PHP (https://cloud.google.com/php/) Guides

Getting started with PHP on Compute Engine

This tutorial shows how to get started with Compute Engine. Follow this tutorial by deploying a Hello World PHP web app to Compute Engine. For help getting started with App Engine, see the App Engine standard environment (https://cloud.google.com/php/getting-started/tutorial-app).

Objectives

- Use Cloud Shell to download and deploy a Hello World sample app.
- Deploy a Hello World sample app to a single Compute Engine instance.

Costs

This tutorial uses the following billable components of Google Cloud:

- Compute Engine (https://cloud.google.com/compute/all-pricing)

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

Before you begin

1. Sign in (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).

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/PROJECTSELECT

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).

4. Enable the Compute Engine API.

ENABLE THE API (HTTPS://CONSOLE.CLOUD.GOOGLE.COM/FLOWS/ENABLEAPI?APIID=COMPUTE)

5. In the Google Cloud Console, open the app in Cloud Shell (https://cloud.google.com/shell).

GO TO CLOUD SHELL (HTTPS://CLOUD.GOOGLE.COM/CONSOLE/CLOUDSHELL/OPEN?GIT_BRANCH

Cloud Shell provides command-line access to your cloud resources directly from the browser. Open Cloud Shell in your browser and click **Proceed** to download the sample code and change into the app directory.

6. In Cloud Shell, configure the `gcloud` tool to use your new Google Cloud project:

```
# Configure gcloud for your project
gcloud config set project YOUR_PROJECT_ID
```

Running the app in Cloud Shell

1. Install your app dependencies using `composer`:

```
composer install
```

2. Run the PHP built-in web server
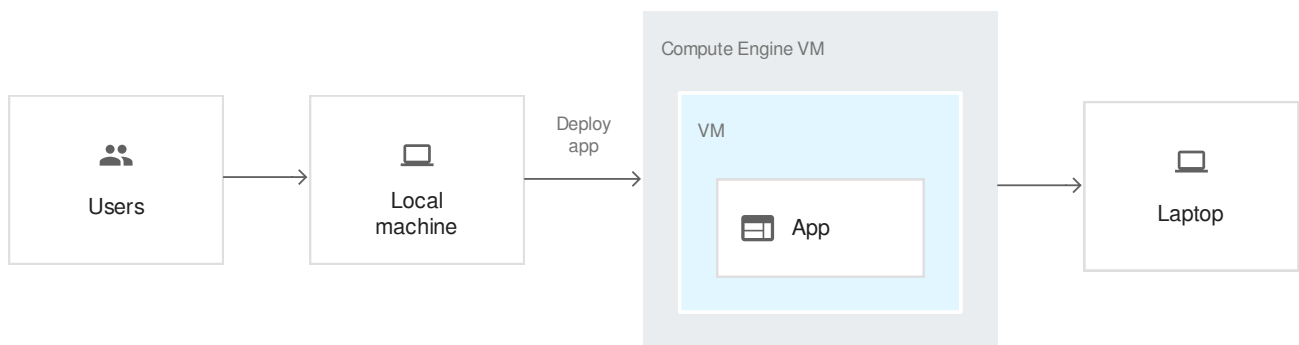(https://www.php.net/manual/en/features.commandline.webserver.php):

```
php -S localhost:8080
```

3. In Cloud Shell, click **Web preview** ⊡, and select **Preview on port 8080**. This opens a new window with your running app.

4. To stop the local web server, press `Control+C`.

Deploying to a single instance

This section walks you through running a single instance of your app on Compute Engine.

From Cloud Shell, you can deploy to a single Compute Engine instance virtual machine (VM) which runs your app.

Use a startup script to initialize an instance

You need a way to instruct your instance to download and run your code. An instance can have a startup script that runs whenever the instance is started or restarted.

[gce/scripts/startup-script.sh](https://github.com/GoogleCloudPlatform/getting-started-php/blob/master/gce/scripts/startup-script.sh) (https://github.com/GoogleCloudPlatform/getting-started-php/blob/master/gce/scripts/startup-script.sh)

GOOGLECLOUDPLATFORM/GETTING-STARTED-PHP/BLOB/MASTER/GCE/SCRIPTS/STARTUP-SCRIPT.SH)

```
set -e
export HOME=/root

# Install PHP and dependencies from apt
apt-get update
```

```
apt-get install -y git nginx php7.2 php7.2-fpm php7.2-mysql php7.2-dev \
    php7.2-mbstring php7.2-zip php-pear pkg-config

# Install Composer
curl -sS https://getcomposer.org/installer | \
    /usr/bin/php -- \
    --install-dir=/usr/local/bin \
    --filename=composer

# Get the application source code
git clone https://github.com/googlecloudplatform/getting-started-php /opt/src
ln -s /opt/src/gce /opt/app

# Run Composer
composer install -d /opt/app --no-ansi --no-progress --no-dev

# Disable the default NGINX configuration
rm /etc/nginx/sites-enabled/default

# Enable our NGINX configuration
cp /opt/app/config/nginx/helloworld.conf /etc/nginx/sites-available/helloworld.conf
ln -s /etc/nginx/sites-available/helloworld.conf /etc/nginx/sites-enabled/helloworld
cp /opt/app/config/nginx/fastcgi_params /etc/nginx/fastcgi_params

# Start NGINX
systemctl restart nginx.service

# Install Fluentd
curl -s "https://storage.googleapis.com/signals-agents/logging/google-fluentd-instal

# Enable our Fluentd configuration
cp /opt/app/config/fluentd/helloworld.conf /etc/google-fluentd/config.d/helloworld.c

# Start Fluentd
service google-fluentd restart &
```

The startup script performs these tasks:

- Installs PHP 7 and PHP 7 FPM.

- Installs and configures NGINX.

- Downloads Composer and runs it for the app code.

- Installs the Stackdriver Logging agent and configures it to monitor the app logs. This
  means that the logging configured in the previous steps of this tutorial are uploaded just

as if you were using App Engine.

Create and configure a Compute Engine instance

1. Create a Compute Engine instance:

   [gce/scripts/deploy.sh](https://github.com/GoogleCloudPlatform/getting-started-php/blob/master/gce/scripts/deploy.sh)
   (https://github.com/GoogleCloudPlatform/getting-started-php/blob/master/gce/scripts/deploy.sh)

   UB.COM/GOOGLECLOUDPLATFORM/GETTING-STARTED-PHP/BLOB/MASTER/GCE/SCRIPTS/DEPLOY.SH)

   ```
   MY_INSTANCE_NAME='my-app-instance'
   ZONE=us-central1-f

   gcloud compute instances create $MY_INSTANCE_NAME \
       --image-family=ubuntu-1804-lts \
       --image-project=ubuntu-os-cloud \
       --machine-type=g1-small \
       --scopes userinfo-email,cloud-platform \
       --metadata-from-file startup-script=scripts/startup-script.sh \
       --zone $ZONE \
       --tags http-server
   ```

   Replace the zone with a development zone, for example `us-central1-a`. For more information on regions and zones, see Geography and regions (https://cloud.google.com/docs/geography-and-regions).

   This creates a new instance, allows it to access Google Cloud services, and runs your startup script. The instance name is `my-app-instance`.

2. Check the progress of the instance creation:

   ```
   gcloud compute instances get-serial-port-output my-app-instance --zone YOUR_ZONE
   ```

   Replace *YOUR_ZONE* with the zone you deployed your instance to.

   When the startup script is complete, you see the following message:

   ```
   startup-script: INFO Finished running startup scripts.
   ```

3. Create a firewall rule to allow traffic to your instance:

```
gcloud compute firewall-rules create default-allow-http-80 \
    --allow tcp:80 \
    --source-ranges 0.0.0.0/0 \
    --target-tags http-server \
    --description "Allow port 80 access to http-server"
```

4. Get the external IP address of your instance:

```
gcloud compute instances list
```

5. To see your app running, enter this URL in your browser:

```
http://YOUR_INSTANCE_IP
```

Replace `YOUR_INSTANCE_IP` with the external IP address of your instance.

Manage and monitor an instance

You can use the Cloud Console to monitor and manage your instance.

1. In the Cloud Console, go to the **VM instances** page.

   GO TO THE VM INSTANCES PAGE (HTTPS://CONSOLE.CLOUD.GOOGLE.COM/COMPUTE/INSTANCES

2. In the list of virtual machine instances, click **SSH** in the row of the instance that you want to connect to.

3. To view all of the logs generated by your Compute Engine resources, go to the **Logs viewer** page.

   GO TO LOGS VIEWER (HTTPS://CONSOLE.CLOUD.GOOGLE.COM/LOGS?SERVICE=COMPUTE.GOOGL

   Stackdriver Logging (https://cloud.google.com/logging) is automatically configured to gather logs from various common services, including `syslog`.

Cleaning up

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

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/PRC

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.

Delete the individual resources

```
gcloud compute instances delete my-app-instance --zone=YOUR_ZONE --delete-disks=all
gcloud compute firewall-rules delete default-allow-http-80
```

What's next

- Create managed instance groups
  (https://cloud.google.com/compute/docs/instance-groups/creating-groups-of-managed-instances).

- Create a load balancer
  (https://cloud.google.com/load-balancing/docs/https/https-load-balancer-example).

- Handle sessions with Firestore
  (https://cloud.google.com/php/getting-started/session-handling-with-firestore).

- Add background processing with Pub/Sub and Cloud Run
  (https://cloud.google.com/php/getting-started/background-processing).

- Deploy an app to Google Kubernetes Engine
  (https://cloud.google.com/kubernetes-engine/docs/quickstarts/deploying-a-language-specific-app).

- Authenticate users with Identity-Aware Proxy
  (https://cloud.google.com/php/getting-started/authenticate-users).

Last updated January 7, 2020.