App Engine standard environment

- Easily deploy your applications
- Autoscale workloads
- Free daily quota
- Usage based pricing

Of the two App Engine Environments, Standard is the simpler. It offers a simpler deployment experience than the Flexible environment and fine-grained auto-scale. Like the Standard Environment, it also offers a free daily usage quota for the use of some services.

What's distinctive about the Standard Environment though, is that low utilization applications might be able to run at no charge.

Google provides App Engine software development kits in several languages, so that you can test your application locally before you upload it to the real

App Engine standard environment

- SDKs for development, testing and deployment

App Engine service. The SDKs also provide simple commands for deployment. Now, you may be wondering what does my code actually run on?

I mean what exactly is the executable binary? App Engine's term for this kind of binary is the runtime. In App Engine Standard Environment, you use a runtime provided by Google. We'll see your choices shortly.

App Engine Standard Environment provides runtimes for specific versions of Java, Python, PHP and Go. The runtimes also include libraries that support App Engine APIs. And for many applications, the Standard

App Engine standard environment: Requirements

- Specific versions of Java, Python, PHP, and Go are supported

App Engine Standard Environment provides runtimes for specific versions of Java, Python, PHP and Go. The runtimes also include libraries that support App Engine APIs. And for many applications, the Standard Environment runtimes and libraries may be all you need.

If you want to code in another language, Standard Environment is not right for you. You'll want to consider the Flexible Environment. The Standard Environment also enforces restrictions on your code by making it run in a so-called "Sandbox."

That's a software construct that's independent of the hardware

App Engine standard environment: Requirements

Sandbox constraints:

- No writing to local files
- All requests time out at 60s
- Limits on third-party software

That's a software construct that's independent of the hardware, operating system, or physical location of the server it runs on. The Sandbox is one of the reasons why App Engine Standard Environment can scale and manage your application in a very fine-grained way.

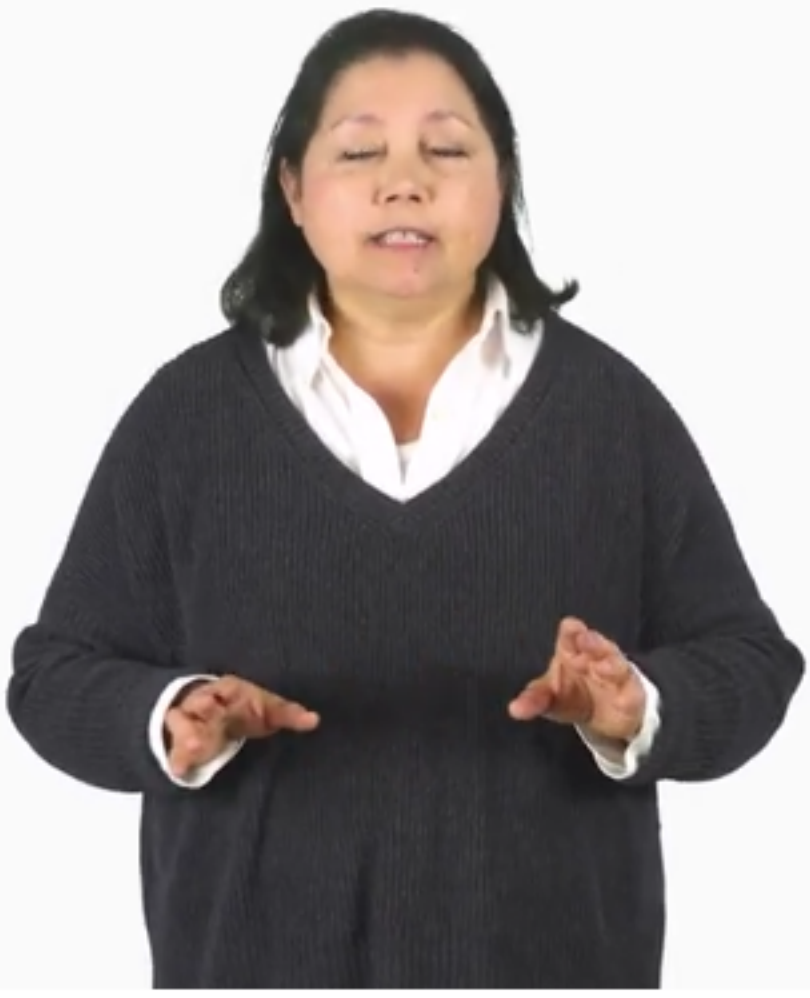
Like all Sandboxes, it imposes some constraints. For example, your application can't write to the local file system. It'll have to write to a database service instead if it needs to make data persistent.

Also, all the requests your application receives has a 60-second timeout, and

App Engine standard environment: Requirements

Sandbox constraints:

- No writing to local files
- All requests time out at 60s
- Limits on third-party software



Also, all the requests your application receives has a 60-second timeout, and you can't install arbitrary third party software. If these constraints don't work for you, that would be a reason to choose the Flexible Environment.

Here's a diagram of how you'll use App Engine Standard Environment in practice. You'll develop your application and run a test version of it locally using the App Engine SDK. Then when you're ready, you'll use the SDK to deploy it. Each App Engine application runs in a GCP project.

App Engine automatically provisions server instances and scales and load

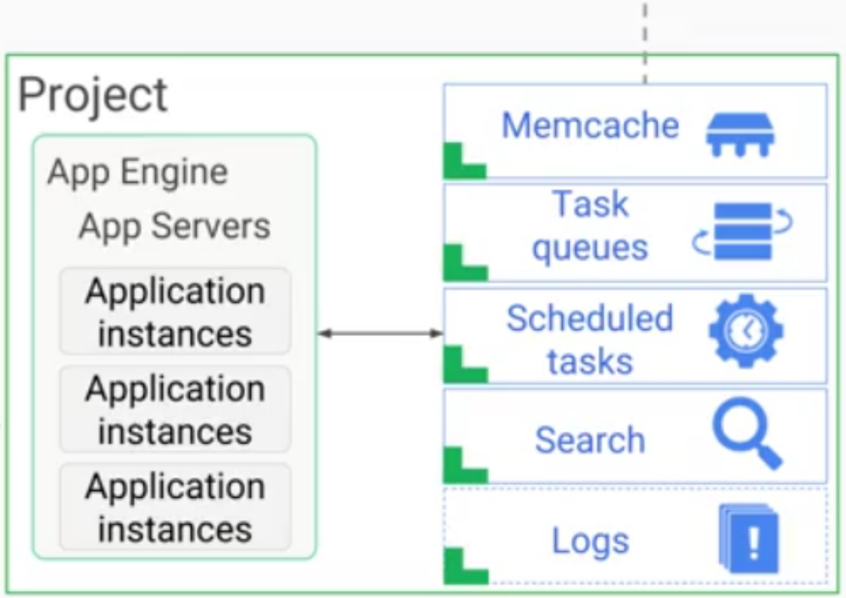
Example App Engine standard workflow: Web applications

1 Develop & test the web application locally

2 Use the SDK to deploy to App Engine

3 App Engine automatically scales & reliably serves your web application

App Engine can access a variety of services using dedicated APIs



Here are a few examples: a NoSQL data store to make data persistent, caching of that data using Memcache, searching logging, user logging, and the ability to launch actions not triggered by direct user requests, like task queues and a task scheduler.

App Engine flexible environment

- Build and deploy containerized apps with a click
- No sandbox constraints
- Can access App Engine resources

Suppose you've decided that the restrictions of App Engine standard environment's sandbox model don't work for you, but you still want to take advantage of the benefits of App Engine. That's what App Engine flexible environment is for.

Instead of the sandbox, App Engine flexible environment lets you specify the container your App Engine runs in. Yes, containers. Your application runs inside Docker containers on Google Compute Engine Virtual Machines, VMs. App Engine manages these Compute Engine machines for you.

They're health checked, healed as necessary, and you get to choose which geographical region they run in, and critical backward-compatible updates to their operating systems are automatically applied. All this so that you can just focus on your code.

App Engine flexible environment apps use standard run times, can access App Engine services such as data store, memcached, task queues, and so on. Here's a side-by-side comparison of Standard and Flexible.

Notice that Standard environment starts up instances of your application

Comparing the App Engine environments

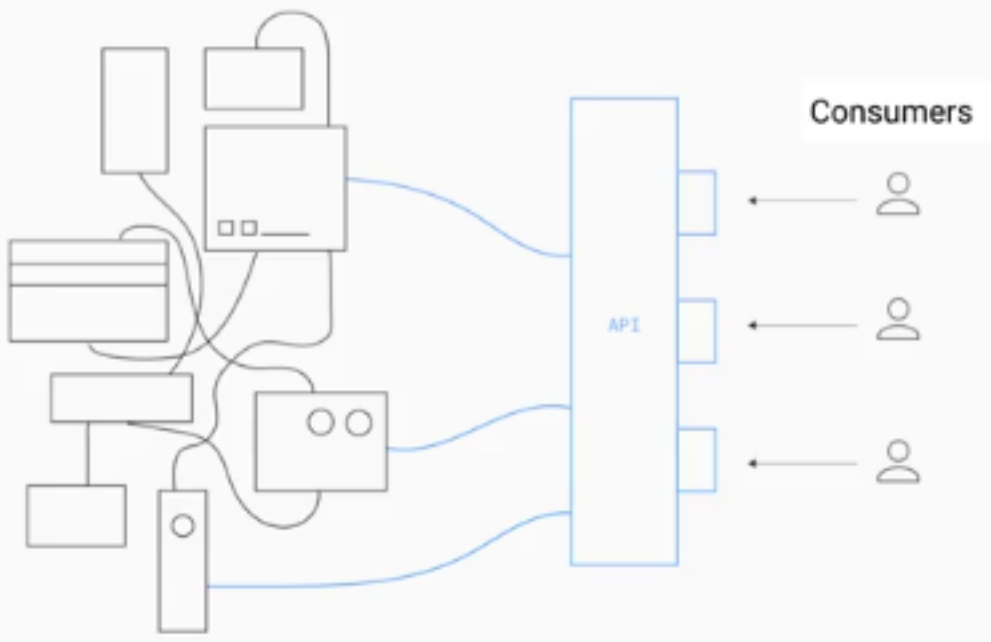
	Standard Environment	Flexible Environment
Instance startup	Milliseconds	Minutes
SSH access	No	Yes (although not by default)
Write to local disk	No	Yes (but writes are ephemeral)
Support for 3rd-party binaries	No	Yes
Network access	Via App Engine services	Yes
Pricing model	After free daily use, pay per instance class, with automatic shutdown	Pay for resource allocation per hour; no automatic shutdown

Deploying Apps: Kubernetes Engine vs App Engine

	Kubernetes Engine	**App Engine Flexible**	**App Engine Standard**
Language support	Any	Any	Java, Python, Go, PHP
Service model	Hybrid	PaaS	PaaS
Primary use case	Container-based workloads	Web and mobile applications, container-based workloads	Web and mobile applications

Toward managed infrastructure *Toward dynamic infrastructure*

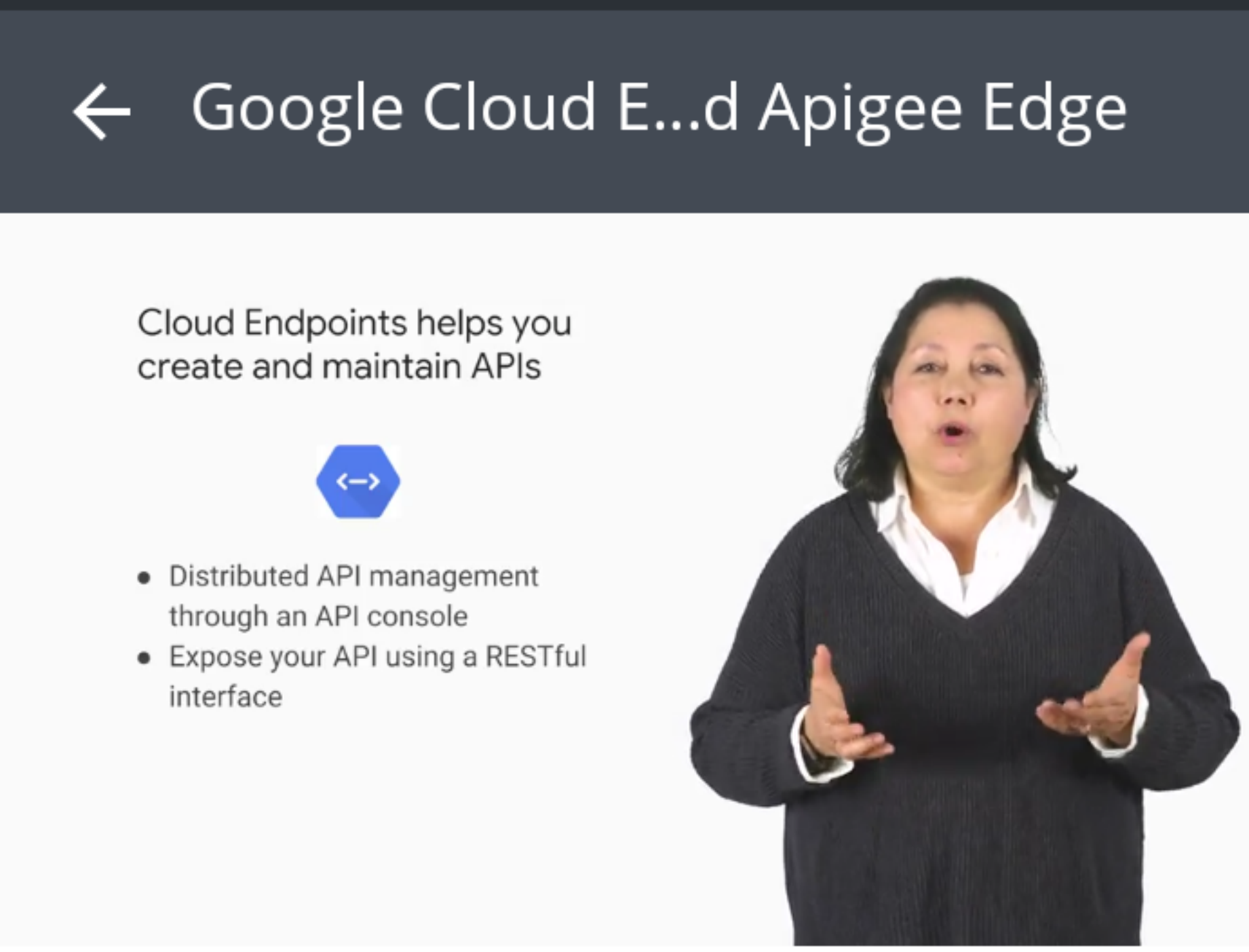
Application Programming Interfaces hide detail, enforce contracts



in this course. Let's be precise about what an API is. A software services implementation can be complex and changeable. What if to use that service, other pieces of software had to know internal details about how they worked?

That would be a mess. So instead application developers structure the software they write so that it presents a clean, well-defined interface that abstracts away needless details and then they document that interface. That's an API.

The underlying implementation can change as long as the interface doesn't and other pieces of software



particular strength. Suppose you're developing a software service and one of GCP's backends. You'd like to make it easy to expose this API. You'd like to make sure it's only consumed by other developers whom you trust.

You'd like an easy way to monitor and log its use. You'd like for the API to have a single coherent way for it to know which end user is making the call. That's when you use Cloud Endpoints.

It implements these capabilities and more using an easy to deploy proxy in front of your software service, and it provides an API console to

Cloud Endpoints helps you create and maintain APIs

- Control access and validate calls with JSON Web Tokens and Google API keys
  - Identify web, mobile users with Auth0 and Firebase Authentication
- Generate client libraries

It implements these capabilities and more using an easy to deploy proxy in front of your software service, and it provides an API console to wrap up those capabilities in an easy-to-manage interface.

Cloud Endpoints supports applica-tions running in GCP's compute plat-forms in your choice of languages and your choice of client technologies. Apigee Edge is also a platform for developing and managing API proxies. It has a different orientation though.

It has a focus on business problems like rate limiting, quotas, and analytics. Many users of Apigee Edge

Cloud Endpoints: Supported platforms

Runtime environment
App Engine Flexible Environment
Kubernetes Engine
Compute Engine

Clients
Android
iOS
Javascript

Cloud Endpoints supports applications running in GCP's compute platforms in your choice of languages and your choice of client technologies. Apigee Edge is also a platform for developing and managing API proxies. It has a different orientation though.

It has a focus on business problems like rate limiting, quotas, and analytics. Many users of Apigee Edge are providing a software service to other companies and those features come in handy.

Because of the backend services for Apigee Edge need not be in GCP, engineers often use it when they are "taking apart" a legacy application

Apigee Edge helps you secure and monetize APIs

apigee

- A platform for making APIs available to your customers and partners
- Contains analytics, monetization, and a developer portal