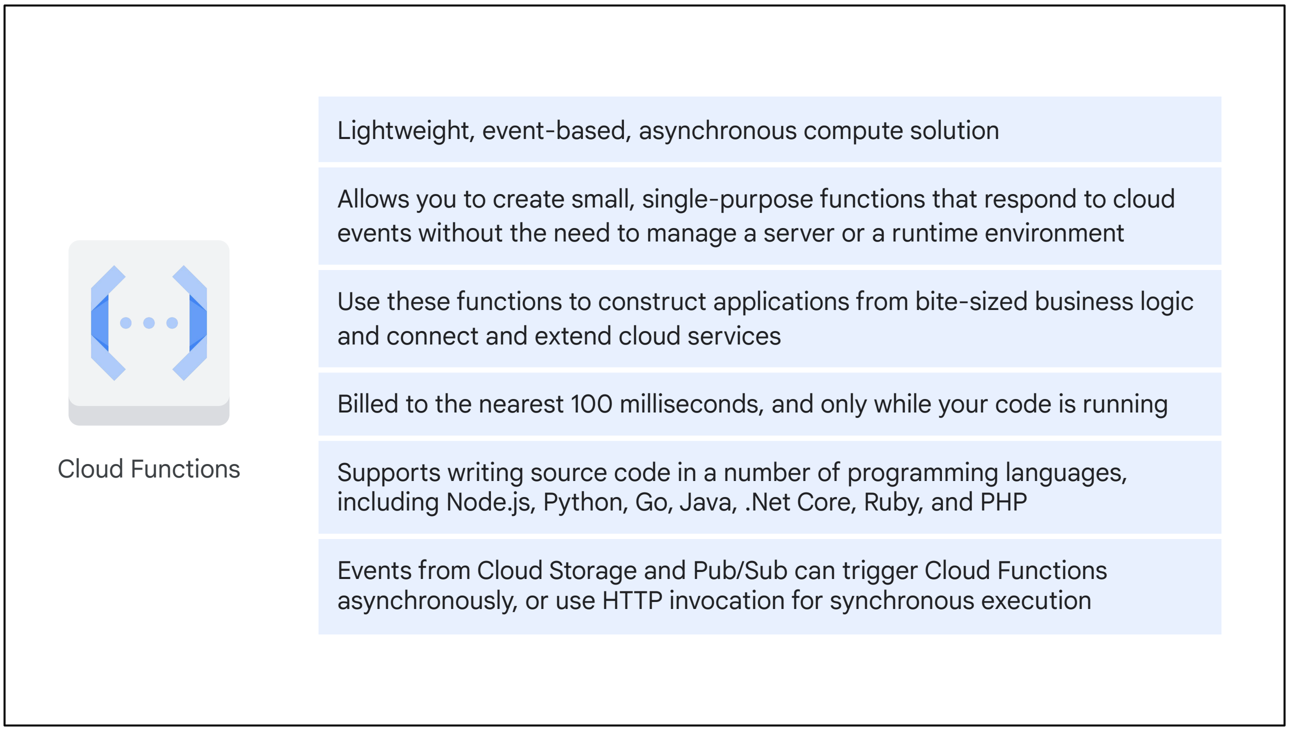


With Cloud Source Repositories, you can have any number of private Git repositories. This allows code associated with a cloud project to be organized the way you choose. It also allows Google Cloud diagnostics tools, like Debugger and Error Reporting, to use the code from Git repositories to track down issues to specific errors in deployed code *without* slowing down your users.

If your code is already in GitHub or BitBucket repositories, it can be migrated into your cloud project and used just like any other repository, including browsing and diagnostics.



When that event takes place, the image might need to be processed in a few different ways, converting a thumbnail into different sizes, and storing each new file in a repository. You could integrate this function into your application, but then you'd have to provide compute resources for it–whether it happens once a millisecond or once a day.



Cloud Functions

Lightweight, event-based, asynchronous compute solution

Allows you to create small, single-purpose functions that respond to cloud events without the need to manage a server or a runtime environment

Use these functions to construct applications from bite-sized business logic and connect and extend cloud services

Billed to the nearest 100 milliseconds, and only while your code is running

Supports writing source code in a number of programming languages, including Node.js, Python, Go, Java, .Net Core, Ruby, and PHP

Events from Cloud Storage and Pub/Sub can trigger Cloud Functions asynchronously, or use HTTP invocation for synchronous execution

Cloud Functions is a lightweight, event-based, asynchronous compute solution that allows you to create small, single-purpose functions that respond to cloud events, without the need to manage a server or a runtime environment. You can use these functions to construct application workflows from individual business logic tasks.

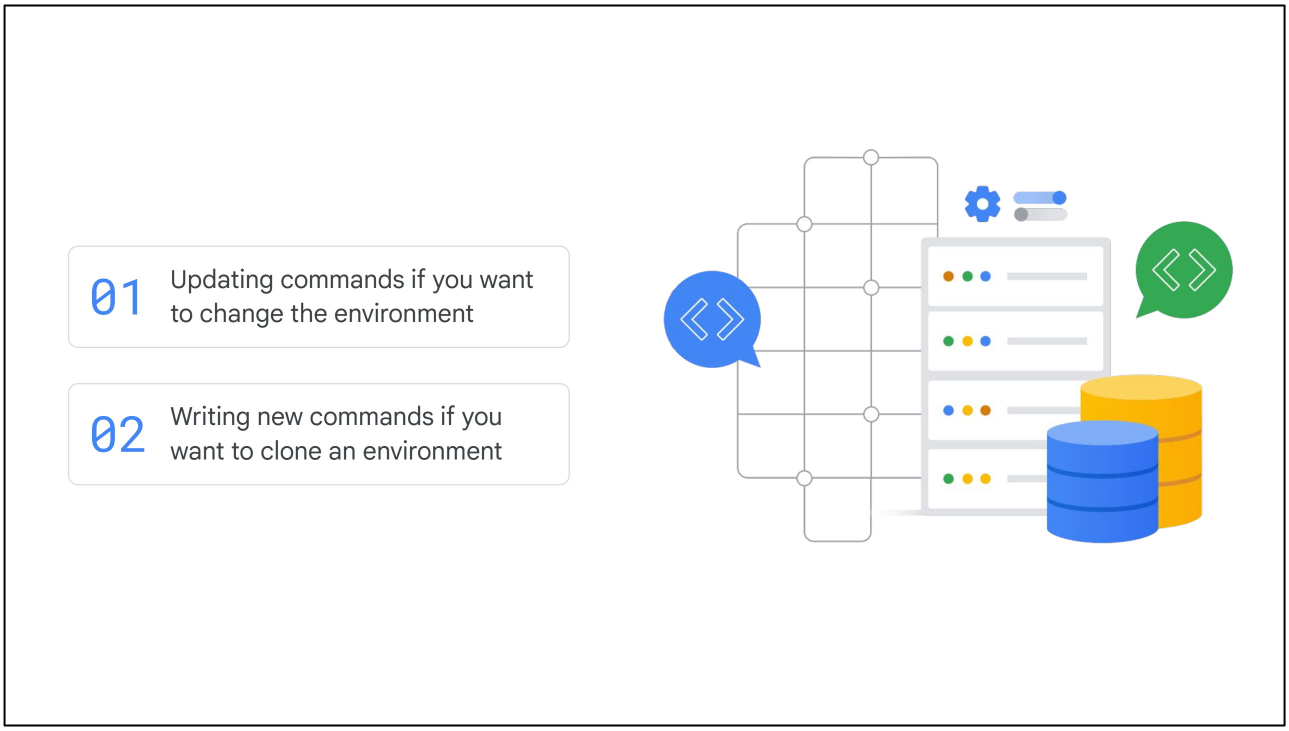
You can also use Cloud Functions to connect and extend cloud services.

You're billed to the nearest 100 milliseconds, but only while your code is running.

Cloud Functions supports writing source code in a number of programming languages. These include Node.js, Python, Go, Java, .Net Core, Ruby, and PHP.

For more information about the supported specific versions, refer to the runtimes documentation.

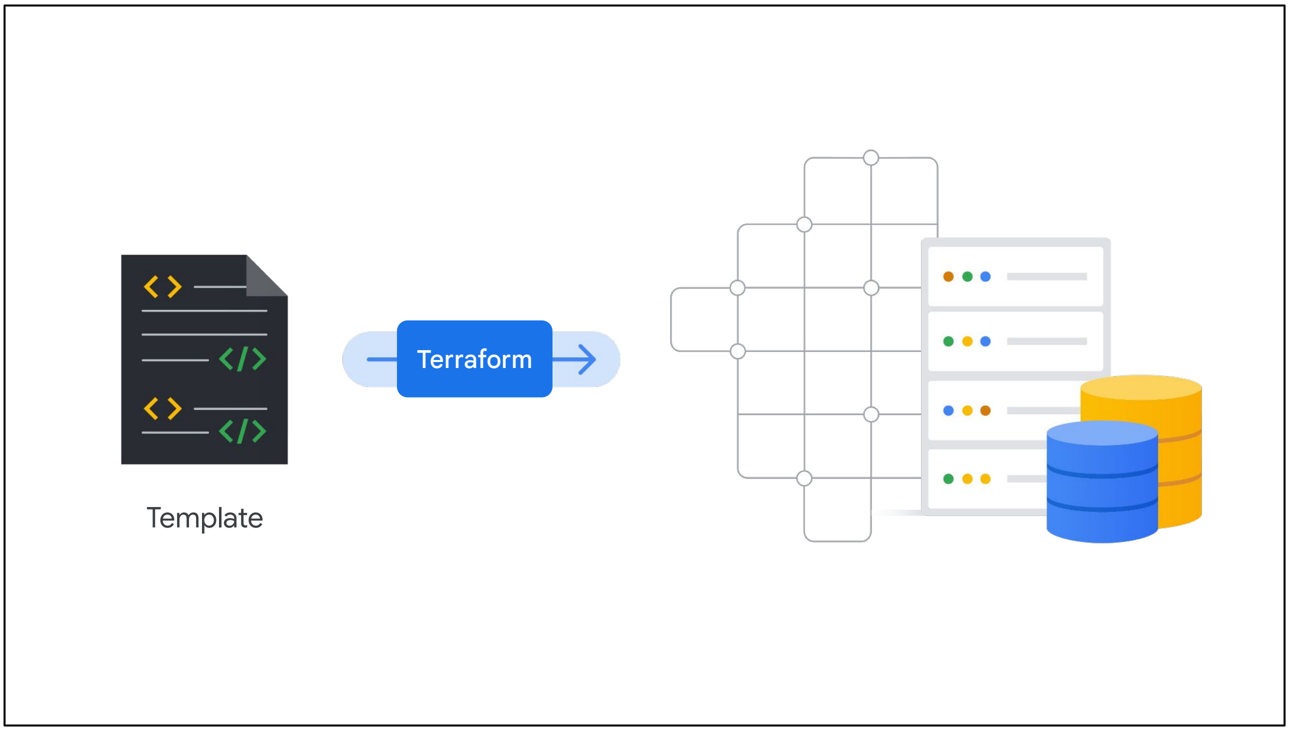
Events from Cloud Storage and Pub/Sub can trigger Cloud Functions asynchronously, or you can use HTTP invocation for synchronous execution.



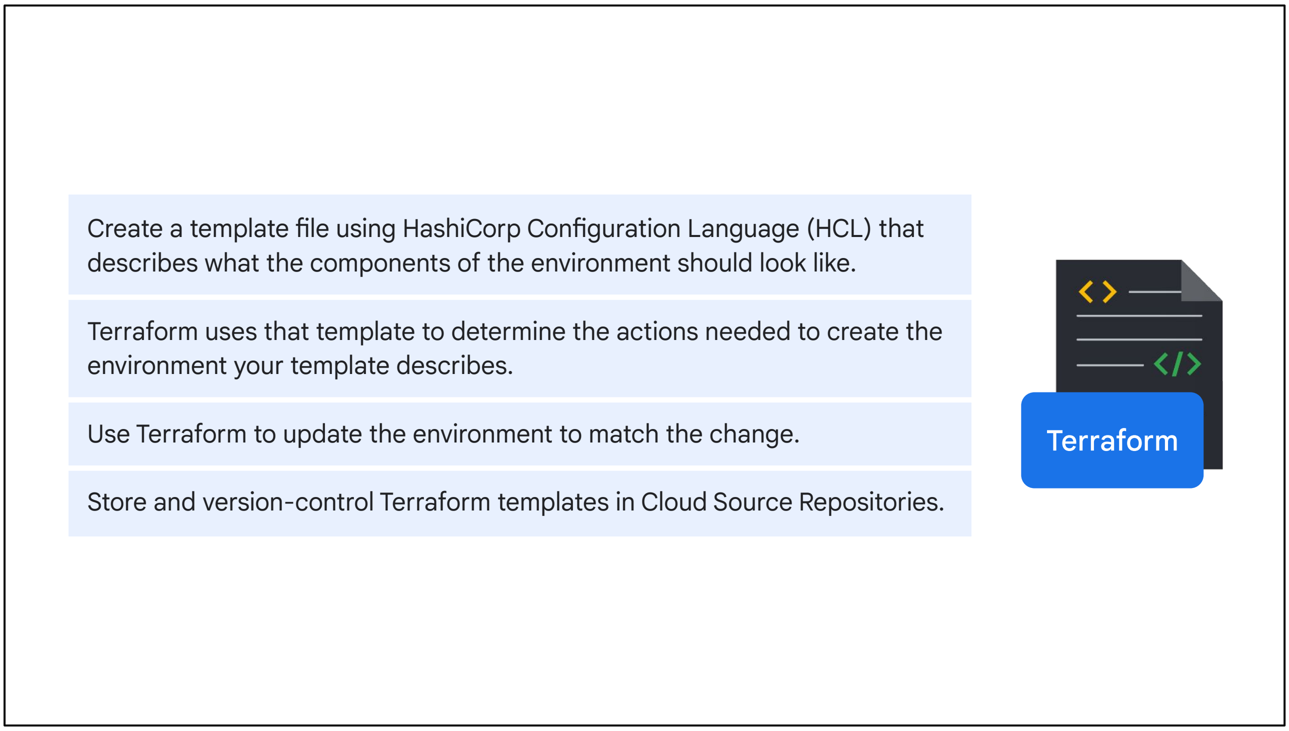
01 Updating commands if you want to change the environment

02 Writing new commands if you want to clone an environment

However, this is labor-intensive and requires updating commands if you want to change the environment or manually writing new commands if you want to clone an environment.



It’s more efficient to use a template. Using a template allows you to write the specifications for your application environment in the same way you’d write a configuration file, but your template can then be deployed in a scaled environment to quickly create as many identical application environments as needed. This can be done with **Terraform.**


Create a template file using HashiCorp Configuration Language (HCL) that describes what the components of the environment should look like.

Terraform uses that template to determine the actions needed to create the environment your template describes.

Use Terraform to update the environment to match the change.

Store and version-control Terraform templates in Cloud Source Repositories.

To use Terraform, you create a template file, using HashiCorp Configuration Language (HCL), that describes what you want the components of your environment to look like.

Terraform then uses that template to determine the actions needed to create the environment your template describes.

If you need to change the environment, you can edit your template and then use Terraform to update the environment to match the change.

You can store and version-control your Terraform templates in Cloud Source Repositories.