

[Python](https://cloud.google.com/python/) (<https://cloud.google.com/python/>) [Guides](#)

Setting an uptime alert

This tutorial shows how to set up an uptime alert for the [Python Hello World app](https://cloud.google.com/python/getting-started/hello-world) (<https://cloud.google.com/python/getting-started/hello-world>) running on App Engine flexible environment using [Stackdriver Monitoring](https://cloud.google.com/monitoring/docs/) (<https://cloud.google.com/monitoring/docs/>). Uptime alerts let you know when your app is not serving traffic. You can also set uptime alerts for apps running on [Compute Engine](https://cloud.google.com/compute/) (<https://cloud.google.com/compute/>) or [Google Kubernetes Engine \(GKE\)](https://cloud.google.com/kubernetes-engine/) (<https://cloud.google.com/kubernetes-engine/>).

Objectives

- Run a basic Hello World app.
- Create an uptime check that monitors whether the app returns an HTTP '200' status code.
- Create an alert that sends an email message to you when the uptime check fails.
- Restart the app to trigger the alert.

Costs

Important: This tutorial uses the following billable components of Google Cloud:

- [App Engine](https://cloud.google.com/appengine/pricing) (<https://cloud.google.com/appengine/pricing>)
- [Stackdriver Monitoring](https://cloud.google.com/stackdriver/pricing) (<https://cloud.google.com/stackdriver/pricing>)

To generate a cost estimate based on your projected usage, use the [pricing calculator](https://cloud.google.com/products/calculator) (<https://cloud.google.com/products/calculator>). New Google Cloud users might be eligible for a [free trial](https://cloud.google.com/free-trial) (<https://cloud.google.com/free-trial>).

Monitoring is currently offered to beta users at no charge.

Before you begin

1. [Sign in](https://accounts.google.com/Login) (https://accounts.google.com/Login) to your Google Account.

If you don't already have one, [sign up for a new account](https://accounts.google.com/SignUp) (https://accounts.google.com/SignUp).

2. In the Cloud Console, on the project selector page, select or create a Google Cloud project.

Note: If you don't plan to keep the resources that you create in this procedure, create a project instead of selecting an existing project. After you finish these steps, you can delete the project, removing all resources associated with the project.

[GO TO THE PROJECT SELECTOR PAGE](https://console.cloud.google.com/projectselector) (HTTPS://CONSOLE.CLOUD.GOOGLE.COM/PROJECTSELECTOR)

3. Make sure that billing is enabled for your Google Cloud project. [Learn how to confirm billing is enabled for your project](https://cloud.google.com/billing/docs/how-to/modify-project) (https://cloud.google.com/billing/docs/how-to/modify-project).

Cloning the sample app

The sample app is available on GitHub at [GoogleCloudPlatform/getting-started-python](https://github.com/GoogleCloudPlatform/getting-started-python) (https://github.com/GoogleCloudPlatform/getting-started-python).

1. Clone the repository.

```
git clone https://github.com/GoogleCloudPlatform/getting-started-python.git
```

2. Go to the sample directory.

```
cd getting-started-python/1-hello-world
```

3. Because the app only returns "Hello World!", it requires no configuration, and you can run it right away.

```
gcloud app deploy
```

4. To see the returned message, enter the following address. Replace [YOUR_PROJECT_ID] with your Google Cloud project ID.

```
https://[YOUR_PROJECT_ID].appspot.com
```



Adding your project to a Workspace

After your app is deployed, you can use Monitoring to create an uptime check. The check continually pings your deployed app to ensure that it's returning a healthy response. To use Stackdriver, your project must be in a Workspace.

1. From the Cloud Console, go to **Monitoring**.

GO TO MONITORING ([HTTPS://CONSOLE.CLOUD.GOOGLE.COM/MONITORING](https://console.cloud.google.com/monitoring))

2. If the **Add your project to a Workspace** dialog is displayed, create a new Workspace by selecting your Google Cloud project under **New Workspace** and then clicking **Add**. In the following image, the Google Cloud project name is **Quickstart**:

Add your project to a Workspace

A Workspace lets you monitor resources for up to 100 Google Cloud Platform projects. [Learn more](#)

New Workspace

Quickstart

Existing Workspace

vhamburger-sap-vpc 2 projects

elfs-test 3 projects

Foo Project 20180511 3 projects

The **Add your project to a Workspace** dialog is displayed only when you have at least one existing Workspace available to you. The Workspaces listed under **Existing Workspace** are Workspaces you've created or Workspaces for Google Cloud projects where you have editorial permission. Using this dialog, you can choose between creating a new Workspace and adding your project to an existing Workspace.

Creating an uptime check

1. In the Cloud Console, go to **Monitoring**.

GO TO MONITORING ([HTTPS://CONSOLE.CLOUD.GOOGLE.COM/MONITORING](https://console.cloud.google.com/monitoring))

2. In the navigation pane, click **Uptime Checks**, then click **Create an Uptime Check**.
3. If a dialog opens asking to automatically create an uptime check, click **No thanks**.
4. Give your check a title, such as **Check Hello World**.
5. Because you deployed to App Engine, change **Resource Type** to **App Engine** instead of **URL**. (**URL** is for generating a custom URL on a Compute Engine instance.)
6. Leave **Path** blank to default to the main index page.
7. Change **Check every** to **1** minute.

Creating an alerting policy

After you create the check, you are automatically prompted to create an associated alerting policy. An alerting policy lets you manage how Stackdriver informs you of any incidents, such as a failed uptime check.

1. To follow the prompt, click **Create Alerting Policy**. Give your policy a name, such as **Hello World Uptime Check Policy**.
2. The **Conditions** of the alert are prepopulated with your uptime check. To manually populate the **Conditions** instead, you can click **Add Another Condition > Uptime Check Health**.
3. To receive an email message from this alerting policy, under **Notifications**, make sure that **Email** is selected, and then click **Add Notification** to enter your email address.
4. Click **Save**.

Simulating an outage

Now that the uptime check is created, you can simulate an outage by changing your app to respond with an HTTP **404 Sorry, we can't find that page** error rather than an HTTP **200 OK** response.

1. The following code shows where the Hello World app returns only a 'Hello World!' message, and that the status code of the response defaults to 200 OK. To view this code in the Hello World app, use the view function.

[1-hello-world/main.py](#)

(<https://github.com/GoogleCloudPlatform/getting-started-python/blob/504b3d550b551502cfe96f32542c31b232135eff/1-hello-world/main.py>)

```
STARTED-PYTHON/BLOB/504B3D550B551502CFE96F32542C31B232135EFF/1-HELLO-WORLD/MAIN.PY
```

```
from flask import Flask

app = Flask(__name__)

@app.route('/')
def hello():
    """Return a friendly HTTP greeting."""
    return 'Hello World!'

if __name__ == '__main__':
    app.run(host='127.0.0.1', port=8080)
```

2. To cause the Hello World app to return an HTTP 404 error code, change the return line by adding a 404 value to the second part of the return value.

```
return 'Hello World', 404
```

3. Deploy the new, intentionally broken app.

```
gcloud app deploy
```

Within half an hour, you'll receive an email message that states that your uptime check is failing.

Cleaning up

To avoid incurring charges to your Google Cloud Platform account for the resources used in this tutorial:

The easiest way to eliminate billing is to delete the project that you created for the tutorial.

To delete the project:


Caution: Deleting a project has the following effects:

- **Everything in the project is deleted.** If you used an existing project for this tutorial, when you delete it, you also delete any other work you've done in the project.
- **Custom project IDs are lost.** When you created this project, you might have created a custom project ID that you want to use in the future. To preserve the URLs that use the project ID, such as an **appspot.com** URL, delete selected resources inside the project instead of deleting the whole project.

If you plan to explore multiple tutorials and quickstarts, reusing projects can help you avoid exceeding project quota limits.

1. In the Cloud Console, go to the **Manage resources** page.

[GO TO THE MANAGE RESOURCES PAGE](https://console.cloud.google.com/iam-admin/projects) ([HTTPS://CONSOLE.CLOUD.GOOGLE.COM/IAM-ADMIN/PRO](https://console.cloud.google.com/iam-admin/projects)

2. In the project list, select the project you want to delete and click **Delete** .

3. In the dialog, type the project ID, and then click **Shut down** to delete the project.

[< PREV](#)

[NEXT >](https://cloud.google.com/python/monitor-and-debug/reporting-errors) ([HTTPS://CLOUD.GOOGLE.COM/PYTHON/MONITOR-AND-DEBUG/REPORTING-ERRORS](https://cloud.google.com/python/monitor-and-debug/reporting-errors))

Except as otherwise noted, the content of this page is licensed under the [Creative Commons Attribution 4.0 License](https://creativecommons.org/licenses/by/4.0/) (<https://creativecommons.org/licenses/by/4.0/>), and code samples are licensed under the [Apache 2.0 License](https://www.apache.org/licenses/LICENSE-2.0) (<https://www.apache.org/licenses/LICENSE-2.0>). For details, see our [Site Policies](https://developers.google.com/terms/site-policies) (<https://developers.google.com/terms/site-policies>). Java is a registered trademark of Oracle and/or its affiliates.

Last updated January 6, 2020.