



#GlobalAzure

#CloudGenVerona

@cloudgen_verona

Thanks to all the sponsors



PREMIUM SPONSOR



BASIC SPONSOR



CODICEPLASTICO





TOPIC

Azure Functions Deep Dive

Who I am



ATosato86



andreatosato



andreatosato

Agenda



Azure Function Internal

From v1 to v2

Programming Languages

Deployment

Bindings

Azure Functions Host

Custom Binding

Durable Functions

Premium Plan and scalability

Azure Functions



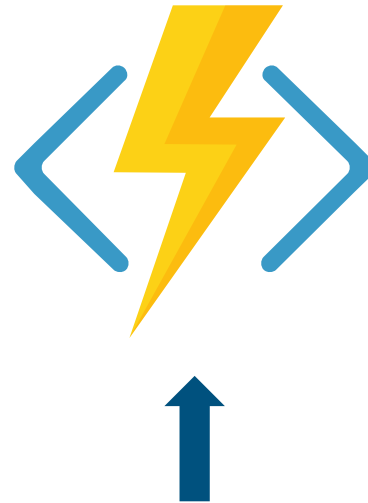
Events



React to timers, HTTP, or events from your favorite Azure services, with more on the way

Code

Author functions in C#, F#, Node.JS, Java, and more



... why not? some inputs

Outputs



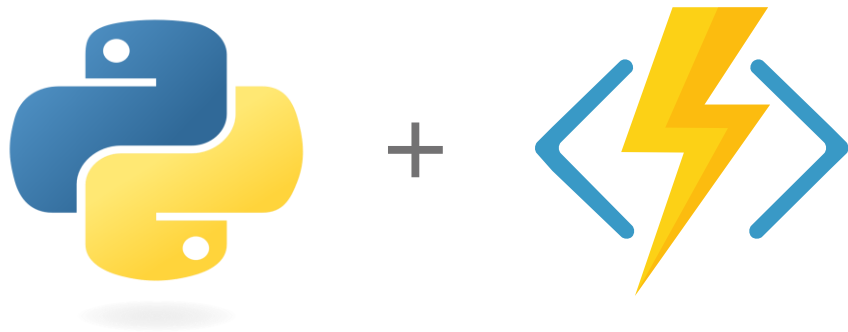
Send results to an ever-growing collection of services

Functions runtime 1.0 vs 2.0



	Functions 1.0	Functions 2.0
.NET Support	.NET Framework 4.7.1	.NET Core 2.1
Assembly isolation	No	Yes
Bindings versions	Runtime versions	User controlled
Language options	Limitations in languages and versions	Languages are external to the host
Node.js version	Node.js 6 only	Node.js 8 & 10 + future versions
Node.js native modules	Not supported	Supported
HTTP triggers	HTTP and specialized Webhooks	HTTP (supports Webhooks)
Language Runtime	Multiple languages per function app	Single language per function app
Functions Proxies	GA	GA
OpenAPI definition	Preview	Not supported
Observability	Application Insights/WebJobs dashboard	App Insights

Azure Functions now supports Python!

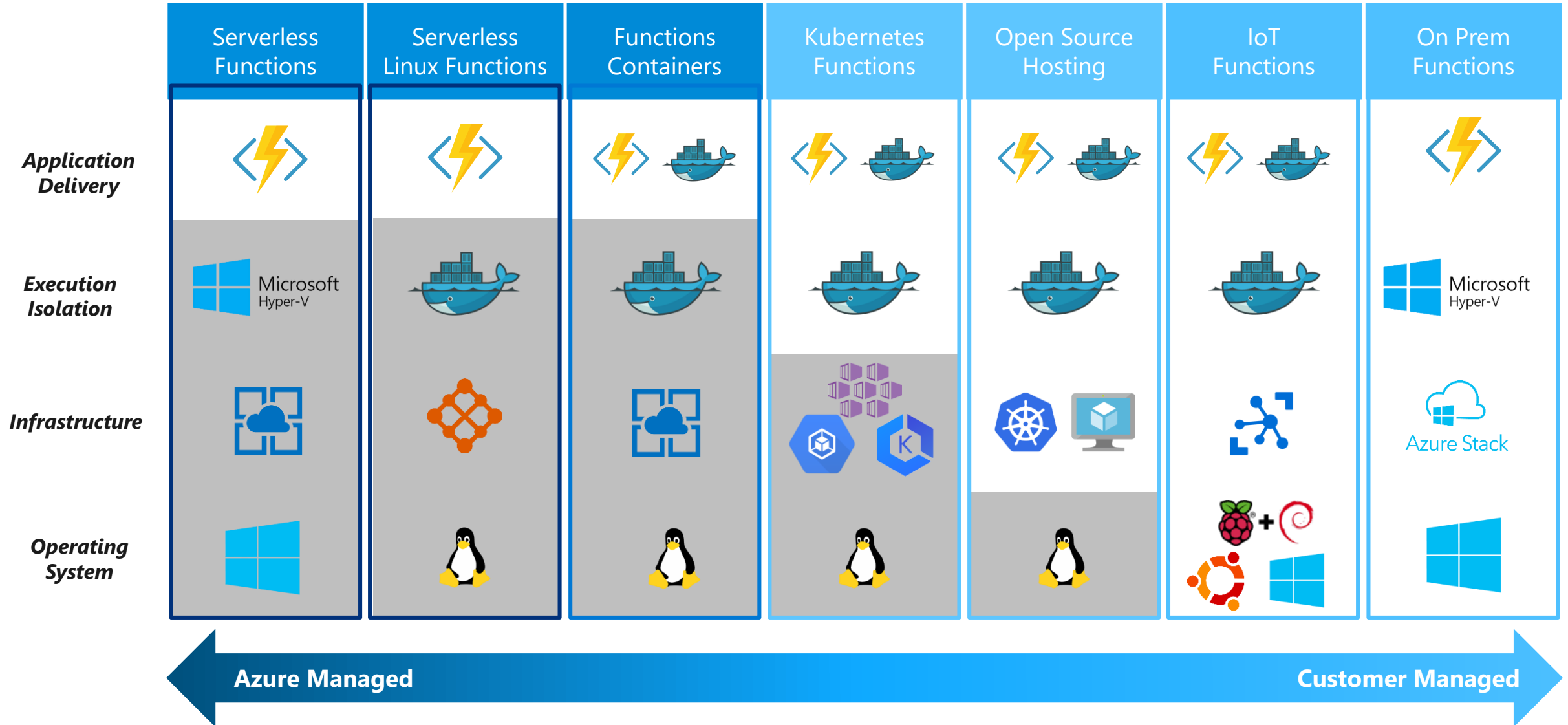


Develop using **Python3.6** on the **Functions v2** runtime

Publish to the **Serverless Linux hosting** platform in Azure

Build, test, debug and publish using Visual Studio Code and the Azure Functions Core Tools (CLI)

Deployment (Hosting) Options



Dockerized Functions Runtime



Type	1.x	2.x ¹	Trigger	Input	Output
Blob Storage	✓	✓	✓	✓	✓
Cosmos DB	✓	✓	✓	✓	✓
Event Grid	✓	✓	✓		
Event Hub	✓	✓	✓		✓
HTTP e Webhook	✓	✓	✓		✓
Microsoft Graph Events		✓	✓	✓	✓
Queue Storage	✓	✓	✓		✓
Service Bus	✓	✓	✓		✓
SignalR		✓		✓	✓
Table Storage	✓	✓		✓	✓
Timer	✓	✓	✓		

`docker run -d -p 5010:80 mcr.microsoft.com/azure-functions/dotnet:latest`
https://hub.docker.com/_/microsoft-azure-functions-base

Bindings and integrations



Functions 1.0

Microsoft.NET.Sdk.Functions
(.NET Framework 4.6)

- HTTP
- Timer
- Storage
- Service Bus
- EventHubs
- Cosmos DB

Functions 2.0

Microsoft.NET.Sdk.Functions
(.NET Standard 2.0 - .NET Core 2.2)

- HTTP
- Timer

Microsoft.Azure.WebJobs.Extensions.Storage

Microsoft.Azure.WebJobs.Extensions.ServiceBus

Microsoft.Azure.WebJobs.Extensions.EventHubs

Microsoft.Azure.WebJobs.Extensions.CosmosDB

Microsoft.Azure.WebJobs.Extensions.EventGrid

Microsoft.Azure.WebJobs.Extensions.MicrosoftGraph

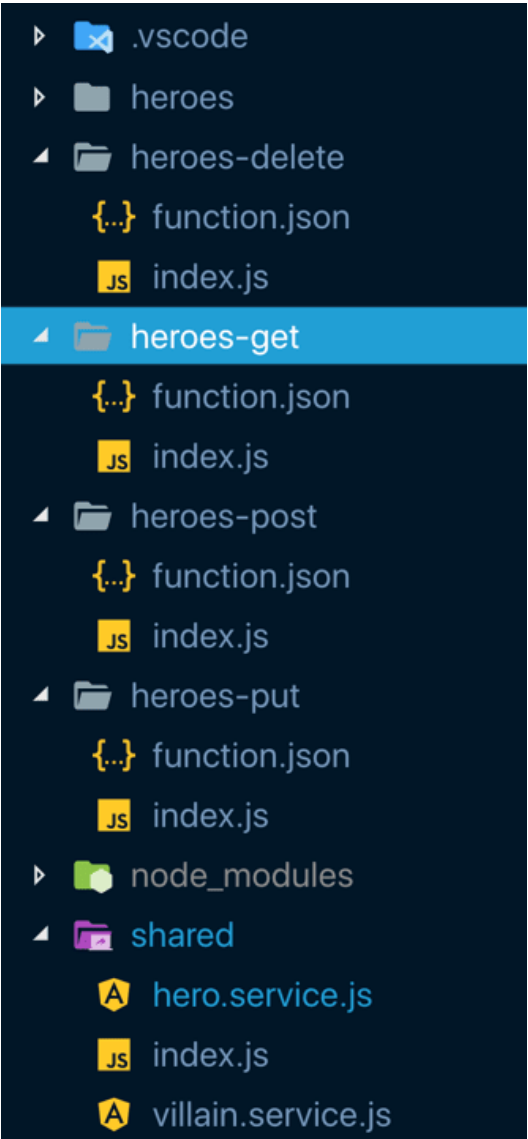
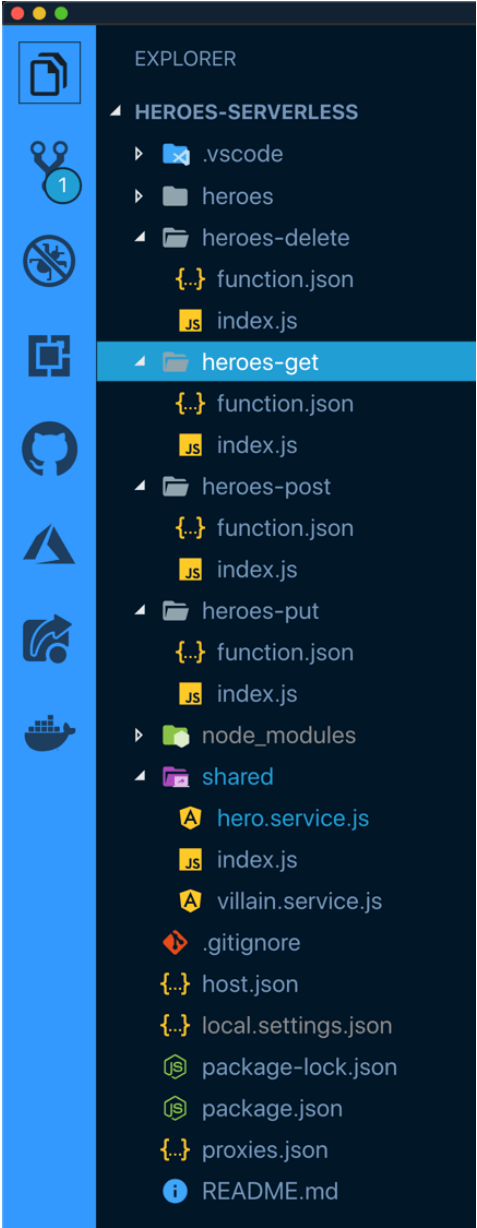
Microsoft.Azure.WebJobs.Extensions.DurableTask

Microsoft.Azure.WebJobs.Extensions.SignalRService

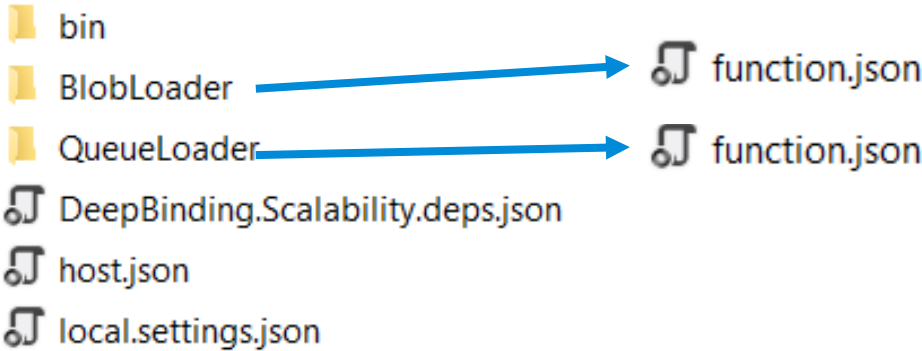
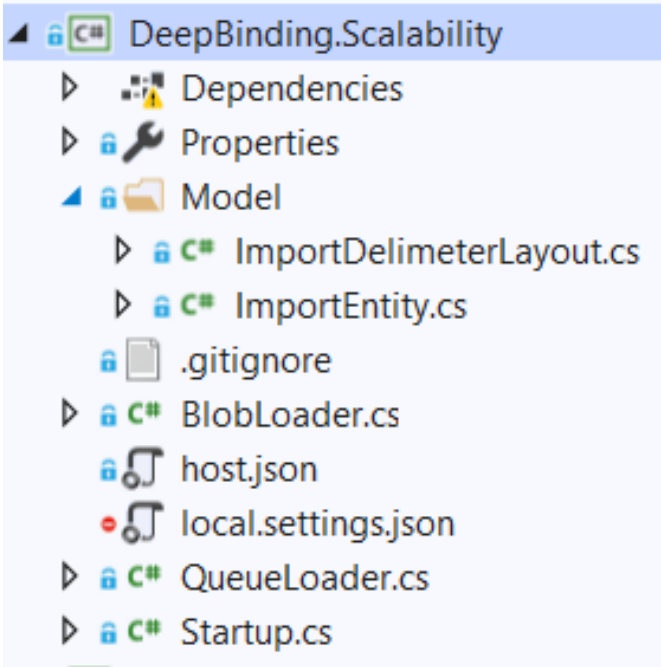
Folder and programming languages



JS



C#



Assembly Isolation - Bindings: 1.0 Model



Azure Functions Host – Functions 1.0

Host Assembly Load Context (default)

Job host

Bindings

Runtime dependencies

WindowsAzure.Storage 7.2.0

LoadFrom Context

Functions Assemblies

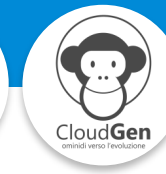
Functions dependencies

WindowsAzure.Storage 9.3.2

```
public static Run(..., CloudBlockBlob blob,...)
{
    //Function code...
}
```



Assembly Isolation - Bindings: 2.0 Model



Azure Functions Host – Functions 2.0

Host Assembly Load Context (default)

Job host

Runtime dependencies

WindowsAzure.Storage 9.3.1

Function Assembly Load Context

Functions Assemblies

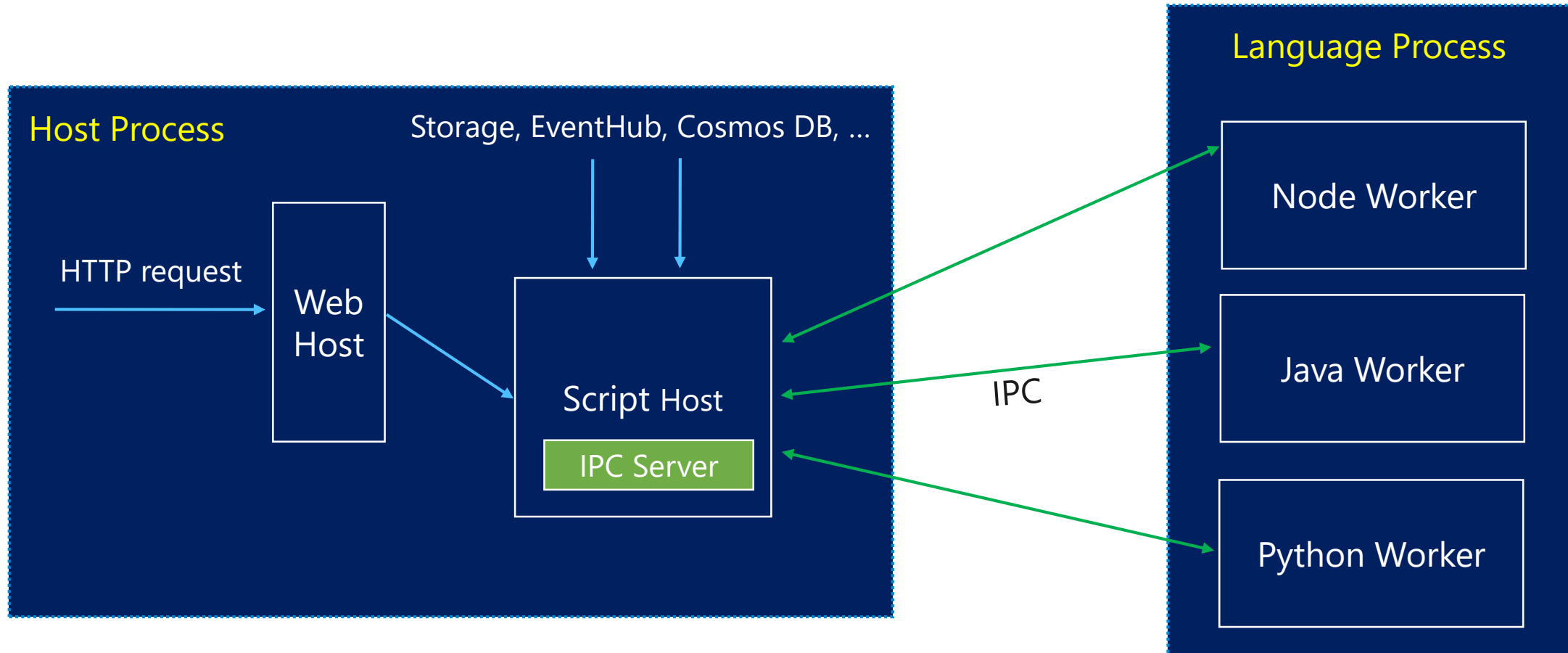
Bindings

Functions dependencies

WindowsAzure.Storage 9.5.0

```
public static Run(..., CloudBlockBlob blob,...)
{
    //Function code...
}
```

Language Extensibility



- Worker and host broken into 2 separate processes
- Development of new language workers can happen independently
- Worker process crashes doesn't bring down the host

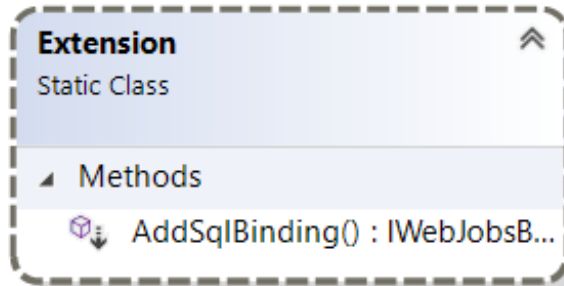


Demo

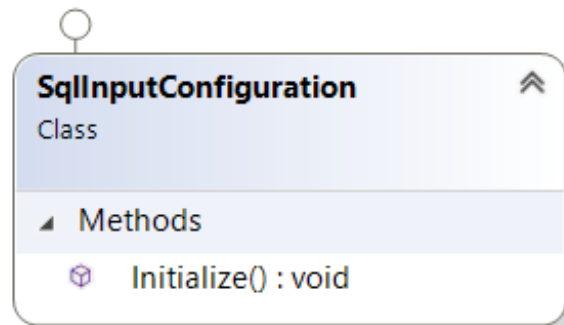
Custom Binding

<https://github.com/andreatosato/GAB2019>

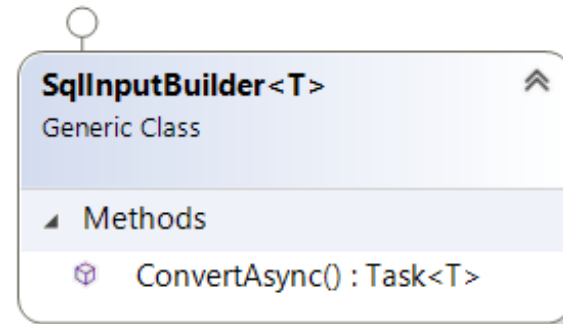
SQL Input Binding



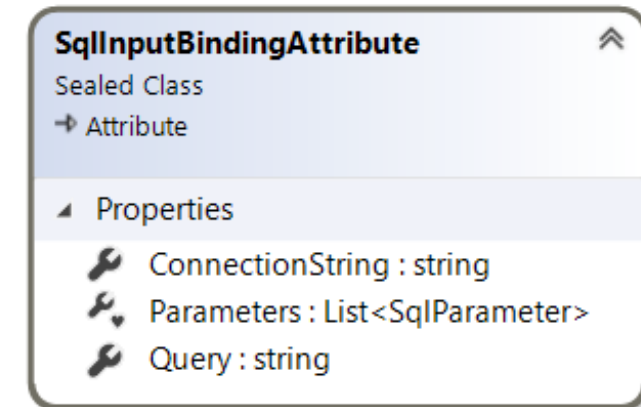
Create Method for easy registration



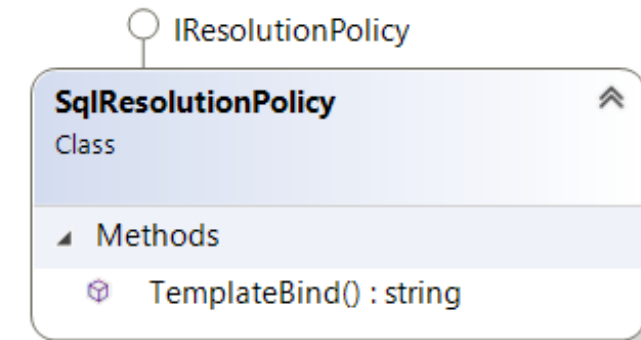
Register attribute, validation, and business logics



Take attribute parameters and return Entity from DB



Define attribute for Azure Functions



Check parameter property of attribute

SQL Output Binding



Extension
Static Class

Methods

AddSqlBinding() : IWebJobsBuilder

Create Method for easy registration

Register attribute, validation, and business logics

SqlOutputConfiguration
Class

Methods

BuildFromAttribute() : IAsyncCollector<IWriterEntity>

Initialize() : void

SqlOutputAsyncCollector<T>
Generic Class

Fields

attr : SqlOutputBindingAttribute

Methods

AddAsync() : Task

FlushAsync() : Task

SqlOutputAsyncCollector()

Take attribute parameters and insert entities into DB

SqlOutputBindingAttribute
Sealed Class
Attribute

Properties

ConnectionString : string

Define attribute for Azure Functions

Interface for define Entity restriction

IWriterEntity
Interface

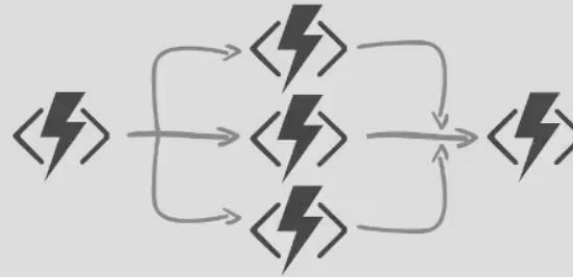
Properties

Id : int

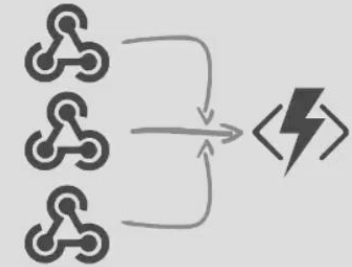
Durable Functions



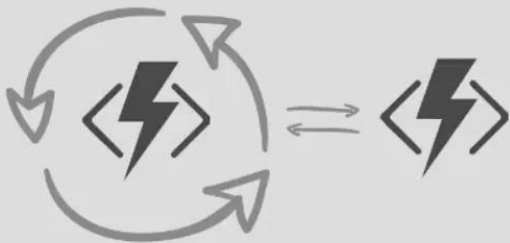
Manageable Sequencing
+ Error Handling / Compensation



Fanning-out & Fanning-in



External Events Correlation



Flexible Automated Long-running
Process Monitoring





Http-based
Async Long-running APIs



Human Interaction

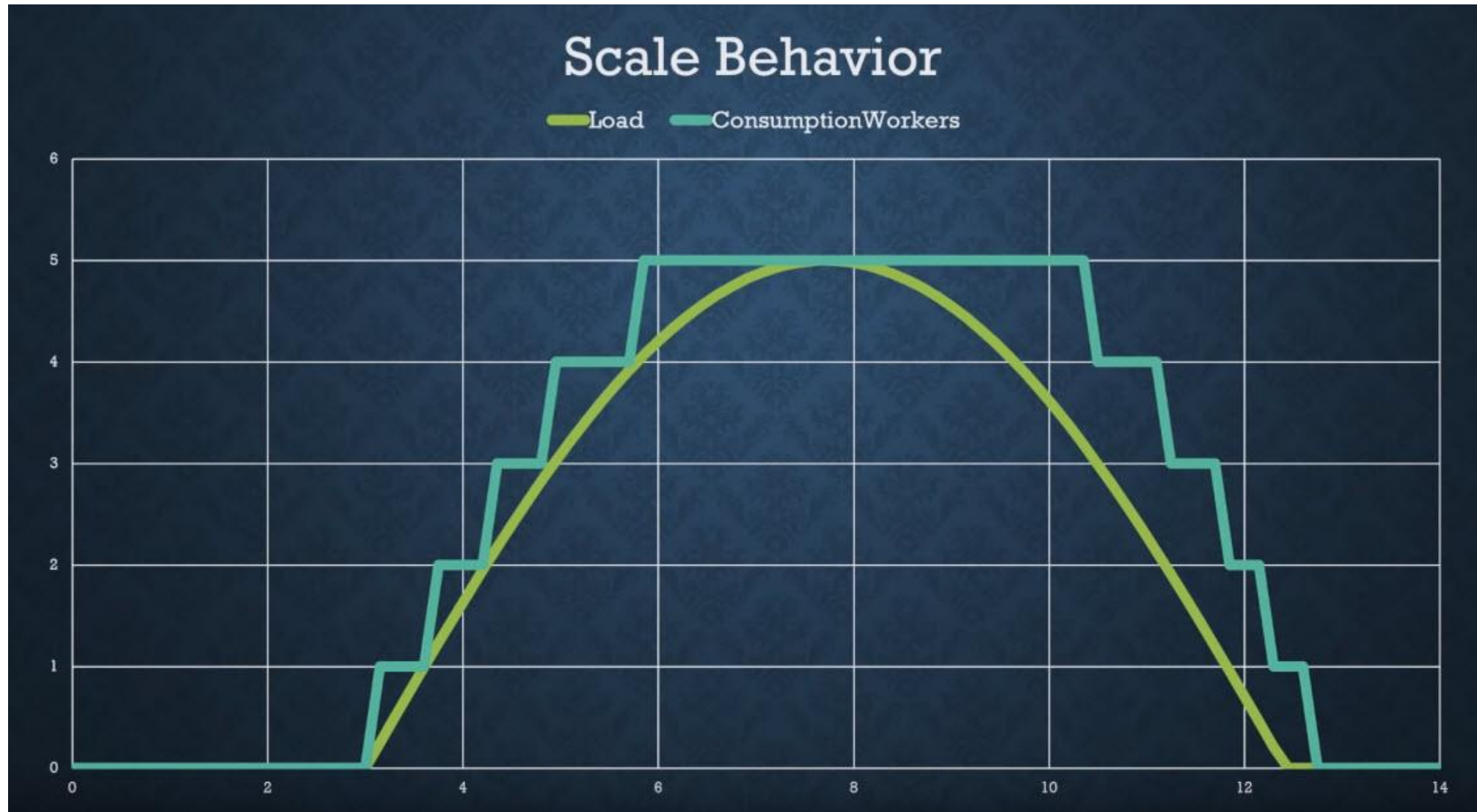
Premium Plan



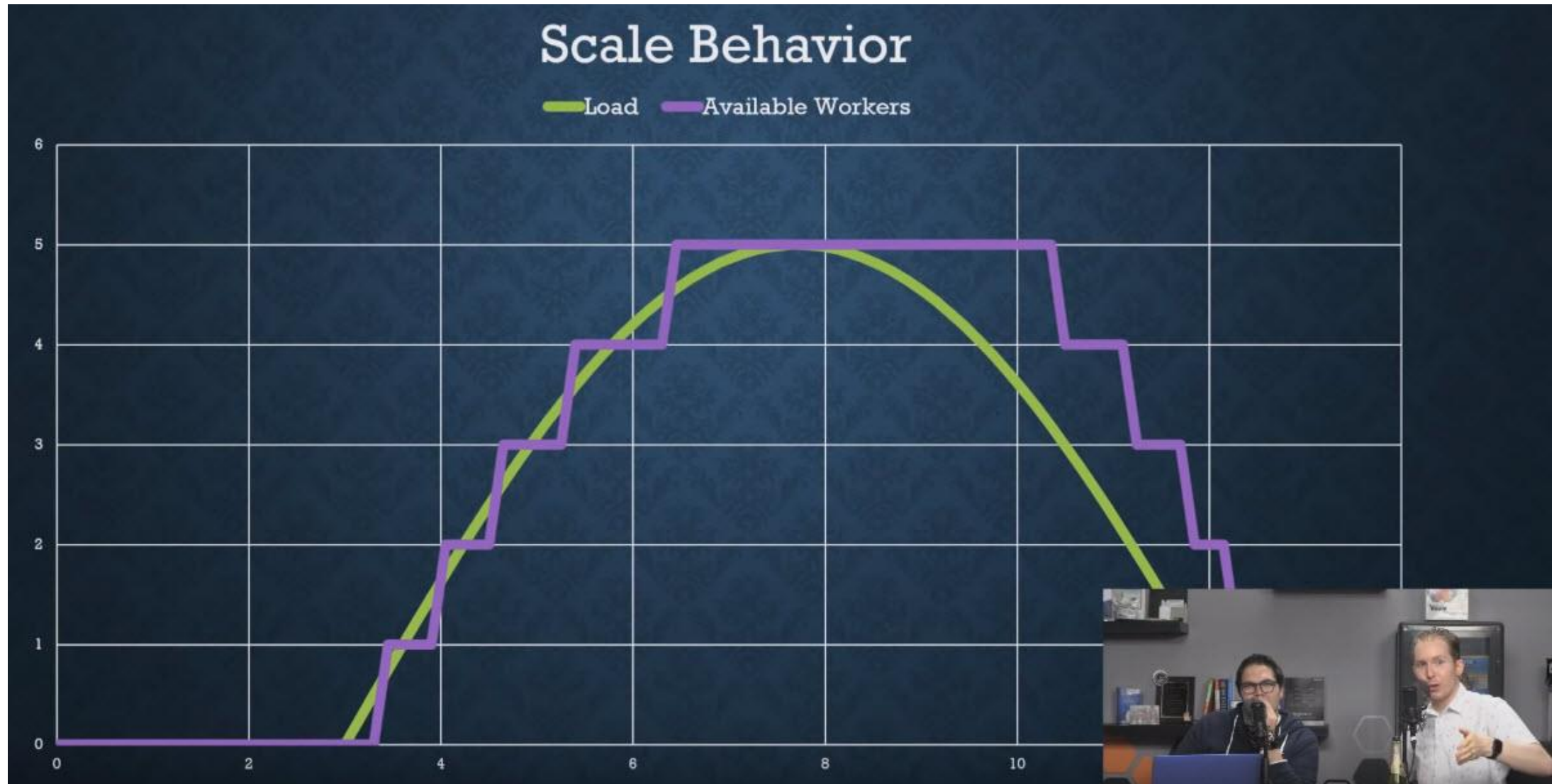
	Consumption Plan 	-New- Premium Plan (Preview) 
Instance Size	Fixed at one core and 1.5Gb of memory	Configurable up to 4 cores and 14Gb of memory
Scaling	Event driven scaling	Event driven scaling
Scale Controls	None	Set min and max instances
Private Networking	None	VNET integration
Warmup Time (Cold Start)	Your app must be loaded after it is inactive	No delay after your app is inactive and scale instantly to pre-warmed instances
Cost	Consumption	Consumption and at least 1 pre-warmed instance per plan

- **Maximum Instances**
allows you to predict your maximum possible bill each month
- **Minimum Instances**
- **VNET**
connect to a VNET and securely access resources in a private network (already in App Service Plan)

Scale Behavior



Scale Behavior

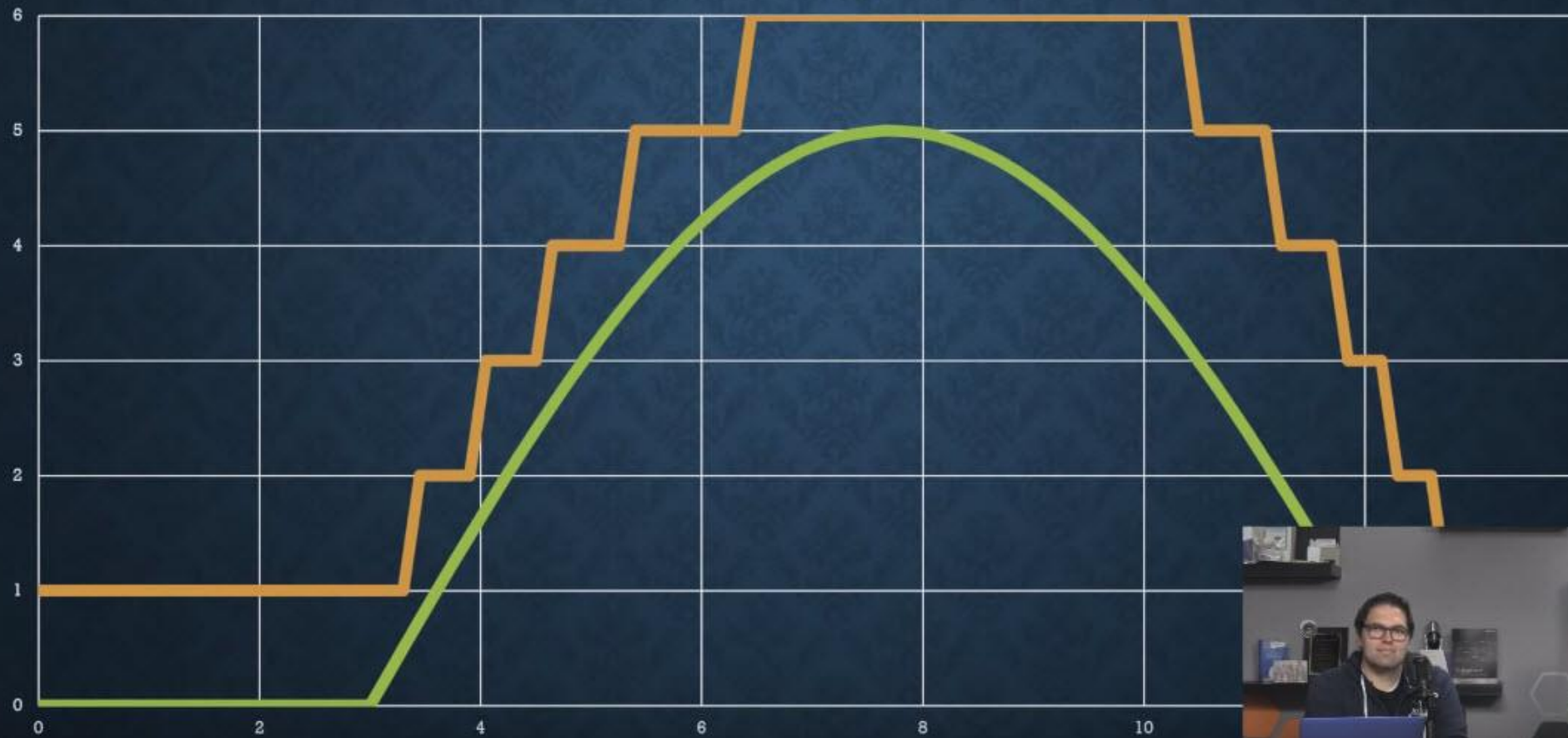


Scale Behavior



Scale With One Pre-Warmed Instance

Load Elastic Premium Workers (1 Dedicated)



Premium Plan



Sviluppo/test

Per i carichi di lavoro meno complessi



Produzione

Per la maggior parte dei carichi di lavoro di produzione



Isolato

Scalabilità e rete avanzate

Piani tariffari consigliati

S1

100 ACU totali
1.75 GB di memoria
Equivalente alle risorse di calcolo della serie A
62.72 EUR/mese (costi stimati)

P1V2

210 ACU totali
3.5 GB di memoria
Equivalente alle risorse di calcolo della serie Dv2
125.44 EUR/mese (costi stimati)

P2V2

420 ACU totali
7 GB di memoria
Equivalente alle risorse di calcolo della serie Dv2
250.95 EUR/mese (costi stimati)

P3V2

840 ACU totali
14 GB di memoria
Equivalente alle risorse di calcolo della serie Dv2
501.90 EUR/mese (costi stimati)

EP1

210 ACU totali
3.5 GB di memoria
Equivalente alle risorse di calcolo della serie Dv2
125.44 EUR/mese (costi stimati)

EP2

420 ACU totali
7 GB di memoria
Equivalente alle risorse di calcolo della serie Dv2
250.95 EUR/mese (costi stimati)

EP3

840 ACU totali
14 GB di memoria
Equivalente alle risorse di calcolo della serie Dv2
501.90 EUR/mese (costi stimati)

Visualizza opzioni aggiuntive

EP1

210 total ACU
3.5 GB memory
Dv2-Series compute equivalent

EP2

420 total ACU
7 GB memory
Dv2-Series compute equivalent

EP3

840 total ACU
14 GB memory
Dv2-Series compute equivalent

 Scalabilità orizzontale elastica
 premiumplangab2019

In the Premium plan there is the ability to specify a number of **pre-warmed** instances that are kept warm with your code ready to execute.

When your application needs to scale, it first uses a pre-warmed instance with no cold start.

Your app immediately pre-warms another instance in the background to replenish the buffer of pre-warmed instances.

This model allows you to **avoid any delay on the execution for the first request** to an idle app, and also at each scaling point.



“ Serverless computing refers to a cloud-computing execution model in which the **cloud provider** runs the server, and **dynamically** manages the allocation of machine resources.

WIKI



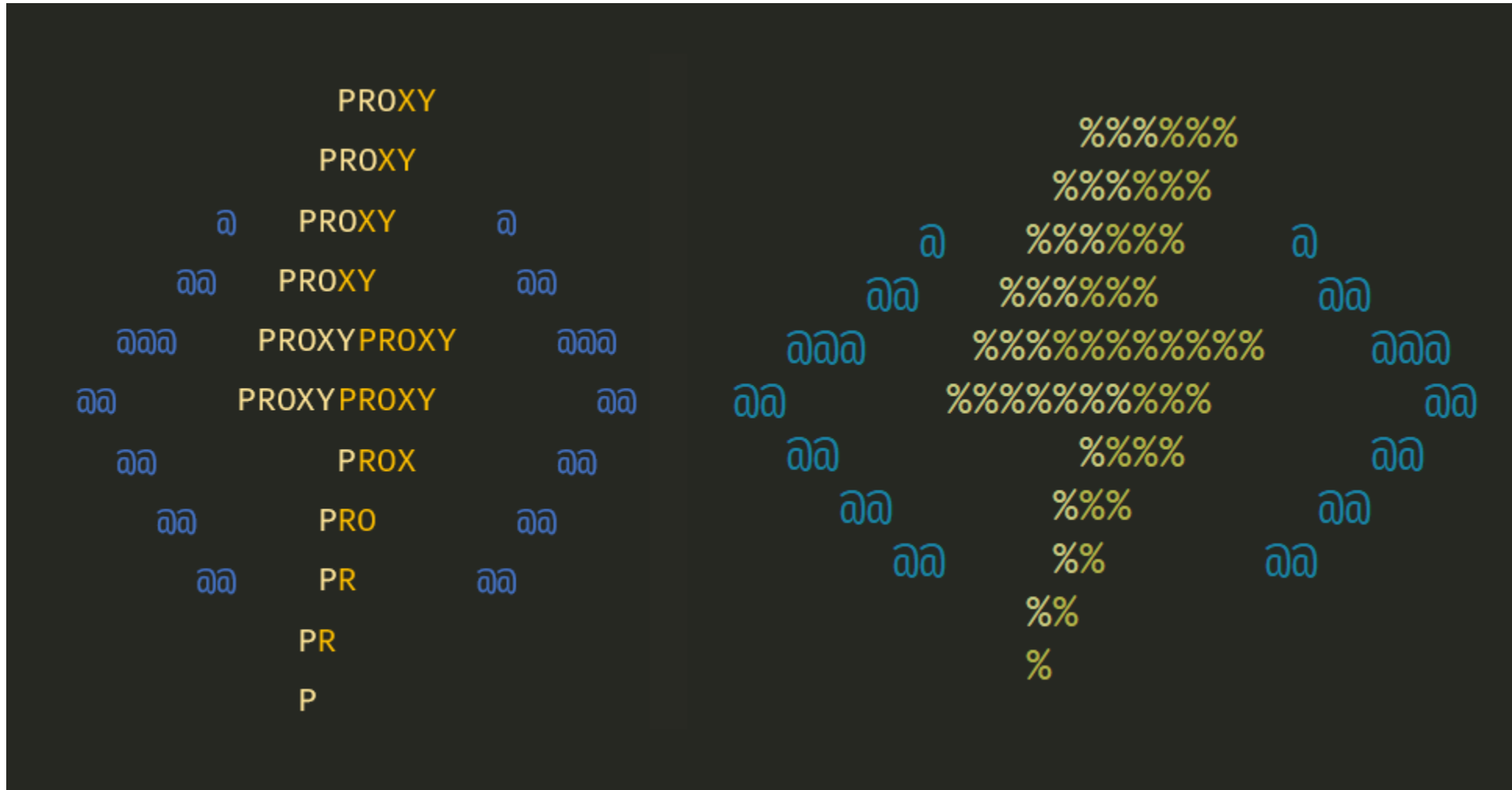
Demo Premium plan



SwaggerTM

Supported by **SMARTBEAR**

Proxy





Classic Deployment Issues:

1. Not atomic => inconsistent files
2. Files in use get locked
3. Multi-region inconsistencies
4. Difficult rollback

Solutions:

1. Externally hosted zip file
2. Zip file hosted within your app

Deployment Slot



Portal



Function Monkey

Write testable more elegant Azure Functions with less boilerplate, more consistency, and support for REST APIs.

 Getting Started

 View on GitHub

<https://functionmonkey.azurefromthetrenches.com/index.html>

Azure Functions – What's coming



1.Bundles for extensions with runtime – Bundles takes all of the extensions that you will install in current V2 functions to use any bindings. So, if you want any trigger type or binding other than HTTP, like Storage, Cosmos DB, Event Hub, Event Grid those all require an extension. These extensions require a nuget install, so with bundles, it takes all of those packages and put them in one functions platform bundle that you can reference.

2..NET Dependency injection (so soon!).

3.Azure DevOps improvements

4.PowerShell support for v2 – Available in *Private preview now*.

5.Super secret stuff for Build – The team is building a super secret feature for Build.



Thanks

Questions?



andreatosato



ATosato86



andreatosato