Microsoft

Microsoft Ignite The Tour

Microsoft

BRK30167
The art of Azure Functions (unit) testing and monitoring

Massimo Bonanni

Paranormal Trainer, with the head in the Cloud and all the REST in microservices!

massimo.bonanni@microsoft.com

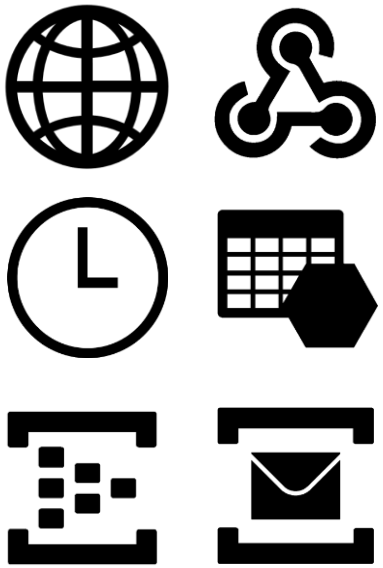
@massimobonanni

The issue….

If you want to use **Azure Functions** as a components of your **Enterprise solutions,** you **must** to test and monitor !!!

What are Azure Functions

Events

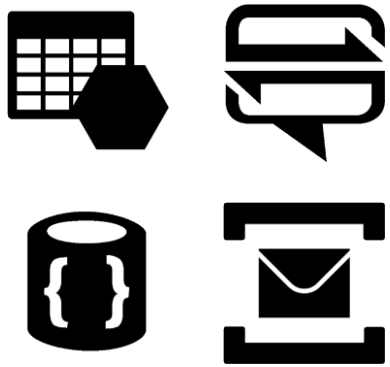


React to timers, HTTP, or events from your favorite Azure services, with more on the way

Code

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

Outputs



Send results to an ever-growing collection of services

What is a Unit Test

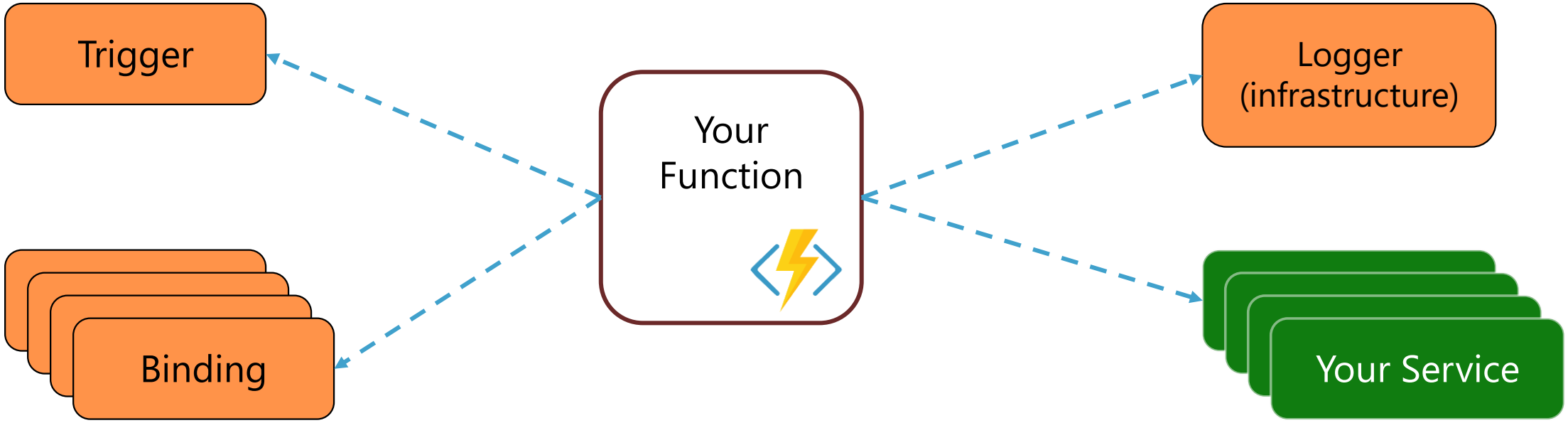
In a **unit test** you invoke a piece of your code with a set of parameters and you checks the correctness its behavior.

In a **unit test** you must substitute all your external reference with a **mock** or **stub**.

Mock *is for the software what a* ***dummy*** *is for a car crash test (you don't test a car with a human being inside…I Hope!!)*

Azure Functions Dependencies

You **should implement** your Azure Functions to allow you to use mock/stub for all external reference!



Azure Function … untestable!!

```
public static class MortgageFunctions
{
    private static readonly IMortgageCalculator mortgageCalculator =
            new MortgageCalculator(null);

    [FunctionName(FunctionNames.MortgageCalculatorFunction + "STATIC")]
    0 references | Massimo Bonanni, 168 days ago | 2 authors, 2 changes
    public static async Task<IActionResult> Run(
        [HttpTrigger(AuthorizationLevel.Function, "get", Route = null)] HttpRequest req,
        [Table("executionsTable", Connection = "StorageAccount")] ICollector<ExecutionRow> outputTable,
        ILogger log)
    {
        log.LogInformation($"{FunctionNames.MortgageCalculatorFunction} start");

        // Retrieve loan, interest and numberOfPayments from HTTP Request
        [ Retrieve request parameters ]

        var calculatorResult =
            await mortgageCalculator.CalculateMontlyRateAsync(loan, interest, nPayments);

        [ Create the response ]

        if (calculatorResult.Succeed)
        {
            return new OkObjectResult(calculatorResult.Result);
        }

        return new BadRequestObjectResult(calculatorResult.Error.Message);
    }

    [ Private Methods ]
}
```

Azure Function … trigger!!

```
public static class MortgageFunctions
{
    private static readonly IMortgageCalculator mortgageCalculator =
            new MortgageCalculator(null);

    [FunctionName(FunctionNames.MortgageCalculatorFunction + "STATIC")]
    0 references | Massimo Bonanni, 168 days ago | 2 authors, 2 changes
    public static async Task<IActionResult> Run(
        [HttpTrigger(AuthorizationLevel.Function, "get", Route = null)] HttpRequest req,
        [Table("executionsTable", Connection = "StorageAccount")] ICollector<ExecutionRow> outputTable,
        ILogger log)
    {
        log.LogInformation($"{FunctionNames.MortgageCalculatorFunction} start");

        // Retrieve loan, interest and numberOfPayments from HTTP Request
        [ Retrieve request parameters ]

        var calculatorResult =
            await mortgageCalculator.CalculateMontlyRateAsync(loan, interest, nPaym

        [ Create the response ]

        if (calculatorResult.Succeed)
        {
            return new OkObjectResult(calculatorResult.Result);
        }
```

Trigger
You can mock it because the trigger payload is a POCO class

Azure Function … bindings!!

```
public static class MortgageFunctions
{
    private static readonly IMortgageCalculator mortgageCalculator =
            new MortgageCalculator(null);

    [FunctionName(FunctionNames.MortgageCalculatorFunction + "STATIC")]
    0 references | Massimo Bonanni, 168 days ago | 2 authors, 2 changes
    public static async Task<IActionResult> Run(
        [HttpTrigger(AuthorizationLevel.Function, "get", Route = null)] HttpRequest req,
        [Table("executionsTable", Connection = "StorageAccount")] ICollector<ExecutionRow> outputTable,
        ILogger log)
    {
        log.LogInformation($"{FunctionNames.MortgageCalculatorFunction} start");

        // Retrieve loan, interest and numberOfPayments from HTTP Request
        [ Retrieve request parameters ]

        var calculatorResult =
            await mortgageCalculator.CalculateMontlyRateAsync(loan, interest, nPaym

        [ Create the response ]

        if (calculatorResult.Succeed)
        {
            return new OkObjectResult(calculatorResult.Result);
        }
```

Binding
You can mock it because the binding payload is an interface

Azure Function … logger!!

```
public static class MortgageFunctions
{
    private static readonly IMortgageCalculator mortgageCalculator =
            new MortgageCalculator(null);

    [FunctionName(FunctionNames.MortgageCalculatorFunction + "STATIC")]
    0 references | Massimo Bonanni, 168 days ago | 2 authors, 2 changes
    public static async Task<IActionResult> Run(
        [HttpTrigger(AuthorizationLevel.Function, "get", Route = null)] HttpRequest req,
        [Table("executionsTable", Connection = "StorageAccount")] ICollector<ExecutionRow> outputTable,
        ILogger log)
    {
        log.LogInformation($"{FunctionNames.MortgageCalculatorFunction} start");

        // Retrieve loan, interest and numberOfPayments from HTTP Request
        [ Retrieve request parameters ]

        var calculatorResult =
            await mortgageCalculator.CalculateMontlyRateAsync(loan, interest, nP

        [ Create the response ]

        if (calculatorResult.Succeed)
        {
            return new OkObjectResult(calculatorResult.Result);
        }
```

Logger
(infrastructural stuffs)
You can mock it because the logger is an interface

Azure Function … your service!!

```
public static class MortgageFunctions
{
    private static readonly IMortgageCalculator mortgageCalculator =
            new MortgageCalculator(null);

    [FunctionName(FunctionNames.MortgageCalculatorFunction + "STATIC")]
    0 references | Massimo Bonanni, 168 days ago | 2 authors, 2 changes
    public static async Task<IActionResult> Run(
        [HttpTrigger(AuthorizationLevel.Function, "get", Route = null)] HttpRequest
        [Table("executionsTable", Connection = "StorageAccount")] ICollector<Execut
        ILogger log)
    {
        log.LogInformation($"{FunctionNames.MortgageCalculatorFunction} start");

        // Retrieve loan, interest and numberOfPayments from HTTP Request
        [ Retrieve request parameters ]

        var calculatorResult =
            await mortgageCalculator.CalculateMontlyRateAsync(loan, interest, nPayments);

        [ Create the response ]

        if (calculatorResult.Succeed)
        {
            return new OkObjectResult(calculatorResult.Result);
        }
```

External service
You **cannot** substitute it with your mock because it is created inside the Azure Function and you **haven't a way** to substitute it

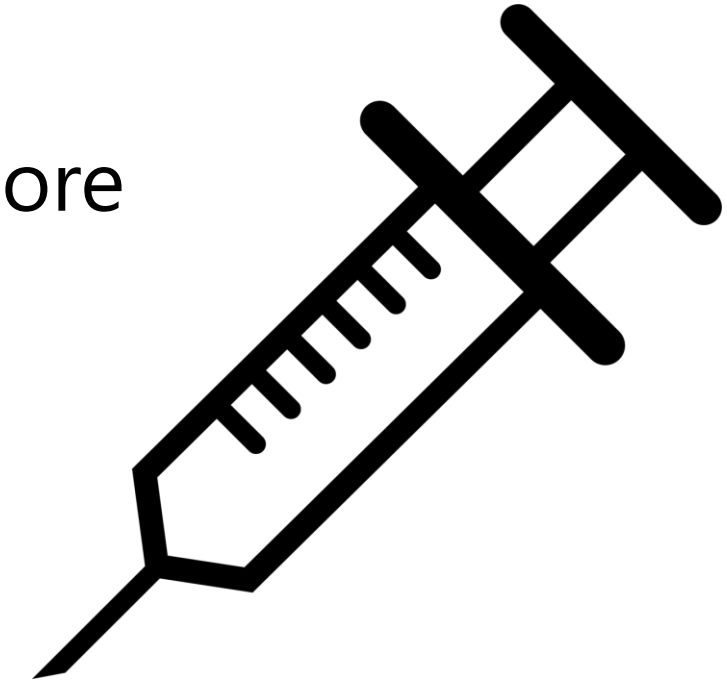
Make your Azure Function testable!!!

The solution of your problem is: **Dependency Injection !!**

Azure Functions Runtime is based on .NET Core.

Azure Functions support the same ASP.NET Core Dependency Injection!!!

Using Dependency Injection you provide a way to substitute your Services with a mock!



Azure Function … testable!!

```
public class MortgageFunctions
{
    private readonly IMortgageCalculator mortgageCalculator;

    0 references | Massimo Bonanni, 197 days ago | 1 author, 1 change
    public MortgageFunctions(IMortgageCalculator mortgageCalculator)
    {
        if (mortgageCalculator == null)
            throw new ArgumentNullException(nameof(mortgageCalculator));

        this.mortgageCalculator = mortgageCalculator;
    }

    [FunctionName(FunctionNames.MortgageCalculatorFunction)]
    0 references | Massimo Bonanni, 168 days ago | 2 authors, 4 changes
    public async Task<IActionResult> Run(
        [HttpTrigger(AuthorizationLevel.Function, "get", Route = null)] Ht
        [Table("executionsTable", Connection = "StorageAccount")] ICollect
        ILogger log)
    {
        log.LogInformation($"{FunctionNames.MortgageCalculatorFunction} st

        // Retrieve loan, interest and numberOfPayments from HTTP Request
        [ Retrieve request parameters ]

        var calculatorResult =
            await this.mortgageCalculator.CalculateMontlyRateAsync(loan, i
```

Constructor Injection

You can choose what kind of actual service you want to use when you instantiate the function.

In a test you can substitute it with a mock!!

Azure Function … how to use mock!!

```
public class MortgageFunctions
{
    private readonly IMortgageCalculator mortgageCalculator;

    0 references | Massimo Bonanni, 197 days ago | 1 author, 1 change
    public MortgageFunctions(IMortgageCalculator mortgageCalculator)
    {
        if (mortgageCalculator == null)
            throw new ArgumentNullException(nameof(mortgageCalculator));

        this.mortgageCalculator = mortgageCalculator;
    }

    [FunctionName(FunctionNames.MortgageCalculatorFunction)]
    0 references | Massimo Bonanni, 168 days ago | 2 authors, 4 changes
    public async Task<IActionResult> Run(
        [HttpTrigger(AuthorizationLevel.Function, "get", Route = null)] HttpRequest req,
        [Table("executionsTable", Connection = "StorageAccount")] ICollector<ExecutionRow> outputTable,
        ILogger log)
    {
        log.LogInformation($"{FunctionNames.MortgageCalculatorFunction} start");

        // Retrieve loan, interest and numberOfPayments from HTTP Request
        [ Retrieve request parameters ]

        var calculatorResult =
            await this.mortgageCal

        [ Create the response ]

        if (calculatorResult.Succe
        {
            return new OkObjectRes
        }

        return new BadRequestObjec
    }

    [ Private Methods ]
}
```

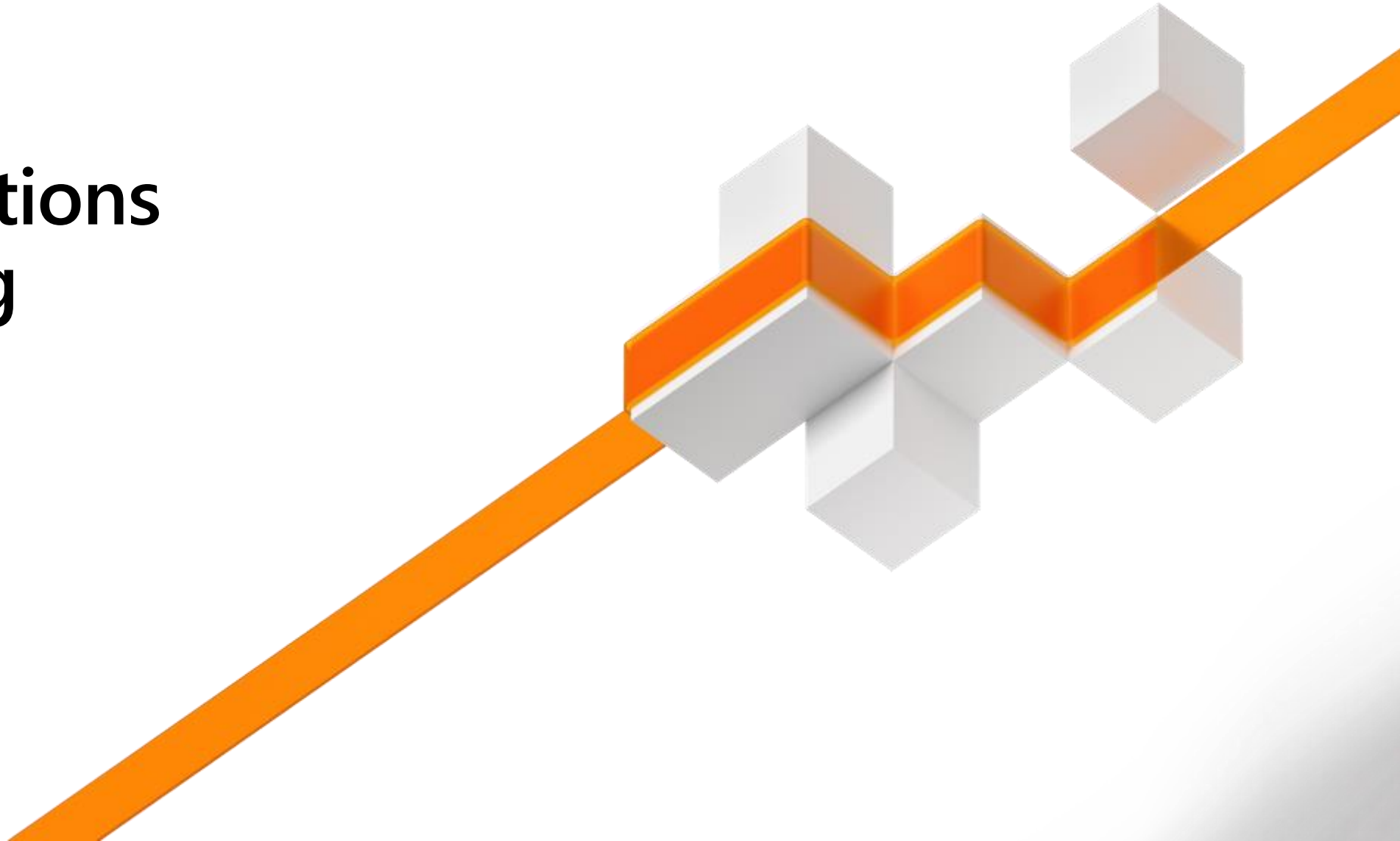
Mock

Create a mock to use in the test!!

```
var mortgageCalculator = new Mock<IMortgageCalculator>();
mortgageCalculator
    .Setup(c => c.CalculateMontlyRateAsync(mortgageLoan, annualInterest, numberOfPayments))
    .ReturnsAsync(new CalculatorResult() { Result = rate });

var target = new MortgageFunctions(mortgageCalculator.Object);
```

DEMO
Azure Functions
Unit Testing



Monitoring Azure Functions

Once you deploy your Azure Functions on Azure, you need to monitor them to check when something goes wrong.

The signature of an Azure Function Run method provides the instance of **ILogger** that you can use to log information about your code.

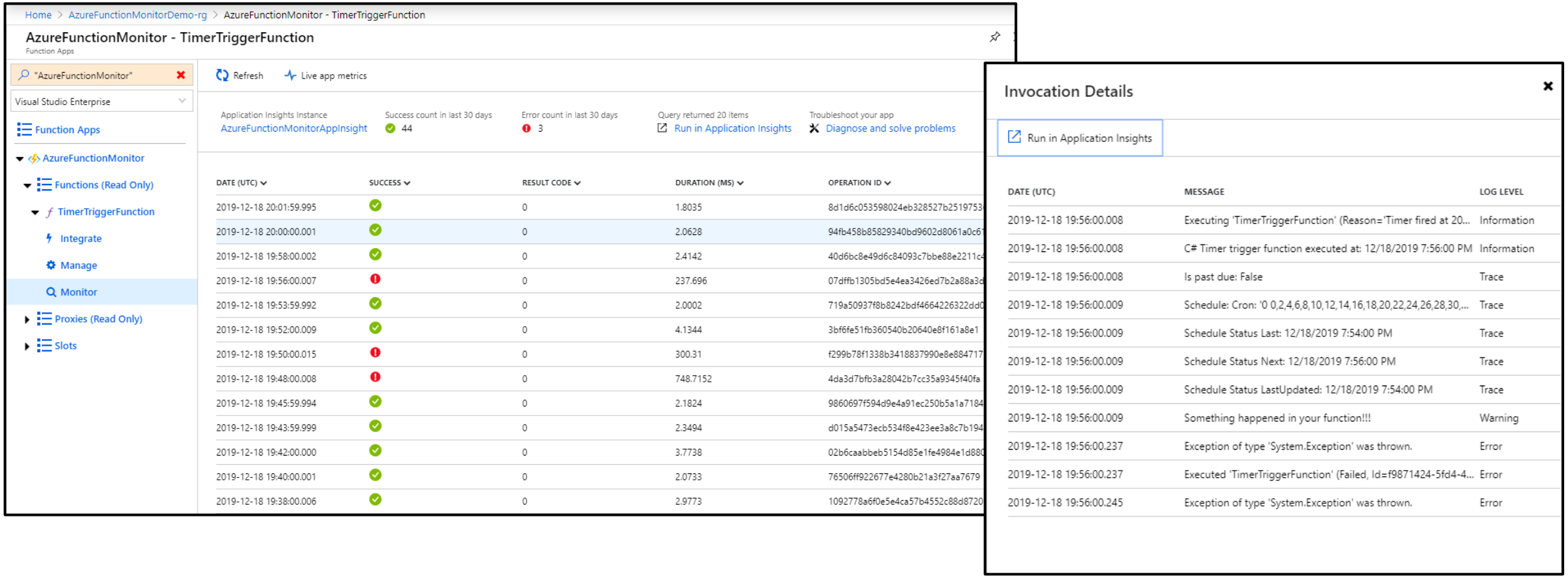
Using **ILogger**, you can collect information from your code execution to monitor and triage errors and exceptions.

```
public static class MonitoringFunctions
{
    [FunctionName("TimerTriggerFunction")]
    0 references | Massimo Bonanni, 196 days ago | 1 author, 1 change
    public static void Run([TimerTrigger("0 */2 * * * *")]TimerInfo myTimer, ILogger log)
    {
        var executionTimestamp = DateTime.Now;
        log.LogInformation($"C# Timer trigger function executed at: {executionTimestamp}");
    }
}
```

Azure Functions Monitor

Azure Functions provide out-of-the-box monitor feature.

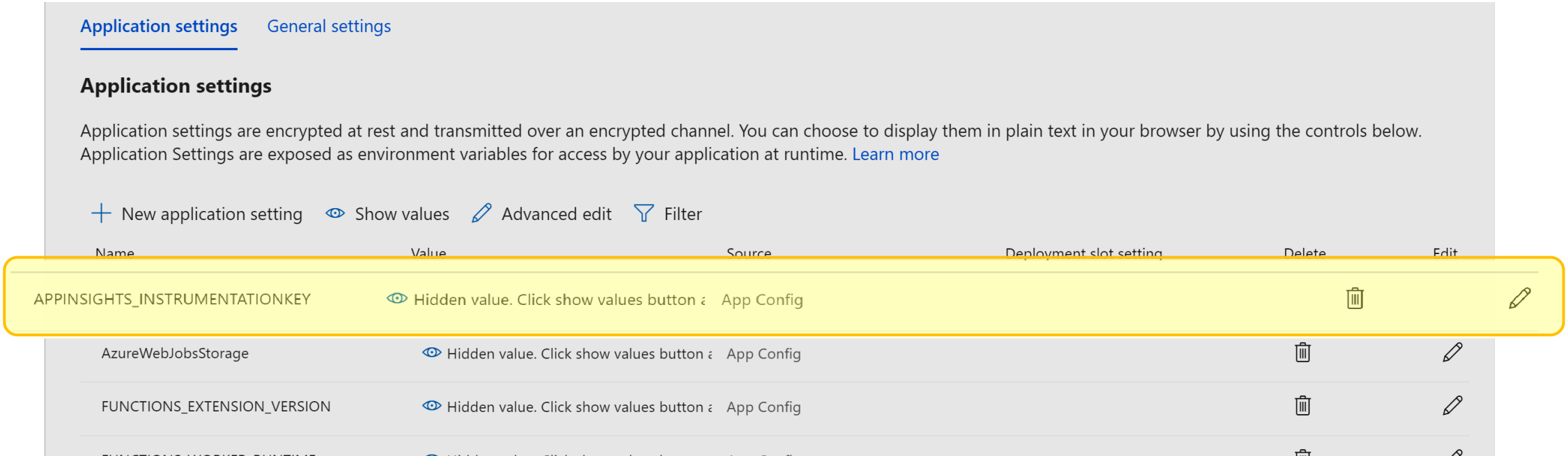
For each Function, you can have info about every function execution.



Azure Functions and Application Insight

The Azure Functions platform offers built-in integration with Azure Application Insights.

Put the **Application Insights instrumentation key** in the function app settings.



Custom Metric

Azure Function SDK provides you extension methods to log custom metrics.

```
log.LogMetric("MyCustomMetric", CalculateMyCustomMetric());
```

```
customMetrics
| where name == "MyCustomMetric"
| order by timestamp desc
```

Completed. Showing partial results from the last 30 minutes. 00:00

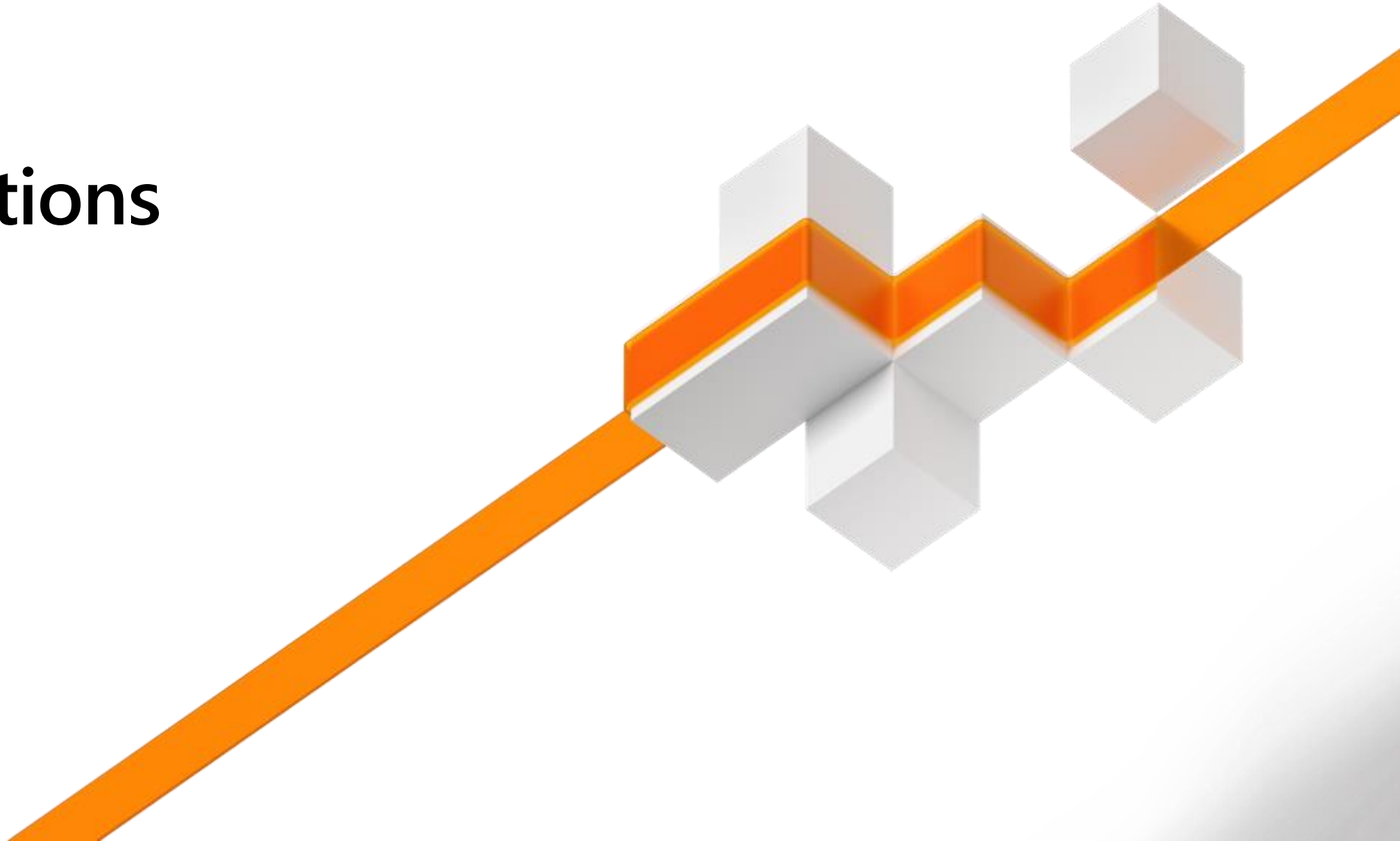
Table | Chart | Columns | Display time (UTC+01:00)

Drag a column header and drop it here to group by that column

timestamp [Amsterdam, Berlin, Bern, R...	name	value	valueCount	valueSum	valueMin
22/12/2019, 19:07:59.992	MyCustomMetric	0,425	1	0,425	0,425
22/12/2019, 19:03:59.999	MyCustomMetric	0,162	1	0,162	0,162
22/12/2019, 19:02:00.000	MyCustomMetric	0,602	1	0,602	0,602

timestamp [UTC]	2019-12-22T18:02:00.0002467Z
name	MyCustomMetric
value	0.60233813272898
valueCount	1
valueSum	0.60233813272898
valueMin	0.60233813272898

DEMO
Azure Functions Monitoring



Take away

- Write an Azure Functions is **simple**!
- Testing Azure Functions is **simple**!
- Monitoring Azure Functions is **simple**!
- .... then ....

KEEP CALM AND USE

AZURE FUNCTIONS

BRK30167

Thanks for your attention!!!!!

Q&A

Massimo Bonanni

Azure Technical Trainer @ Microsoft

massimo.bonanni@microsoft.com
@massimobonanni

Connect with me on LinkedIn

linkedin.com/in/massimobonanni/

References

- Azure Functions Documentation

  https://docs.microsoft.com/en-US/azure/azure-functions/

- Azure Functions Code Samples

  https://azure.microsoft.com/en-us/resources/samples/?service=functions&sort=0

- Azure Updates

  https://azure.microsoft.com/en-us/roadmap/?category=compute

- Demo MortgageCalculator – GitHub

  http://bit.ly/TestAZFunc

- Demo Monitor Azure Functions – GitHub

  http://bit.ly/MonitorAZFunc