Introduction to Azure Functions

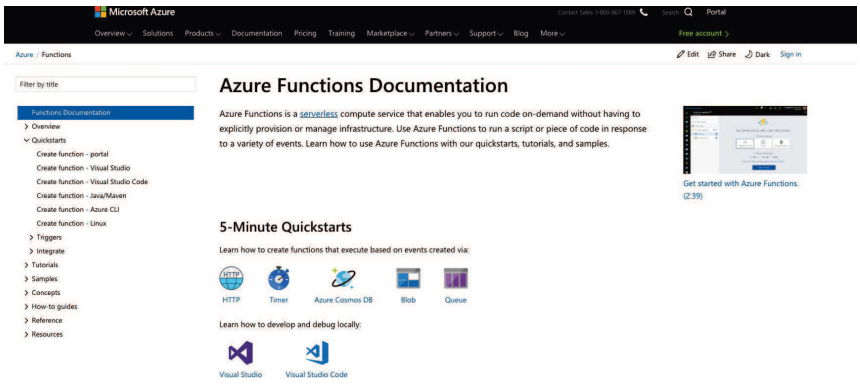
TCSS562
Feng Gao; Xiaola Ye; Jiaqi Wang

Azure Functions

Azure Functions is a solution for easily running "**functions**" **in the cloud**.

Development language:

C#, F#, JavaScript.



Reference: https://docs.microsoft.com/en-us/azure/azure-functions/functions-overview

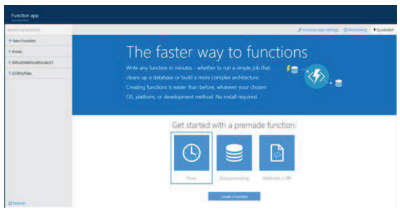
History of Azure Functions

In October 2008, Azure was announced and started with codename "Project Red Dog".

On February 1, 2010, Azure was released as "Windows Azure" .

On March 25, 2014, Azure was renamed "Microsoft Azure".

On March 31, 2016, Azure Functions was introduced by Microsoft.



History of Azure Functions

- **Cloud development is evolving**
- highly-scalable solutions built on ever-compressed timetables.
- Implement on demand or scheduled batch jobs in real-time.
- Simplified and scalable event-driven solutions.

- **Azure Functions is an innovative way**
- Implement code **triggered** by events occurring in Azure or third party service as well as on-premises systems.
- Build **HTTP-based API endpoints** accessible by mobile and IoT devices.
- Azure Functions is **scale-based** and **on-demand**, users pay only for the resources they consume.

History of Azure Functions

There are two major versions of the Azure Functions: 1.x and 2.x.

1.x hosting on Azure portal on Windows

2.x runs on .NET core 2, extend to macOS and Linux

Instead Azure Application Insights by WebJobs dashboard;

100 concurrent requests per instance;

Adding Graph Excel tables, OneDrive files, Outlook email;

Language	1.x	2.x
C#	GA (.NET Framework 4.7)	GA (.NET Core 2)
JavaScript	GA (Node 6)	GA (Node 8 & 10)
F#	GA (.NET Framework 4.7)	GA (.NET Core 2)
Java	N/A	Preview (Java 8)
Python	Experimental	N/A
TypeScript	Experimental	Supported through transpiling to JavaScript
PHP	Experimental	N/A
Batch (.cmd, .bat)	Experimental	N/A
Bash	Experimental	N/A
PowerShell	Experimental	N/A

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

Key Features

- Language - Write functions using your choice of C#, F#, or Javascript.
- Pay-per-use - Pay only for the time spent running your code.
- Own dependencies
- Integrated security - Protect HTTP-triggered functions with OAuth providers
- Simplified integration
- Flexible development
- Open-source - The Functions runtime is open-source and available on GitHub.

Reference: https://docs.microsoft.com/en-us/azure/azure-functions/functions-overview

Example Application Cases

FUJIFILM

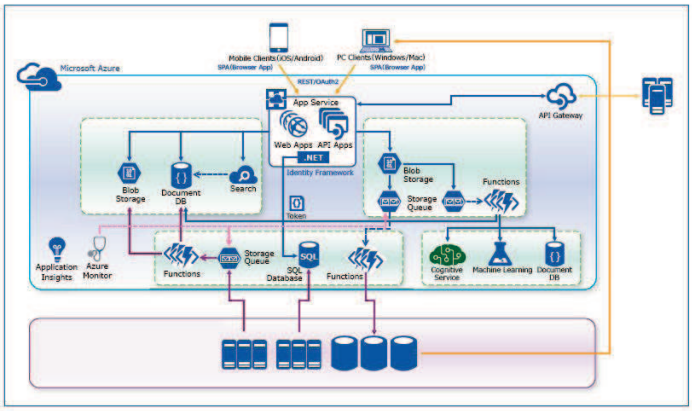
Speed Development and Accurate Debug

FUJIFILM's use of Azure App Service to develop the new IMAGE WORKS and then connect them with Azure Functions;

Azure Functions to implement **microservices**—— for service delivery stability.

Easily maintenance for a single microservice fails——waterfall development, but these Azure features supported adoption of **agile** development for IMAGE WORKS.

The move from waterfall to agile cut development time for new functions by **75 percent.**



Azure Functions Advantages

As serverless:

- **Avoidance of managing servers**
- **Allowing for dynamic scaling:** scale out automatically, even during periods of high load
- **Micro-Pricing:** pay only when your functions are running

As Azure functions:

- **Easy to test and debug:** you can test and debug your application on your local computer
- **Easy to deploy:** allow continuous integration by using GitHub or Bitbucket, etc.
- **Cheaper than lambda:** only $0.000016/GBs while lambda requires $0.00001667/GBs
- **Runs in two different plans:** Consumption plan and Azure App Service plan

http://www.mamicode.com/info-detail-2087014.html
http://resource.onlinetech.com/aws-lambda-vs-azure-functions-key-differences/

Azure Functions Disadvantages

As serverless:

- **Latency:** deal with cold starts, not the best idea to use FaaS for high-performance applications
- **Memory Limits:** only 1.5GB memory on Azure functions' Consumption Plan
- **Execution Time Limits:** maximum timeout for Azure functions on Consumption Plan is 10 min
- **New technologies risks:** lack of documents, stability of frameworks, libraries, etc.

As Azure functions:

- **Vendor lock-in:** depend on Azure. What if Microsoft goes bankrupt in 3 years ?
- **Limited Supported Languages compared with lambda:** python not supported, Java in preview version 2.x runtime.

https://assist-software.net/blog/pros-and-cons-serverless-computing-faas-comparison-aws-lambda-vs-azure-functions-vs-google

Cost Discussion

Consumption Plan Pricing:

Monthly billing would be calculated as follows:

- Memory used
- Number of executions

METER	PRICE	FREE GRANT (PER MONTH)
Execution Time*	$0.000016/GB-s	400,000 GB-s
Total Executions*	$0.20 per million executions	1 million executions

https://azure.microsoft.com/en-us/pricing/details/functions/

Cost Example

Suppose you have a function running as Azure functions Consumption Plan:

- each execution has **a duration of one second** and **consumes 512MB memory**.
- executes **3,000,000 times** during the month.

Monthly resource consumption cost:

- 3 million executions X 1 second = 3 million seconds
- 512 MB / 1,024 MB X 3 million seconds = 1.5 million GB-s
- 1.5 million GB-s - 400,000 GB-s = 1.1 million GB-s
- 1.1 million GB-s X $ 0.000016 / GB-s = $ 17.6

Monthly executions cost:

- 3 million executions - 1 million executions = 2 million executions
- 2 million executions X $0.2 / million executions = $ 0.4

Total monthly cost = Monthly resource consumption cost + Monthly executions cost = $17.60 + $0.40 = $18

Comparison

Feature/Providers	Languages support	Pricing	Limits
AWS Lambda	Java, Go, PowerShell, Node.js, C#, Python, and Ruby	0.00001667/GBs, 400,000 GBs/month for free 0.2/million executions, with 1 million executions/month for free	Maximum timeout is 15 minutes Maximum memory limit is 3008 MB
Google Functions	Node.js, Python	0.0000025/GBs, 400,000 GBs/month for free 0.4/million executions, with 2 million executions/month for free	Maximum timeout is 9 minutes Maximum memory limit is 2.0 GB
Azure Functions	Node.js, C#, F#, Java (in preview in 2.x Runtime) ~~Python~~ (not supported in 2.x Runtime)	Consumption Plan: 0.000016/GBs, 400,000 GBs/month for free 0.2/million executions, with 1 million executions/month for free App Service plan: Different pricing strategy: https://azure.microsoft.com/en-us/pricing/details/app-service/windows/	Consumption Plan: maximum timeout is 10 minutes Maximum memory limit is 1.5 GB App Service plan: Offer more cpu and memory no limitations on Maximum timeout.

https://assist-software.net/blog/pros-and-cons-serverless-computing-faas-comparison-aws-lambda-vs-azure-functions-vs-google

Conclusion

- **Serverless:** you don't have to deal with servers, but you also lose control of servers
- **Cost saving:** significantly reduces server cost (70-90%), because you don't pay for idle
- **Numerous ways to deploy your code**
- **Rapid development**
- **Cheaper than lambda**
- **Less mature than lambda:** for version 2.x runtime, java in preview, and python not supported

Suggestions:

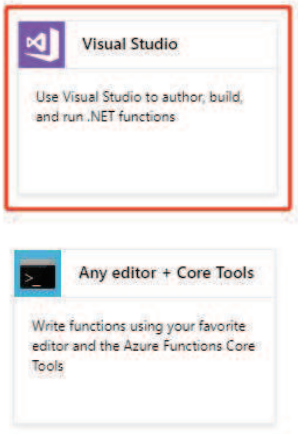
If developing in C# or F# (languages developed by Microsoft): azure functions might be a good choice

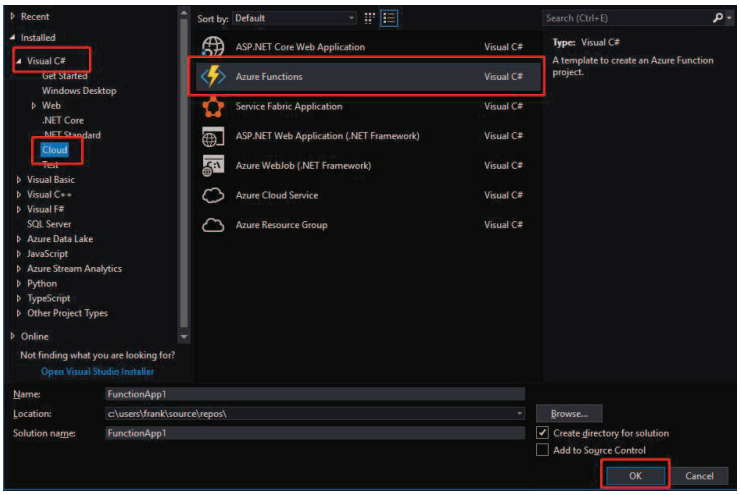
If developing in Java or python: Lambda would be a better choice for now

https://markheath.net/post/azure-functions-choices

Demonstration

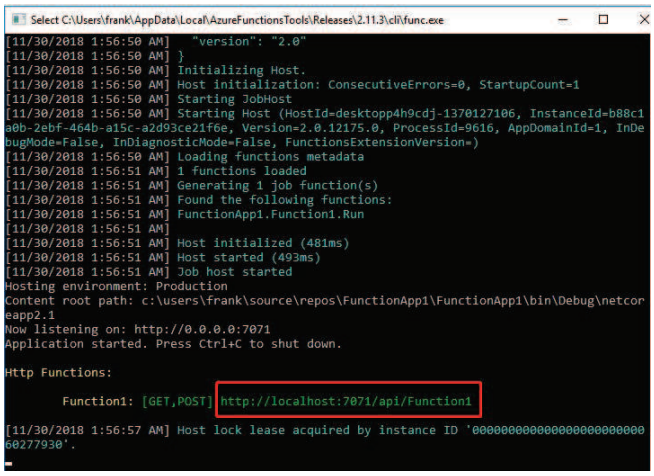
Demo: Visual studio 2017 + C#

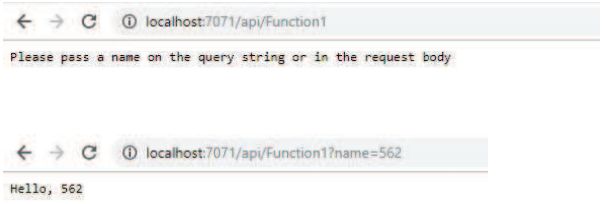




Demonstration

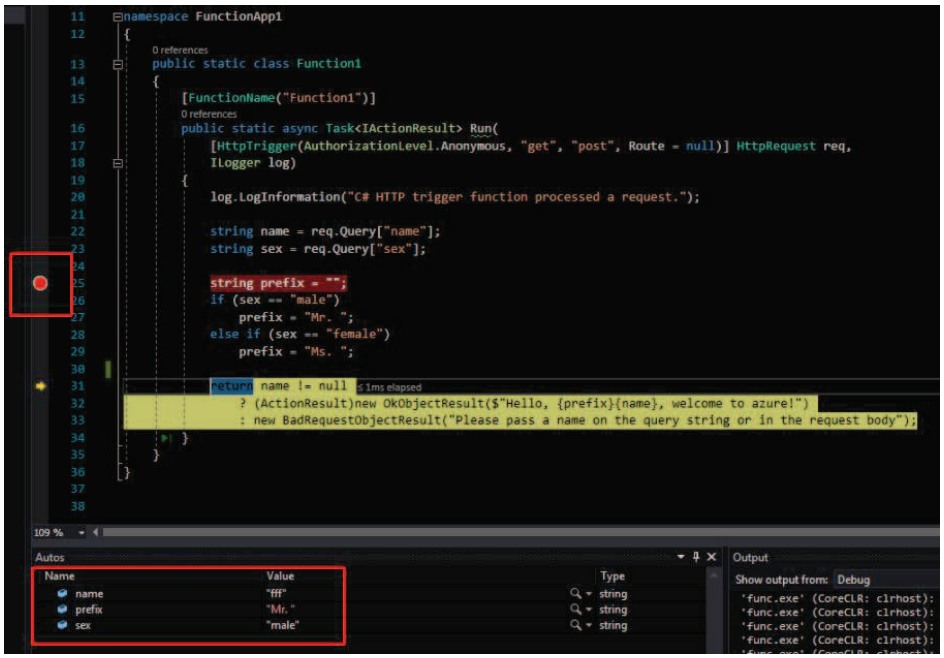
Run locally





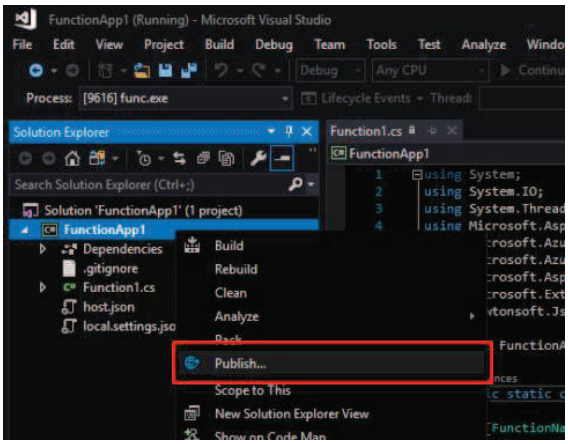
Demonstration

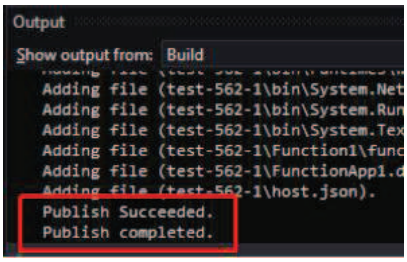
Breakpoint & Debug

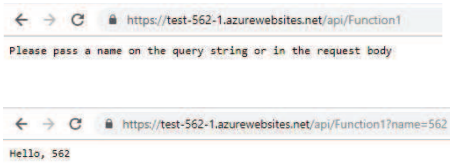


Demonstration

Publish







Questions & Answers

Thank You !