Azure DevOps Pipeline CI for Azure Functions

About Eriawan Kusumawardhono

Principal Software Engineer by day, and also software developer community servant, and Microsoft MVP since 2013

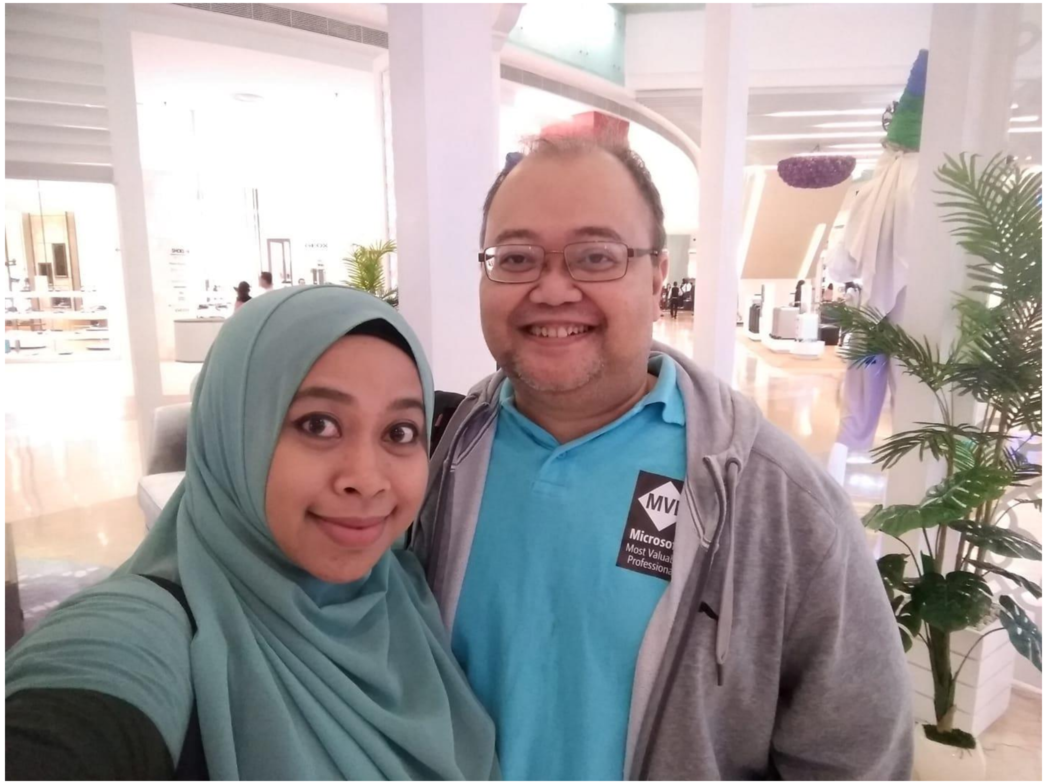
MUGI Jadetabek chairman

Azure DevOps Indonesia chairman

Github: eriawan

Twitter: @erikuma

Email: eriawan@live.com



Main topics

Overview of Azure Functions development

Preparing Azure DevOps CI pipelines for Azure Functions

Common best practices for Azure Functions CI/CD

Overview of Azure Functions development

Quick overview of Azure Functions

One of SaaS service from Azure

Focus on one concern of function at one time. For example: GetCurrentNYMEXMarketPrice to get current NYMEX market price for raw crude oil

Development platform specific. For example: Java, .NET, and other supported platforms.

Does not care about infrastructure. Infrastructure is not a main concern here, as the actual concern is more about the actual executions and cost incurred later. See next slide.

For more information, visit https://docs.microsoft.com/azure/azure-functions/

Assessing TCO plan of Azure Functions

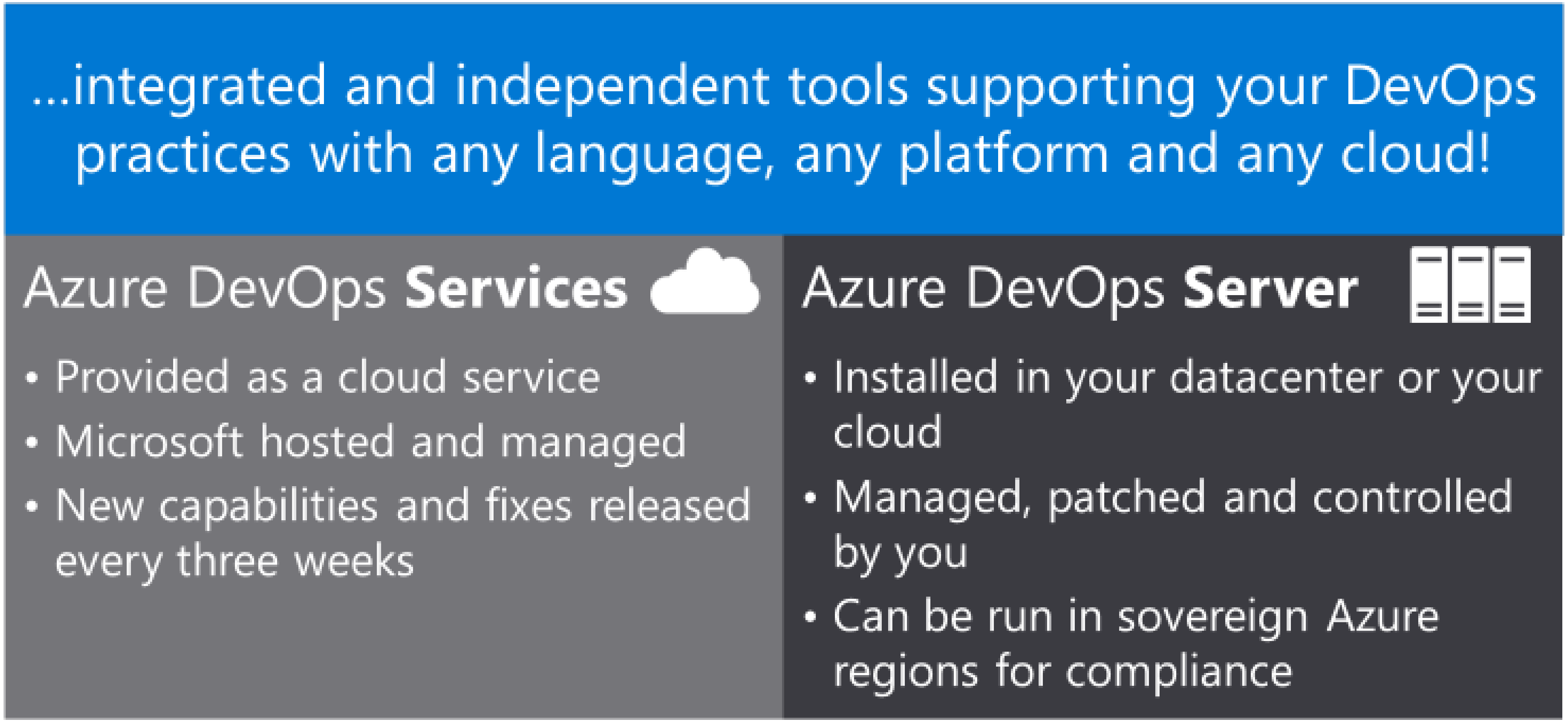
Azure Functions has three pricing plans that will always affect the overall production activities: Consumption plan, Premium plan, and App Service plan.

If the main concern is the actual execution time and number of executions, Consumption plan is the best, especially for the initial development as Azure Functions consumption is free for the first 1 million execution in Consumption plan.

If performance is the main concern, Premium plan provides auto scaling and throughput boost buffering for initial warm start and provide more information about the scaling.

Preparing Azure DevOps CI pipelines for Azure Functions

Azure DevOps is…



…integrated and independent tools supporting your DevOps practices with any language, any platform and any cloud!

Azure DevOps **Services**

- Provided as a cloud service
- Microsoft hosted and managed
- New capabilities and fixes released every three weeks

Azure DevOps **Server**

- Installed in your datacenter or your cloud
- Managed, patched and controlled by you
- Can be run in sovereign Azure regions for compliance

Introducing Azure DevOps

Azure Boards

Deliver value to your users faster using proven agile tools to plan, track, and discuss work across your teams.

Azure Pipelines

Build, test, and deploy with CI/CD that works with any language, platform, and cloud. Connect to GitHub or any other Git provider and deploy continuously.

Azure Repos

Get unlimited, cloud-hosted private Git repos and collaborate to build better code with pull requests and advanced file management.

Azure Test Plans

Test and ship with confidence using manual and exploratory testing tools.

Azure Artifacts

Create, host, and share packages with your team, and add artifacts to your CI/CD pipelines with a single click.

https://azure.com/devops

Quick glance of Azure DevOps Pipelines

Provides Continuous Integrations (Build) and Continuous Deployment (Release)

Support CI creation using visual build designer and YAML (since Azure DevOps Server 2019)

Support CD creation using visual release sequence designer (since Azure DevOps Server 2019)

Support CD creation using YAML (since Azure DevOps Server 2020)

Pipelines build support for Azure DevOps

Support for CI pipeline depends on language and/or platform support. However, language support comes first. For example: developing Azure Functions using C# is only available on top of .NET Framework and .NET Core

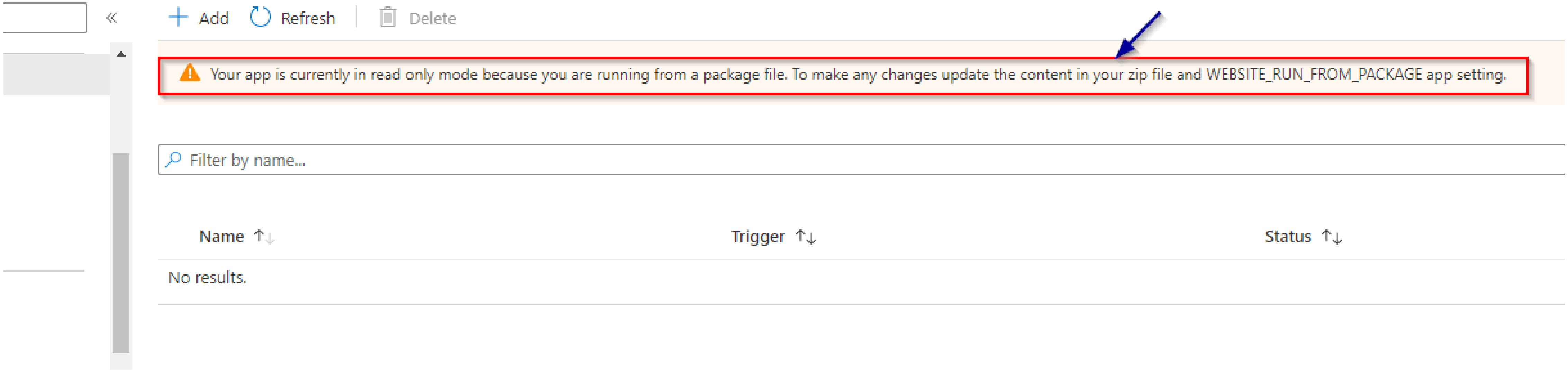
Azure DevOps Pipelines provide publishing of build result as “build artifact”. For CI build, it Is better to have build artifact after fully building or compiling, as this provide historical build source for us to compare later

Demo: Azure Functions CI using Azure’s DevOps Starter template

Setting Azure Functions configuration to manage function app setting

Further configuration can be updated from Azure Functions web in Azure portal.

For example: by default, Azure Functions created using DevOps Starter is using run from package as part of the deployment. You will see this:

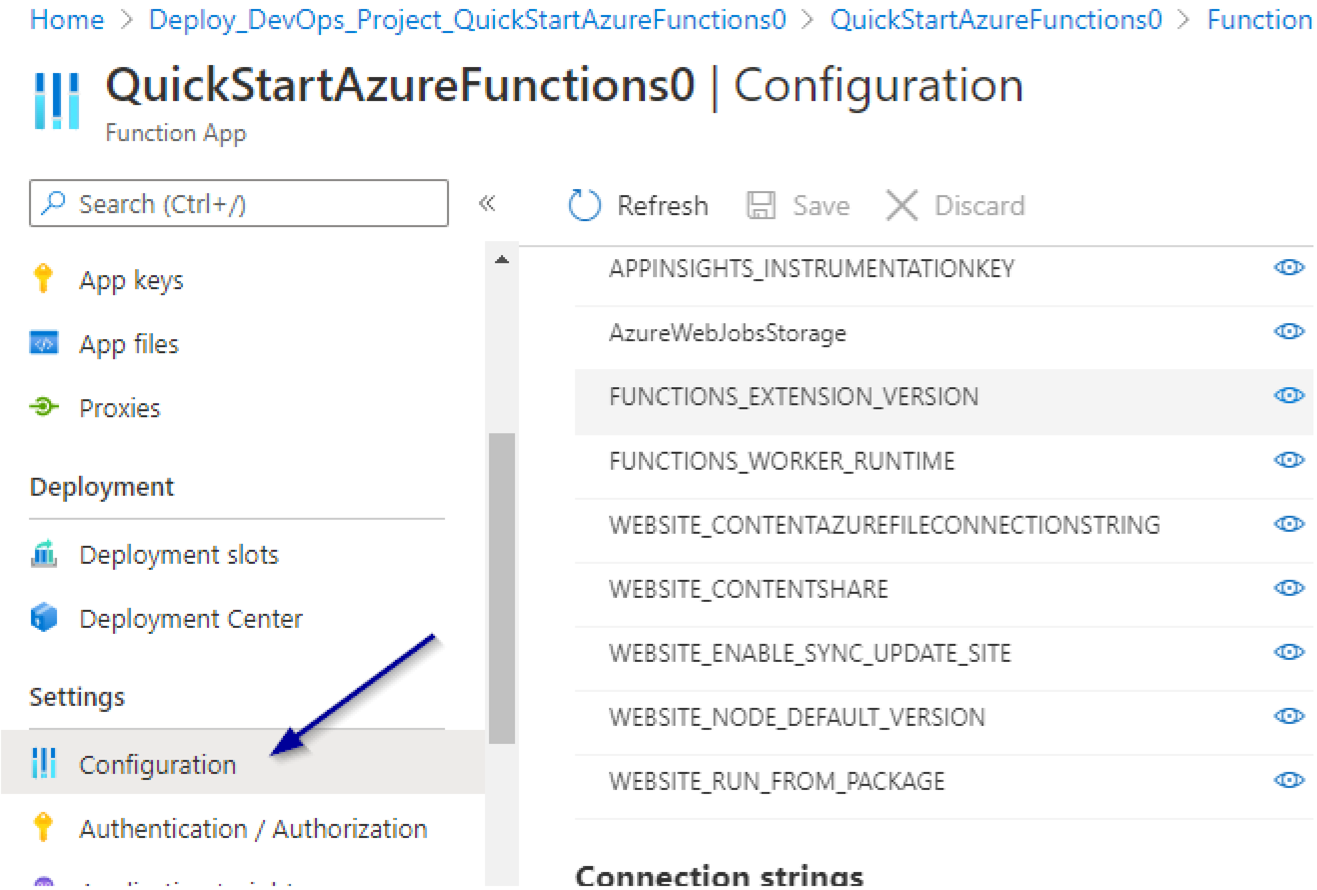


Your app is currently in read only mode because you are running from a package file. To make any changes update the content in your zip file and WEBSITE_RUN_FROM_PACKAGE app setting.

Name ↑↓	Trigger ↑↓	Status ↑↓
No results.		

Use Azure Function in Azure portal’s to change WEBSITE_RUN_FROM_PACKAGE. But this is not

Azure Function configuration UI



Home > Deploy_DevOps_Project_QuickStartAzureFunctions0 > QuickStartAzureFunctions0 > Function
QuickStartAzureFunctions0 | Configuration
Function App
Search (Ctrl+/)
«
Refresh
Save
Discard
App keys
App files
Proxies
Deployment
Deployment slots
Deployment Center
Settings
Configuration
Authentication / Authorization

APPINSIGHTS_INSTRUMENTATIONKEY
AzureWebJobsStorage
FUNCTIONS_EXTENSION_VERSION
FUNCTIONS_WORKER_RUNTIME
WEBSITE_CONTENTAZUREFILECONNECTIONSTRING
WEBSITE_CONTENTSHARE
WEBSITE_ENABLE_SYNC_UPDATE_SITE
WEBSITE_NODE_DEFAULT_VERSION
WEBSITE_RUN_FROM_PACKAGE

Connection strings

Common best practices

Pay attention on the main concerns: executions vs performance!

The function should be as simple as possible and focus on one concern, although the implementation detail may does one or more things.

Function should be as quick and as fast as possible, as it is focusing on once concern. Therefore any long running function should be identified first and may not fit on what kind of the initial trigger the function will be. Long running function should not be created to be triggered by HTTP requests, whereas simple but concise function that does not need to be long running may be triggered by HTTP requests.

Deploy the function in package, because this has more benefits especially for production environment. See docs: https://docs.microsoft.com/en-us/azure/azure-functions/run-functions-from-deployment-package

Always pay attention on the underlying language and platform. Put it simply, platform development support has to be top priority for maintenance. For example: .NET Core in October 2020 only provide support for v2.1 and 3.1, because 2.2 and 3.0 is not supported anymore. This also impacts Azure DevOps Pipeline build support after this point.

For Azure DevOps Server, maintenance is on your own. On Azure DevOps Service, task/step to build depends on Microsoft maintenance and support based on the lifecycle of the platform used to develop Azure Functions.

Questions?

End.