TECHORAMA

DEEP KNOWLEDGE IT CONFERENCE

October 1-3 | 2018 Ede, The Netherlands

HOW FAR CAN ‘SERVERLESS’ ACTUALLY GO?

TECHORAMA

DEEP KNOWLEDGE IT CONFERENCE

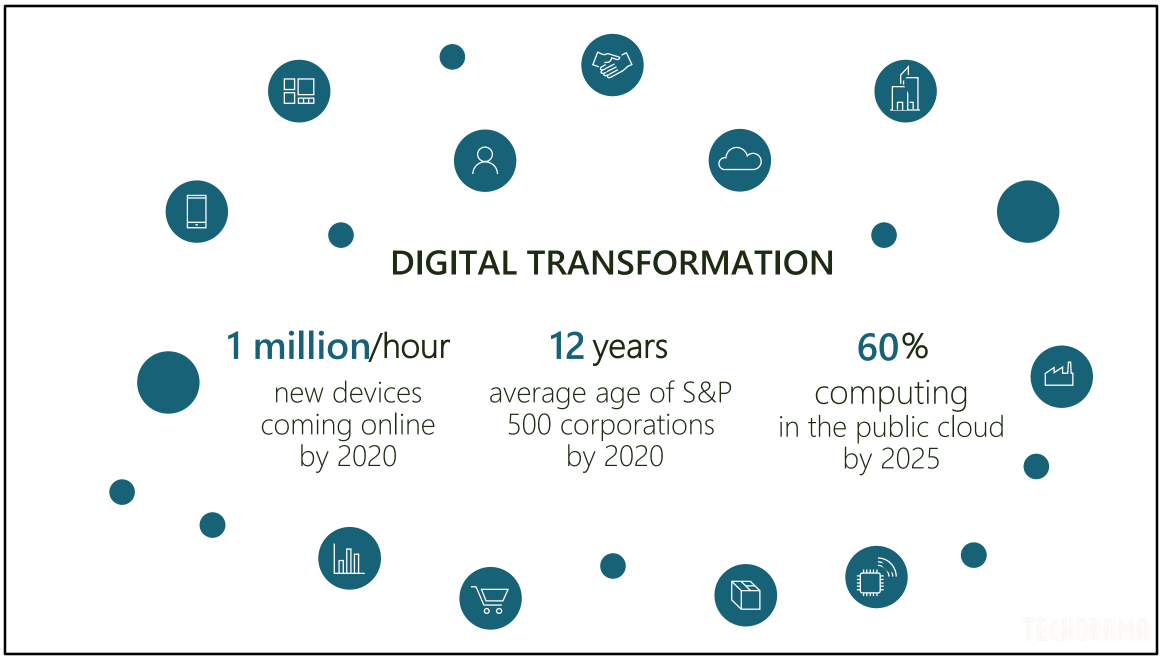
October 1-3 | 2018 Ede, The Netherlands

Alex Mang

Azure MVP & Advisor
MCSD, MCSE, PSM

@iAmAlexMang
www.alexmang.com

You!

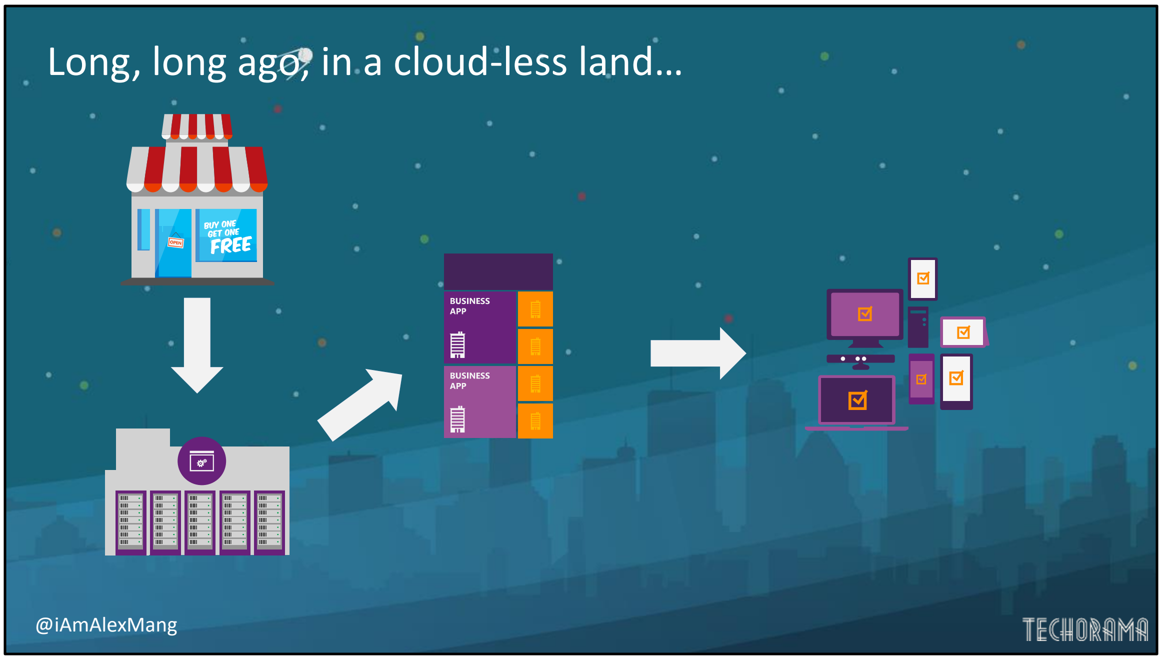


DIGITAL TRANSFORMATION

1 million/hour
new devices
coming online
by 2020

12 years
average age of S&P
500 corporations
by 2020

60%
computing
in the public cloud
by 2025



Long, long ago, in a cloud-less land…

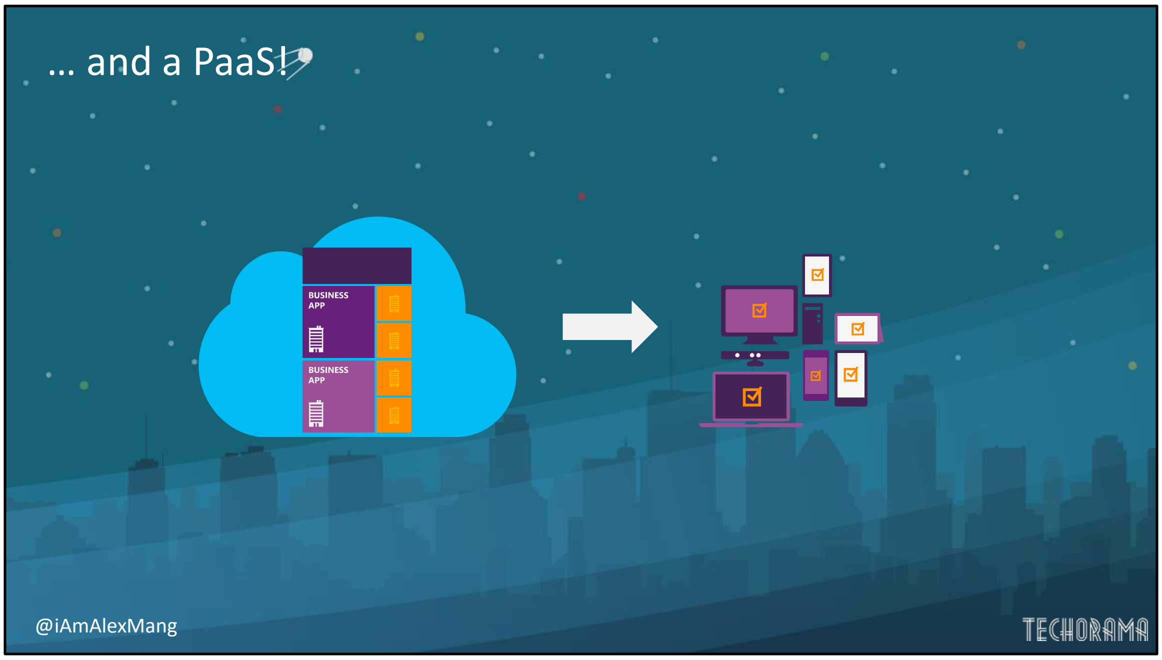
@iAmAlexMang

TECHORAMA

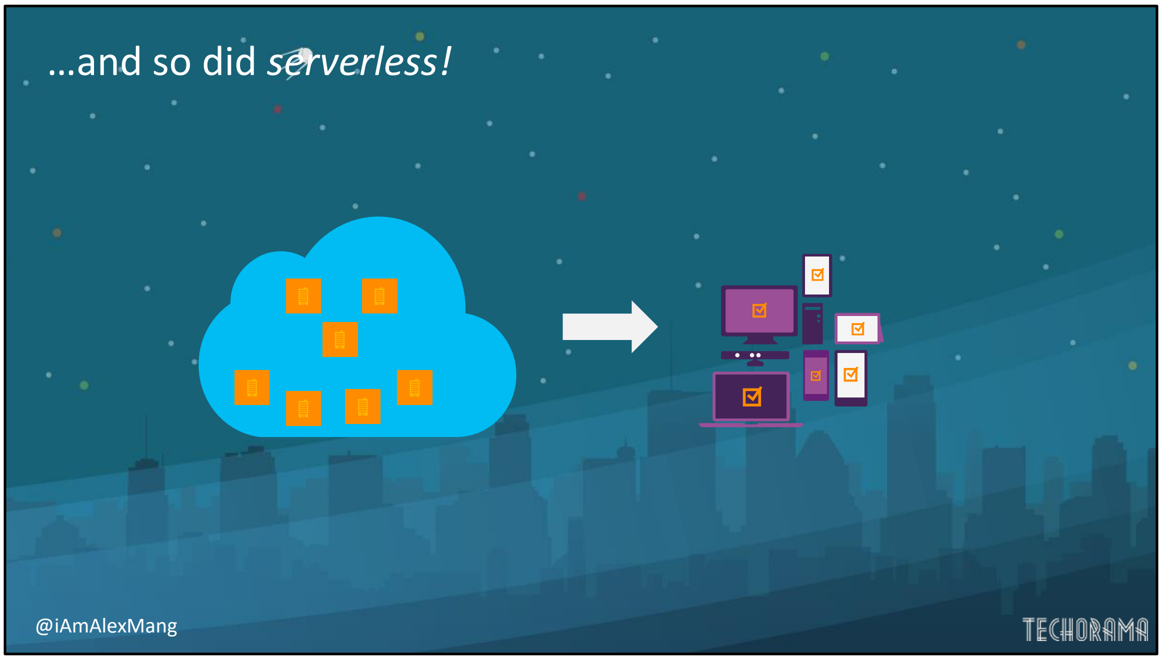
Long, long ago, in a cloud-less land… ☺ **this is how the IT bedtime stories will start in a few years from now**. Especially when we'll refer to those days when we were selling complete custom-tailored solutions (not apps or HW, but solutions – the full package), when we knew exactly what the software was supposed to run on. Such as IBM x3650M2 with 8GB of RAM and whatever crazy Intel Xeon CPU. But really – did the software developer ever feel happy about taking those numbers into consideration? Or even the provisioning process? Of course not.

Then a wild IaaS appeared…

And that's when **IaaS popped up** – basically, the underlying HW infrastructure was stripped away, and we were merely responsible of the OS, software solution stack and whatever sits above that, up to the application we were developing, of course.


… and a PaaS!

Then PaaS, or Platform as a Service started to become more and more popular. **But truth be told, it's the PaaS which really locks you in**. I've always said it – there's no lock-in, like in the cloud lock in. Because once you've chosen your provider, you're forced to take their requirements into account, you're forced to use their SDK, their tools or the 3rd party tools especially designed for your provider and so on. So as I was saying – no other lock-in like the cloud lock-in. And it's the same whether we'll talk about Azure, AWS, GCE or even OpenStack. Yes, OpenStack!

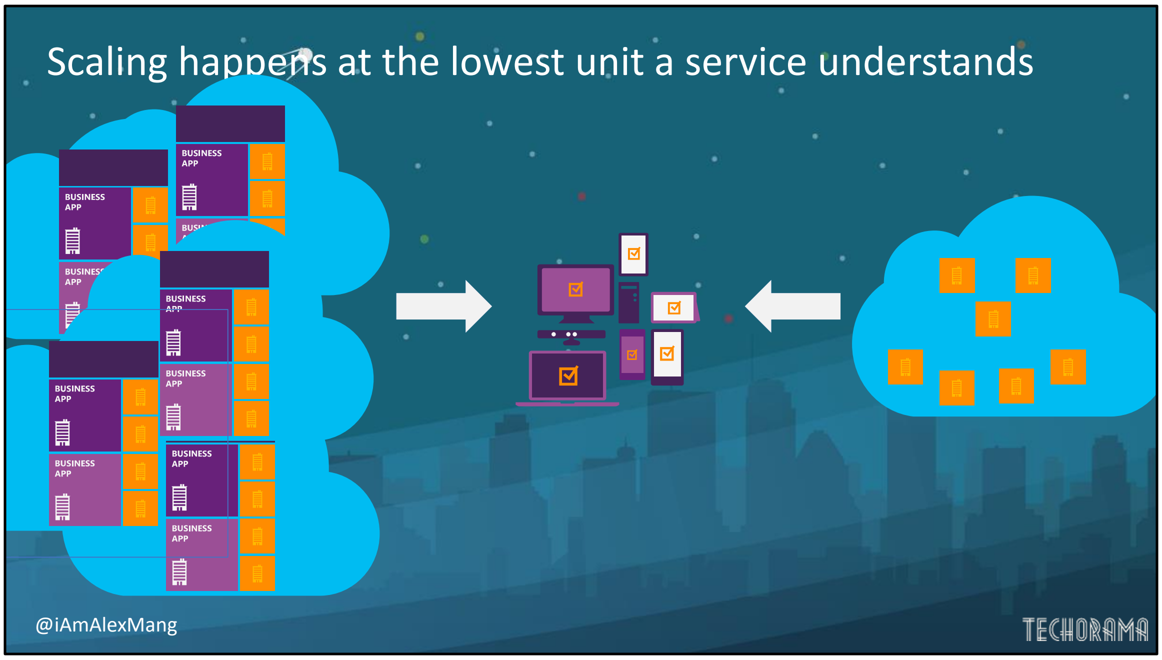


…and so did *serverless!*

@iAmAlexMang

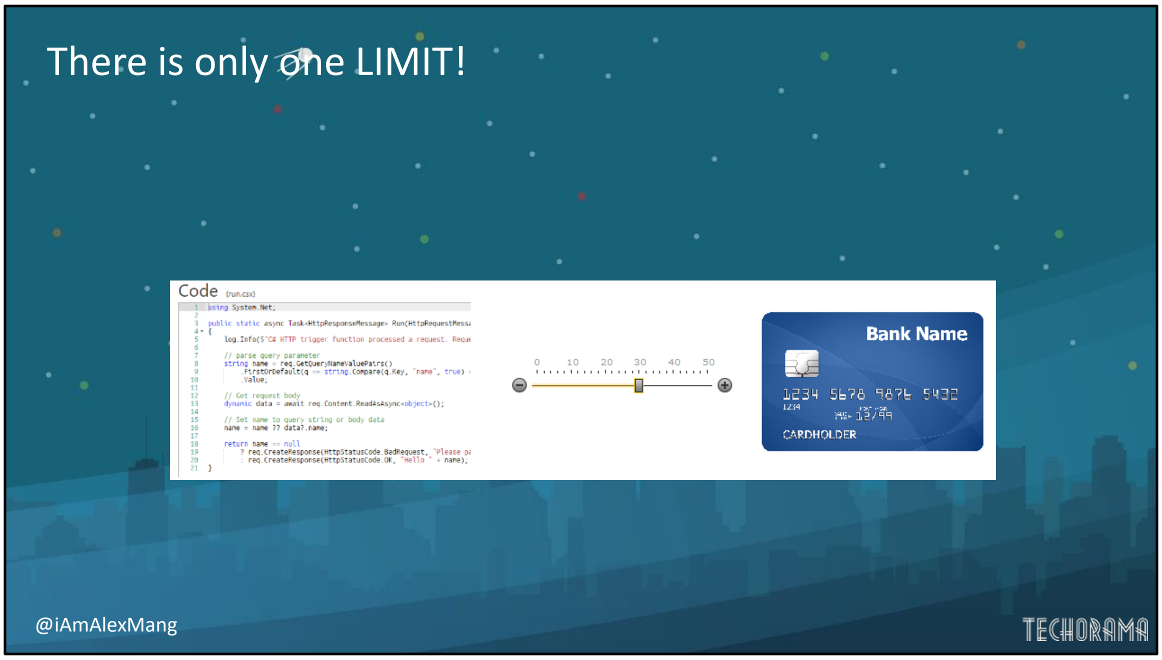
TECHORAMA

Last, but most likely not least, serverless. What is it? It's that awesome place, almost like an utopia, where all we care about is code. You push the code – emphasize on code, not binaries or executable or anything alike – just the text which represents your code, and it runs. But not only does it run, as it also scales.
How much, you might wonder? As much as it wants to. Really ☺.



Scaling happens at the lowest unit a service understands

Look, here ☺.
Really much - limitless.



There is only one LIMIT!

Code (run.csx)

```
using System.Net;

public static async Task<HttpResponseMessage> Run(HttpRequestMessa
{
    log.Info($"C# HTTP trigger function processed a request. Reque

    // parse query parameter
    string name = req.GetQueryNameValuePairs()
        .FirstOrDefault(q => string.Compare(q.Key, "name", true) =
        .Value;

    // Get request body
    dynamic data = await req.Content.ReadAsAsync<object>();

    // Set name to query string or body data
    name = name ?? data?.name;

    return name == null
        ? req.CreateResponse(HttpStatusCode.BadRequest, "Please pa
        : req.CreateResponse(HttpStatusCode.OK, "Hello " + name);
}
```

@iAmAlexMang

TECHORAMA

What is '*serverless*'?

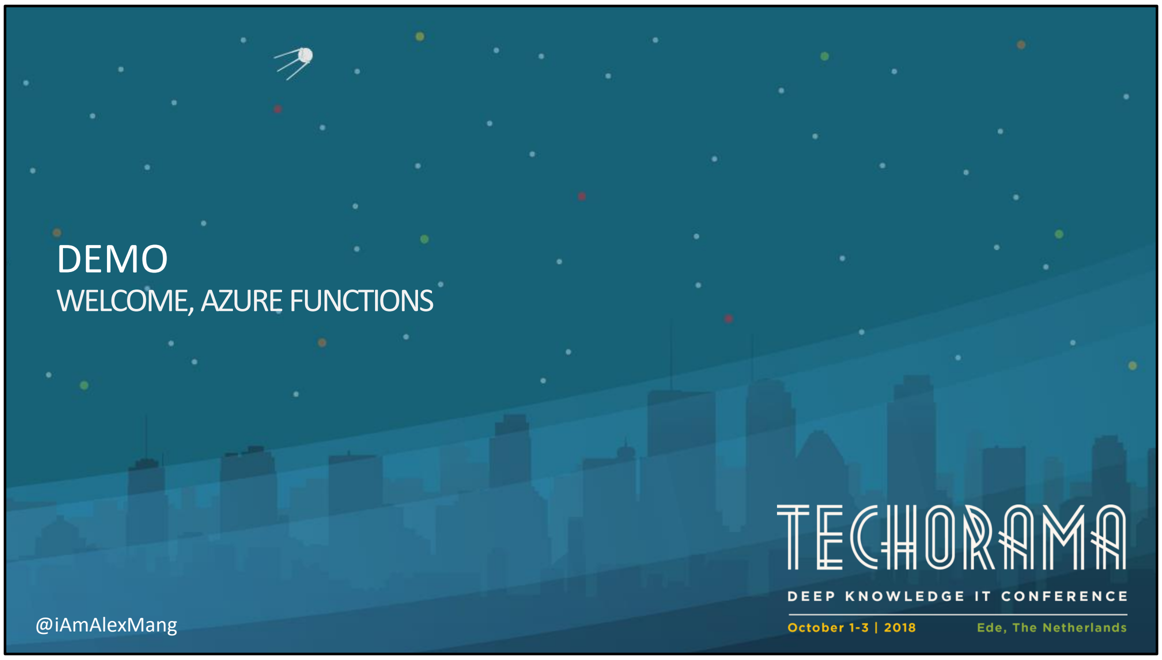
Abstraction of servers

Event-driven/ instant scale

Micro-billing

@iAmAlexMang

TECHORAMA



DEMO

WELCOME, AZURE FUNCTIONS

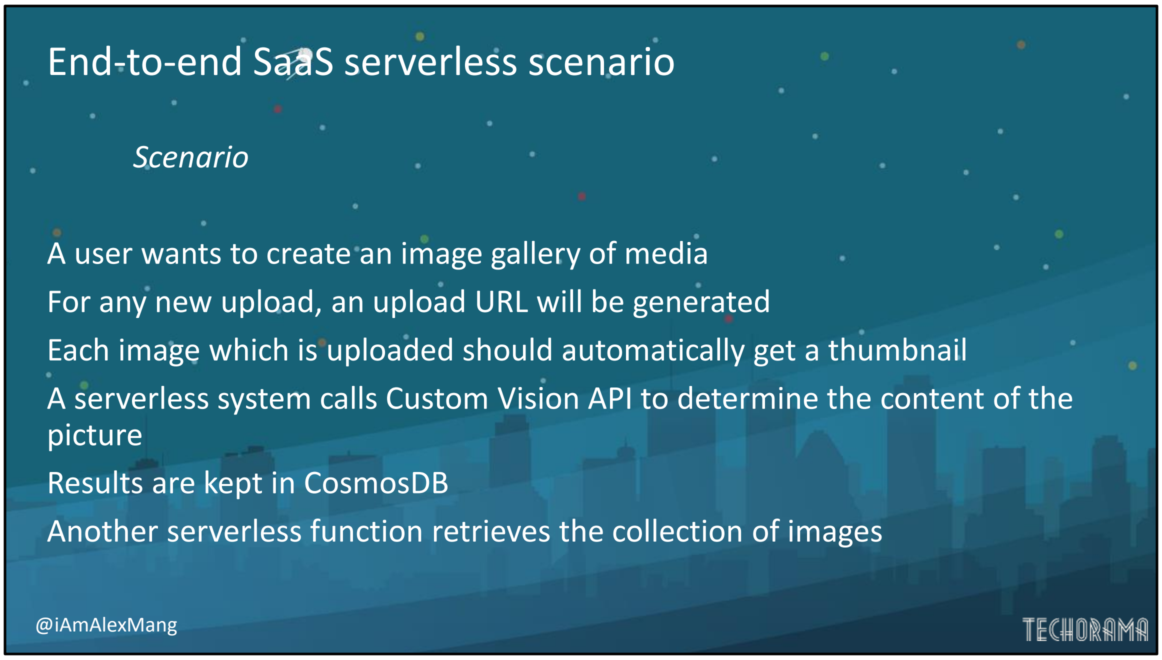
@iAmAlexMang

TECHORAMA
DEEP KNOWLEDGE IT CONFERENCE
October 1-3 | 2018
Ede, The Netherlands

WHAT ABOUT SOME REAL PRODUCTION APPLICATIONS?

@iAmAlexMang

TECHORAMA
DEEP KNOWLEDGE IT CONFERENCE
October 1-3 | 2018 Ede, The Netherlands



End-to-end SaaS serverless scenario

Scenario

A user wants to create an image gallery of media

For any new upload, an upload URL will be generated

Each image which is uploaded should automatically get a thumbnail

A serverless system calls Custom Vision API to determine the content of the picture

Results are kept in CosmosDB

Another serverless function retrieves the collection of images

@iAmAlexMang

TECHORAMA

DEMO

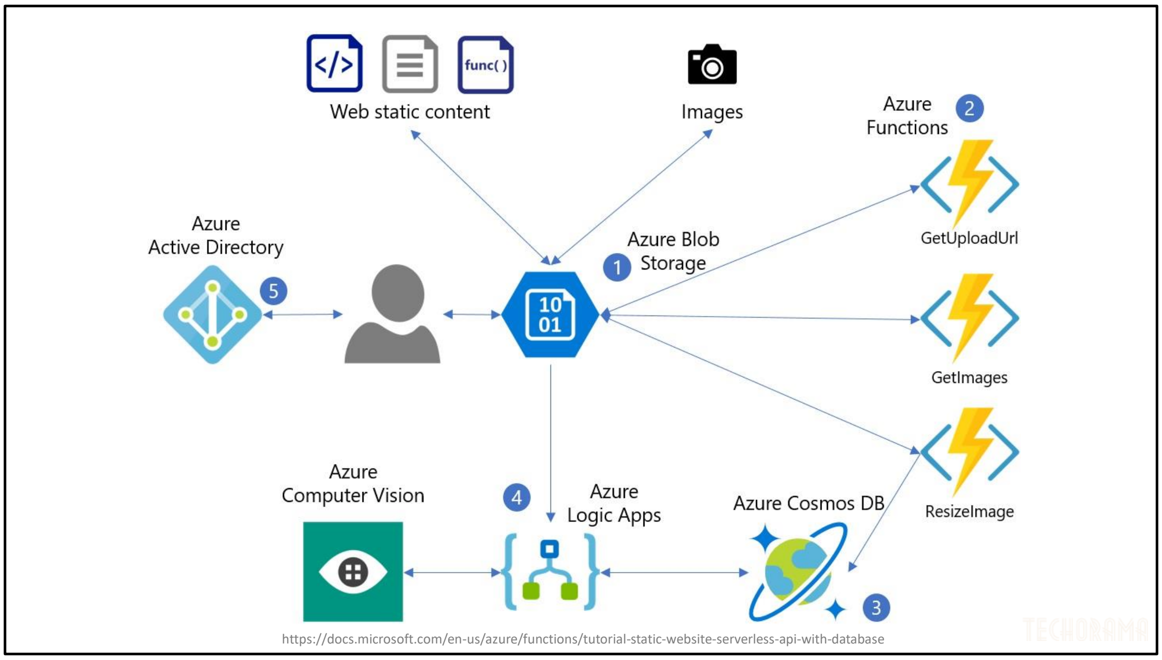
E2E FUNCTIONS APPLICATION

@iAmAlexMang

TECHORAMA

DEEP KNOWLEDGE IT CONFERENCE

October 1-3 | 2018 Ede, The Netherlands


Web static content
func( )
Images
Azure Functions
2
GetUploadUrl
GetImages
ResizeImage
Azure Active Directory
5
Azure Blob Storage
1
10 01
Azure Computer Vision
4
Azure Logic Apps
Azure Cosmos DB
3
https://docs.microsoft.com/en-us/azure/functions/tutorial-static-website-serverless-api-with-database
TECHORAMA

HOW SCALABLE, REALLY?

@iAmAlexMang

TECHORAMA
DEEP KNOWLEDGE IT CONFERENCE
October 1-3 | 2018 Ede, The Netherlands

DEMO

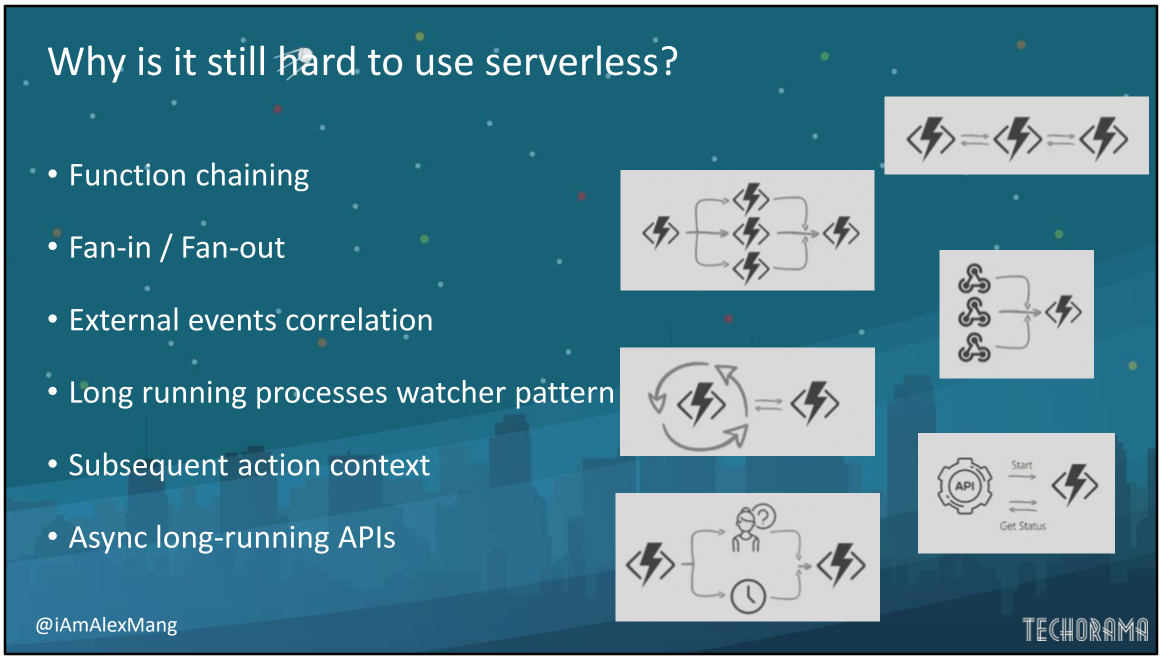
SCALABILITY PERFORMANCE

@iAmAlexMang

TECHORAMA

DEEP KNOWLEDGE IT CONFERENCE

October 1-3 | 2018 Ede, The Netherlands



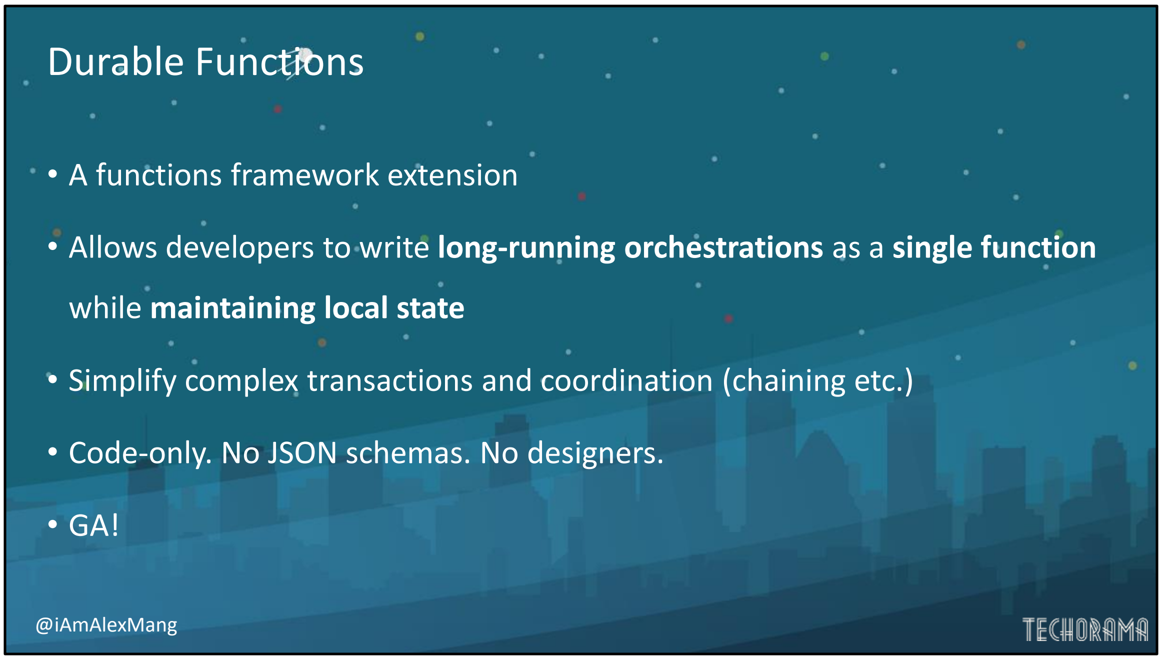
Why is it still hard to use serverless?

- Function chaining
- Fan-in / Fan-out
- External events correlation
- Long running processes watcher pattern
- Subsequent action context
- Async long-running APIs

Start
API
Get Status

@iAmAlexMang

TECHORAMA

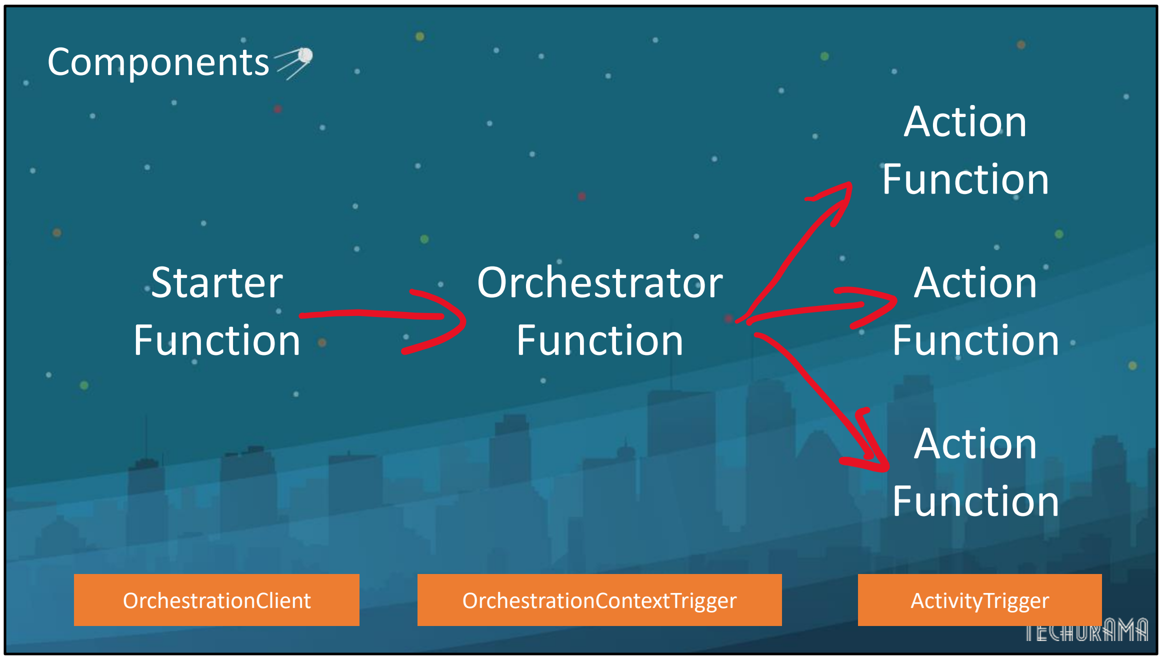


Durable Functions

- A functions framework extension
- Allows developers to write **long-running orchestrations** as a **single function** while **maintaining local state**
- Simplify complex transactions and coordination (chaining etc.)
- Code-only. No JSON schemas. No designers.
- GA!

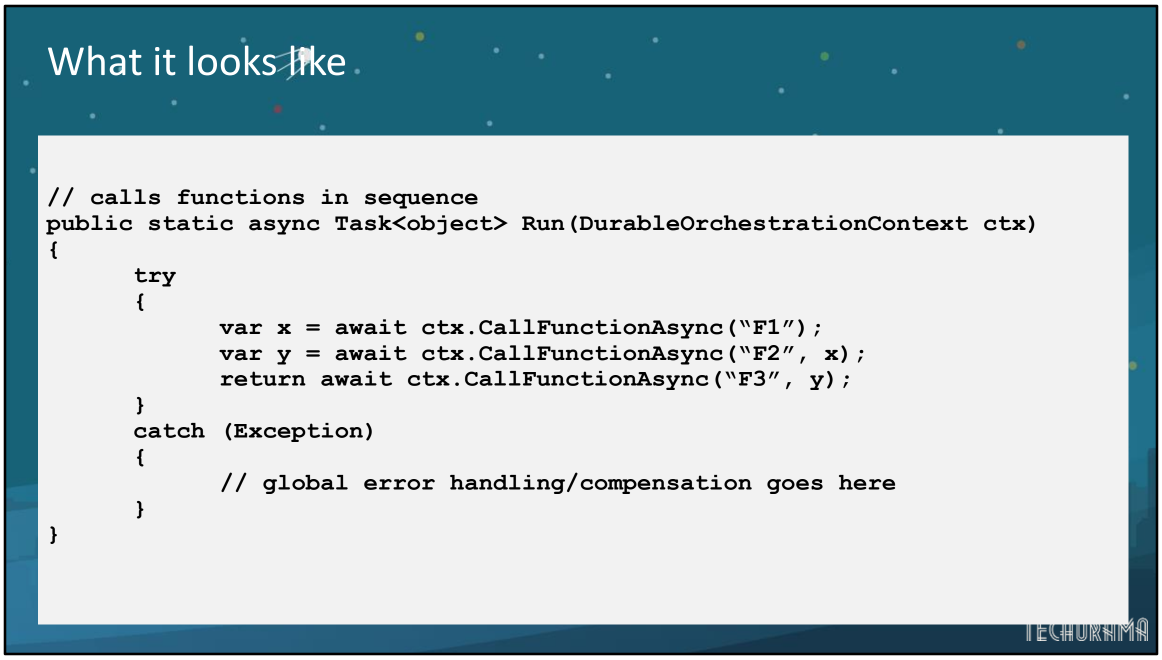
@iAmAlexMang

TECHORAMA



Components

Starter Function
Orchestrator Function
Action Function
Action Function
Action Function
OrchestrationClient
OrchestrationContextTrigger
ActivityTrigger
TECHORAMA



What it looks like

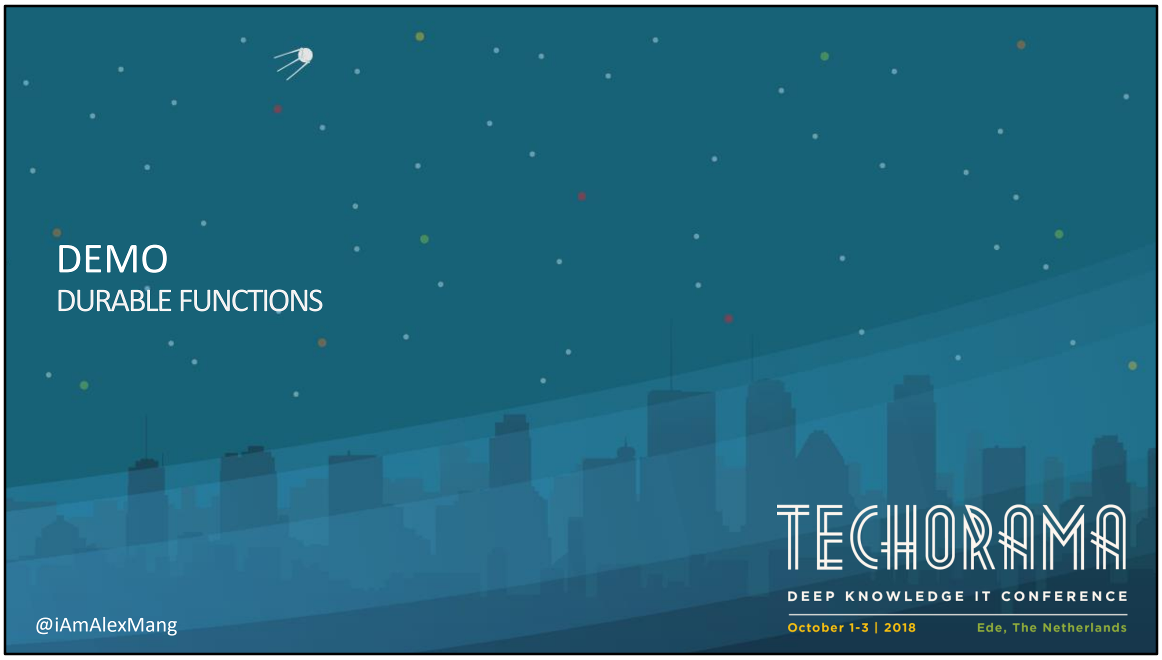
```
// calls functions in sequence
public static async Task<object> Run(DurableOrchestrationContext ctx)
{
      try
      {
            var x = await ctx.CallFunctionAsync(“F1”);
            var y = await ctx.CallFunctionAsync(“F2”, x);
            return await ctx.CallFunctionAsync(“F3”, y);
      }
      catch (Exception)
      {
            // global error handling/compensation goes here
      }
}
```

What it looks like

```
var outputs = new List<string>();
outputs.Add(
      await context.CallActivityAsync<string>(
            "SayHello",
            "awesome crowd")
      );
return outputs;
```

History Table
Orchestrator started
Execution started
Task scheduled, SayHello, “awesome crowd”
Orchestrator completed
Task Completed, “Hello, awesome crowd!”
Orchestrator started
Execution completed, [(“Hello, awesome crowd”)]
Orchestrator completed

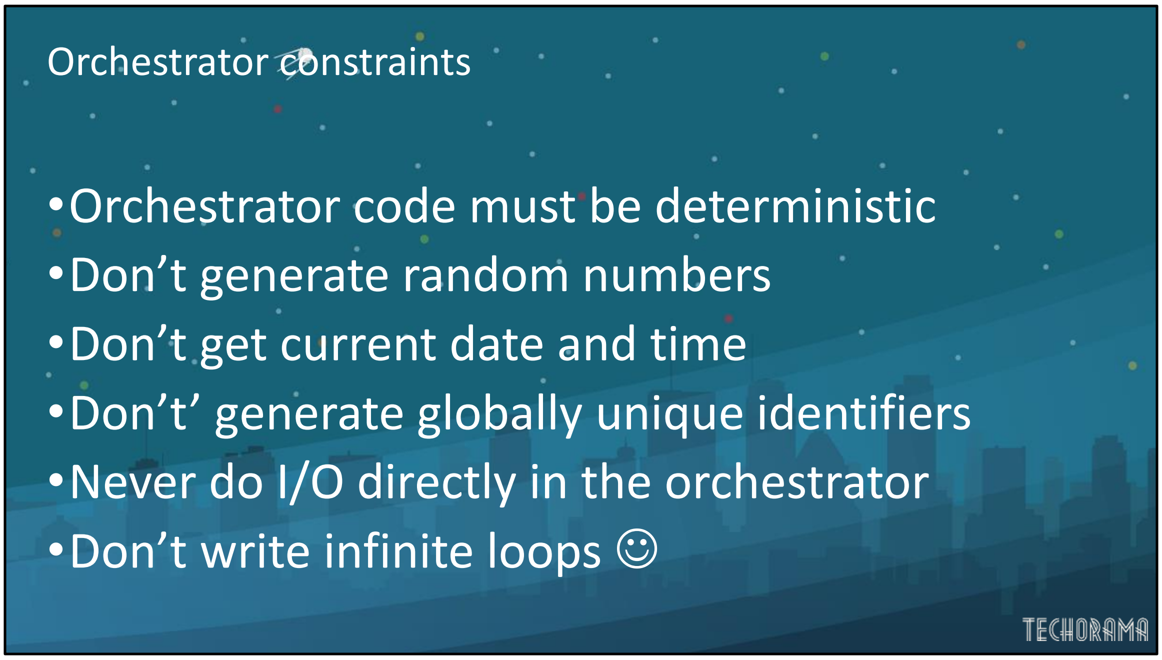
TECHORAMA



DEMO
DURABLE FUNCTIONS

@iAmAlexMang

TECHORAMA
DEEP KNOWLEDGE IT CONFERENCE
October 1-3 | 2018 Ede, The Netherlands



Orchestrator constraints

- Orchestrator code must be deterministic
- Don’t generate random numbers
- Don’t get current date and time
- Don’t’ generate globally unique identifiers
- Never do I/O directly in the orchestrator
- Don’t write infinite loops ☺

TECHORAMA

Q&A

@iAmAlexMang

TECHORAMA

DEEP KNOWLEDGE IT CONFERENCE

October 1-3 | 2018 Ede, The Netherlands