Running in Production

Mark Heath
MICROSOFT MVP

@mark_heath   www.markheath.net

Overview

How can I deploy my functions to Azure?

Are my functions secure?

How can I monitor my functions?

How can I upgrade my functions?

Demo applications overview

Deploying Durable Functions

Azure Functions Fundamentals

Visual Studio right-click publish

Git integration with Kudu

Upload a zip with Kudu API

Zip Deployment

Create a Function App

- Manually in the portal
- Automate using ARM templates

Zip up the build artefacts

Deploy with the Kudu API

- https://markheath.net/post/deploy-azure-webapp-kudu-zip-api

Simple deployment with Azure CLI

```
az functionapp deployment source config-zip `
    --src $zipFile -n $appName -g $resourceGroup
```

Deploying Durable Functions

Visual Studio right-click Publish

Git integration with Kudu

Upload a zip with Kudu API

Securing Durable Functions

Orchestrator and activity functions cannot be called directly

Orchestrations are triggered by “starter” functions

- Secured in the usual way
- e.g. authorization key for HTTP triggered starter

Durable Functions REST API

- Secret key required
- Don’t give key directly to end users

Monitoring Durable Functions

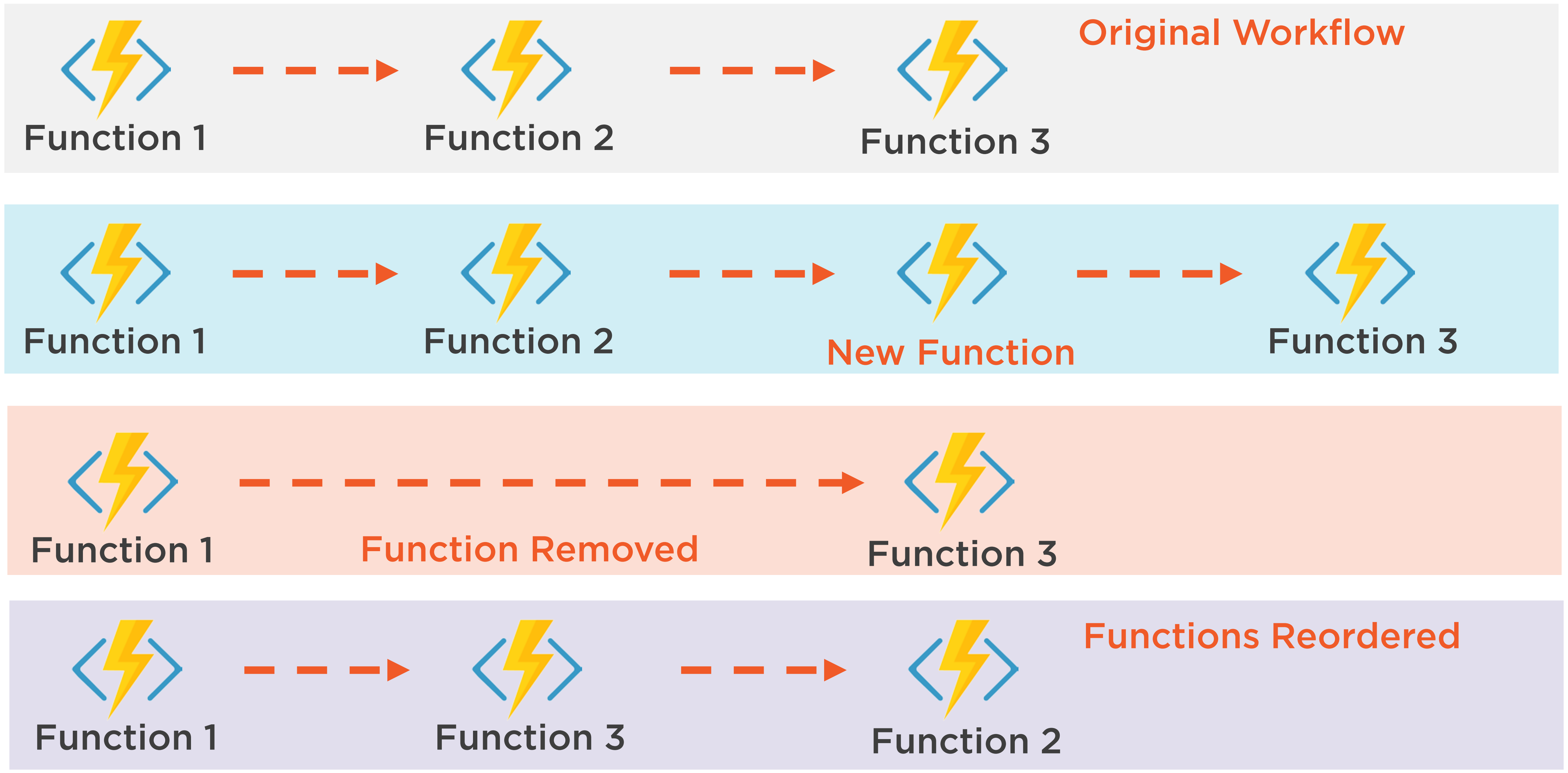
Log messages stored in Table Storage

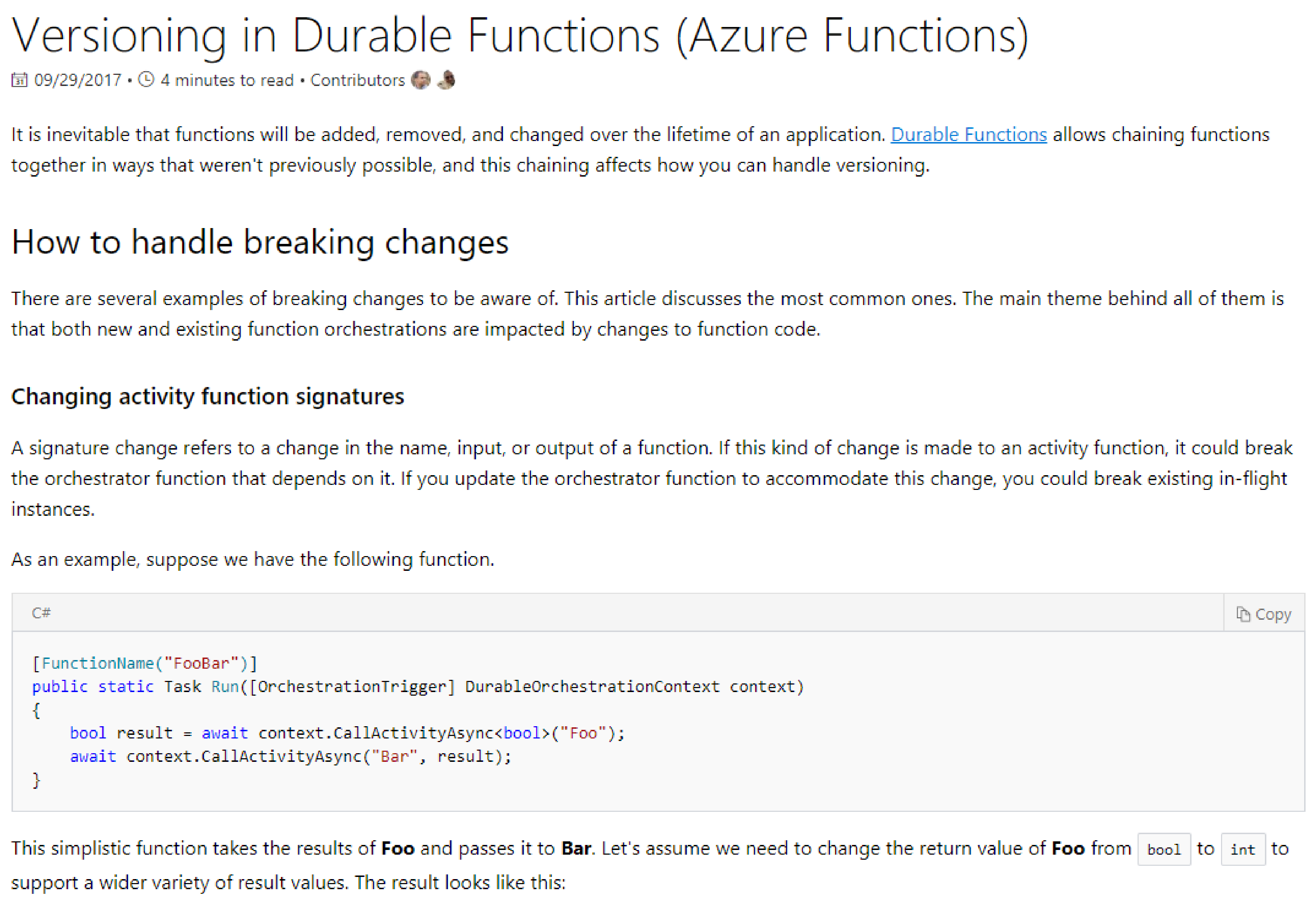
- Invocation history and log output can be viewed in the Azure Portal for each function

Application Insights

- Create an instance of Application Insights
- Get the instrumentation key
- APPINSIGHTS_INSTRUMENTATIONKEY

Upgrading Orchestrator Functions





Versioning in Durable Functions (Azure Functions)

09/29/2017 • 4 minutes to read • Contributors

It is inevitable that functions will be added, removed, and changed over the lifetime of an application. Durable Functions allows chaining functions together in ways that weren't previously possible, and this chaining affects how you can handle versioning.

How to handle breaking changes

There are several examples of breaking changes to be aware of. This article discusses the most common ones. The main theme behind all of them is that both new and existing function orchestrations are impacted by changes to function code.

Changing activity function signatures

A signature change refers to a change in the name, input, or output of a function. If this kind of change is made to an activity function, it could break the orchestrator function that depends on it. If you update the orchestrator function to accommodate this change, you could break existing in-flight instances.

As an example, suppose we have the following function.

C#   Copy

```csharp
[FunctionName("FooBar")]
public static Task Run([OrchestrationTrigger] DurableOrchestrationContext context)
{
    bool result = await context.CallActivityAsync<bool>("Foo");
    await context.CallActivityAsync("Bar", result);
}
```

This simplistic function takes the results of **Foo** and passes it to **Bar**. Let's assume we need to change the return value of **Foo** from `bool` to `int` to support a wider variety of result values. The result looks like this:

https://docs.microsoft.com/en-us/azure/azure-functions/durable-functions-versioning

Upgrade Strategies

Do nothing

- Allow workflows in progress to fail

Stop orchestrations

- Wait for them to finish
- Sending termination request

Side by side deployments

- Create new versions of orchestrator functions (e.g. OrchestratorV2)
  - Keep the V1 orchestrator until all V1 workflows have completed
- Deploy a copy of the function app with different task hub name

Summary

Deploying Durable Function apps

- Deploy from Visual Studio
- Git integration
- Deploy a zip using the Kudu API

Keep the REST API keys secret

Monitor Durable Functions

- Azure portal
- Application Insights

Versioning workflows

- Allow in-flight workflows to fail
- Deploying side by side

Additional Content

Azure Functions v2 demo implementation (.NET Core)

Real transcoding demo using FFMPEG

Thanks for watching!

Recommended Resources

Course Download Materials

- Azure Functions v1 implementation (.NET Framework)
- Azure Functions v2 implementation (.NET Core)
- Actual transcoding with FFMPEG (turn off “demo mode”)

Azure Functions Useful Links

- https://github.com/markheath/azure-functions-links
- Lots more Durable Functions content!