Azure Functions hosting plans comparison

06/12/2017 • 6 minutes to read • Contributors all

In this article

Introduction

You can run Azure Functions in two different modes: Consumption plan and Azure App Service plan. The Consumption plan automatically allocates compute power when your code is running, scales out as necessary to handle load, and then scales down when code is not running. So, you don't have to pay for idle VMs and don't have to reserve capacity in advance. This article focuses on the Consumption plan, a serverless app model. For details about how the App Service plan works, see the Azure App Service plans in-depth overview.

If you aren't familiar with Azure Functions, see the Azure Functions overview.

When you create a function app, you must configure a hosting plan for functions that the app contains. In either mode, an instance of the *Azure Functions host* executes the functions. The type of plan controls:

- How host instances are scaled out.
- The resources that are available to each host.

Currently, you must choose the type of hosting plan during the creation of the function app. You can't change it afterward.

On an App Service plan you can scale between tiers to allocate different amount of resources. On the Consumption plan, Azure Functions automatically handles all resource allocation.

Consumption plan

When you're using a Consumption plan, instances of the Azure Functions host are dynamically added and removed based on the number of incoming events. This plan scales

automatically, and you are charged for compute resources only when your functions are running. On a Consumption plan, a function can run for a maximum of 10 minutes.

> ⓘ Note
>
> The default timeout for functions on a Consumption plan is 5 minutes. The value can be increased to 10 minutes for the Function App by changing the property `functionTimeout` in host.json.

Billing is based on number of executions, execution time, and memory used. Billing is aggregated across all functions within a function app. For more information, see the Azure Functions pricing page.

The Consumption plan is the default hosting plan and offers the following benefits:

- Pay only when your functions are running.
- Scale out automatically, even during periods of high load.

App Service plan

In the App Service plan, your function apps run on dedicated VMs on Basic, Standard, Premium, and Isolated SKUs, similar to Web Apps, API Apps, and Mobile Apps. Dedicated VMs are allocated to your App Service apps, which means the functions host is always running.

Consider an App Service plan in the following cases:

- You have existing, underutilized VMs that are already running other App Service instances.
- You expect your function apps to run continuously, or nearly continuously. In this case, an App Service Plan can be more cost-effective.
- You need more CPU or memory options than what is provided on the Consumption plan.
- You need to run longer than the maximum execution time allowed on the Consumption plan (of 10 minutes).
- You require features that are only available on an App Service plan, such as support for App Service Environment, VNET/VPN connectivity, and larger VM sizes.

A VM decouples cost from number of executions, execution time, and memory used. As a result, you won't pay more than the cost of the VM instance that you allocate. For details about how the App Service plan works, see the Azure App Service plans in-depth overview.

With an App Service plan, you can manually scale out by adding more VM instances, or you can enable autoscale. For more information, see Scale instance count manually or automatically. You can also scale up by choosing a different App Service plan. For more information, see Scale up an app in Azure.

If you are planning to run JavaScript functions on an App Service plan, you should choose a plan that has fewer cores. For more information, see the JavaScript reference for Functions.

Always On

If you run on an App Service plan, you should enable the **Always On** setting so that your function app runs correctly. On an App Service plan, the functions runtime will go idle after a few minutes of inactivity, so only HTTP triggers will "wake up" your functions. This is similar to how WebJobs must have Always On enabled.

Always On is available only on an App Service plan. On a Consumption plan, the platform activates function apps automatically.

Storage account requirements

On either a Consumption plan or an App Service plan, a function app requires a general Azure Storage account that supports Azure Blob, Queue, Files, and Table storage. Internally, Azure Functions uses Azure Storage for operations such as managing triggers and logging function executions. Some storage accounts do not support queues and tables, such as blob-only storage accounts (including premium storage) and general-purpose storage accounts with zone-redundant storage replication. These accounts are filtered from the **Storage Account** blade when you're creating a function app.

To learn more about storage account types, see Introducing the Azure Storage services.

How the Consumption plan works

In the Consumption plan, the scale controller automatically scales CPU and memory resources by adding additional instances of the Functions host, based on the number of events that its functions are triggered on. Each instance of the Functions host is limited to 1.5 GB of memory.

When you use the Consumption hosting plan, function code files are stored on Azure Files shares on the function's main storage account. When you delete the main storage account of the function app, the function code files are deleted and cannot be recovered.

ⓘ Note

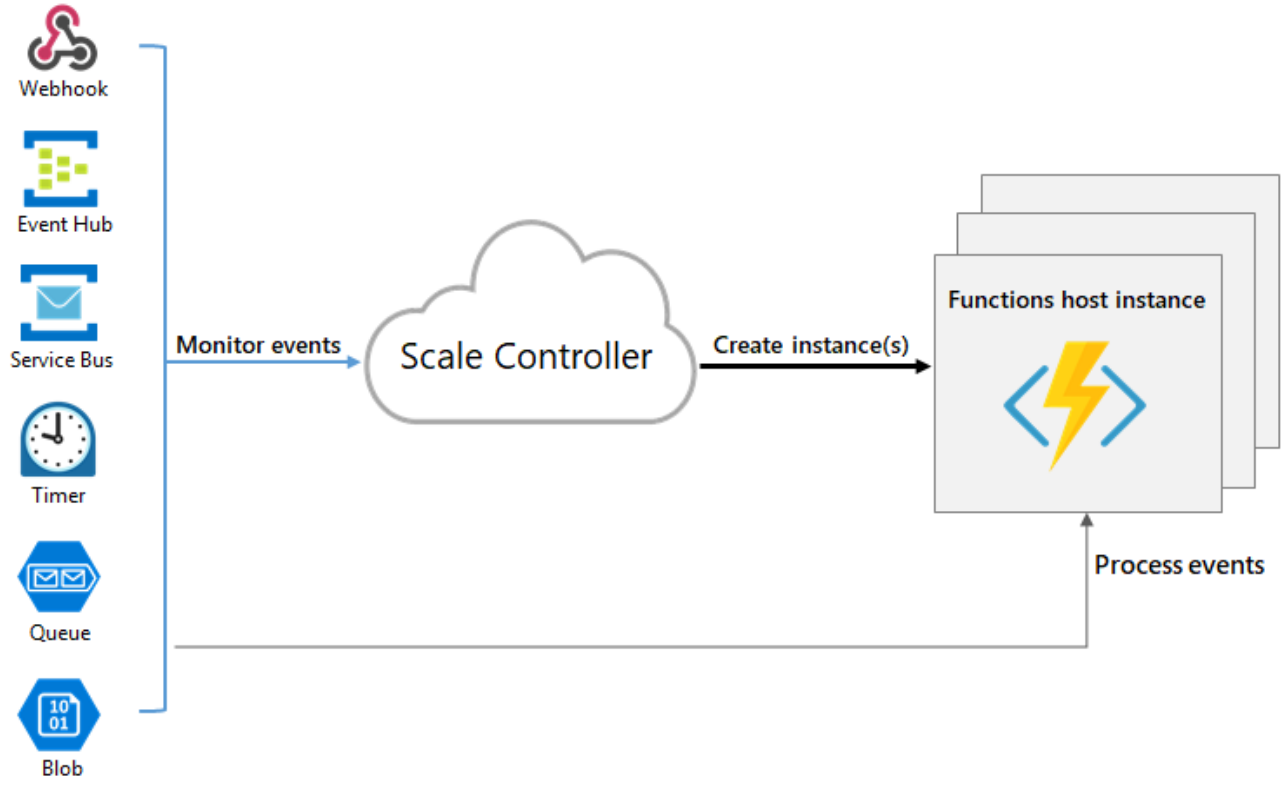
When you're using a blob trigger on a Consumption plan, there can be up to a 10-minute delay in processing new blobs if a function app has gone idle. After the function app is running, blobs are processed immediately. To avoid this initial delay, consider one of the following options:

- Host the function app on an App Service plan, with Always On enabled.
- Use another mechanism to trigger the blob processing, such as a queue message that contains the blob name. For an example, see Queue trigger with blob input binding.

Runtime scaling

Azure Functions uses a component called the *scale controller* to monitor the rate of events and determine whether to scale out or scale in. The scale controller uses heuristics for each trigger type. For example, when you're using an Azure Queue storage trigger, it scales based on the queue length and the age of the oldest queue message.

The unit of scale is the function app. When the function app is scaled out, additional resources are allocated to run multiple instances of the Azure Functions host. Conversely, as compute demand is reduced, the scale controller removes function host instances. The number of instances is eventually scaled down to zero when no functions are running within a function app.



Billing model