

[Serverless Computing](https://cloud.google.com/products/serverless/) (https://cloud.google.com/products/serverless/)

[Cloud Run: Serverless Computing](https://cloud.google.com/run/) (https://cloud.google.com/run/)

[Documentation](https://cloud.google.com/run/docs/) (https://cloud.google.com/run/docs/) [Guides](#)

Using system packages tutorial

This tutorial shows how to build a custom [Cloud Run](https://cloud.google.com/run/) service that transforms a graph description input parameter into a diagram in the PNG image format. It uses [Graphviz](http://www.graphviz.org/) and is installed as a system package in the service's container environment. Graphviz is used via command-line utilities to serve requests.

You can use this tutorial with Cloud Run (fully managed) or Cloud Run for Anthos on Google Cloud.

Objectives

- Write and build a [custom container](https://www.docker.com/resources/what-container) with a [Dockerfile](https://docs.docker.com/engine/reference/builder/)
- Write, build, and deploy a Cloud Run service
- Use [Graphviz dot](http://www.graphviz.org/documentation/) utility to generate diagrams
- Test the service by posting a DOT syntax diagram from the collection or your own creation

Costs

This tutorial uses billable components of Cloud Platform, including:

- [Cloud Build](https://cloud.google.com/cloud-build/)
- [Container Registry](https://cloud.google.com/container-registry/)
- [Cloud Run or Cloud Run for Anthos on Google Cloud](https://cloud.google.com/run/)

Use the [Pricing Calculator](https://cloud.google.com/products/calculator/) to generate a cost estimate based on your projected usage.

New Cloud Platform users might be eligible for a [free trial](https://cloud.google.com/free).

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).

- 4.

[ENABLE THE CLOUD RUN API](https://console.cloud.google.com/apis/library/run.googleapis.com) (HTTPS://CONSOLE.CLOUD.GOOGLE.COM/APIS/LIBRARY/RUN.GOOGLE.CLOUD.COM)

5. [Install and initialize](https://cloud.google.com/sdk/docs/) (https://cloud.google.com/sdk/docs/) the Cloud SDK.

6. For Cloud Run for Anthos on Google Cloud install the gcloud kubectl component:

```
gcloud components install kubectl
```



7. Update components:

```
gcloud components update
```



8. Install [curl](https://curl.haxx.se/dlwiz/?type=bin) (https://curl.haxx.se/dlwiz/?type=bin) to try out the service
9. If you are using Cloud Run for Anthos on Google Cloud, create a new cluster using the instructions in [Setting up Cloud Run for Anthos on Google Cloud](https://cloud.google.com/run/docs/gke/setup) (https://cloud.google.com/run/docs/gke/setup).

Setting up gcloud defaults

To configure gcloud with defaults for your Cloud Run service:

1. Set your default project:

```
gcloud config set project PROJECT-ID
```

Replace **PROJECT-ID** with the name of the project you created for this tutorial.

2. If you are using Cloud Run (fully managed), configure gcloud for your chosen region:

```
gcloud config set run/region REGION
```

Replace **REGION** with the supported Cloud Run [region](#) ([#follow-cloud-run](#)) of your choice.

3. If you are using Cloud Run for Anthos on Google Cloud, configure gcloud for your cluster:

```
gcloud config set run/cluster CLUSTER-NAME  
gcloud config set run/cluster_location REGION
```

Replace

- **CLUSTER-NAME** with the name you used for your cluster,
- **REGION** with the supported cluster location of your choice.

Retrieving the code sample

To retrieve the code sample for use:

1. Clone the sample app repository to your local machine:

[NODE.JS](#) [PYTHON](#) [GO](#) [JAVA](#)

```
git clone https://github.com/GoogleCloudPlatform/nodejs-docs-samples.git
```

Alternatively, you can [download the sample](#)

(<https://github.com/GoogleCloudPlatform/nodejs-docs-samples/archive/master.zip>) as a zip file and extract it.

2. Change to the directory that contains the Cloud Run sample code:

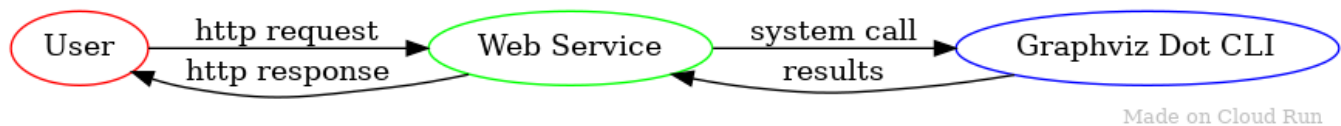
[NODE.JS](#) [PYTHON](#) [GO](#) [JAVA](#)

```
cd nodejs-docs-samples/run/system-package/
```



Visualizing the architecture

The basic architecture looks like this:



For the diagram source, see the [DOT Description](#)

(<https://cloud.google.com/run/docs/tutorials/resources/diagram-as-a-service.dot.txt>)

The user makes an HTTP request to the Cloud Run service which executes a Graphviz utility to transform the request into an image. That image is delivered to the user as the HTTP response.

Understanding the code

Defining your environment configuration with the `Dockerfile`

Key Point: Use your `Dockerfile` to customize your environment with libraries, utilities, environment variables, configuration files, and startup processes. This is the container-based approach for "server configuration".

Your `Dockerfile` is specific to the language and base operating environment, such as Ubuntu, that your service will use.

The [Build and Deploy Quickstart](#) (<https://cloud.google.com/run/docs/quickstarts/build-and-deploy>) shows various `Dockerfiles` that can be used as a starting point to build a `Dockerfile` for other services.

This service requires one or more additional system packages not available by default.

1. Open the `Dockerfile` in an editor.
2. Look for a `Dockerfile RUN` (<https://docs.docker.com/engine/reference/builder/#run>) statement. This statement allows running arbitrary shell commands to modify the environment. If the

Dockerfile has multiple stages, identified by finding multiple **FROM** statements, it will be found in the last stage.

The specific packages required and the mechanism to install them varies by the operating system declared inside the container.

To get instructions for your operating system or base image, click the appropriate tab.

The screenshot shows a GitHub repository page for a Dockerfile. At the top, there are two tabs: 'DEBIAN/UBUNTU' (which is selected and underlined) and 'ALPINE'. Below the tabs, the Dockerfile content is displayed in a blue box. The content includes a link to the Dockerfile on GitHub: [run/system_package/Dockerfile](https://github.com/GoogleCloudPlatform/golang-samples/blob/master/run/system_package/Dockerfile) (https://github.com/GoogleCloudPlatform/golang-samples/blob/master/run/system_package/Dockerfile). Below this, the repository path is shown: 'GOOGLECLOUDPLATFORM/GOLANG-SAMPLES/BLOB/MASTER/RUN/SYSTEM_PACKAGE/DOCKERFILE'. On the right side of the code block, there is a 'FEEDBACK (#)' button. At the bottom of the code block, there is a terminal window showing the following Dockerfile snippet:

```
RUN apt-get update -y && apt-get install -y \
  graphviz \
  && apt-get clean
```

To determine the operating system of your container image, check the name in the **FROM** statement or a README associated with your base image. For example, if you extend from `node`, you can find documentation and the parent **Dockerfile** on [Docker Hub](http://hub.docker.com/_/node) (http://hub.docker.com/_/node).

3. Test your customization by building the image, using [docker build locally](https://cloud.google.com/run/docs/building/containers#building_locally_and_pushing_using_docker) (https://cloud.google.com/run/docs/building/containers#building_locally_and_pushing_using_docker) or [Cloud Build](#) ([#build-container](#)).

Handling incoming requests

The sample service uses parameters from the incoming HTTP request to invoke a system call that executes the appropriate `dot` utility command.

In the HTTP handler below, a graph description input parameter is extracted from the `dot` querystring variable.

Graph descriptions can include characters which must be [URL encoded](https://wikipedia.org/wiki/Query_string#URL_encoding) (https://wikipedia.org/wiki/Query_string#URL_encoding) for use in a querystring.

NODE.JSPYTHONGOJAVA

[run/system-package/app.js](https://github.com/GoogleCloudPlatform/nodejs-docs-samples/blob/master/run/system-package/app.js)
(<https://github.com/GoogleCloudPlatform/nodejs-docs-samples/blob/master/run/system-package/app.js>)

[/GOOGLECLOUDPLATFORM/NODEJS-DOCS-SAMPLES/BLOB/MASTER/RUN/SYSTEM-PACKAGE/APP.JS](https://github.com/GoogleCloudPlatform/nodejs-docs-samples/blob/master/run/system-package/app.js)

FEEDBACK (#)

```
app.get('/diagram.png', (req, res) => {
  try {
    const image = createDiagram(req.query.dot);
    res.setHeader('Content-Type', 'image/png');
    res.setHeader('Content-Length', image.length);
    res.setHeader('Cache-Control', 'public, max-age=86400');
    res.send(image);
  } catch (err) {
    console.error(`error: ${err.message}`);
    const errDetails = (err.stderr || err.message).toString();
    if (errDetails.includes('syntax')) {
      res.status(400).send(`Bad Request: ${err.message}`);
    } else {
      res.status(500).send('Internal Server Error');
    }
  }
});
```

You'll need to differentiate between internal server errors and invalid user input. This sample service returns an Internal Server Error for all dot command-line errors unless the error message contains the string `syntax`, which indicates a user input problem.

Generating a diagram

The core logic of diagram generation uses the dot command-line tool to process the graph description input parameter into a diagram in the PNG image format.

NODE.JSPYTHONGOJAVA

[run/system-package/app.js](https://github.com/GoogleCloudPlatform/nodejs-docs-samples/blob/master/run/system-package/app.js)[\(https://github.com/GoogleCloudPlatform/nodejs-docs-samples/blob/master/run/system-package/app.js\)](https://github.com/GoogleCloudPlatform/nodejs-docs-samples/blob/master/run/system-package/app.js)

/GOOGLECLOUDPLATFORM/NODEJS-DOCS-SAMPLES/BLOB/MASTER/RUN/SYSTEM-PACKAGE/APP.JS

FEEDBACK (#)

```
// Generate a diagram based on a graphviz DOT diagram description.
const createDiagram = dot => {
  if (!dot) {
    throw new Error('syntax: no graphviz definition provided');
  }

  // Adds a watermark to the dot graphic.
  const dotFlags = [
    '-Glabel="Made on Cloud Run"',
    '-Gfontsize=10',
    '-Glabeljust=right',
    '-Glabelloc=bottom',
    '-Gfontcolor=gray',
  ].join(' ');

  const image = execSync(`/usr/bin/dot ${dotFlags} -Tpng`, {
    input: dot,
  });
  return image;
};
```

Designing a secure service

Any vulnerabilities in the `dot` tool are potential vulnerabilities of the web service. You can mitigate this by using up-to-date versions of the `graphviz` package through re-building the container image on a regular basis.

If you extend the current sample to accept user input as command-line parameters, you should protect against [command-injection attacks](https://wikipedia.org/wiki/Code_injection#Shell_injection) (https://wikipedia.org/wiki/Code_injection#Shell_injection)

. Some of the ways to prevent injection attacks include:

- Mapping inputs to a dictionary of supported parameters
- Validating inputs match a range of known-safe values, perhaps using regular expressions

- Escaping inputs to ensure shell syntax is not evaluated

Shipping the code

To ship your code, you build with Cloud Build, and upload to Container Registry, and deploy to Cloud Run or Cloud Run for Anthos on Google Cloud:

1. Run the following command to build your container and publish on Container Registry.

```
gcloud builds submit --tag gcr.io/PROJECT-ID/graphviz
```

Where **PROJECT-ID** is your GCP project ID, and **graphviz** is the name you want to give your service.

Upon success, you will see a SUCCESS message containing the ID, creation time, and image name. The image is stored in Container Registry and can be re-used if desired.

2. Deploy using the following command:

```
gcloud run deploy graphviz-web --image gcr.io/PROJECT-ID/graphviz
```

Where **PROJECT-ID** is your GCP project ID, and **graphviz** is the name of the container from above and **graphviz-web** is the name of the service.

If deploying to Cloud Run, answer Y to the "allow unauthenticated" prompt. See [Managing Access](https://cloud.google.com/run/docs/securing/managing-access) (<https://cloud.google.com/run/docs/securing/managing-access>) for more details on IAM-based authentication.

Wait until the deployment is complete: this can take about half a minute. On success, the command line displays the service URL.

3. If you want to deploy a code update to the service, repeat the previous steps. Each deployment to a service creates a new revision and automatically starts serving traffic when ready.

Try it out

Try out your service by sending HTTP **POST** requests with DOT syntax descriptions in the request payload.

1. Send an HTTP request to your service.

Copy the URL into your browser URL bar and update [SERVICE_DOMAIN]:

```
https://SERVICE_DOMAIN/diagram.png?dot=digraph Run { rankdir=LR Code -> Build
```

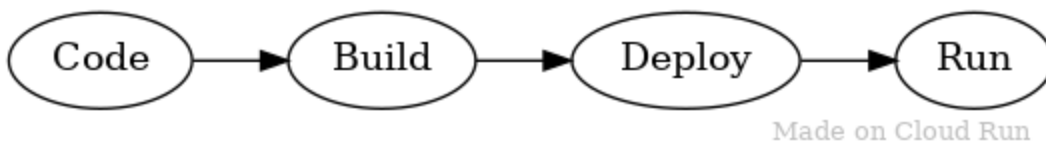
You can embed the diagram in a web page:

```
 Build -> Deploy
  > diagram.png"

```

2. Open the resulting diagram.png file in any application that supports PNG files, such as Chrome.

It should look like this:



Source: [DOT Description](#)

(https://github.com/GoogleCloudPlatform/golang-samples/blob/master/run/system_package/library/hello-cloud-run.dot)

You can explore a small collection of [ready-made diagram descriptions](#)

(https://github.com/GoogleCloudPlatform/golang-samples/tree/master/run/system_package/library).

1. Copy the contents of the selected .dot file

2. Paste it into a `curl` command similar to the above:

```
https://SERVICE_DOMAIN/diagram.png?dot=SELECTED DOTFILE CONTENTS
```



Cleaning up

If you created a new project for this tutorial, delete the project (#delete-project). If you used an existing project and wish to keep it without the changes added in this tutorial, delete resources created for the tutorial (#delete-resources).

Deleting the project

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/PROJ

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.

Deleting tutorial resources

1. Delete the Cloud Run service you deployed in this tutorial:

```
gcloud run services delete SERVICE-NAME
```



Where **SERVICE-NAME** is your chosen service name.

You can also delete Cloud Run services from the [Google Cloud Console](https://console.cloud.google.com/run) (<https://console.cloud.google.com/run>).

2. Remove the gcloud default configurations you added during tutorial setup.

If you use Cloud Run (fully managed), remove the region setting:

```
gcloud config unset run/region
```



If you use Cloud Run for Anthos on Google Cloud, remove the cluster configuration:

```
gcloud config unset run/cluster run/cluster  
gcloud config unset run/cluster run/cluster_location
```



3. Remove the project configuration:

```
gcloud config unset project
```



4. Delete other Google Cloud resources created in this tutorial:

- [Delete the container image](https://cloud.google.com/container-registry/docs/managing#deleting_images) (https://cloud.google.com/container-registry/docs/managing#deleting_images) named `gcr.io/<var>PROJECT-ID</var>/graphviz` from Container Registry.

What's next

- Experiment with your graphviz app:
 - Add support for other graphviz utilities which apply different algorithms to diagram generation.
 - Save diagrams to [Cloud Storage](https://cloud.google.com/storage) (<https://cloud.google.com/storage>). Do you want to save the image or the DOT syntax?
 - Implement content abuse protection with [Cloud Natural Language API](https://cloud.google.com/natural-language) (<https://cloud.google.com/natural-language>).

- See another example of a system package in the [Image Processing with Cloud Run tutorial](https://cloud.google.com/run/docs/tutorials/image-processing) (<https://cloud.google.com/run/docs/tutorials/image-processing>).
- Try out other Google Cloud Platform features for yourself. Have a look at our [tutorials](https://cloud.google.com/docs/tutorials) (<https://cloud.google.com/docs/tutorials>).

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 December 10, 2019.