

This quickstart shows how to automatically deploy an app stored in Cloud Source Repositories to App Engine after a new commit.

1. Complete the steps from [Quickstart: Create a repository](/source-repositories/docs/quickstart/) (/source-repositories/docs/quickstart/).

After you complete that quickstart, you have an app you can deploy to App Engine.

2. Enable the App Engine Admin, Cloud Build APIs.

[Enable the APIs](https://console.cloud.google.com/flows/enableapi?apiid=appengine.googleapis.com,cloudbuild.googleapis.com) (https://console.cloud.google.com/flows/enableapi?apiid=appengine.googleapis.com,cloudbuild.googleapis.com)

Cloud Build uses a service account to deploy your code. The default permissions for this account don't allow certain actions, such as deploying to [App Engine](/appengine/docs/) (/appengine/docs).

Enable your service account to deploy to App Engine by granting the account additional [Cloud Identity and Access Management \(Cloud IAM\) roles](/iam/docs/understanding-roles/) (/iam/docs/understanding-roles):

1. In the Google Cloud console, open the Cloud Build Settings page:

[Open the Cloud Build Settings page](https://console.cloud.google.com/cloud-build/settings) (https://console.cloud.google.com/cloud-build/settings)

You'll see the **Service account permissions** page:

Settings

Service account permissions

Cloud Build executes builds with the permissions granted to the [Cloud Build service account](#) tied to the project. You can grant additional roles to the service account to allow Cloud Build to interact with other GCP services.

Service account email: [REDACTED]@cloudbuild.gserviceaccount.com

GCP Service	Role [?]	Status
Cloud Functions	Cloud Functions Developer	DISABLED ▾
Cloud Run	Cloud Run Admin	DISABLED ▾
App Engine	App Engine Admin	DISABLED ▾
Kubernetes Engine	Kubernetes Engine Developer	DISABLED ▾
Compute Engine	Compute Instance Admin (v1)	DISABLED ▾
Cloud KMS	Cloud KMS CryptoKey Decrypter	DISABLED ▾
Service Accounts	Service Account User	DISABLED ▾

Roles not listed here can be managed in the [IAM section](#)

2. Set the status of the **App Engine Admin** role to **Enable**.

1. In a terminal window, go to the directory containing the repository:

2. Deploy the sample app:

3. Verify that your app is running:

If your app is running, the browser displays the message `Hello, World!`.

1. In a terminal window, go to the directory containing the repository:

2. Using a text editor, create a file named `cloudbuild.yaml`, and then paste the following configuration information:

1. Add `cloudbuild.yaml` to the repository:

2. Commit the file with a comment describing the history of this action:

3. Using the `git push` command, add the contents of the local Git repository to Cloud Source Repositories:

1. In the GCP Console, open the Cloud Build **Triggers** page.

[Open the Triggers page \(https://console.cloud.google.com/cloud-build/triggers\)](https://console.cloud.google.com/cloud-build/triggers)

2. If your Google Cloud project isn't selected, click **Select a project**, and then click the name of your Google Cloud project.

3. Click **Create Trigger**.

The **Create trigger** page opens.

4. In the **Repository** list, select the `hello-world` repository.

5. Fill out the following options:

- In the **Name** field, type `app-engine-test`.
- In the **Trigger type** list, select **Branch**.
- In the **Build configuration** list, select **Cloud Build configuration file**.
- In the **Cloud Build configuration file location** field, type `cloudbuild.yaml` after the `/`.

6. Click **Create trigger**.

1. In a terminal window, use a text editor to update the `main.py` file by pasting the following code:

2. Add the file to Git:

3. Commit the file with a comment describing the history of this action:

4. Using the `git push` command, add the contents of the local Git repository to Cloud Source Repositories:

1. In the GCP Console, open the Cloud Build **Triggers** page.

[Open the Triggers page](https://console.cloud.google.com/cloud-build/triggers) (<https://console.cloud.google.com/cloud-build/triggers>)

2. If your Google Cloud project isn't selected, click **Select a project**, and then click the name of your Google Cloud project.

3. Click **History**.

A list of all builds opens. At the top is a new entry that represents the build that began after you pushed your change to Cloud Source Repositories. When the build is ready, a green check mark appears next to the build entry.

In a terminal window, open your app:

The browser now displays the message `I update automatically!`.


Note: If you don't see your changes yet, wait a minute or two and refresh your browser window.

To avoid incurring charges to your Google Cloud account for the resources used in this quickstart, follow these steps.

1. In the GCP Console, open the Cloud Build **Triggers** page.


[Open the Triggers page \(https://console.cloud.google.com/cloud-build/triggers\)](https://console.cloud.google.com/cloud-build/triggers)

2. If your Google Cloud project isn't selected, click **Select a project**, and then click the name of your Google Cloud project.


3. On the same line as the trigger you want to delete, click **More** , and then click **Delete**.

1. In the GCP Console, open the **All repositories** page for Cloud Source Repositories.

[Open Cloud Source Repositories \(https://source.cloud.google.com/repos\)](https://source.cloud.google.com/repos)

2. Hold the pointer over the repository you want to delete and click **Settings** .

The **General settings** page opens.

3. Click **Delete this repository** .

The **Remove repository** dialog opens.

4. Type the name of the repository you want to delete.

5. Click **Delete**.

- Learn more about [Cloud Build](/cloud-build/docs/) (/cloud-build/docs/).