

Série de Certificação de GCP: 3.3 Implantando e implementando recursos do App Engine e do Cloud Functions



Prashanta Paudel

26 de outubro de 2018 · 14 minutos de leitura

Google App Engine	
	
Developer(s)	Google
Initial release	April 7, 2008; 10 years ago
Stable release	1.9.63 / 27 February 2018
Written in	Python, Java, Go, PHP, Node.js
Operating system	linux (glibc), Windows
Platform	little-endian 32bits
Type	Web framework, cloud computing platform
License	Proprietary, LGPL
Website	cloud.google.com/appengine/

Referência: https://en.wikipedia.org/wiki/Google_App_Engine



App Engine

O Google App Engine é uma *plataforma como serviço (PaaS)* do Google. Isso significa que o Google fornecerá toda a infraestrutura e o software necessários para você escrever o código, e o Google lidará com o restante. O Google App Engine é a plataforma do Google como uma oferta de serviços que permite que desenvolvedores e empresas criem e executem aplicativos usando a infraestrutura avançada do Google. Também requer o uso da linguagem de consulta do Google e que o banco de dados usado é o Google Big Table. Os aplicativos devem obedecer a esses padrões, portanto, os aplicativos devem ser desenvolvidos com o GAE em mente ou modificados para atender aos requisitos. O Google fará o dimensionamento para cima e para baixo automaticamente para atender à necessidade e garantirá que seu código atenda os usuários.



Você pagará apenas pelos recursos que você usa e não mais.

O mecanismo de aplicativos é perfeito para empresas que não querem se preocupar com a infraestrutura subjacente e só se preocupam em disponibilizar o serviço o tempo todo.

Para uma startup, esta é uma solução perfeita.

O App Engine geralmente é usado para aplicativos da Web, em vez de aplicativos para dispositivos móveis.

O App Engine tem serviços integrados, como balanceamento de carga, verificações de integridade e registro de aplicativos, o que é muito útil para a administração de aplicativos.

O App Engine oferece várias opções de armazenamento, como cloud SQL e NoSQL, por meio de APIs.

Os aplicativos são colocados em sandbox e executados em vários servidores. O App Engine oferece dimensionamento automático para aplicativos da web, o que significa que, conforme o número de solicitações aumenta para um aplicativo, o Google App Engine aloca automaticamente mais recursos para o aplicativo da Web para lidar com tráfego adicional.

This embedded content is from a site that does not comply with the Do Not Track (DNT) setting now enabled on your browser.

Please note, if you click through and view it anyway, you may be tracked by the website hosting the embed.

Learn More about Medium's DNT policy

This embedded content is from a site that does not comply with the Do Not Track (DNT) setting now enabled on your browser.


Please note, if you click through and view it anyway, you may be tracked by the website hosting the embed.

Learn More about Medium's DNT policy

Fully managed serverless application platform

Build and deploy applications on a fully managed platform. Scale your applications seamlessly from zero to planet scale without having to worry about managing the underlying infrastructure. With zero server management and zero configuration deployments, developers can focus only on building great applications without the management overhead. App Engine enables developers to stay more productive and agile by supporting popular development languages and a wide range of developer tools.

Referência: <https://cloud.google.com/appengine/>




Open & familiar languages and tools


Quickly build and deploy applications using many of the popular languages like Java, PHP, Node.js, Python, C#, .Net, Ruby and Go or bring your own language runtimes and frameworks if you choose. Get started quickly with zero configuration deployments in App Engine. Manage resources from the command line, debug source code in production and run API backends easily using industry leading tools such as Cloud SDK, Cloud Source Repositories, IntelliJ IDEA, Visual Studio and Powershell.

Just add code

Focus just on writing code, without the worry of managing the underlying infrastructure. With capabilities such as automatic scaling-up and scaling-down of your application between zero and planet scale, fully managed patching and management of your servers, you can offload all your infrastructure concerns to Google. Protect your applications from security threats using App Engine firewall capabilities, Identity and Access Management (IAM) rules, and managed SSL/ TLS certificates.



Referência: <https://cloud.google.com/appengine/>



Pay only for what you use

Choose to run your applications in a serverless environment without the worry of over or under provisioning. App Engine automatically scales depending on your application traffic and consumes resources only when your code is running. You will only need to pay for the resources you consume.

Referência: <https://cloud.google.com/appengine/>

APP ENGINE FEATURES
A powerful platform to build apps and scale automatically

<p>Popular Languages Build your application in Node.js, Java, Ruby, C#, Go, Python, or PHP—or bring your own language runtime</p> <p>Open & Flexible Custom runtimes allow you to bring any library and framework to App Engine by supplying a Docker container</p> <p>Fully Managed A fully managed environment lets you focus on code while App Engine manages infrastructure concerns</p> <p>Monitoring, Logging & Diagnostics Google Stackdriver gives you powerful application diagnostics to debug and monitor the health and performance of your app</p>	<p>Application Versioning Easily host different versions of your app, easily create development, test, staging, and production environments</p> <p>Traffic Splitting Route incoming requests to different app versions, A/B test and do incremental feature rollouts</p> <p>Application Security Help safeguard your application by defining access rules with App Engine firewall and leverage managed SSL/TLS certificates* by default on your custom domain at no additional cost</p> <p>Services Ecosystem Tap a growing ecosystem of GCP services from your app including an excellent suite of cloud developer tools</p>
---	--

Referência: <https://cloud.google.com/appengine/>

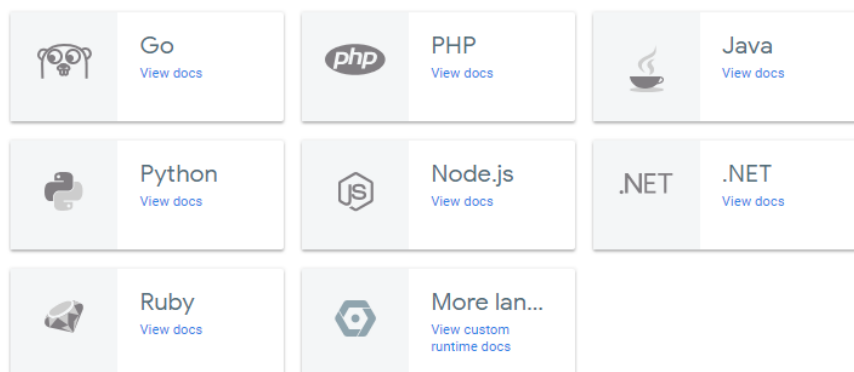
Idiomas Suportados

O ambiente padrão do App Engine oferece suporte para idiomas



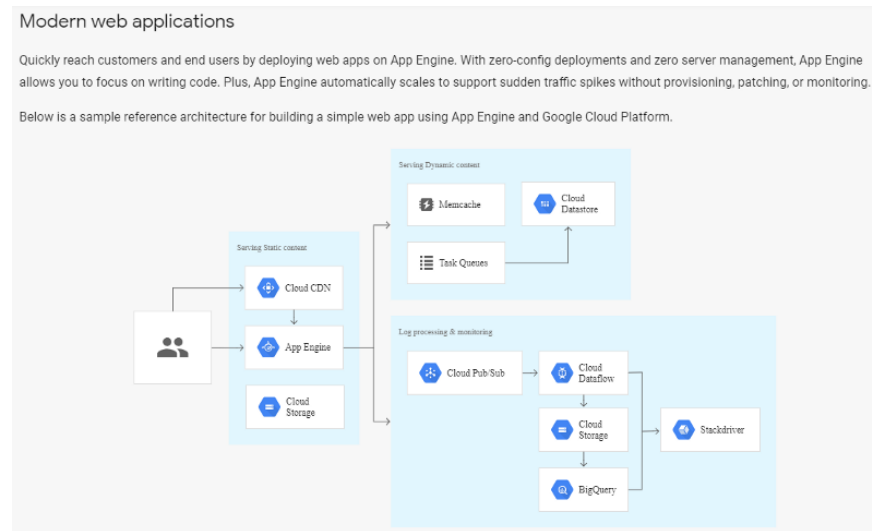
referência: <https://cloud.google.com/appengine/>

- Java
- Python
- PHP
- Vai
- C #
- Rubi
- Node.js
- .Líquido



O ambiente flexível também oferece outros tempos de execução de linguagem. Comece rapidamente com implantações de configuração

zero no Google App Engine. Gerencie recursos a partir da linha de comando, depure o código-fonte em produção e execute back-ends de API facilmente usando ferramentas líderes do setor, como Cloud SDK, Repositórios de Fonte em Nuvem, IntelliJ IDEA, Visual Studio e Powershell.



Referência: <https://cloud.google.com/appengine/>

O ambiente padrão do Google App Engine

O ambiente padrão do App Engine é baseado em instâncias de contêiner em execução na infraestrutura do Google. Os contêineres são pré-configurados com um dos vários tempos de execução disponíveis.

O ambiente padrão do App Engine facilita a criação e implantação de um aplicativo que é executado de forma confiável mesmo sob carga pesada e com grandes quantidades de dados.

Os aplicativos são executados em um ambiente seguro e em área restrita, permitindo que o ambiente padrão do App Engine distribua solicitações em vários servidores e dimensionando servidores para atender às demandas de tráfego. Seu aplicativo é executado dentro de seu próprio ambiente seguro e confiável, independente do hardware, do sistema operacional ou da localização física do servidor.

Aulas de instância

Cada aplicativo em execução no ambiente padrão possui uma *classe de instância*, que determina seus recursos de computação e preço. Esta tabela resume os limites de memória e CPU das várias classes de instância. Veja a página de preços para informações de faturamento.

Instance Class	Memory Limit	CPU Limit	Supported Scaling Types
F1 (default)	128 MB	600 MHz	automatic
F2	256 MB	1.2 GHz	automatic
F4	512 MB	2.4 GHz	automatic
F4_1G	1024 MB	2.4 GHz	automatic
B1	128 MB	600 MHz	manual, basic
B2 (default)	256 MB	1.2 GHz	manual, basic
B4	512 MB	2.4 GHz	manual, basic
B4_1G	1024 MB	2.4 GHz	manual, basic
B8	1024 MB	4.8 GHz	manual, basic

Referência: <https://cloud.google.com/appengine/docs/standard/>

Cotas e limites

O ambiente padrão do Google App Engine oferece 1 GB de armazenamento de dados e tráfego de graça, o que pode ser aumentado com a ativação de aplicativos pagos. No entanto, alguns recursos impõem limites não relacionados às cotas para proteger a estabilidade do sistema. Para mais detalhes sobre cotas, incluindo como você pode editá-las para atender às suas necessidades, consulte a página [Cotas](#).

Idiomas e tempos de execução padrão do ambiente

- Java 7 runtime
- Java 8 Runtime
- Tempo de execução em Python
- Tempo de execução PHP
- Ir tempo de execução

Ambiente flexível do App Engine

O App Engine permite que os desenvolvedores se concentrem em fazer o que fazem melhor, escrevendo código. Com base no Google Compute Engine, o ambiente flexível do App Engine dimensiona automaticamente seu aplicativo para cima e para baixo enquanto equilibra a carga. Microservices, autorização, bancos de dados SQL e NoSQL, divisão de tráfego, criação de log, controle de versão, verificação de segurança e redes de distribuição de conteúdo são todos suportados nativamente. Além disso, o ambiente flexível do App Engine permite que você personalize o tempo de execução e até mesmo o sistema operacional de sua máquina virtual usando Dockerfiles. Aprenda sobre as diferenças entre o ambiente padrão e o ambiente flexível.

- **Tempos de execução** - O ambiente flexível inclui suporte nativo para Java 8 (sem estrutura de veiculação da Web), Eclipse Jetty 9, Python 2.7 e Python 3.6, Node.js, Ruby, PHP, núcleo .NET e Go. Os desenvolvedores podem personalizar esses tempos de execução ou fornecer seu próprio tempo de execução, fornecendo uma imagem do Docker personalizada ou Dockerfile da comunidade de código aberto.
- **Personalização de infraestrutura** - como as instâncias de VM no ambiente flexível são máquinas virtuais do Google Compute Engine, você pode aproveitar as bibliotecas personalizadas, usar o SSH para depuração e implantar seus próprios contêineres do Docker.
- **Desempenho** - aproveite uma grande variedade de configurações de CPU e memória. Você pode especificar quanto de CPU e memória cada instância de seu aplicativo precisa e o ambiente flexível fornecerá a infraestrutura necessária para você.

O App Engine gerencia suas máquinas virtuais, garantindo que:

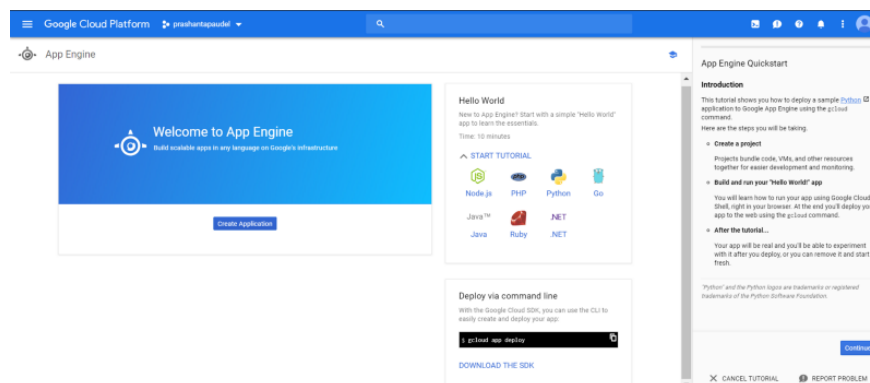
- As instâncias são verificadas pela saúde, curadas conforme necessário e co-localizadas com outros serviços dentro do projeto.
- Atualizações críticas e compatíveis com versões anteriores são aplicadas automaticamente ao sistema operacional subjacente.

- As instâncias de VM são localizadas automaticamente por região geográfica de acordo com as configurações do seu projeto. Os serviços de gerenciamento do Google garantem que todas as instâncias de VM de um projeto estejam localizadas para um ótimo desempenho.
- As instâncias de VM são reiniciadas semanalmente. Durante as reinicializações, os serviços de gerenciamento do Google aplicarão qualquer sistema operacional e atualizações de segurança necessárias.
- Você sempre tem acesso root às instâncias de VM do Compute Engine. O acesso SSH às instâncias de VM no ambiente flexível é desativado por padrão. Se preferir, você pode ativar o acesso root às instâncias de VM do seu aplicativo.

Implantando um aplicativo no App Engine

Veremos como implantar um aplicativo simples no App Engine

Primeiro de tudo, acesse o painel de controle do Google Cloud e depois calcule o mecanismo. Dentro do mecanismo de computação, você verá o App Engine. Clique no App Engine



Painel do App Engine

Depois de clicar no App Engine, você verá o painel de controle do Google App Engine

Etapa 1: criar um projeto

Em quase todas as tarefas do GCP, você deve ter pelo menos um projeto para começar.

Então, primeiro crie um projeto, o que já fiz.

App Engine Quickstart

Project setup

To deploy an application you need to first create a project. Google Cloud Platform organizes resources into projects. This allows you to collect all the related resources for a single application in one place.

Create project Select an existing project

prashantapaudel

prashantapaudel-219410

Back **Confirm Project**

selecionar projeto

Etapa 2: copiar / clonar o código de fora para o App Engine

Lembre-se de que o App Engine exige o uso de um dos idiomas suportados pelo App Engine.

Aqui, eu seleciono **Python** para o meu projeto de teste.

Usando o Google Cloud Shell

O Cloud Shell é uma ferramenta de linha de comando integrada para o console. Nós vamos usar o Cloud Shell para implantar nosso aplicativo.

Abra o Google Cloud Shell

Abra o Cloud Shell clicando no ícone do shell na barra de navegação na parte superior do console.

Clone o código de amostra

Use o Cloud Shell para clonar e navegar para o código "Hello World". O código de amostra é clonado do seu repositório de projetos para o Cloud Shell.

Nota: Se o diretório já existir, remova os arquivos anteriores antes de clonar.

No Cloud Shell, digite:

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

Em seguida, mude para o diretório do tutorial:

```
cd python-docs-samples/appengine/standard_python37/hello_world
```

Dê uma olhada na estrutura do aplicativo de amostra - a `website` pasta contém o conteúdo do site e `app.yaml` é o arquivo de configuração do aplicativo.

- O conteúdo do seu site deve ir dentro da `website` pasta, e sua página de destino deve ser chamada `index.html`, mas, além disso, ela pode ser de qualquer forma que você desejar.
- O `app.yaml` arquivo é um arquivo de configuração que informa ao App Engine como mapear URLs para seus arquivos estáticos. Você não precisa editá-lo.

Etapa 3: configurando sua implantação

Você está agora no diretório principal para o código de amostra. Vamos ver os arquivos que configuram seu aplicativo.

Explorando o aplicativo

Digite o seguinte comando para visualizar o código do seu aplicativo:

```
cat main.py
```

O aplicativo é um aplicativo Python simples que usa o framework da Web Flask. Este aplicativo Python responde a uma solicitação com um cabeçalho HTTP e a mensagem `Hello World!`.

Explorando sua configuração

O Google App Engine usa arquivos YAML para especificar a configuração de uma implantação. `app.yaml` Os arquivos contêm informações sobre seu aplicativo, como o ambiente de tempo de execução, variáveis de ambiente e muito mais.

Digite o seguinte comando para visualizar seu arquivo de configuração:

```
cat app.yaml
```

Este arquivo contém a quantidade mínima de configuração necessária para um aplicativo Python 3: o `runtime` campo especifica o `python37` ambiente de tempo de execução.

A sintaxe deste arquivo é YAML. Para uma lista completa de opções de configuração, consulte a [app.yaml](#) referência.

Etapa 4: testando seu aplicativo

O Cloud Shell permite que você teste seu aplicativo antes de implantá-lo para garantir que ele esteja sendo executado conforme o esperado, assim como a depuração em sua máquina local.

Para testar seu aplicativo, primeiro crie um ambiente virtual isolado. Isso garante que seu aplicativo não interfira em outros aplicativos Python que possam estar disponíveis em seu sistema.

```
virtualenv --python python3 \  
~/envs/hello_world
```

Ative seu ambiente virtual recém-criado:

```
source \  
~/envs/hello_world/bin/activate
```

Use `pip` para instalar dependências do projeto. Este aplicativo "Hello World" depende do microframework do Flask:

```
pip install -r requirements.txt
```

Por fim, execute seu aplicativo no Cloud Shell usando o servidor de desenvolvimento do Flask:

```
python main.py
```

Visualize seu aplicativo com "visualização da Web"

Seu aplicativo está sendo executado no Cloud Shell. Você pode acessar o aplicativo usando a visualização da Web

para se conectar ao port8080.

Terminando a instância de visualização

Encerre a instância do aplicativo pressionando `ctrl+c` o Cloud Shell.

Etapa 5: implantação no Google App Engine

Para implantar seu aplicativo, você precisa criar um aplicativo em uma região:

```
gcloud app create
```

Nota: Se você já criou um aplicativo, pode pular esta etapa.

Implantando com o Cloud Shell

Você pode usar o Cloud Shell para implantar seu aplicativo. Para implantar seu aplicativo, digite:

```
gcloud app deploy app.yaml --project \
prashantapaudel-219410
```

Visite seu aplicativo

Parabéns! Seu aplicativo foi implantado. O URL padrão do seu aplicativo é prashantapaudel-219410.appspot.com . Clique no URL

para visitá-lo.

=====

=====

Veja também: [https://developer.mozilla.org/pt-BR/docs/Learn/Common_questions/How do you host your website on Google App Engine](https://developer.mozilla.org/pt-BR/docs/Learn/Common_questions/How_do_you_host_your_website_on_Google_App_Engine)

Implantando seu serviço da Web

Use a `gcloud` ferramenta do Cloud SDK para implantar seu serviço da web no App Engine.

Embora essa versão inicial do serviço da Web não tenha o Cloud Datastore ou a autenticação do Firebase, você pode implantá-lo no App Engine neste estágio para testar e garantir que ele funcione conforme o esperado.

Antes de você começar

Se você concluiu todas as etapas anteriores deste guia, pule esta seção. Caso contrário, conclua um dos seguintes procedimentos:

- Comece por criar um aplicativo Python 3.7 e conclua todas as etapas que levam a este.
- Se você já tem um projeto do GCP, pode continuar fazendo o download de uma cópia do serviço da web:
 1. Faça o download do repositório do aplicativo de amostra usando o Git
 - `git clone https://github.com/GoogleCloudPlatform/python-docs-samples`
 1. Como alternativa, você pode baixar a amostra como um arquivo zip e extraí-lo.
 2. Navegue até o diretório que contém uma cópia dos arquivos da etapa anterior

- `cd python-docs-samples/appengine/standard_python37/building-an-app/building-an-app-1`

Implantando seu serviço

Para implantar seu serviço da web, você executa o `gcloud app deploy` comando no diretório raiz do seu projeto, onde o `app.yaml` arquivo está localizado:

```
gcloud app deploy
```

Toda vez que você implanta seu serviço da Web, uma nova versão desse aplicativo é criada no App Engine. Durante a implantação, uma imagem de contêiner é criada usando o serviço Cloud Build e, em seguida, uma cópia é carregada no Google Cloud Storage antes de ser executada no Google App Engine.

Para obter mais informações sobre como implantar no App Engine, consulte Testando e implantando seu aplicativo .

Nota: Os arquivos listados em seu `.gcloudignore` arquivo não são enviados ao App Engine durante a implantação.

Visualizando seu serviço

Para iniciar rapidamente seu navegador e acessar seu serviço da web em `https://GCP_PROJECT_ID.appspot.com` , digite o seguinte comando:

```
gcloud app browse
```

Dica: Se você quiser alterar o URL do seu serviço da web para algo diferente do `appspot.com` URL padrão , poderá adicionar um domínio personalizado .

Gerenciando serviços e versões

Você acaba de implantar uma versão do serviço da web no App Engine. Cada vez que você implanta uma versão do seu código, essa versão é criada em um serviço. A implantação inicial no App Engine deve ser criada no `default` serviço, mas, para implantações subsequentes, você pode especificar o nome do seu serviço no `app.yaml` arquivo .

Você pode atualizar um serviço a qualquer momento executando o `gcloud app deploy` comando e implantando novas versões para esse serviço. Cada vez que você atualiza um serviço, o tráfego é roteado automaticamente para a última versão implementada. No entanto, você pode incluir `gcloud` señalizadores para alterar o comportamento do comando de implantação.

Use o Console do GCP para gerenciar e visualizar os serviços e as versões que você implanta no App Engine:

- Use o Console do GCP para visualizar seus serviços do App Engine:
- [IR PARA A PÁGINA DE SERVIÇOS](#)
- Use o Console do GCP para visualizar suas versões:
- [IR PARA A PÁGINA DE VERSÕES](#)

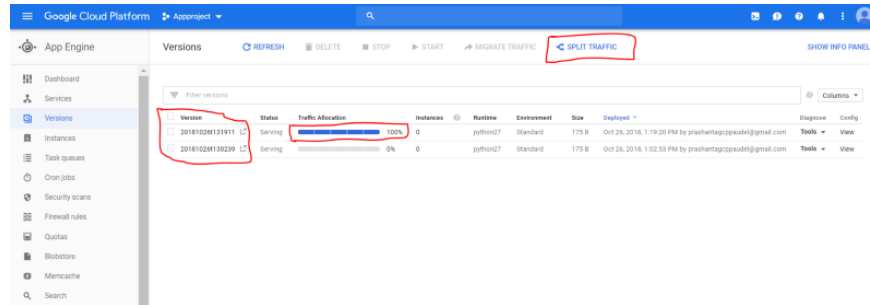
Para mais informações sobre o padrão de design de vários serviços, consulte Visão geral do App Engine. Para saber como enviar solicitações para serviços e versões específicos, consulte Dividindo o tráfego.

Divisão de tráfego em várias versões

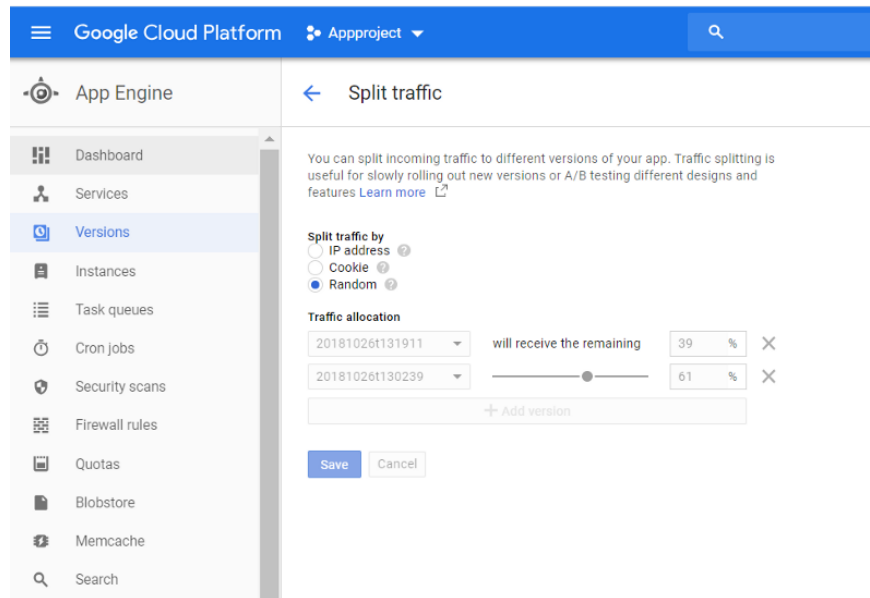
Quando tiver especificado duas ou mais versões para divisão, você deverá escolher se deseja dividir o tráfego usando um endereço IP ou um cookie HTTP. É mais fácil configurar uma divisão de endereço IP, mas uma divisão de cookie é mais precisa. Para mais informações, consulte Divisão de endereço IP e Divisão de cookie.

Quando você atualiza ou adiciona outra versão do aplicativo no Google App Engine, você verá duas versões na guia Versão do App Engine

Agora selecione estas duas versões e clique em SPLIT TRAFFIC no topo da página da versão

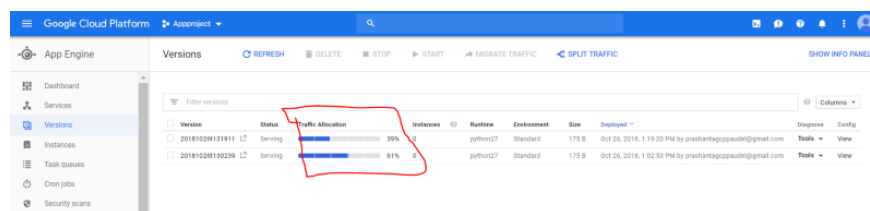


Página de versão



Dividindo o tráfego por aleatório

Agora você pode ver as diferentes versões que servem



duas versões

Funções do Google Cloud



Funções de nuvem

Plataforma de computação sem servidor orientada a eventos

- A maneira mais simples de executar seu código na nuvem
- Automaticamente escala, altamente disponível e tolerante a falhas
- Nenhum servidor para provisionar, gerenciar, corrigir ou atualizar
- Pague apenas enquanto seu código é executado
- Conecta e estende os serviços em nuvem

This embedded content is from a site that does not comply with the Do Not Track (DNT) setting now enabled on your browser.

Please note, if you click through and view it anyway, you may be tracked by the website hosting the embed.

[Learn More about Medium's DNT policy](#)

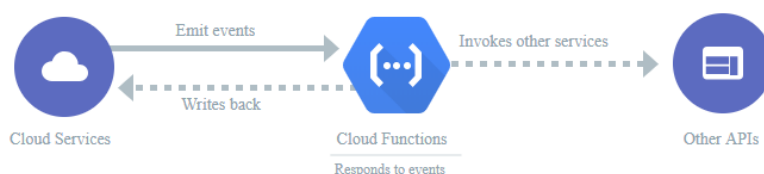
As funções da nuvem do Google (GCF) são uma nova oferta que permite aos desenvolvedores executar uma função específica com base em eventos do acionador. Isso é ainda mais alto em nível do que o App Engine em termos de infraestrutura independente.

O GCP é o principal produto que representa o conceito sem servidor no Google Cloud Platform.

As funções de nuvem permitem funções específicas que o desenvolvedor gravou para serem acionadas por algum evento.

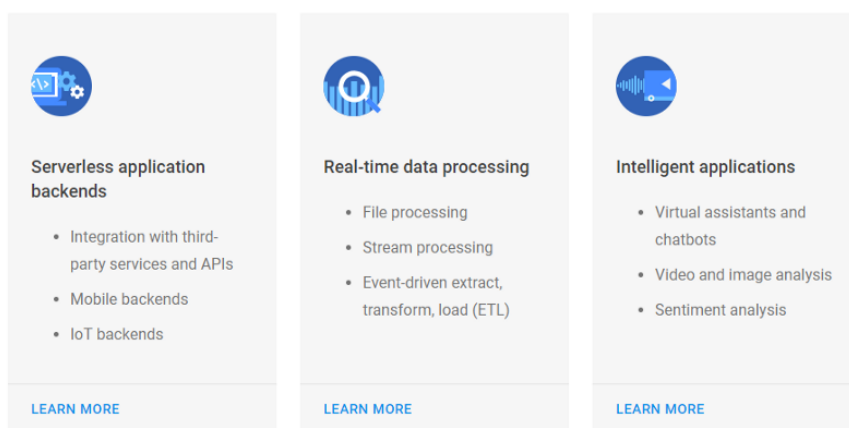
Por exemplo, quando um sensor recebe um sinal, algumas funções podem ser chamadas para executar alguma tarefa. Quando alguém se inscrever para o e-mail mesmo será enviado para alguém.

How it works



Referência: <https://cloud.google.com/functions/>

What you can build with Cloud Functions



Referência: <https://cloud.google.com/functions/>

Microservices over monoliths

Your development agility comes from building systems composed of small, independent units of functionality focused on doing one thing well. Cloud Functions lets you build and deploy services at the level of a single function, not at the level of entire applications, containers, or VMs.



Referência: <https://cloud.google.com/functions/>

Key Features

[ALL FEATURES](#)



No server management



Scales automatically



Pay only while your code runs



Runs code in response to events



Open and familiar



Connects and extends cloud services

Referência: <https://cloud.google.com/functions/>

Supported event sources

Cloud Pub/Sub

You can invoke Cloud Functions in response to messages published to Cloud Pub/Sub topics. Cloud Pub/Sub is a globally distributed message bus that automatically scales as you need it and provides a foundation for building your own robust, global services.



[TUTORIAL](#)

Cloud Storage

You can invoke Cloud Functions in response to change notifications from Cloud Storage such as object addition (create), update (modify), or deletion.



[TUTORIAL](#)

HTTP

You can invoke Cloud Functions directly over HTTP(S). Each function is given a dedicated domain and a dynamically generated SSL/TLS certificate for secure communication. Function execution result is returned in response to an HTTP request.



[TUTORIAL](#)

Stackdriver Logging

You can invoke Cloud Functions in response to log changes in Stackdriver Logging. Stackdriver Logging allows you to store, search, analyze, monitor, and alert on log data and events from Google Cloud Platform and Amazon Web Services (AWS).



[TUTORIAL](#)

Firebase

You can invoke Cloud Functions in response to mobile-related events from Firebase such as changes to data in the Realtime Database, new user sign-ups via Auth, and conversion events in Analytics.

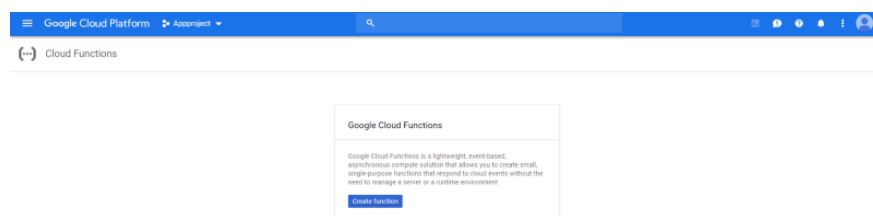


[TUTORIAL](#)

Fontes de gatilho

Implantando uma nuvem que recebe eventos do Google Cloud

Primeiro painel Goto Cloud Functions



Painel de funções da nuvem

Clique em Create Functions

Google Cloud Platform

Appproject

Cloud Functions

Create function

Name

testfunctionstorage

Memory allocated

256 MB

Trigger

Cloud Storage

Event Type

Finalize/Create

Bucket

bucket

Browse

Source code

☒ Inline editor

☐ ZIP upload

☐ ZIP from Cloud Storage

☐ Cloud Source repository

Runtime

Node.js 6

index.js

package.json

```
1 /**
2  * Triggered from a change to a Cloud Storage bucket.
3  *
4  * @param {!Object} event Event payload and metadata.
5  * @param {!Function} callback Callback function to signal compl
6  */
7 exports.helloGCS = (event, callback) => {
8   console.log('Processing file: ${event.data.name}');
9   callback();
10 };
11
```

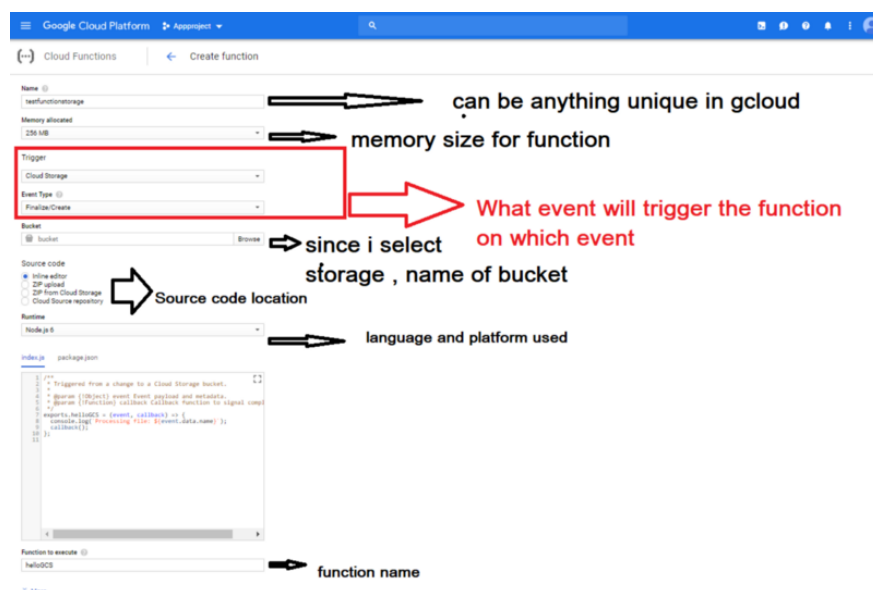
Function to execute

helloGCS

More

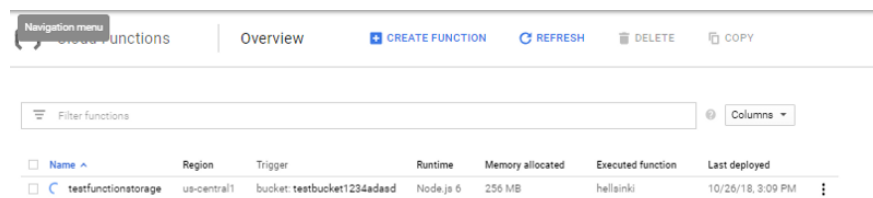
Criar função

Você terá várias opções para selecionar

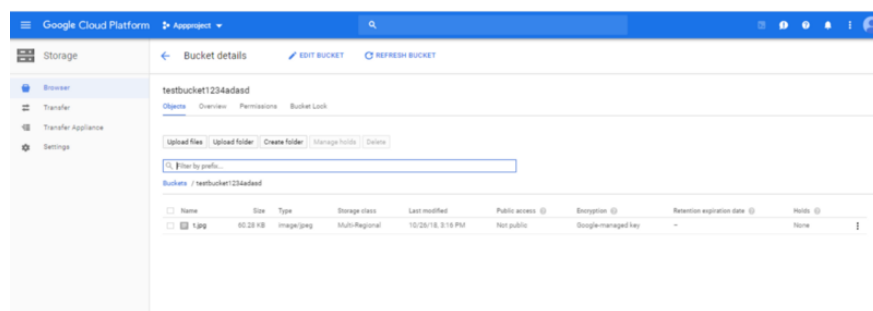


Opções na criação da função

Levará algum tempo para criar a função



Agora carregue o arquivo no balde



Após o upload, use o shell da nuvem ou o SDK para verificar a função disparada

```
Logs de funções $ gcloud são lidos --limit 50
```

```
D function-1 278457471527247 2018-10-26 12: 16: 45.259
Execução da função iniciada
I function-1 278457471527247 2018-10-26 12: 16: 45.269
Processando o arquivo: t.jpg.
D function-1 278457471527247 2018-10-26 12: 16: 45.355 A
execução da função levou 97 ms e terminou com o status: 'ok'
```

Armazenamento em Nuvem com o Tutorial de Função

Este tutorial simples demonstra como escrever, implantar e acionar uma função de nuvem em segundo plano com um acionador do Cloud Storage.

Objetivos

- Escreva e implemente uma função de nuvem em segundo plano.
- Acione a função fazendo o upload de um arquivo para o Cloud Storage.

Custos

Este tutorial usa componentes faturáveis do Cloud Platform, incluindo:

- Funções do Google Cloud
- Google Cloud Storage

Use a calculadora de preços para gerar uma estimativa de custo com base em seu uso projetado.

Os novos usuários do Cloud Platform podem ser qualificados para uma avaliação gratuita.

Preparando o aplicativo

1. Crie um intervalo do Cloud Storage para fazer upload de um arquivo de teste, no qual `YOUR_TRIGGER_BUCKET_NAME` é um nome de depósito exclusivo globalmente:

- `gsutil mb gs://YOUR_TRIGGER_BUCKET_NAME`

1. Clone o repositório de aplicativos de amostra em sua máquina local:

- NODE.JS 6
- NODE.JS 8 (BETA)
- PYTHON (BETA)
- `git clone https://github.com/GoogleCloudPlatform/python-docs-samples.git`

1. Alternativamente, você pode baixar o exemplo como um arquivo zip e extraí-lo.

2. Altere para o diretório que contém o código de amostra Cloud Functions:

- NODE.JS 6
- NODE.JS 8 (BETA)
- PYTHON (BETA)

```
cd python-docs-samples / functions / gcs /
```

Implantando e acionando a função

Atualmente, as funções do Cloud Storage são baseadas em notificações de Pub / Sub do Cloud Storage e suportam tipos de eventos semelhantes:

- finalizar
- excluir
- arquivo

- atualização de metadados

As seções a seguir descrevem como implantar e acionar uma função para cada um desses tipos de eventos.

Objeto Finalizar

Eventos de finalização de objeto são acionados quando uma “gravação” de um objeto do Cloud Storage é finalizada com sucesso. Em particular, isso significa que criar um novo objeto ou sobrescrever um objeto existente aciona esse evento. As operações de atualização de arquivamento e metadados são ignoradas por esse acionador.

Object Finalize: implantando a função

Dê uma olhada na função de amostra, que lida com eventos do Cloud Storage:

[funções / gcs / main.py](#)

[VISTA NO GITHUB](#)

```
def hello_gcs_generic (dados, contexto):
    """ Função em nuvem em segundo plano a ser acionada pelo
    Cloud Storage.
        Essa função genérica registra dados relevantes quando
        um arquivo é alterado.

    Args:
        data (dict): a carga útil do evento Cloud Functions.
        context (google.cloud.functions.Context): Metadados
        do evento de disparo.
    Retorna:
        Nenhum; a saída é gravada no Stackdriver Logging
    """

    print ('ID do evento: {}'. format (contexto.event_id))
    print ('Tipo de evento: {}'. format
(contexto.event_type))
    print ('Bucket: {}'. format (data ['bucket' ]))
    print ('Ficheiro: {}'. format (data ['name']))
    print ('Metageneration: {}'. format (data
['metageneration']))
    print ('Criado: {}'. format (data ['timeCreated']))
    print ('Atualizado: {}'. format (data ['updated']))
```

O Cloud Functions procura seu código em um arquivo chamado

```
main.py .
```

Para implantar a função, execute o seguinte comando no diretório em que o código de amostra está localizado:

PYTHON (BETA)

```
gcloud funções implantar hello_gcs_generic --runtime
python37 --trigger-resource YOUR_TRIGGER_BUCKET_NAME --
trigger-event google.storage.object.finalize
```

onde `YOUR_TRIGGER_BUCKET_NAME` está o nome do intervalo do Cloud Storage que aciona a função.

Object Finalize: acionando a função

Para acionar a função:

1. Crie um `gcf-test.txt` arquivo vazio no diretório em que o código de amostra está localizado.

2. Carregue o arquivo no Cloud Storage para acionar a função:

- `gsutil cp gcf-test.txt gs://YOUR_TRIGGER_BUCKET_NAME`

1. onde `YOUR_TRIGGER_BUCKET_NAME` está o nome do seu repositório do Cloud Storage no qual você fará o upload de um arquivo de teste.

2. Verifique os logs para se certificar de que as execuções foram concluídas:

- `gcloud functions logs read --limit 50`

```
ctions_v1beta2/worker.py", line 103, in call_user_function
    event_context.Context(**request_or_event.context))
    File "/User_code/main.py", line 4, in hello_gcs
    print(f"Processing file: {file['name']}".)
    KeyError: 'name'
D function-1 a50sh81gbidt 2018-10-26 11:21:52.663 Function execution took 73 ms, finished with status: 'crash'
D hello_gcs_generic 278319835660757 2018-10-26 11:56:41.747 Function execution started
I hello_gcs_generic 278319835660757 2018-10-26 11:56:41.780 Event ID: 278319835660757
I hello_gcs_generic 278319835660757 2018-10-26 11:56:41.780 Event type: google.storage.object.finalize
I hello_gcs_generic 278319835660757 2018-10-26 11:56:41.780 Bucket: testmybucket1234321
I hello_gcs_generic 278319835660757 2018-10-26 11:56:41.780 File: gcf-test.txt
I hello_gcs_generic 278319835660757 2018-10-26 11:56:41.780 Metageneration: 1
I hello_gcs_generic 278319835660757 2018-10-26 11:56:41.780 Created: 2018-10-26T11:56:40.824Z
I hello_gcs_generic 278319835660757 2018-10-26 11:56:41.780 Updated: 2018-10-26T11:56:40.824Z
D hello_gcs_generic 278319835660757 2018-10-26 11:56:41.782 Function execution took 36 ms, finished with status: 'ok'
```

Dessa forma, podemos usar o App Engine e o Cloud Functions.

