Introduction To Development With Azure Functions

Rob Windsor
rob@robwindsor.com
@robwindsor

About Me

Rob Windsor

.NET/Microsoft 365 developer, trainer, author
Microsoft MVP | Microsoft 365 Development

Twitter: https://twitter.com/robwindsor
LinkedIn: https://www.linkedin.com/in/rwindsor
Blog: https://robwindsor.hashnode.dev
YouTube: https://www.youtube.com/@RobWindsor
GitHub: https://github.com/rob-windsor

Overview

According to the Microsoft documentation:

> *Azure Functions is a serverless solution that allows you to write less code, maintain less infrastructure, and save on costs. Instead of worrying about deploying and maintaining servers, the cloud infrastructure provides all the up-to-date resources needed to keep your applications running.*
>
> *You focus on the code that matters most to you, in the most productive language for you, and Azure Functions handles the rest.*
>
> *https://learn.microsoft.com/en-us/azure/azure-functions/functions-overview*

The above may or may not be true depending on the choice you make for hosting.

Hosting Plans

- Consumption (Serverless)
  - Only pay when functions are used
  - Can be delays due to cold worker starts
- Flex Consumption (Preview)
  - Similar to plan above
  - Optionally configure one or more pre-warmed instances
- Premium
  - Requires an Azure App Service
  - One or more pre-warmed workers
  - Additional workers added as needed
  - Can deploy multiple Function Apps to same App Service

Hosting Plans 2

- App Service (Dedicated)
  - Requires an Azure App Service
  - You manage hosting
- Container Apps
  - Requires an Azure App Service
  - Function Apps are containerized
  - You manage hosting

Supported Languages

Depending on the hosting plan, you can use the following languages:

- C#
- TypeScript, JavaScript
- PowerShell
- Java
- Python

.NET Runtimes

Azure Functions currently supports two versions of the runtime host

- v1
  - Supported only for C# apps that must use .NET Framework
  - Support will end in September 2026
- v4
  - Recommended runtime version for functions in all languages

.NET: In-process vs isolated

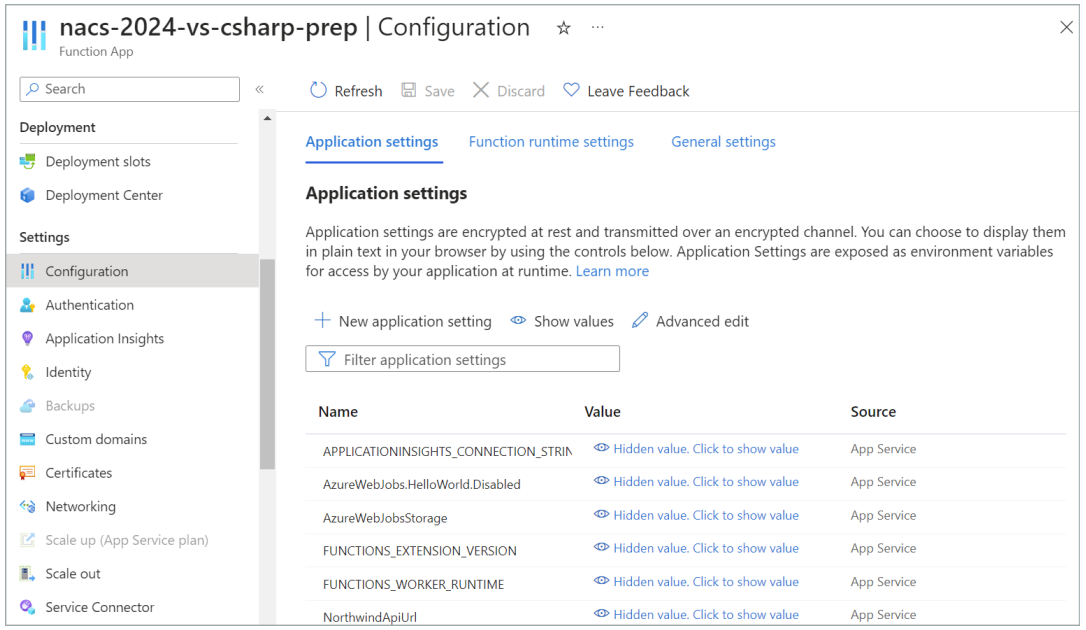
- In-process
  - Function code runs in the same process as the Functions host process
  - Support for in-process model ends in November 2026
- Isolated
  - Your function code runs in a separate .NET worker process

Triggers and Bindings

- Triggers
  - Causes a function to run
  - Each function must have exactly one trigger
- Bindings
  - Declaratively connecting another resource to the function
  - May be input, output or both
  - Data from bindings is provided to the function as parameters

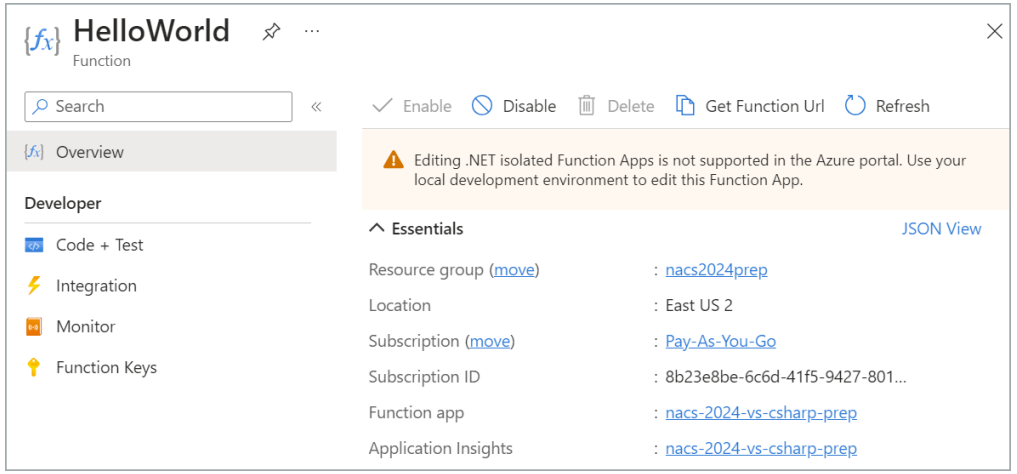
App Settings (Environment Variables)

- Running locally
  - Configured in local.settings.json file in project
- Running from Azure host
  - Configured in Azure portal

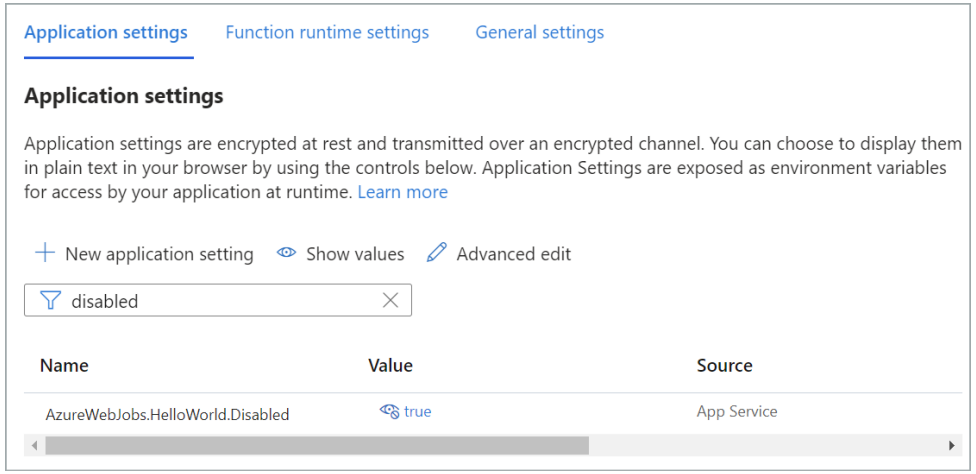


Disable Function

Function overview page



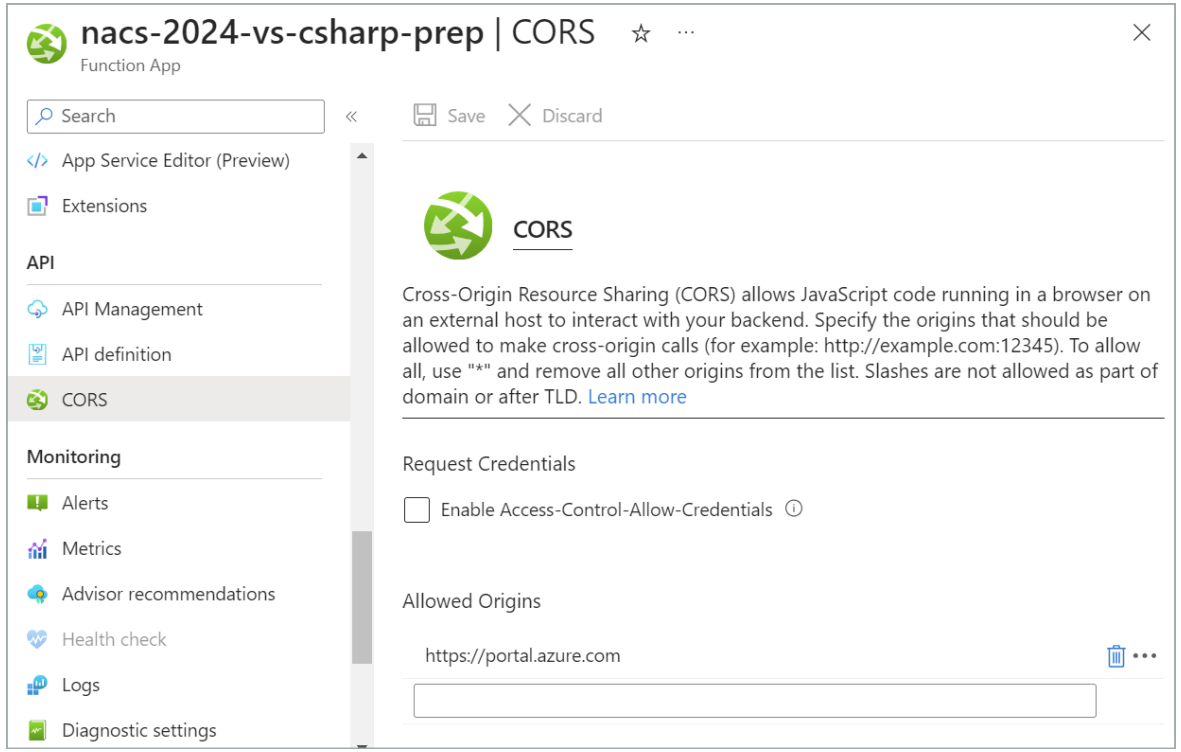
Environment variable



CORS

Cross-Origin Resource Sharing (CORS) allows JavaScript code running in a browser on an external host to interact with your backend.

- Configured in Azure portal



Durable Functions

An extension of Azure Functions that lets you write stateful functions even in a serverless compute environment.

- Define stateful workflows by writing orchestrator functions
- Define stateful entities by writing entity functions
- Extension manages state, checkpoints, and restarts for you

Resources

Azure Functions Documentation
https://learn.microsoft.com/en-us/azure/azure-functions/

Azure Functions Core Tools
https://learn.microsoft.com/en-us/azure/azure-functions/functions-run-local

Azure CLI
https://learn.microsoft.com/en-us/cli/azure/