



365<sup>+</sup>  
*Saturday*

# URDU HINDI MICROSOFT 365 BOOTCAMP

19th December 2020

[powercommunity.com](http://powercommunity.com)



+

A man with a beard and glasses, wearing a dark blazer over a white shirt and dark shorts, is running from left to right. He is positioned on a red wavy background that resembles a desert landscape. His shadow is cast on the wall behind him.

Track – Dataverse / Azure

# Extending Dataverse using Azure Functions



# Allan De Castro

CRM Power Platform Consultant

Paris Power Platform Happy Hour

[Linkedin.com/in/allandecastro/](https://www.linkedin.com/in/allandecastro/)

[twitter.com/decastroallan](https://twitter.com/decastroallan)

[www.blog.allandecastro.com](http://www.blog.allandecastro.com)



# Activities & timeline

Here is the schedule of this Extending Dataverse with Azure Function session! Hopefully, you will like it...

What is an Azure Function?

10 Minutes

- Serverless Concept & Components
- Azure Function Definition
- Benefits

When to use an Azure Function and how much it costs ?

15 Minutes

- Dataverse constraints
- Pricing Model

Creating an Azure Function from Azure Portal & VS2019

20 Minutes

- Live Demo
- Azure Portal
- Visual Studio 2019 set up

Scenario And Trigger From Dataverse

20 Minutes

- Scenario
- Dataverse Plugin
- WebHook

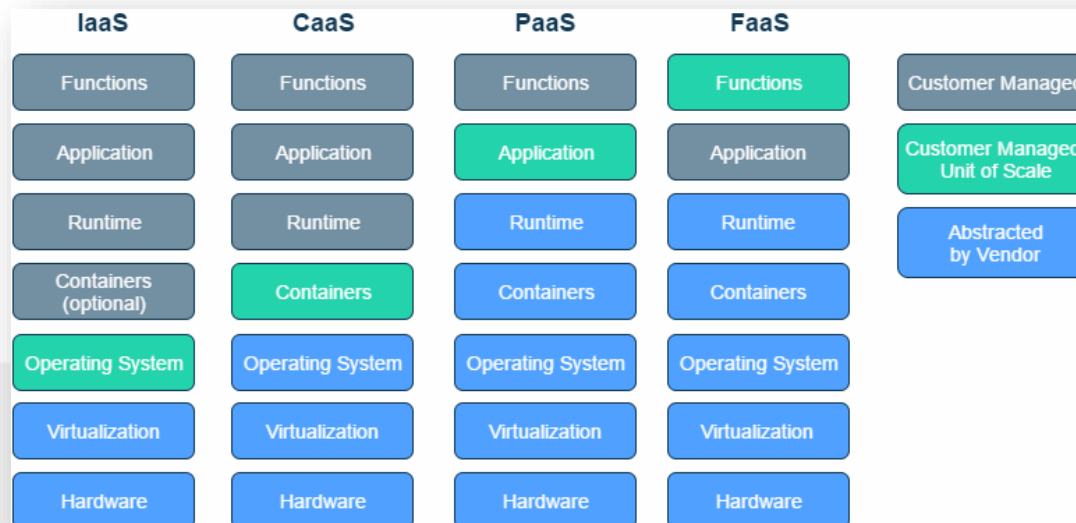


# What is an Azure Function?

# Serverless Concept

Executing a “piece of code” without really caring about managing the allocation of resources used on the machines.

- Cloud provider (here Azure) runs the server and adapts the allocation of machine resources.
- Piece of code is called “FaaS” for **Function as a Service**






# Azure Serverless Components

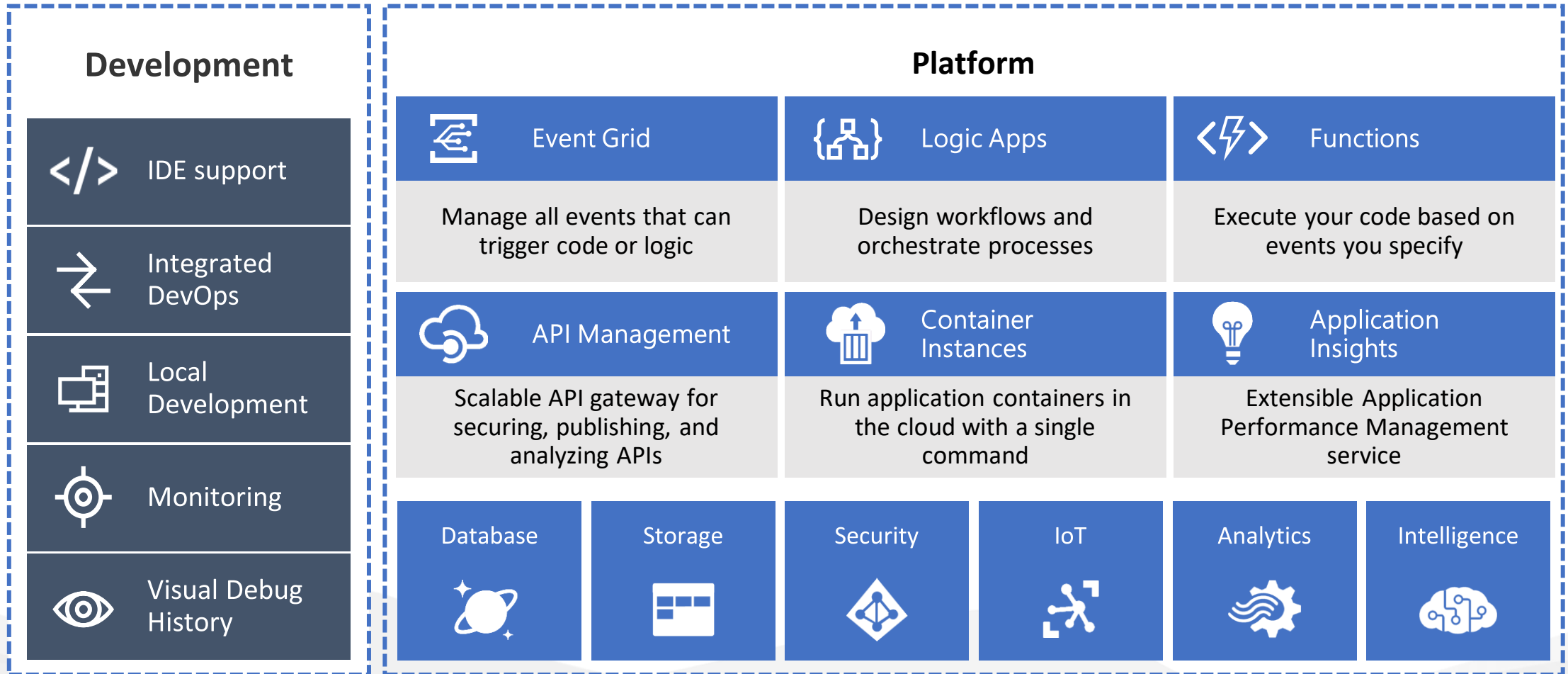
**Azure Function:** Used to host business logic implemented in a specific language by a developer and executed without managing the infrastructure.

**Azure Logic Apps (not covered):** Allows creating workflows using connectors.

**Event Grid (not covered):** A messaging service built to easily build application with event-based architectures.

 Event Grid	 Logic Apps	 Functions
Manage all events that can trigger code or logic	Design workflows and orchestrate processes	Execute your code based on events you specify

# Azure Serverless Application Platform





# Azure Function

*Azure Functions is a solution for easily running **small pieces of code**, or “functions,” in the cloud. You can write just the code you need for the problem at hand, **without worrying** about a whole application or the infrastructure to run it.*

Executed **on demand** or for **specific events** such as the arrival of a message in a Service Bus for example.

**Trigger(s):** How a function is invoked, only one trigger.

**Code:** Implement the business logic using a specific language.

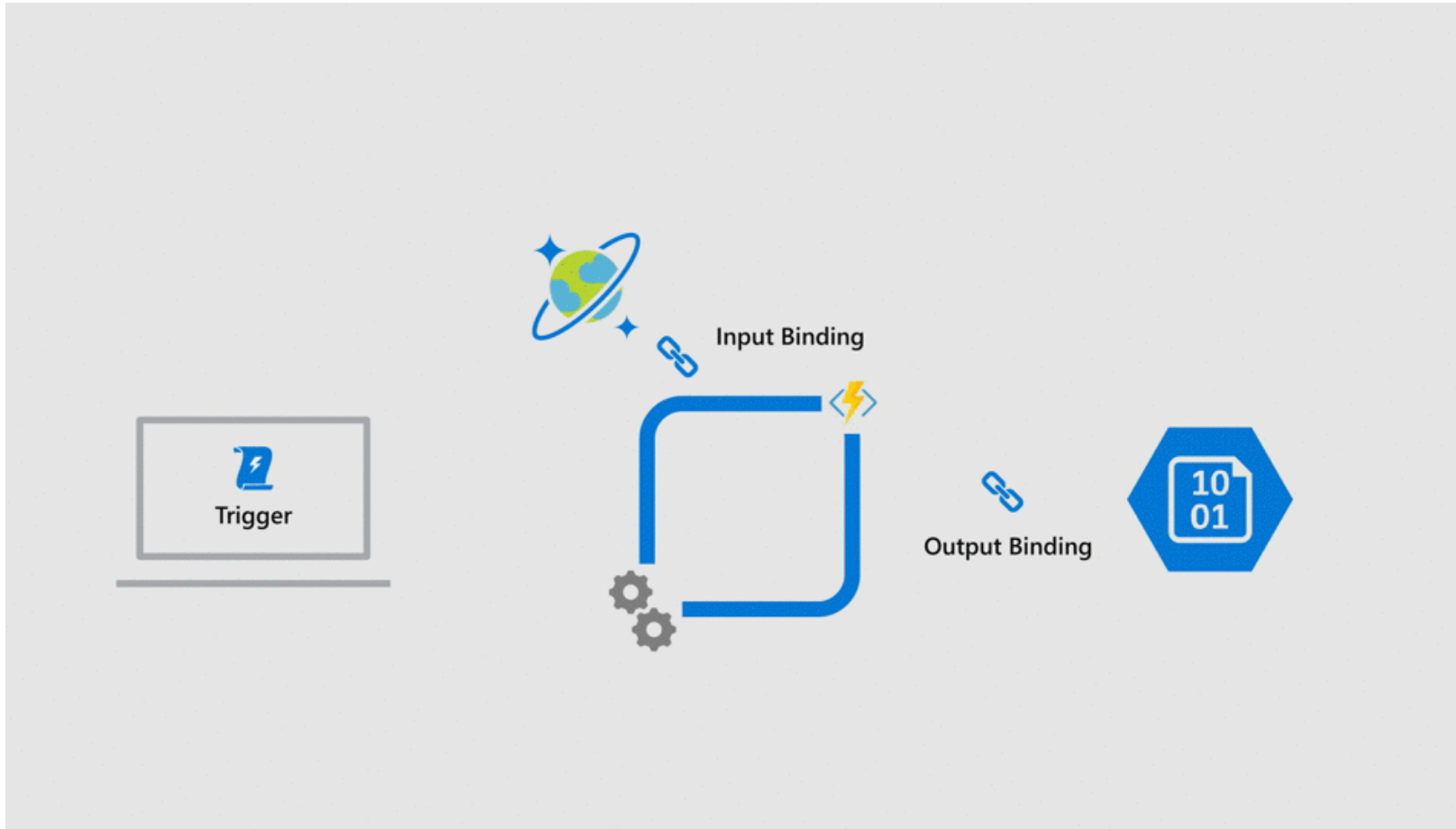
**Bindings:** Optional. Connecting another resource to the function.



# Azure Function: Trigger and Binding

Example scenario	Trigger	Input binding	Output binding
A new queue message arrives which runs a function to write to another queue.	Queue	None	Queue
A scheduled job reads Blob Storage contents and creates a new Cosmos DB document.	Timer	Blob Storage	Cosmos DB
The Event Grid is used to read an image from Blob Storage and a document from Cosmos DB to send an email.	Event Grid	Blob Storage and Cosmos DB	SendGrid
A webhook that uses Microsoft Graph to update an Excel sheet.	HTTP	None	Microsoft Graph

# Azure Function: Trigger and Binding





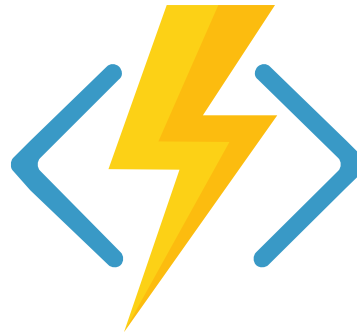
# Azure Function

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

Outputs



Send results to an ever-growing collection of services

# Azure Function Benefits

**Event-Driven:** they can be called in many ways via triggers like HTTP Trigger, a message arriving in a queue or a topic ...

**Stateless:** they can be executed, consumed, and destroyed on demand.

**Timeout:** Depending on the chosen consumption plan they can run with a timeout of up to 10 minutes.

**Scalability:** At any time, you can scale out or down depending on load.

**Easily Integrate:** It is quite easy to integrate it with other services, especially when exposed as an HTTP API endpoint.

- In Dataverse, you can also register it as a WebHook using Plugin Registration Tool.

**Languages:** You can use your development language of choice, such as C#, F#, Node.js, Python or PHP.

**Retry:** Implement a retry pattern to counter a momentary loss of network connectivity to components and services, temporary service unavailability, or timeouts that arise when a service is busy.

# When to use an Azure Function and how much it costs ?

- Examples
- Pricing Model





# Examples

**Reusable Business Logic:** As mentioned above, it can be a good idea to implement a business logic via an Azure Function if the Azure Function can be used in different systems, so you don't have to implement the same logic in all of them!

**Dataverse development constraint:** Within a Dataverse environment, there are several possibilities to implement business logic but there are some limitations:

- Timeout of 2 minutes.
- Only HTTP/HTTPS protocols are allowed.
- Limited use of assemblies.
- Runs on the resources of the Dataverse server.
- Scheduled Workflows

**Integration:** In integration projects between different systems they are often used for their ability to be triggered to insert or send data, either directly in the system or to another Azure component.



# Pricing Model

## **App Service plan (same as App Service apps):**

- Scale between tiers to allocate a different amount of resources.
  - Run on VMs (basic, standard, premium..)
- Ensure to enable the "Always On" setting. Runtime may become inactive after a few minutes of inactivity, so that only the HTTP triggers will "wake up" your functions.
- No additional cost if you already have an App Service for another application.
- To be used when there is a need for continuous operation, or if there is any limitation in the consumption plan.



# Pricing Model

## Consumption Plan:

- Allocates computation power while your code is running. With this, you are billed only when your function is executed.
- Azure function never goes idle when running in Consumption Plan.
- Two components in consideration:
  - **Number of Execution** : any time a function is executed (i.e. triggered), it counts as an execution
  - **Resource consumption per second**: Amount of memory (RAM) that is used and for how long translated into a number of Gigabytes-seconds (GB-s) (hard to predict)

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



# Live demonstration

- Creating an Azure Function from Azure Portal
- Set up VS2019 Project
- Scenario And Trigger From Dataverse
  - Scenario
  - Dataverse Plugin
  - WebHook



# Scenario

Simple scenario whose aim will be to manage accounts creation.

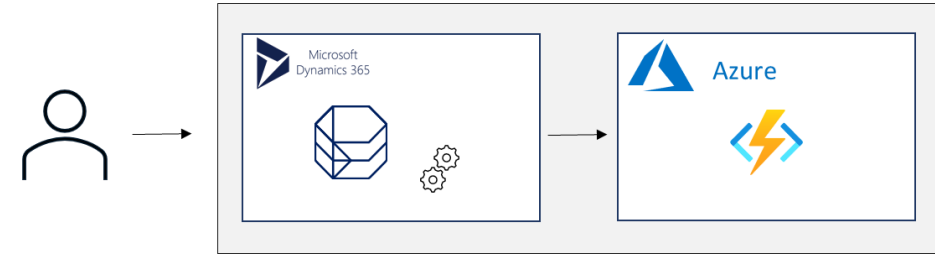
- When an account is created an Azure Function must be triggered.

Two ways of triggering an Azure Function from a Dataverse event, in our case the creation of an Account, using Plugin and WebHook.

Information from Plugin call :

- Account Name: Required Field
- Account Number: Required Field
- Account Type: Required Field
- Created By (Name): Required Field
- Website: Not Required

# Scenario



## Call using **Dataverse Plugin**:

- Make sure that all required fields are present in the Target Entity and the Post Image.
- Make sure that the mandatory fields contain data.
- Create the message using the Target entity and Post Image data.
- Send the message by calling the Azure Function

## Call using **Webhook** registration

- Registering WebHook (Function Key)
- Observation
- Deserialize RemoteExecutionContext





**DEMO**

# Closing

Depends on the context and the different issues you encounter.

- Plugin:
  - Custom JSON (control what we sent, reusable logic...)
  - Minimize the information sent
  - Minimize parsing in Azure Function
- Webhooks:
  - Additional ALM steps
  - Security Issues
  - Complex code ( retrieved related entities)

# Links / References

**Articles Series on my blog** (<https://blog.allandecastro.com>):

**Episode 1:** [Extending Common Data Service using Azure Function – Part 1: Introduction](#)

**Episode 2:** [Extending Common Data Service using Azure Function – Part 2: Outgoing Scenario](#)

**MS Learn - Create Serverless Logic with Azure Function:** <https://docs.microsoft.com/en-us/learn/modules/create-serverless-logic-with-azure-functions/>

**MS Learn - Execute Azure Function with triggers:** <https://docs.microsoft.com/fr-fr/learn/modules/execute-azure-function-with-triggers/>

**Pricing Details:** <https://azure.microsoft.com/en-us/pricing/details/functions/>

**Ebook - Designing Distributed Systems:** <https://azure.microsoft.com/en-us/resources/designing-distributed-systems/>

**Ebook - Azure Serverless Computing Cookbook, Third Edition:** <https://azure.microsoft.com/en-us/resources/azure-serverless-computing-cookbook/>



A pug dog with a fawn-colored face and dark ears is peeking over the back of a dark grey, textured armchair. The dog's eyes are wide and looking towards the camera. The background is a plain, light-colored wall.

# THE END