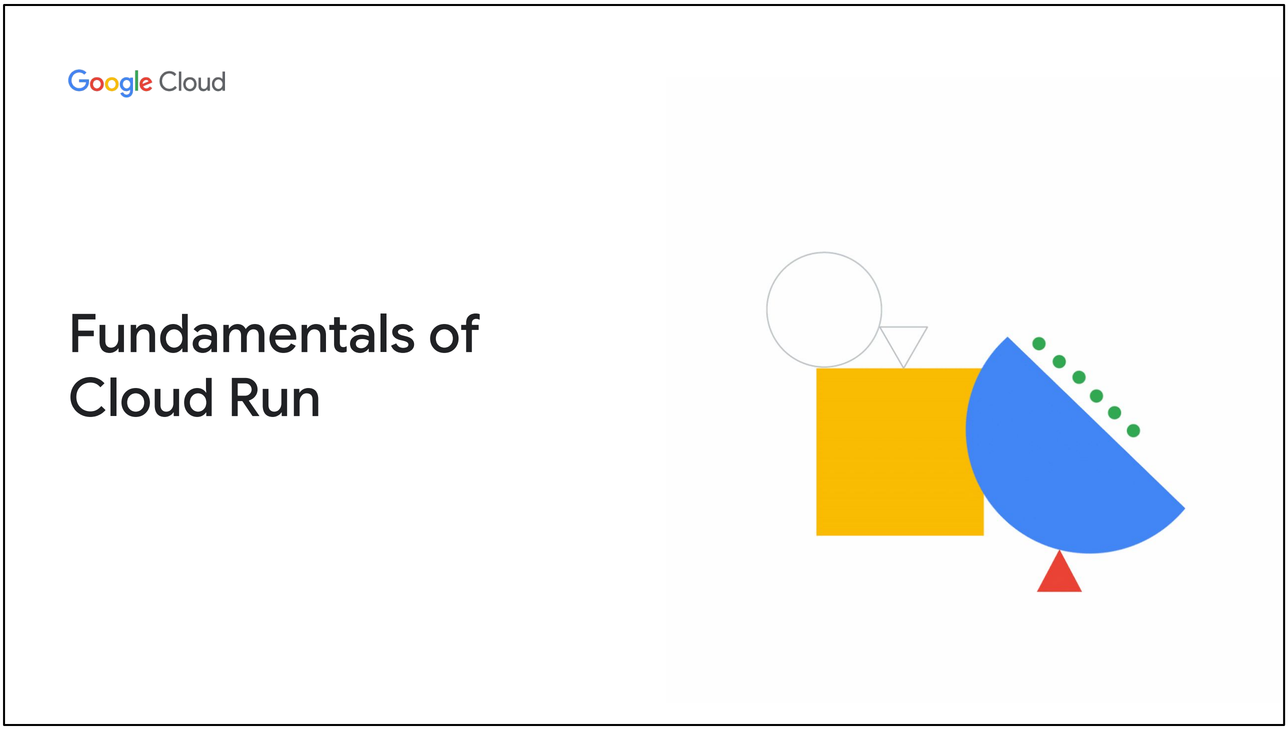
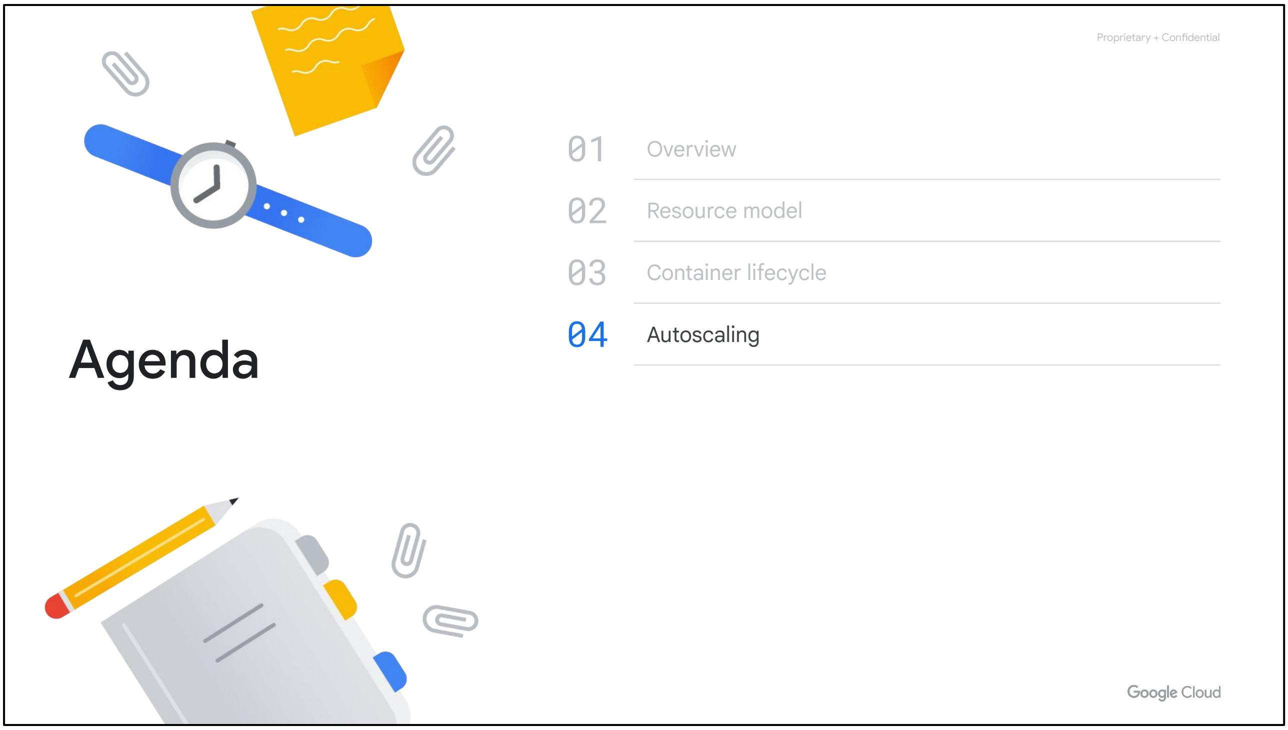


Google Cloud

Fundamentals of Cloud Run

In this module, we discuss the fundamentals of developing and running applications on the Cloud Run platform.



Proprietary + Confidential

Agenda

01 Overview

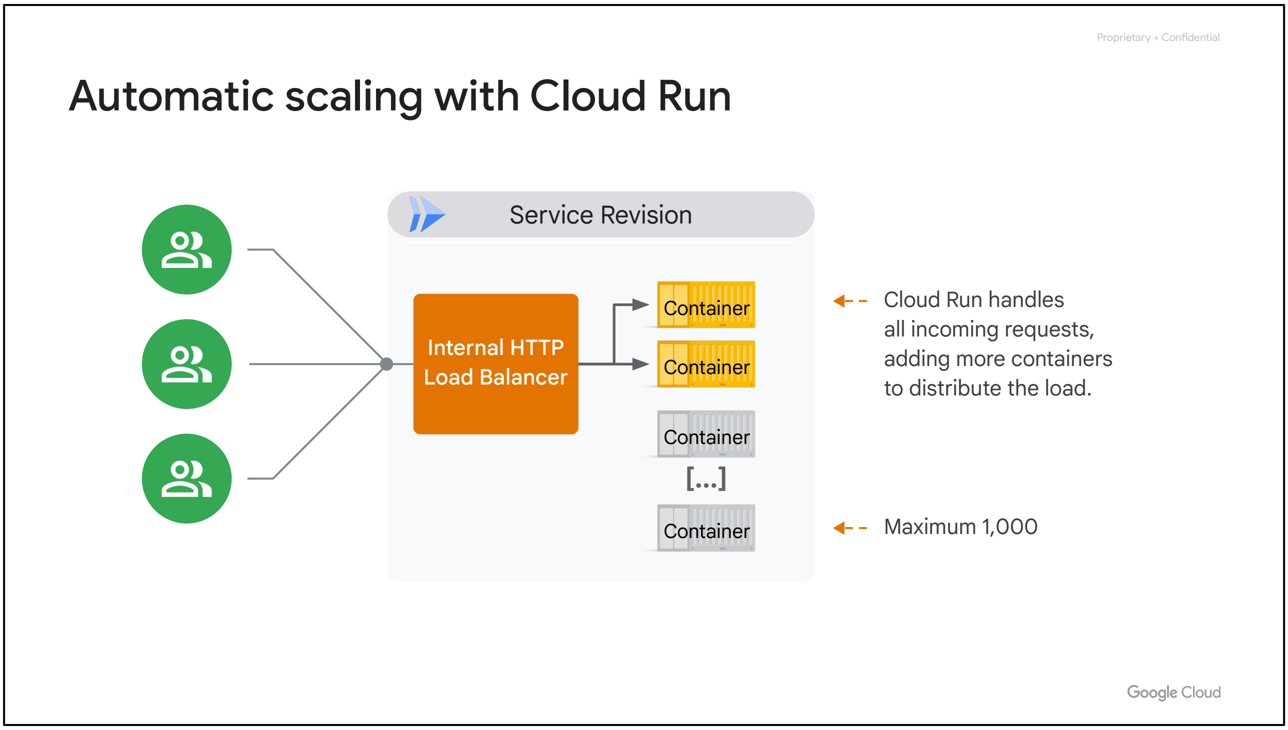
02 Resource model

03 Container lifecycle

04 Autoscaling

Google Cloud

Let’s discuss how Cloud Run autoscales your containers.



Proprietary + Confidential

Automatic scaling with Cloud Run

Service Revision

Internal HTTP Load Balancer

Container

Container

Container

[...]

Container

Cloud Run handles all incoming requests, adding more containers to distribute the load.

Maximum 1,000

Google Cloud

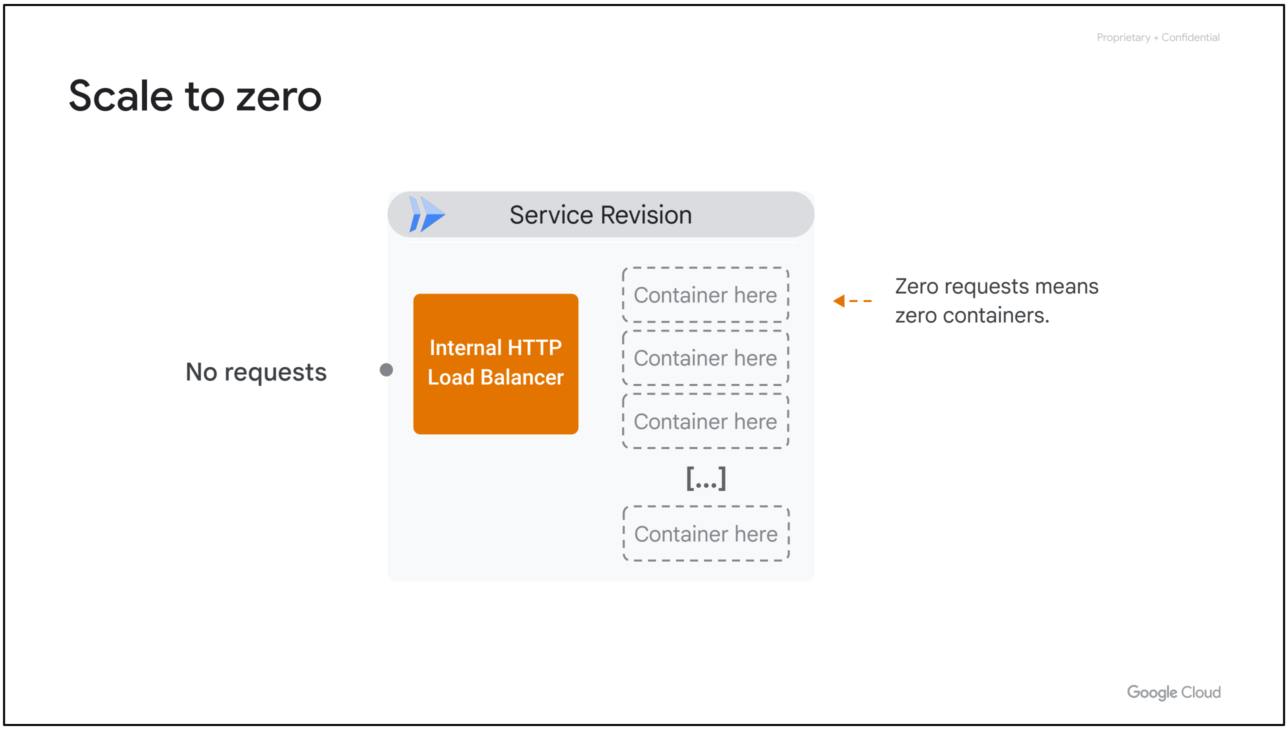
To maintain the capacity to handle incoming requests to your service, Cloud Run automatically increases the number of container instances of a service revision when necessary. This feature is known as autoscaling.

Requests to a service revision are distributed by an internal load balancer across the group of container instances.

- If all container instances are busy, Cloud Run adds additional instances.
- When demand decreases, Cloud Run stops sending traffic to some instances and shuts them down.
- A container instance can receive many requests at the same time. With the concurrency setting, you can set the maximum number of requests that can be sent in parallel to a given container instance.

In addition to the rate of incoming requests to your service, the number of container instances is affected by:

- The CPU utilization of existing instances when they are processing requests (with a target of 60% of utilization).
- The maximum concurrency setting.
- The minimum and maximum number of container instances setting.
  - The number of container instances in a Cloud Run service is limited to 1,000 instances by default. If you need more, you can submit a request for a quota increase.



Proprietary + Confidential

Scale to zero

Service Revision

No requests

Internal HTTP Load Balancer

Container here

Container here

Container here

[...]

Container here

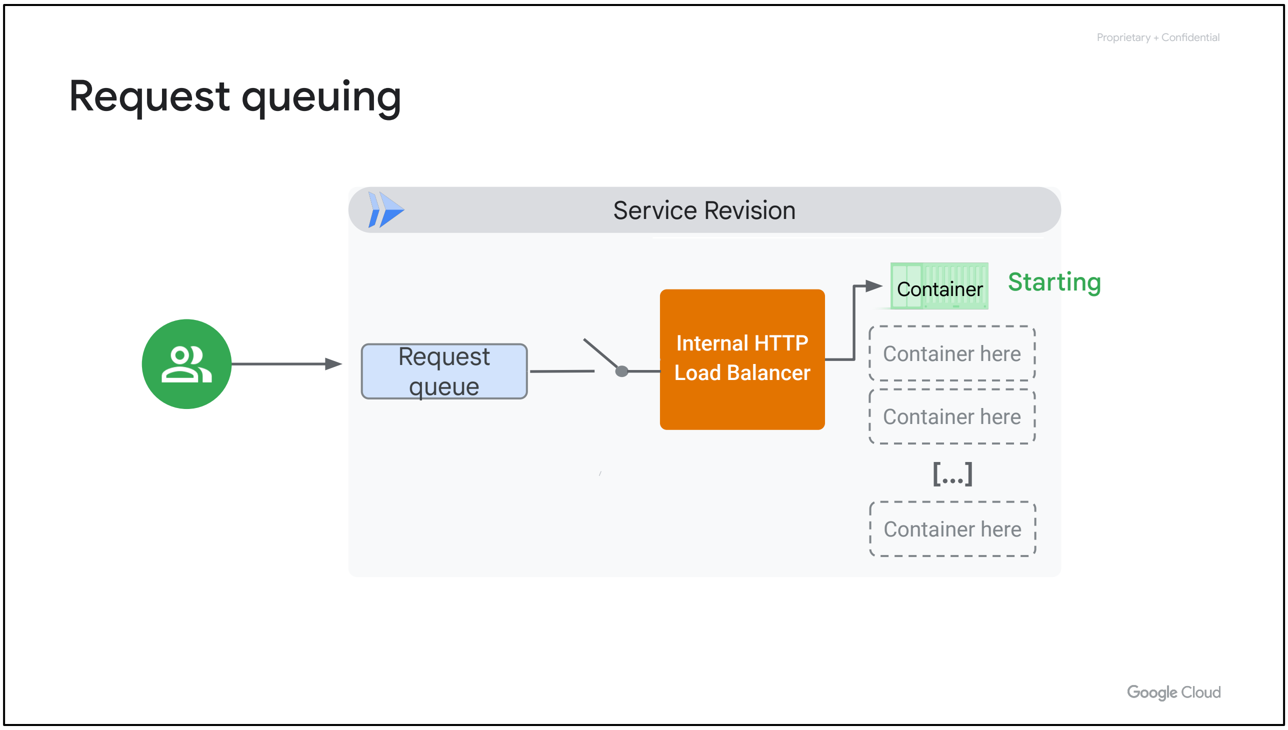
Zero requests means zero containers.

Google Cloud

If there are no incoming requests to your service, even the last remaining container instance will be shut down. This is commonly referred to as scale to zero.
This feature is attractive for economic reasons because you don’t pay for container instances that are idling.

A new container instance will start on demand when a new request is sent to your service.

To reduce the latency of your service when there are no instances, you can configure Cloud Run to keep a minimum number of container instances active.



Proprietary + Confidential

Request queuing

Service Revision

Request queue

Internal HTTP Load Balancer

Container

Starting

Container here

Container here

[...]

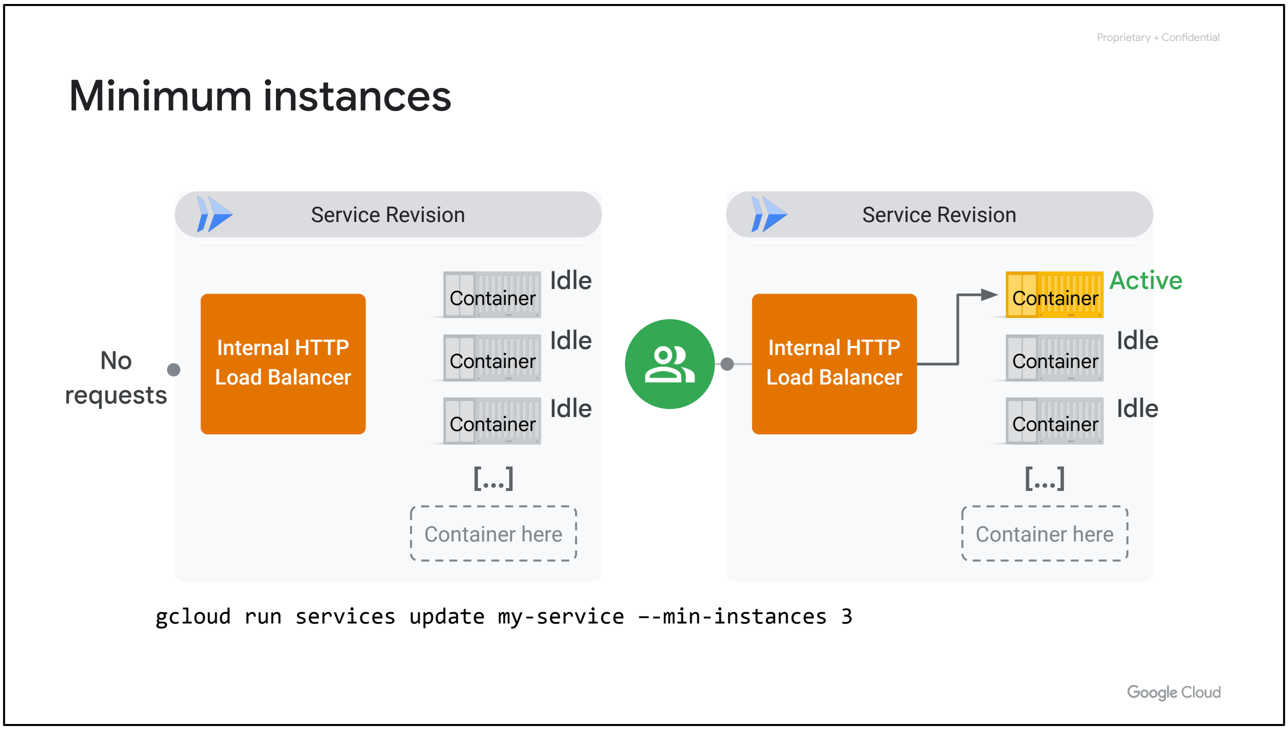
Container here

Google Cloud

The first few requests that come in after your service scaled to zero will queue while the first container instance starts. This is known as a ‘cold start’.

To reduce the latency of your service, you can configure Cloud Run to keep a minimum number of container instances idle.

These instances will be ready to process requests when they are received.



Proprietary + Confidential

Minimum instances

No requests

Service Revision

Internal HTTP Load Balancer

Container Idle

Container Idle

Container Idle

[...]

Container here

Service Revision

Internal HTTP Load Balancer

Container Active

Container Idle

Container Idle

[...]

Container here

```
gcloud run services update my-service --min-instances 3
```

Google Cloud

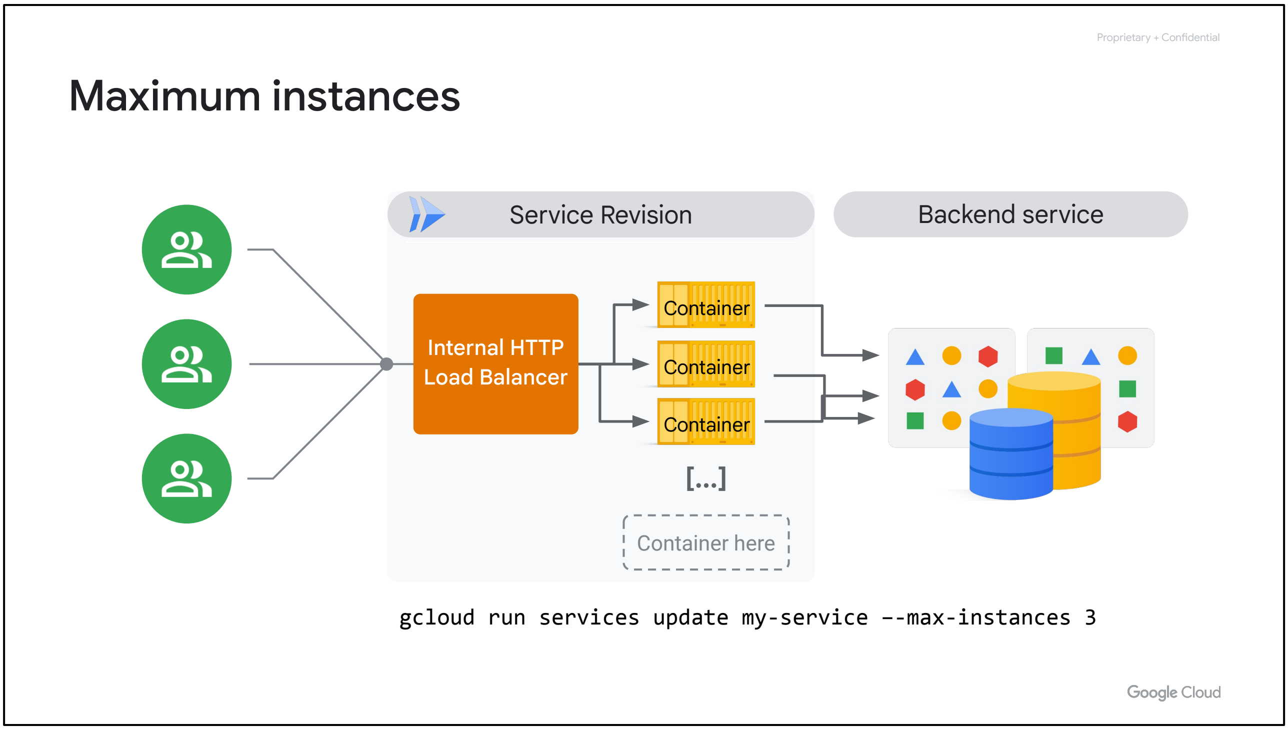
To change the default ‘scale to zero’ behavior, you specify a minimum number of container instances to be kept warm and ready to serve requests.

With minimum instances set, Cloud Run keeps at least the number of minimum instances running, even if they're not serving requests (idle).

As your service receives requests, the number of active instances might increase and the number of idle instances decrease.

Idle instances that are kept running using the minimum instances feature incur billing costs.

You can set or update the minimum instances configuration for your service in the Google Cloud console, using the gcloud CLI, a YAML configuration file, or Terraform.



Proprietary + Confidential

Maximum instances

```
gcloud run services update my-service --max-instances 3
```

Google Cloud

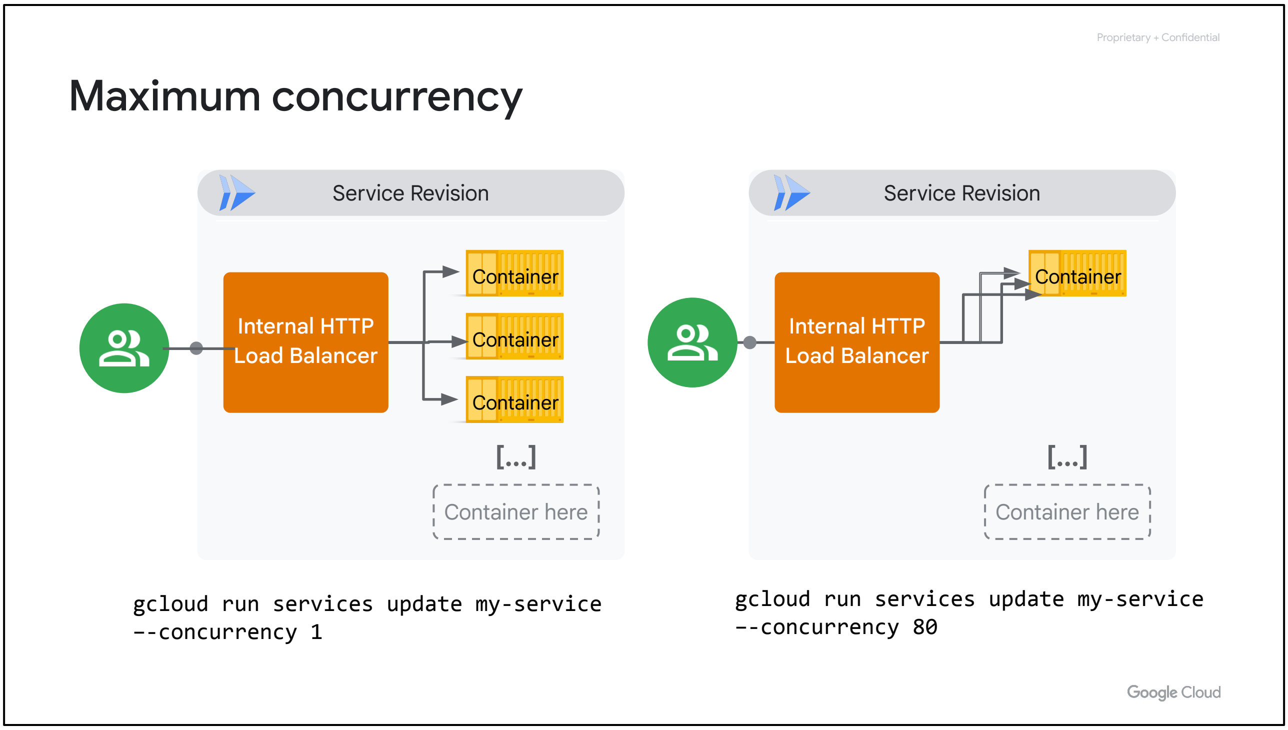
If you deploy a service that scales up to many container instances, you will incur costs for running those containers.

If your Cloud Run service scales up to many container instances in a short time period, your downstream systems might not be able to handle the additional traffic load. You'll need to understand the throughput capacity of those downstream systems when configuring your Cloud Run service.

For example, your Cloud Run service might interact with a database that can only handle a certain number of concurrent open connections.

To limit the total number of container instances that can be started, for cost control reasons, or for better compatibility with other resources used by your service, use the maximum container instances setting for your service revision. Be aware that setting maximum instances too low affects the ability of Cloud Run to scale up to serve all incoming requests.

By default, Cloud Run services are configured to scale out to a maximum of 100 instances. You can set or update the maximum instances configuration for your service in the Google Cloud console, using the gcloud CLI, a YAML configuration file, or Terraform.



Proprietary + Confidential

Maximum concurrency

```
gcloud run services update my-service
--concurrency 1
```

```
gcloud run services update my-service
--concurrency 80
```

Google Cloud

As previously mentioned, each service revision is automatically scaled to the number of container instances needed to handle all incoming requests. When more container instances are processing requests, more CPU and memory will be used, resulting in higher costs.

To give you more control, Cloud Run provides a *maximum concurrent requests per instance* setting, that specifies the maximum number of requests that can be processed simultaneously by a given container instance. Cloud Run automatically adjusts the concurrency up to the configured maximum.

By default, each Cloud Run container instance can receive up to 80 requests at the same time; you can increase this to a maximum of 1000.
Consider setting the maximum concurrency to 1 if:

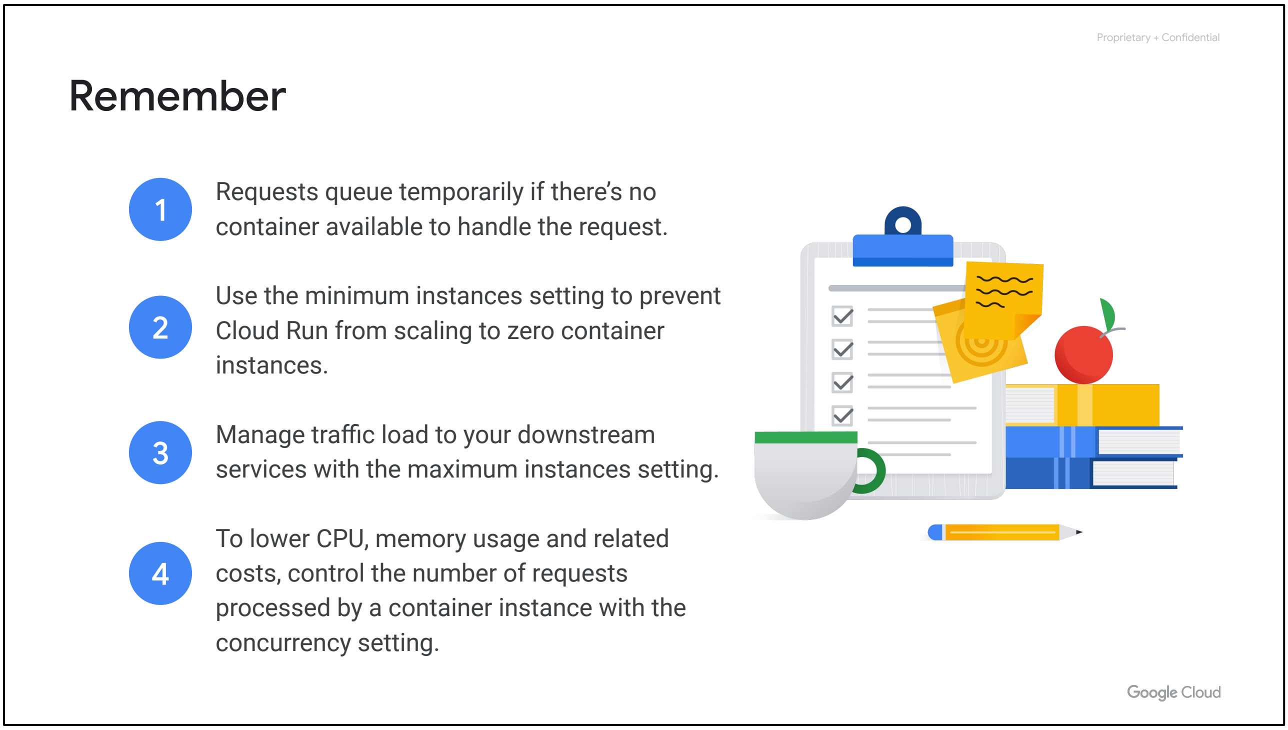
- Each request uses most of the available container CPU or memory.
- You application code is not designed to handle multiple requests at the same time.
- Your application code relies on global states that cannot be shared by multiple requests.

If your container cannot process many requests concurrently, you should consider lowering the maximum concurrency setting.
The number of concurrent requests that each container instance can serve can be limited by the technology stack, and the use of shared resources such as variables and database connections. Setting a higher maximum concurrency could have an

effect on downstream services when there are traffic spikes to your service.
Also, as each request to your service requires some amount of additional memory, a high maximum concurrency setting could increase the overall memory requirement of your container.

You can set or update the maximum concurrency configuration for your service in the Google Cloud console, using the gcloud CLI, a YAML configuration file, or Terraform.

Configure your service with an optimal concurrency configuration to maintain stability under expected load. You can achieve this by load testing your application with tools that support configurable concurrency configuration.



Proprietary + Confidential

Remember

1. Requests queue temporarily if there's no container available to handle the request.
2. Use the minimum instances setting to prevent Cloud Run from scaling to zero container instances.
3. Manage traffic load to your downstream services with the maximum instances setting.
4. To lower CPU, memory usage and related costs, control the number of requests processed by a container instance with the concurrency setting.

In summary:

The first few requests that come in after your service scaled to zero will queue while the first container instance starts.

To reduce the latency of your service when there are no instances, configure Cloud Run to keep a minimum number of container instances active with the minimum instances setting.

For cost control reasons, or for better compatibility with other resources used by your service, use the maximum container instances setting to limit the total number of container instances that can be started.

You can control CPU, memory usage, and related costs by setting the concurrency configuration to allow a container instance to process multiple requests at the same time.