

Showdown of the serverless cloud, orchestrating functions



choose your alliance, go to:
<https://presentain.com/showdown>





Kenny Baas-Schwegler

Software Consultant - Domain-Driven Design

@kenny_baas

Baasie.com

xebia.com/blog/author/kbaas/



Marc Duiker

Azure Consultant - I ❤️ Serverless

@marcduiker

blog.marcduiker.nl

Why do I like serverless?





Why do I like serverless?



A close-up photograph of a dog's face, specifically its eyes and nose. The dog is wearing a pair of bright green goggles. The reflection in the goggles shows a blue and white airplane on a runway, with a hangar and trees in the background under a cloudy sky. The dog's fur is black and white.

Let's look at the specs!

Comparing Cost*

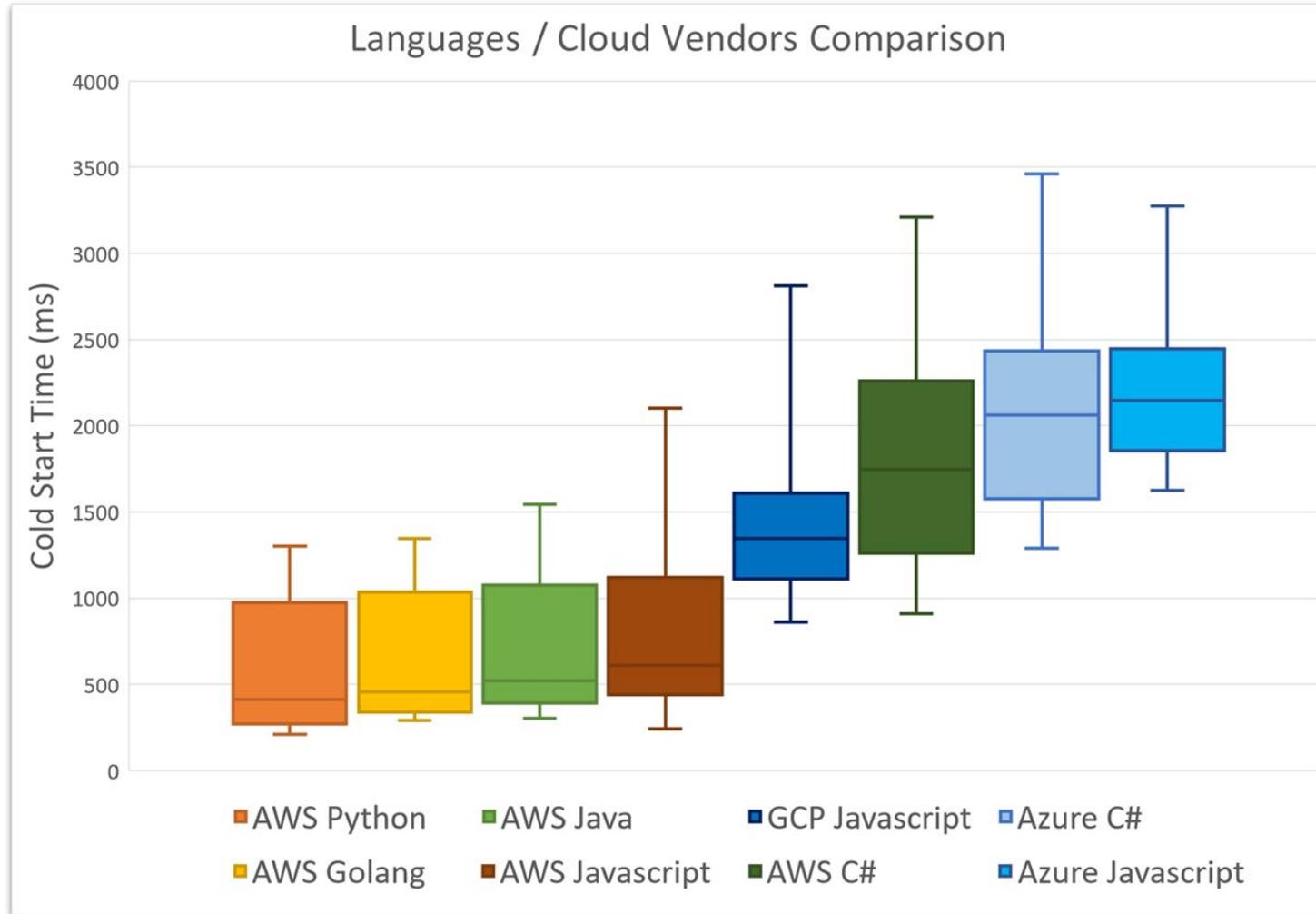
| Type | AWS Lambda | Azure Functions | Google Cloud Functions |
|------------------------------|--------------|-----------------|------------------------|
| Price 1M Executions (\$) | 0.20 | 0.20 | 0.40 |
| Price GB-s (\$) | 0.00001667 | 0.000016 | 0.0000025 |
| Price GHz-s (\$) | - | - | 0.0000100 |
| Free executions/month | 1M | 1M | 2M |
| Free GB-s/month | 400K | 400K | 400K |
| Free GHz-s/month | - | - | 200K |
| Total cost/month (\$) | 78.48 | 75.40 | 92.70 |

* For 10M function executions of 1 sec each using a 512MB / 800MHz machine.

Comparing Runtimes

| Runtimes | AWS Lambda | Azure Functions | Google Cloud Functions |
|----------------|--------------------|-----------------|------------------------|
| Node.js | v6 & v8 | v6, v8, v10 | v6 & v8 (beta) |
| .NET Framework | - | v4.7 | - |
| .NET Core | v1, v2.x | v2.x | - |
| Python | v.2.7, v3.6 & v3.7 | v3.6 (preview) | v3.7.1 (beta) |
| Java | v8 | v8 (preview) | - |
| Ruby | v2.5 | - | - |
| Go | v1.x | - | v1.11(beta) |

Comparing Cold Start

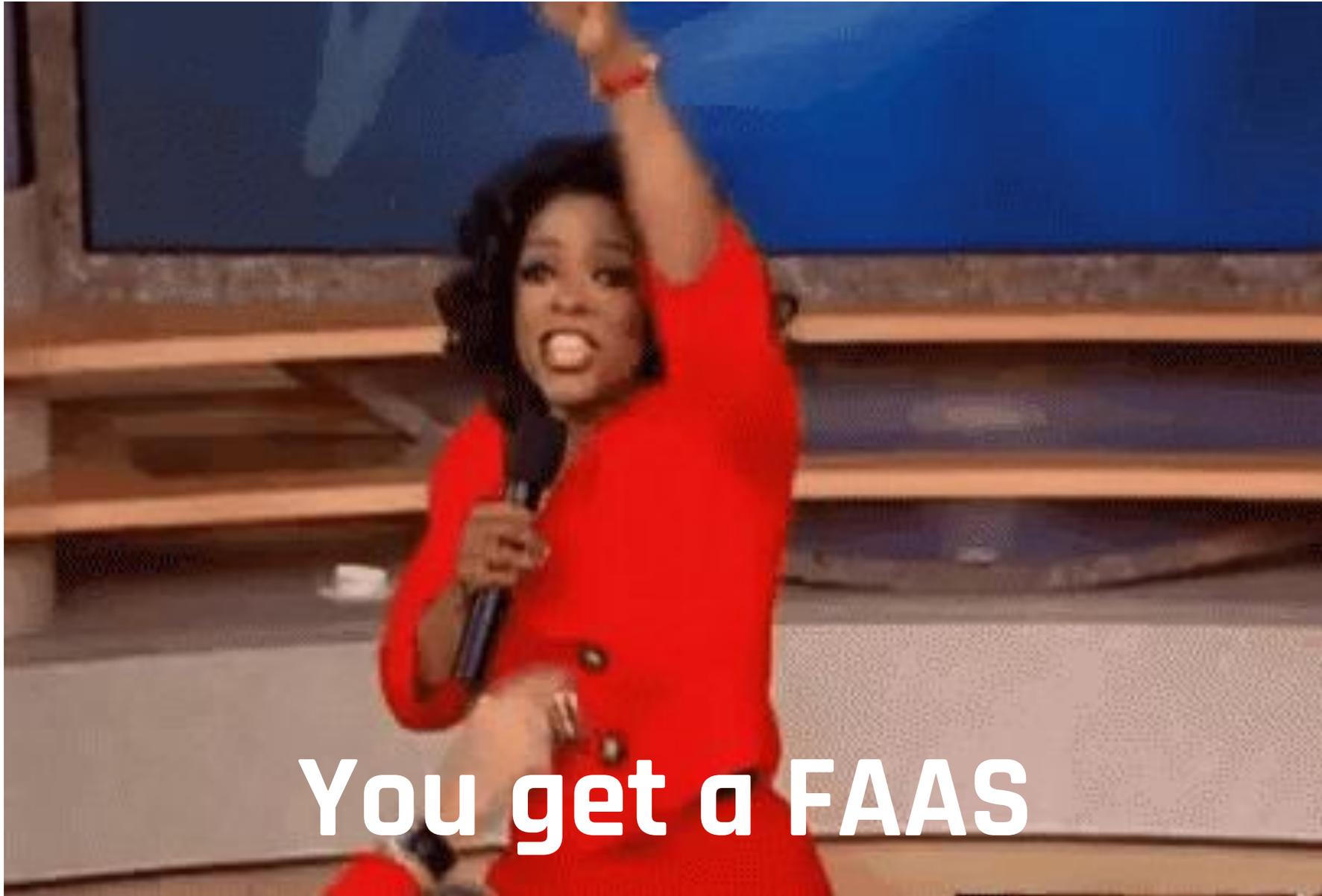


Source: <https://mikhail.io/2018/08/serverless-cold-start-war/>





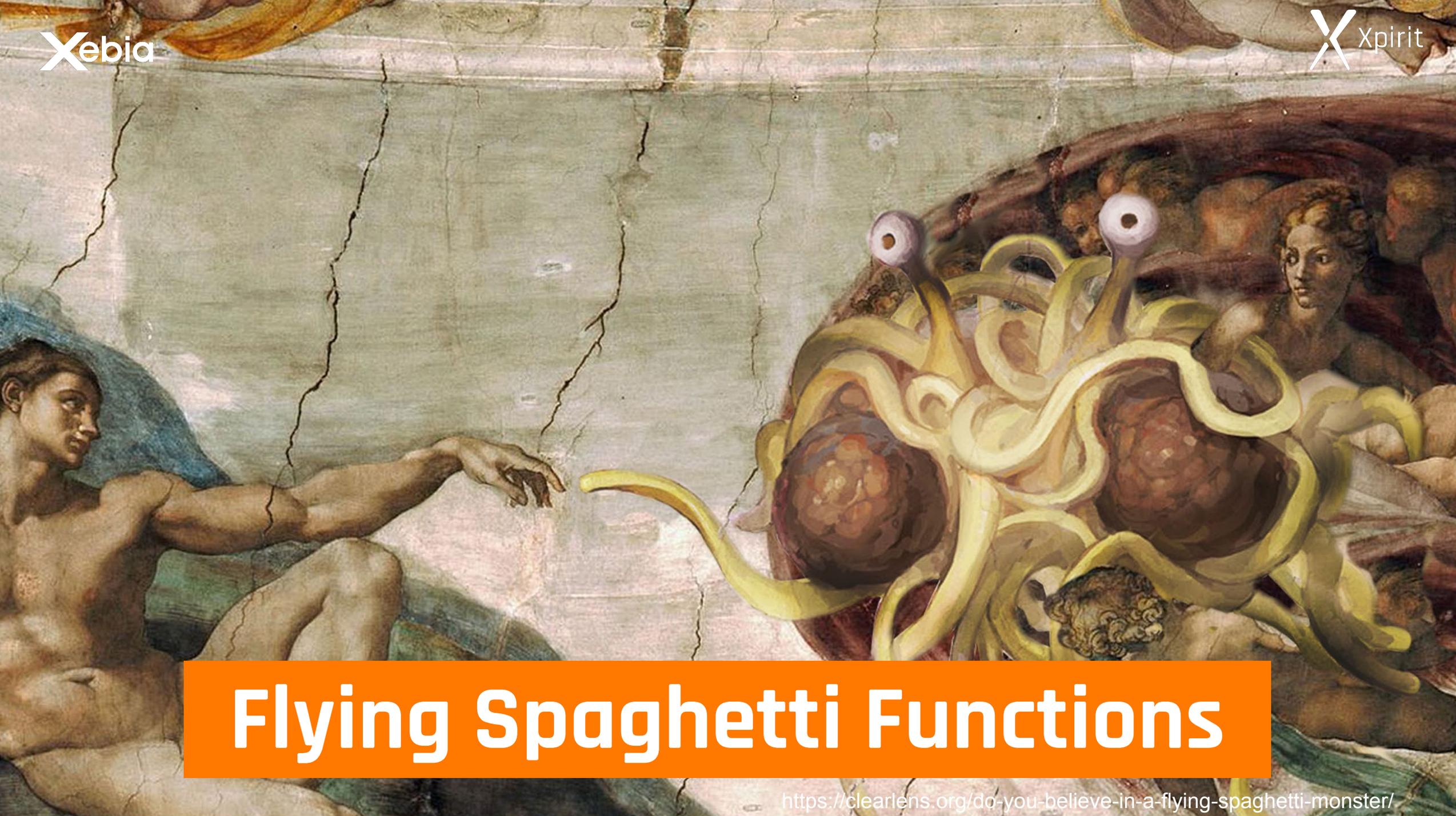
Parking Garage Use Case



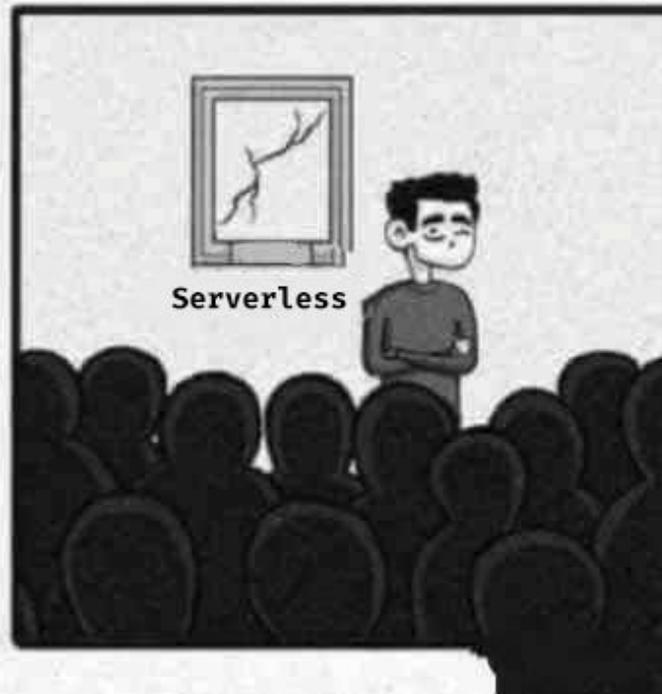
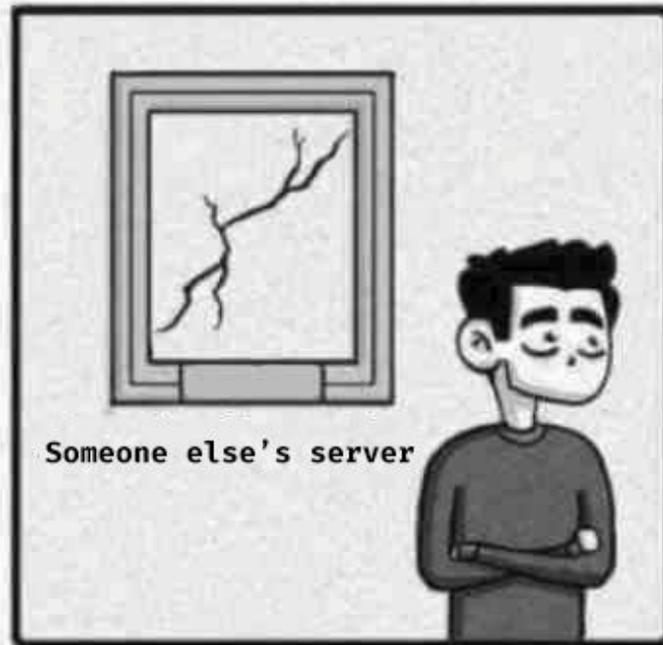
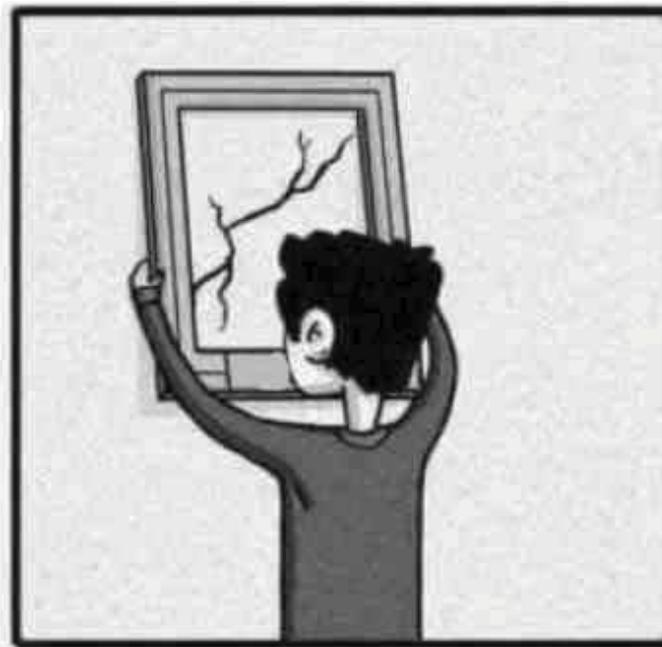
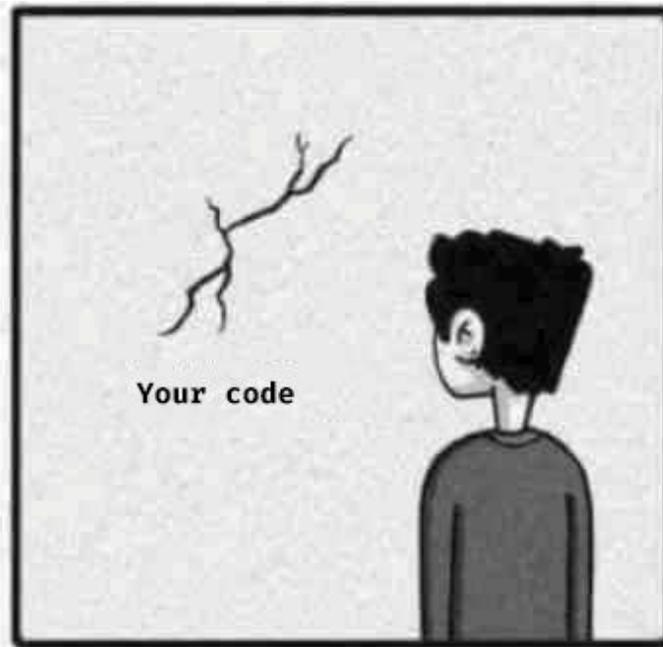
You get a FAAS



Everybody gets a FAAS



Flying Spaghetti Functions



Domain-Driven

DESIGN

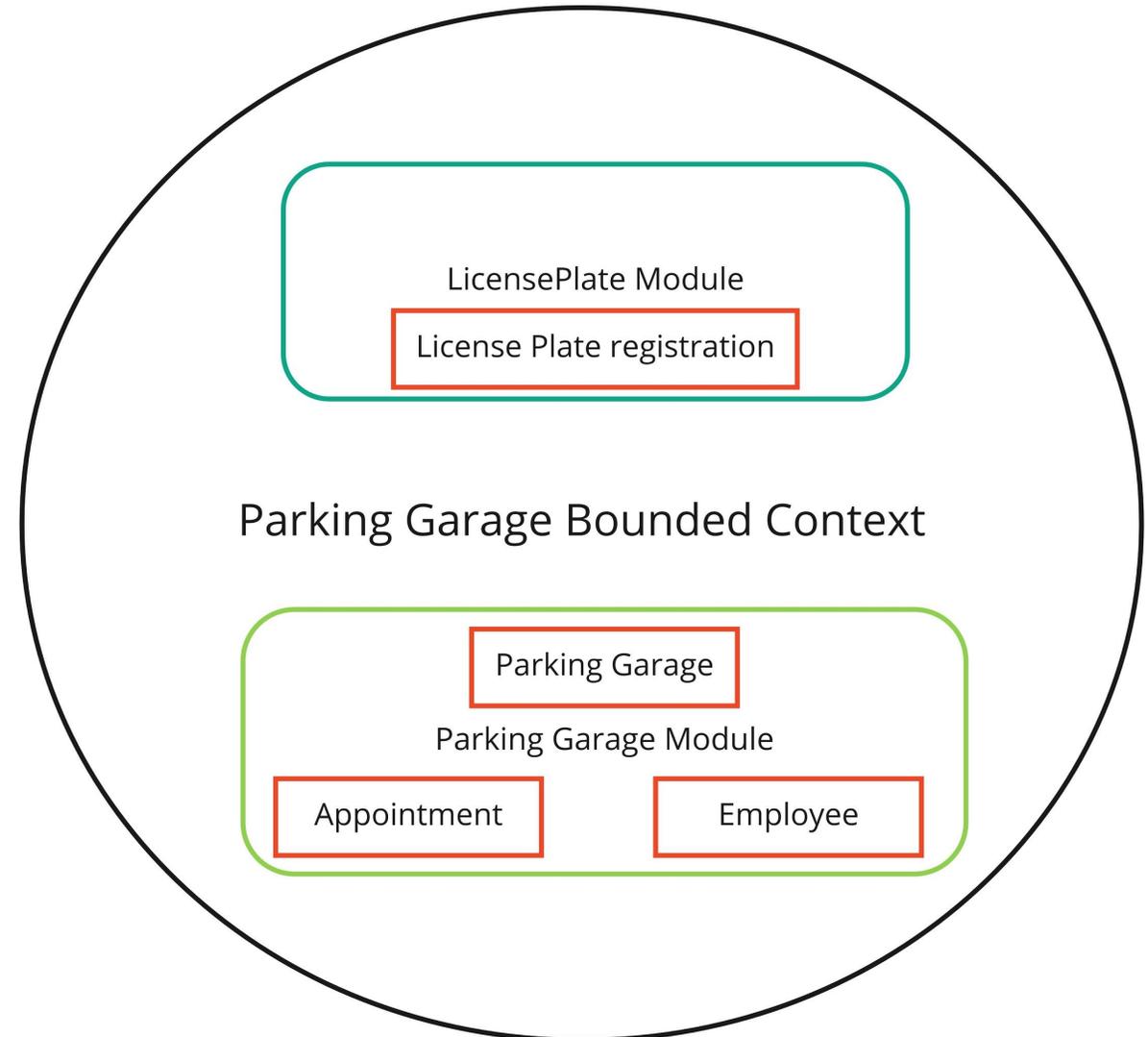
EVENT STORMING



Question: How do we split systems into functions?

Heuristic: Within a bounded context modules are a perfect candidate for splitting up into functions.

Heuristic: Use one source control repository per bounded context.





Portal

Documentation



SDK



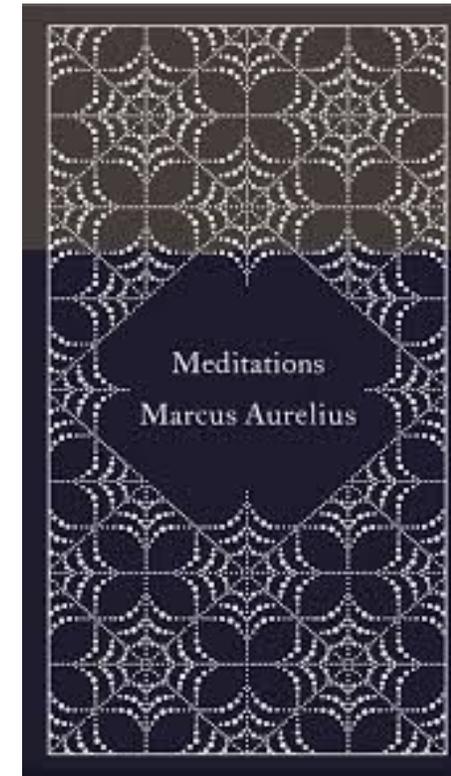
Testing

Orchestration

Monitoring

“Everything we hear is an *opinion*, *not a fact*.
Everything we see is a *perspective*, *not the truth*.”

– Marcus Aurelius , Meditations



Documentation - AWS

The screenshot shows the AWS Lambda Developer Guide page. The top navigation bar includes the AWS logo, social media icons, a language dropdown set to 'English', and a 'Sign In to the Console' button. The left sidebar contains a search bar and a list of navigation links for the AWS Lambda Developer Guide, with 'What Is AWS Lambda?' selected. The main content area features the breadcrumb 'AWS Documentation » AWS Lambda » Developer Guide » What Is AWS Lambda?' and the title 'What Is AWS Lambda?'. The text explains that AWS Lambda is a compute service that runs code without provisioning servers, scales automatically, and charges only for compute time. It lists supported languages (Node.js, Java, C#, Go, Python) and describes how to use Lambda for various use cases, such as responding to events or building data processing pipelines. A 'When Should I Use AWS Lambda?' section highlights its suitability for many application scenarios. A right-hand sidebar contains a 'On this page:' section with links to 'When Should I Use AWS Lambda?' and 'Are You a First-time User of AWS Lambda?'. The footer contains the AWS logo, 'Terms of Use | Privacy | © 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.', and a 'Feedback' button.

aws

English

Sign In to the Console

AWS Lambda
Developer Guide

Documentation - This Guide

Search

What Is AWS Lambda?

- + Getting Started
- + Lambda Functions
- + Configuring Functions
- + Invoking Functions
- + Lambda Runtimes
- + Lambda Applications
- + Use Cases
- + Monitoring
- + Administration
- + Working with Node.js
- + Working with Python
- + Working with Java
- + Working with Go
- + Working with C#
- + Working with PowerShell
- + Working with Ruby
- + API Reference
- Releases
- AWS Glossary

AWS Documentation » AWS Lambda » Developer Guide » What Is AWS Lambda?

What Is AWS Lambda?

AWS Lambda is a compute service that lets you run code without provisioning or managing servers. AWS Lambda executes your code only when needed and scales automatically, from a few requests per day to thousands per second. You pay only for the compute time you consume - there is no charge when your code is not running. With AWS Lambda, you can run code for virtually any type of application or backend service - all with zero administration. AWS Lambda runs your code on a high-availability compute infrastructure and performs all of the administration of the compute resources, including server and operating system maintenance, capacity provisioning and automatic scaling, code monitoring and logging. All you need to do is supply your code in one of the languages that AWS Lambda supports (currently Node.js, Java, C#, Go and Python).

You can use AWS Lambda to run your code in response to events, such as changes to data in an Amazon S3 bucket or an Amazon DynamoDB table; to run your code in response to HTTP requests using Amazon API Gateway; or invoke your code using API calls made using AWS SDKs. With these capabilities, you can use Lambda to easily build data processing triggers for AWS services like Amazon S3 and Amazon DynamoDB, process streaming data stored in Kinesis, or create your own back end that operates at AWS scale, performance, and security.

You can also build [serverless](#) applications composed of functions that are triggered by events and automatically deploy them using AWS CodePipeline and AWS CodeBuild. For more information, see [AWS Lambda Applications](#).

For more information about the AWS Lambda execution environment, see [Lambda Execution Environment and Available Libraries](#). For information about how AWS Lambda determines compute resources required to execute your code, see [Basic AWS Lambda Function Configuration](#).

When Should I Use AWS Lambda?

AWS Lambda is an ideal compute platform for many application scenarios, provided that you can write your application code in languages supported by AWS Lambda (that is, Node.js, Java, Go and C# and Python), and run within the AWS Lambda standard runtime environment and resources provided by Lambda.

When using AWS Lambda, you are responsible only for your code. AWS Lambda manages the compute fleet that offers a balance of memory, CPU, network, and other resources. This is in exchange for flexibility, which means you cannot log in to compute instances, or customize the operating system or language runtime. These constraints enable AWS Lambda to perform operational and administrative activities on your behalf, including provisioning capacity, monitoring fleet health, applying security patches, deploying your code, and monitoring and logging your Lambda functions.

If you need to manage your own compute resources, Amazon Web Services also offers other compute services to meet your needs.

On this page:

- When Should I Use AWS Lambda?
- [Are You a First-time User of AWS Lambda?](#)

Terms of Use | Privacy | © 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Have a question? Try the Forums. Did this page help you? Yes No

Documentation - AWS



Documentation - Azure

The screenshot shows a web browser displaying the Microsoft Azure documentation page. The browser's address bar shows the URL: <https://docs.microsoft.com/en-us/azure/azure-functions/function-quickstarts/create-first-function-visual-studio>. The page header includes the Microsoft Azure logo, contact information (1-800-867-1389), a search bar, and a 'Portal' link. The navigation menu includes 'Overview', 'Solutions', 'Products', 'Documentation', 'Pricing', 'Training', 'Marketplace', 'Partners', 'Support', 'Blog', and 'More'. The current page is 'Azure / Functions'. The main content area features a large heading 'Create your first function using Visual Studio' with a sub-heading 'Studio'. Below the heading is the date '10/17/2018', reading time '5 minutes to read', and contributors. The text describes Azure Functions as a serverless environment and explains that the article teaches how to use Visual Studio 2017 tools to create and test a 'hello world' function. A screenshot of a web browser shows the URL <https://mydemofunctionapp20181016.azurewebsites.net/api/Function1?name=Bill> and the response 'Hello, Bill'. The right sidebar contains a 'In this article' section with links to 'Prerequisites', 'Create a function app project', 'Test the function locally', 'Publish the project to Azure', 'Test your function in Azure', 'Watch the video', and 'Next steps'. The left sidebar shows a 'Filter by title' search box and a list of navigation items, with 'Create function - Visual Studio' highlighted.

Microsoft Azure

Contact Sales: 1-800-867-1389 Search Portal

Overview Solutions Products Documentation Pricing Training Marketplace Partners Support Blog More Free account >

Azure / Functions Feedback Edit Share Dark Sign in

Filter by title

Functions Documentation

- > Overview
- > Quickstarts
 - Create function - Visual Studio
 - Create function - Visual Studio Code
 - Create function - Java/Maven
 - Create function - Python
 - Create function - Azure CLI
 - Create function - portal
 - Create function - Linux
- > Triggers
- > Integrate
- > Tutorials
- > Samples
- > Concepts
- > How-to guides

Create your first function using Visual Studio

10/17/2018 • 5 minutes to read • Contributors all

Azure Functions lets you execute your code in a [serverless](#) environment without having to first create a VM or publish a web application.

In this article, you learn how to use the Visual Studio 2017 tools for Azure Functions to locally create and test a "hello world" function. You then publish the function code to Azure. These tools are available as part of the Azure development workload in Visual Studio 2017.

This topic includes [a video](#) that demonstrates the same basic steps.

Prerequisites

In this article

- [Prerequisites](#)
- [Create a function app project](#)
- [Test the function locally](#)
- [Publish the project to Azure](#)
- [Test your function in Azure](#)
- [Watch the video](#)
- [Next steps](#)

Documentation - GCloud

Compute Products

- Cloud Functions
 - Product Overview
 - [Documentation](#)
- Quickstarts
 - All Quickstarts
 - Using the gcloud Command-Line Tool
 - Using the Console
- How-to Guides
 - All How-to Guides
 - Writing Cloud Functions
 - Deploying Cloud Functions
 - Calling Cloud Functions
 - Securing Cloud Functions
 - Monitoring Cloud Functions
 - Best Practices
 - Connecting to Cloud SQL
 - Using Environment Variables
 - Node.js Emulator 🐘
- APIs & Reference
 - All APIs & References
 - gcloud Command Reference
 - Migration Guide: v1beta2 to v1
 - REST Reference
 - RPC Reference
 - IAM Reference
 - System Packages
- Concepts

Cloud Functions



Google Cloud Functions documentation

[FEEDBACK VERZENDEN](#)

Google Cloud Functions is a lightweight compute solution for developers to create single-purpose, stand-alone functions that respond to Cloud events without the need to manage a server or runtime environment.

| | | |
|---|---|---|
| <p>Quickstart Create & deploy Cloud Functions</p> | <p>How-to guides Perform specific tasks</p> | <p>APIs & reference REST, RPC, and gcloud reference.</p> |
| <p>Concepts Develop a deep understanding of Cloud Functions</p> | <p>Tutorials Walkthroughs of common applications</p> | <p>Support Get assistance with Google Cloud Functions issues</p> |
| <p>Resources Pricing, quotas, release notes, and other resources</p> | | |

Was deze pagina nuttig? Laat ons weten hoe goed we u hebben geholpen:



[FEEDBACK VERZENDEN](#)

Documentation

| | + | - |
|------------------------|---|---|
| AWS Lambda | <ul style="list-style-type: none">● Extensive, detailed documentation.● Everything you need! | <ul style="list-style-type: none">● Extensive wall of text crits you |
| Azure Functions | <ul style="list-style-type: none">● Quickstarts for novices● Detailed reference material for the more experienced. | <ul style="list-style-type: none">● Not 100% complete across all languages. |
| GCP Functions | <ul style="list-style-type: none">● Quickstarts for novices. | <ul style="list-style-type: none">● Hard to find specific information |

SDK - AWS

```
5 export function handle(event: APIGatewayEvent, context: Context, callback: Callback) {  
6  
7     let licensePlate: LicensePlate;  
8     try {  
9         licensePlate = LicensePlate.fromJSON(JSON.parse(event.body as string));  
10    } catch (error) {  
11        console.log("error: "+error);  
12        return callback( error: null, Response.BAD_REQUEST(error));  
13    }  
14  
15    return callback( error: null, Response.OK(licensePlate.number));  
16 }
```

<https://www.npmjs.com/package/@types/aws-lambda>

SDK - AWS

```
export class Response {  
  
    public static OK = (body: string) => new Response( statusCode: 200, body);  
    public static BAD_REQUEST = (body: string) => new Response( statusCode: 400, body);  
    public static CONFLICT = (body: string) => new Response( statusCode: 409, body);  
    public static INTERNAL_SERVER_ERROR = (body: string) => new Response( statusCode: 500, body);  
  
    private headers: { [name: string]: string } = {};  
  
    constructor(public readonly statusCode: number, public readonly body: string) {  
  
    }  
}
```

SDK - AWS

```
public static async saveParkingGarage(parkingGarage: ParkingGarage): Promise<ParkingGarage> {  
  const tableName = await resolveTableName();  
  const params: PutItemInput = {  
    TableName: tableName,  
    Item: parkingGarage.toJSON() as PutItemInputAttributeMap,  
  };  
  return documentClient.put(params).promise().then(  
    onfulfilled: data => Promise.resolve(parkingGarage),  
    onrejected: error => Promise.reject(error)  
  );  
}
```

SDK - AWS

```
public static async findParkingGarage(id: string): Promise<ParkingGarage> {
  const tableName = await resolveTableName();

  const params = {
    TableName: tableName,
    KeyConditionExpression: '#id = :id',
    ExpressionAttributeNames: {
      '#id': 'id',
    },
    ExpressionAttributeValues: {
      ':id': id,
    }
  };

  let data = await documentClient.query(params).promise();
  if (data.Items) {
    return Promise.resolve(ParkingGarage.fromJSON(data.Items.pop() as any));
  }
  return Promise.reject( reason: 'no parking garage found');
}
```

SDK - Azure

```
0 references
public static class Function1
{
    [FunctionName("Function1")]
    public static async Task<IActionResult> Run(
        [HttpTrigger(AuthorizationLevel.Function, "get", "post", Route = null)] HttpRequest req,
        ILogger log)
    {
        log.LogInformation("C# HTTP trigger function processed a request.");

        string name = req.Query["name"];

        string requestBody = await new StreamReader(req.Body).ReadToEndAsync();
        dynamic data = JsonConvert.DeserializeObject(requestBody);
        name = name ?? data?.name;

        return name != null
            ? (ActionResult)new OkObjectResult($"Hello, {name}")
            : new BadRequestObjectResult("Please pass a name on the query string or in the request body");
    }
}
```

SDK - Azure

1 reference | 0 changes | 0 authors, 0 changes

```
public async Task<ParkingGarage> GetByIdAsync(string id)
{
    ParkingGarage result;
    var documentUri = UriFactory.CreateDocumentUri(DatabaseId, CollectionId, id);
    var requestOptions = new RequestOptions { PartitionKey = new PartitionKey(PartitionKeyValue) };
    try
    {
        var document = await _documentClient.ReadDocumentAsync<ParkingGarage>(documentUri, requestOptions);
        result = document.Document;
    }
    catch (Exception e)
    {
        throw RepositoryExceptionBuilder.CreateExceptionForDocumentRead(documentUri, e);
    }

    return result;
}
```

SDK - GCP

```
1 import {LicensePlate} from "../entities/license-plate";
2 import {Request, Response} from "express";
3
4 export function handle(request: Request, response: Response) {
5     |   let licensePlate: LicensePlate;
6     |   try {
7         |       licensePlate = LicensePlate.fromJSON(request.body);
8     |   } catch (error) {
9         |       console.log("error: " + error);
10        |       return response.status(400).send(error);
11    |   }
12
13    return response.send(licensePlate.number);
14 }
```

```
export class LicensePlateRepository {  
  
    public static async findLicensePlate(license: string): Promise<LicensePlateRegistration> {  
        const query = datastore.createQuery( kind: 'License').filter( property: "license", operator: "=", license);  
  
        const [licensePlateRegistrations] = await datastore.runQuery(query);  
        if (licensePlateRegistrations.length > 0) {  
            return Promise.resolve(LicensePlateRegistration.fromJSON(licensePlateRegistrations.pop() as any));  
        }  
        return Promise.resolve(new LicensePlateRegistration(license, LicensePlateType.UNKNOWN));  
    }  
}
```

https://www.npmjs.com/package/@types/google-cloud__datastore

SDK - GCP

express()

Application

Request

Properties

req.app
req.baseUrl
req.body
req.cookies
req.fresh
req.hostname
req.ip
req.ips
req.method
req.originalUrl
req.params
req.path
req.protocol
req.query
req.route
req.secure
req.signedCookies
req.stale
req.subdomains
req.xhr

Methods

req.accepts()
req.acceptsCharsets()
req.acceptsEncodings()
req.acceptsLanguages()
req.get()
req.is()
req.param()
req.range()

Response

Router

Request

The `req` object represents the HTTP request and has properties for the request query string, parameters, body, HTTP headers, and so on. In this documentation and by convention, the object is always referred to as `req` (and the HTTP response is `res`) but its actual name is determined by the parameters to the callback function in which you're working.

For example:

```
app.get('/user/:id', function(req, res) {  
  res.send('user ' + req.params.id);  
});
```

But you could just as well have:

```
app.get('/user/:id', function(request, response) {  
  response.send('user ' + request.params.id);  
});
```

The `req` object is an enhanced version of Node's own request object and supports all [built-in fields and methods](#).

Properties

In Express 4, `req.files` is no longer available on the `req` object by default. To access uploaded files on the `req.files` object, use multipart-handling middleware like [busboy](#), [multer](#), [formidable](#), [multiparty](#), [connect-multiparty](#), or [pez](#).

req.app

This property holds a reference to the instance of the Express application that is using the middleware.

If you follow the pattern in which you create a module that just exports a middleware function and `require()` it in your main file, then the middleware can access the Express instance via `req.app`

For example:

```
//index.js  
app.get('/viewdirectory', require('./middleware.js'))
```

| | + | - |
|------------------------|--|--|
| AWS Lambda | <ul style="list-style-type: none">● Flexible SDK, extensive.● Can also create your own runtime. | <ul style="list-style-type: none">● Quirks in the gateway body as string. Need to do validation yourself, also typed. |
| Azure Functions | <ul style="list-style-type: none">● Easy to use SDK for C#. | <ul style="list-style-type: none">● Limited usage of certain NuGet package versions the framework is using (e.g. Newtonsoft.Json). |
| GCP Functions | <ul style="list-style-type: none">● Using standard 3th party libraries. | |

Orchestration - AWS

```
Type: "AWS::StepFunctions::StateMachine"
Properties:
  StateMachineName: serverless-showdown-state-machine
  DefinitionString: !Sub |
    {
      "Comment": "Car Request",
      "StartAt": "FindMatchingLicensePlate",
      "Version": "1.0",
      "States": {
        "FindMatchingLicensePlate": {
          "Type": "Task",
          "Resource": "${FindMatchingLicensePlate.Arn}",
          "Next": "HandleParking"
        },
        "HandleParking": {
          "Type": "Choice",
          "Choices": [
            {
              "Variable": "$.type",
              "StringEquals": "NoLicensePlateRegistered",
              "Next": "CallOfficeSupport"
            }
          ]
        }
      }
    }
```

```
RoleParkingAccessStateMachine:
  Type: 'AWS::IAM::Role'
  Properties:
    AssumeRolePolicyDocument:
      Version: '2012-10-17'
      Statement:
        - Effect: Allow
          Principal:
            Service: !Sub 'states.${AWS::Region}.amazonaws.com'
          Action: 'sts:AssumeRole'
    Policies:
      - PolicyName: lambda
        PolicyDocument:
          Statement:
            - Effect: Allow
              Action: 'lambda:InvokeFunction'
              Resource:
                - !GetAtt 'FindMatchingLicensePlate.Arn'
                - !GetAtt 'ConfirmAppointment.Arn'
                - !GetAtt 'ConfirmEmployee.Arn'
```

<https://github.com/binxio/aws-cfn-update>

Orchestration - AWS

Execution ARN

arn:aws:states:eu-west-1:761563646002:execution:serverless-showdown-state-machine:7642ab7a-7b92-6464-638c-b1be2e273f5b

▶ Input

End Time

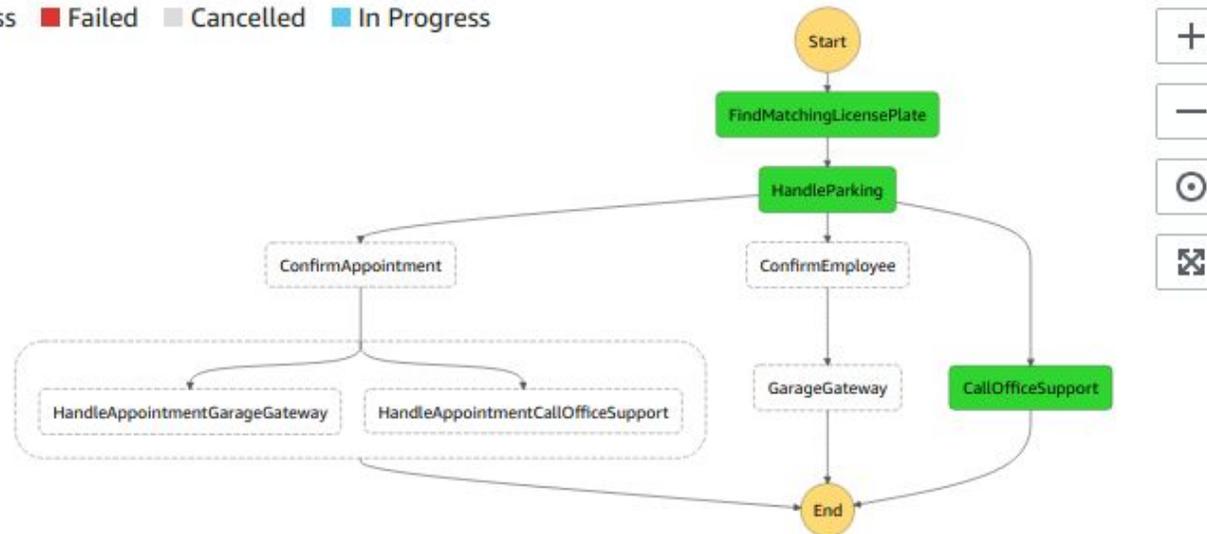
Feb 5, 2019 06:03:05.219 AM

▶ Output

Visual workflow

Code

■ Success ■ Failed ■ Cancelled ■ In Progress



Step details

Select a step to view its details.

Orchestration - AWS

Visual workflow | Code

■ Success
 ■ Failed
 ■ Cancelled
 ■ In Progress

```

            graph TD
                Start((Start)) --> FindMatchingLicensePlate[FindMatchingLicensePlate]
                FindMatchingLicensePlate --> HandleParking[HandleParking]
                HandleParking --> ConfirmAppointment[ConfirmAppointment]
                HandleParking --> ConfirmEmployee[ConfirmEmployee]
                HandleParking --> CallOfficeSupport[CallOfficeSupport]
                ConfirmAppointment --> HandleAppointmentGarageGateway[HandleAppointmentGarageGateway]
                ConfirmAppointment --> HandleAppointmentCallOfficeSupport[HandleAppointmentCallOfficeSupport]
                HandleAppointmentGarageGateway --> End((End))
                HandleAppointmentCallOfficeSupport --> End
                GarageGateway[GarageGateway] --> End
                CallOfficeSupport --> End
                ConfirmEmployee --> End
            
```

Step details (ConfirmEmployee)

Status
⊗ Failed

Resource
[arn:aws:lambda:eu-west-1:761563646002:function:pg-confirmemployee-kb-local](#) | [CloudWatch logs](#)

Input

```

            {
              "number": "170",
              "type": "EmployeeLicensePlateMatched"
            }
        
```

Output

Exception

Error
 Lambda.Unknown

Cause
 The cause could not be determined because Lambda did not return an error type.

Orchestration - Azure

2 references | Marc Duiker, 1 day ago | 1 author, 1 change

```
18 public static class ParkingGarageCarEntryOrchestration
19 {
20     [FunctionName(nameof(ParkingGarageCarEntryOrchestration))]
21     public static async Task<ParkingOrchestrationResponse> Run(
22         [OrchestrationTrigger] DurableOrchestrationContextBase context,
23         ILogger logger)
24     {
25         if (!context.IsReplaying)
26         {
27             logger.LogInformation($"Started {nameof(ParkingGarageCarEntryOrchestration)} with InstanceId: {context.InstanceId}.");
28         }
29
30         var request = context.GetInput<ParkingOrchestrationRequest>();
31
32         var licensePlateResult = await context.CallActivityAsync<LicensePlateRegistration>(
33             nameof(GetLicensePlateRegistration),
34             request.LicensePlateNumber);
35
36
37         var confirmParkingRequest = ConfirmParkingRequestBuilder.Build(request.ParkingGarageName, licensePlateResult);
38         var confirmParkingResponse = await ConfirmParking(confirmParkingRequest, licensePlateResult, context);
39
40         if (confirmParkingResponse.IsSuccess)
41         {
42             await context.CallActivityAsync(
43                 nameof(OpenGate),
44                 confirmParkingResponse.ParkingGarageName);
45         }
46     }
47     else
48     {
```

SDK - Azure

3 references | Marc Duiker, 1 day ago | 1 author, 1 change

```
public static class GetLicensePlateRegistration
```

```
{
```

```
    private static readonly ILicensePlateRegistrationService Service = new LicensePlateRegistrationService();
```

```
    [FunctionName(nameof(GetLicensePlateRegistration))]
```

0 references | Marc Duiker, 1 day ago | 1 author, 1 change

```
    public static async Task<Domain.LicensePlateRegistration> Run(  
        [ActivityTrigger] string licensePlateNumber,  
        ILogger logger)
```

```
    {
```

```
        logger.LogInformation($"Started {nameof(GetLicensePlateRegistration)} with {licensePlateNumber}.");
```

```
        var licenseplate = await Service.GetAsync(licensePlateNumber);
```

```
        return licenseplate;
```

```
    }
```

```
}
```

Orchestration - GCP

Not supported OOTB :(

Orchestration

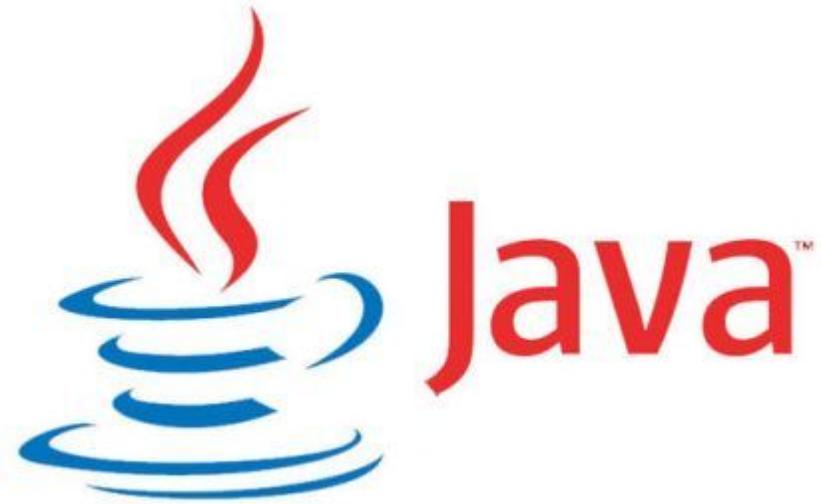
| | + | - |
|------------------------|---|--|
| AWS Lambda | <ul style="list-style-type: none"> ● Visualisation in the portal | <ul style="list-style-type: none"> ● Uses JSON/YAML configuration. |
| Azure Functions | <ul style="list-style-type: none"> ● Flexibility because the orchestration is in code. | <ul style="list-style-type: none"> ● The orchestration needs to be deterministic, so don't use code which is not (e.g. new GUIDs, DateTime, new threads). |
| GCP Functions | <ul style="list-style-type: none"> ● N/A | <ul style="list-style-type: none"> ● N/A |

Deployment Cycle - AWS

```
FindMatchingLicensePlate:
  Type: 'AWS::Serverless::Function'
  Properties:
    FunctionName: !Sub "pg-findmatchinglicenseplate-${Stage}"
    Handler: find-matching-license-plate.handle
    CodeUri: dist/
    Description: FaaS handler for requesting a car entry
    MemorySize: 128
    ReservedConcurrentExecutions: 20
    Environment:
      Variables:
        LICENSE_PLATE_TABLE_NAME: !Ref 'LicensePlateRepository'
    Policies:
      - Version: "2012-10-17"
        Statement:
          - Effect: "Allow"
            Action:
              - "dynamoDb:Query"
            Resource:
              !Sub 'arn:aws:dynamodb:${AWS::Region}:${AWS::AccountId}:table/LicensePlateRepository*'

```

Deployment Cycle - AWS



<https://xebia.com/blog/building-an-elixir-runtime-for-aws-lambda/>

Deployment Cycle - AWS

Stacks (2)

Active

| Stack name | Status | Created time |
|--|--|-------------------------------|
| serverless-showdown-kb-local | ✔ UPDATE_COMPLETE | Mon, 04 Feb 2019 14:23:40 GMT |

| | | | | |
|---------------------|--|----------------------------------|------------------------------|---|
| ▶ 16:46:51 UTC-0500 | UPDATE_ROLLBACK_COMPLETE | AWS::CloudFormation::Stack | serverless-showdown-kb-local | |
| ▶ 16:46:49 UTC-0500 | UPDATE_ROLLBACK_COMPLETE_CLEANUP_IN_PROGRESS | AWS::CloudFormation::Stack | serverless-showdown-kb-local | |
| ▶ 16:46:48 UTC-0500 | UPDATE_COMPLETE | AWS::StepFunctions::StateMachine | ParkingAccessStateMachine | |
| ▶ 16:46:46 UTC-0500 | UPDATE_COMPLETE | AWS::Lambda::Function | GarageGateway | |
| ▶ 16:46:46 UTC-0500 | UPDATE_COMPLETE | AWS::Lambda::Function | ConfirmEmployee | |
| ▶ 16:46:46 UTC-0500 | UPDATE_COMPLETE | AWS::Lambda::Function | FindMatchingLicensePlate | |
| ▶ 16:46:46 UTC-0500 | UPDATE_COMPLETE | AWS::Lambda::Function | ConfirmAppointment | |
| ▶ 16:46:46 UTC-0500 | UPDATE_COMPLETE | AWS::Lambda::Function | ContactNotification | |
| ▶ 16:46:46 UTC-0500 | UPDATE_COMPLETE | AWS::Lambda::Function | RequestCarEntry | |
| ▶ 16:46:46 UTC-0500 | UPDATE_IN_PROGRESS | AWS::Lambda::Function | GarageGateway | |
| ▶ 16:46:46 UTC-0500 | UPDATE_IN_PROGRESS | AWS::Lambda::Function | ConfirmAppointment | |
| ▶ 16:46:46 UTC-0500 | UPDATE_IN_PROGRESS | AWS::Lambda::Function | ConfirmEmployee | |
| ▶ 16:46:46 UTC-0500 | UPDATE_IN_PROGRESS | AWS::Lambda::Function | FindMatchingLicensePlate | |
| ▶ 16:46:46 UTC-0500 | UPDATE_IN_PROGRESS | AWS::Lambda::Function | ContactNotification | |
| ▶ 16:46:46 UTC-0500 | UPDATE_IN_PROGRESS | AWS::Lambda::Function | RequestCarEntry | |
| ▶ 16:46:13 UTC-0500 | UPDATE_ROLLBACK_IN_PROGRESS | AWS::CloudFormation::Stack | serverless-showdown-kb-local | The following resource(s) failed to update: [ParkingAccessStateMachine]. |
| ▶ 16:46:12 UTC-0500 | UPDATE_FAILED | AWS::StepFunctions::StateMachine | ParkingAccessStateMachine | Invalid State Machine Definition: 'MISSING_TRANSITION_TARGET: Missing 'Next' target: Choice at /States/FindMatchingLicensePlate/Next, MISSING_TRANSITION_TARGET: State "HandleParking" is not reachable. at /States/HandleParking' (Service: AWSStepFunctions; Status Code: 400; Error Code: InvalidDefinition; Request ID: 49e6359e-28c6-11e9-83bf-e983c9efb683) |
| ▶ 16:46:10 UTC-0500 | UPDATE_IN_PROGRESS | AWS::StepFunctions::StateMachine | ParkingAccessStateMachine | |

Deployment Cycle - Azure

The screenshot shows the Microsoft Azure portal interface. The browser address bar displays `https://portal.azure.com`. The user is logged in as `marcduiker@gmail.co...` (MARC DUIKER).

The left-hand navigation pane includes the following items:

- Create a resource
- Home
- Dashboard
- All services
- FAVORITES
- Resource groups
- Recent
- All resources
- Function Apps
- Logic Apps
- Cognitive Services
- Azure Cosmos DB
- App Services
- Security Center
- Subscriptions
- Monitor
- Cost Management + Billing
- Help + support
- API Connections

The main content area is titled **serverlessparking-fa** (Function Apps). The **Deployment Center** tab is active, showing a workflow diagram with two steps:

- 1 SOURCE CONTROL**
- 2 CONFIGURE**

Below the diagram, two configuration cards are visible:

- Azure Repos:** Configure continuous integration with an Azure Repo, part of Azure DevOps Services (formerly known as VSTS).
- Github:** Configure continuous integration with a GitHub repo.

The user `marcduiker` is associated with the Github configuration card.

Deployment Cycle - Azure

The screenshot shows the Azure DevOps web interface for configuring a pipeline task. The browser tabs include 'ServerlessParking Build - Azure', 'Serverless Parking Deployment', and 'serverlessparking-rg - Microsoft'. The URL is 'https://marcduiker.visualstudio.com/Personal'. The page title is 'Serverless Parking Deploy...'.

The left sidebar shows the navigation menu with 'Pipelines' selected. The main content area shows the 'Tasks' tab for the 'Development (Azure)' pipeline. A task named 'Deploy Azure Function App' is selected, which uses the 'Azure App Service Deploy' provider.

The configuration panel for 'Azure App Service Deploy' includes the following fields:

- Version:** 3.*
- Display name:** Deploy Azure Function App
- Azure subscription:** Windows Azure MSDN - Visual Studio Premium (41aa9018-8016-418b-ab)
- App type:** Function App
- App Service name:** serverlessparking-fa
- Deploy to slot:** (checkbox)

Deployment Cycle - GCP

```
-/+ google_storage_bucket_object.archive (new resource required)
  id:          "showdown-serverless-deploy-http_trigger.zip" => <computed> (forces new resource)
  bucket:      "showdown-serverless-deploy" => "showdown-serverless-deploy"
  content_type: "application/zip" => <computed>
  crc32c:      "sDzRbA==" => <computed>
  detect_md5hash: "hVC+76p+XCpeDt6ZfpFEiA==" => "different hash" (forces new resource)
  md5hash:      "hVC+76p+XCpeDt6ZfpFEiA==" => <computed>
  name:        "http_trigger.zip" => "http_trigger.zip"
  source:      "../dist/index.zip" => "../dist/index.zip"
  storage_class: "STANDARD" => <computed>
```

Plan: 1 to add, 0 to change, 1 to destroy.

Do you want to perform these actions?
Terraform will perform the actions described above.
Only 'yes' will be accepted to approve.

Enter a value: yes

```
google_storage_bucket_object.archive: Destroying... (ID: showdown-serverless-deploy-http_trigger.zip)
google_storage_bucket_object.archive: Destruction complete after 1s
google_storage_bucket_object.archive: Creating...
  bucket:      "" => "showdown-serverless-deploy"
  content_type: "" => "<computed>"
  crc32c:      "" => "<computed>"
  detect_md5hash: "" => "different hash"
  md5hash:     "" => "<computed>"
  name:        "" => "http_trigger.zip"
  source:      "" => "../dist/index.zip"
  storage_class: "" => "<computed>"
google_storage_bucket_object.archive: Creation complete after 0s (ID: showdown-serverless-deploy-http_trigger.zip)
```

Apply complete! Resources: 1 added, 0 changed, 1 destroyed.

The state of your infrastructure has been saved to the path below. This state is required to modify and destroy your infrastructure, so keep it safe. To inspect the complete state use the `terraform show` command.

State path: ./state/terraform.tfstate
baasie@baasie-Precision-5520:~/gitrepos/prive/serverless-showdown/gcloud/infra\$ terraform destroy

Deployment Cycle

| | + | - |
|------------------------|--|--|
| AWS Lambda | <ul style="list-style-type: none"> ● Stateful Cloudformation. | <ul style="list-style-type: none"> ● Sometimes manual interference when config is false. |
| Azure Functions | <ul style="list-style-type: none"> ● Multiple ways to deploy your functions to the cloud. ● Prefer CLI over ARM. | <ul style="list-style-type: none"> ● Watch out for breaking changes in orchestrations. |
| GCP Functions | <ul style="list-style-type: none"> ● Terraform to the rescue! | <ul style="list-style-type: none"> ● Google Cloud deployment configuration counter intuitive. |

Portal - AWS

AWS Management Console

AWS services

Find services
You can enter names, keyword or acronyms.

▶ Recently visited services

▶ All services

Build a solution
Get started with simple wizards and automated workflows.

| | | | |
|--|---|--|---|
| <p>Launch a virtual machine With EC2 ~2-3 minutes</p> | <p>Build a web app With Elastic Beanstalk ~6 minutes</p> | <p>Build using virtual servers With Lightsail ~1-2 minutes</p> | <p>Connect an IoT device With AWS IoT ~5 minutes</p> |
| <p>Start a development project With CodeStar ~5 minutes</p> | <p>Register a domain With Route 53 ~3 minutes</p> | <p>Deploy a serverless microservice With Lambda, API Gateway ~2 minutes</p> | <p>Create a backend for your mobile app With Mobile Hub ~5 minutes</p> |

Learn to build
Learn to deploy your solutions through step-by-step guides, labs, and videos. [See all](#)

| | | | |
|---|---|---|--|
| <p>Websites and Web Apps 3 videos, 3 tutorials, 3 labs</p> | <p>Storage 3 videos, 3 tutorials, 3 labs</p> | <p>Databases 3 videos, 3 tutorials, 3 labs</p> | <p>DevOps 3 videos, 3 tutorials, 3 labs</p> |
|---|---|---|--|

[Machine Learning](#) [Big Data](#)

Access resources on the go

Access the Management Console using the AWS Console Mobile App. [Learn more](#)

Explore AWS

AWS Marketplace
Find, buy, and deploy popular software products that run on AWS. [Learn more](#)

Amazon RDS
Set up, operate, and scale your relational database in the cloud. [Learn more](#)

Scalable, Durable, Secure Backup & Restore with Amazon S3
Discover how customers are building backup & restore solutions on AWS that save money. [Learn more](#)

Run Serverless Containers with AWS Fargate
AWS Fargate runs and scales your containers without having to manage servers or clusters. [Learn more](#)

Have feedback?

[Submit feedback](#) to tell us about your experience with the AWS Management Console.

aws
Services ▾
Resource Groups ▾
✦

AWS Systems Manager ✕

- ▾ Resource Groups
 - Find Resources
 - Saved Resource Groups
- ▾ Insights
 - Built-In Insights
 - Dashboard by CloudWatch
 - Inventory
 - Compliance
- ▾ Actions
 - Automation
 - Run Command
 - Session Manager
 - Patch Manager
 - Maintenance Windows
 - Distributor
 - State Manager
- ▾ Shared Resources
 - Managed Instances
 - Activations
 - Documents
 - Parameter Store

Create query-based group

Group type
Select a group type to define a group based on resource types and tags, or create a group based on your existing CloudFormation stack.

Tag based
Group resources by specifying tags that are shared by the resources.

CloudFormation stack based
Create a resource group based on an existing CloudFormation stack. The group will have the same logical structure as the stack.

Grouping criteria View group resources

Define a group based on resource types and tags.

Resource types

Select resource types ▾

All supported resource types

Tags

Group resources

< 1 > ⚙

| Name | Service | Type | ID |
|---------------|---------|------|----|
| No resources. | | | |

Portal - Azure

The screenshot shows the Microsoft Azure portal interface. The browser address bar indicates the URL is <https://portal.azure.com/#@marcduikerg>. The user is logged in as 'marcduiker@gmail.co...' (MARC DUIKER). The main content area displays the 'serverlessparking-rg' resource group. The left sidebar contains navigation options like 'Create a resource', 'Home', 'Dashboard', and 'All services'. The right pane shows details for the resource group, including subscription information and a table of resources.

serverlessparking-rg
Resource group

Subscription (change) Windows Azure MSDN - Visual Studio Pre... Subscription ID 41aa9018-8016-418b-ab22-18e300477af7

Deployments 2 Succeeded

Tags (change) Click here to add tags

Filter by name... All types All locations No grouping

5 items Show hidden types

| <input type="checkbox"/> | NAME | TYPE | LOCATION |
|--------------------------|--|----------------------|-------------|
| <input type="checkbox"/> | serverlessparking-ai | Application Insights | West Europe |
| <input type="checkbox"/> | serverlessparking-fa | App Service | West Europe |
| <input type="checkbox"/> | serverlessparkingstorage | Storage account | West Europe |
| <input type="checkbox"/> | serverlessprkngfastorage | Storage account | West Europe |
| <input type="checkbox"/> | WestEuropePlan | App Service plan | West Europe |

Portal - Azure

The screenshot shows the Microsoft Azure portal interface. The browser address bar displays `https://portal.azure.com/#blade/Website`. The user is logged in as `marcduiker@gmail.com` (MARC DUIKER).

The navigation pane on the left includes options like "Create a resource", "Home", "Dashboard", "All services", "FAVORITES", "Resource groups", "Recent", "All resources", "Function Apps", "Logic Apps", "Cognitive Services", "Azure Cosmos DB", "App Services", "Security Center", "Subscriptions", "Monitor", "Cost Management + Billing", and "Help + support".

The main content area shows the "serverlessparking-fa" Function App page. The breadcrumb path is `Home > Resource groups > serverlessparking-rg > serverlessparking-fa`. The page title is "serverlessparking-fa" (Function Apps).

There are two tabs: "Overview" (selected) and "Platform features".

Under the "Overview" tab, there are several action buttons: Stop, Swap, Restart, Get publish profile, Reset publish profile, Download app content, and Delete.

The main content area displays the following information:

- Status:** Running (indicated by a green checkmark)
- Subscription:** Windows Azure MSDN - Visual Studio Premium
- Resource group:** serverlessparking-rg
- Subscription ID:** 41aa9018-8016-418b-ab22-18e300477af7
- Location:** West Europe
- URL:** `https://serverlessparking-fa.azurewebsites.net`
- App Service plan / pricing tier:** WestEuropePlan (Consumption)

At the bottom, there is a section for "Configured features" with two links: "Function app settings" and "Application settings".

Portal - GCloud

Navigation menu
ACTIVITY

Project info

Project name
speeltuyn-kenny-baas

Project ID
speeltuyn-kenny-baas

Project number
943579468162

[Go to project settings](#)

Resources

Storage
1 bucket

Cloud Functions
1 function

Trace

No trace data from the last 7 days

[Get started with Stackdriver Trace](#)

Getting started

API Explore and enable APIs

Deploy a prebuilt solution

Add dynamic logging to a running application

API APIs

Requests (requests/sec)

● Requests: 0.033

[Go to APIs overview](#)

Google Cloud Platform status

All services normal

[Go to Cloud status dashboard](#)

Billing

Estimated charges USD \$0.00

For the billing period 1-4 Feb 2019

[View detailed charges](#)

Error Reporting

No sign of any errors. Have you set up Error Reporting?

[Learn how to set up Error Reporting](#)

News

Design better meeting rooms—new hardware partnerships for Hangouts Meet
8 hours ago

10 tips for building long-running clusters using Cloud Dataproc
3 days ago

Reliable streaming pipeline development with Cloud Pub/Sub's Replay
3 days ago

[Read all news](#)

Portal

| | + | - |
|------------------------|--|---|
| AWS Lambda | <ul style="list-style-type: none"> • Most parts are simple and easy to discover. | <ul style="list-style-type: none"> • No real boundaries of projects. • Some parts are hard to navigate. |
| Azure Functions | <ul style="list-style-type: none"> • Resource Groups are useful containers for multiple services. | <ul style="list-style-type: none"> • It is very slow to navigate & use. |
| GCP Functions | <ul style="list-style-type: none"> • Projects is good concept for grouping services. • Portal is quick to use. | |

Testing - AWS



```
1 import {doesJournalEventExist, LocalDate, putBMICalculatedEvent} from '../src/port.adapter.dynamodb/document-client';
2 import {BASIC_BMICALCULATEDEVENT} from '../domain/bmi-meter-events.fixtures';
3 import {PersonId} from '../src/domain/plastic-categories';
4
5 const mockResolveTableName = jest.fn( implementation: () => Promise.resolve( value: 'table'));
6 jest.mock( moduleName: '../src/support/properties', factory: () => ({
7   resolvePlasticDayEventsTableName: () => mockResolveTableName(),
8 }));
9
10 let mockCount = 1;
11 const mockQueryResult = jest.fn( implementation: () => Promise.resolve( value: {
12   Count: mockCount
13 }));
14 const mockPutResult = jest.fn( implementation: () => Promise.resolve( value: {}));
15 jest.mock( moduleName: 'aws-sdk', factory: () => ({
16   DynamoDB: {
17     DocumentClient: class {
18       query() {
19         return {
20           promise: mockQueryResult
21         };
22       };
23
24       put() {
25         return {
26           promise: mockPutResult
27         };
28       };
29     }
30   }
31 }));
```

Testing - AWS



AWS SAM LOCAL

- Python Versions support
 - Python 2.7
 - Python 3.6
 - Python 3.7
- Supported AWS Lambda Runtimes
 - nodejs
 - nodejs4.3
 - nodejs6.10
 - nodejs8.10
 - java8
 - python2.7
 - python3.6
 - python3.7
 - go1.x
 - dotnetcore1.0
 - dotnetcore2.0
 - dotnetcore2.1
 - ruby2.5
 - Provided
- AWS credential support
- Debugging support
- Inline Swagger support within SAM templates
- Validating SAM templates locally
- Generating boilerplate templates
 - nodejs
 - nodejs4.3
 - nodejs6.10
 - nodejs8.10
 - java8
 - python2.7
 - python3.6
 - python3.7
 - go1.x
 - dotnetcore1.0
 - dotnetcore2.0
 - ruby2.5
 - Provided

Testing - Azure

1 reference | Marc Duiker, 4 days ago | 1 author, 1 change

```
private DurableOrchestrationContextBase CreateFakeContextForUnkownLicensePlate()
{
    var context = A.Fake<DurableOrchestrationContextBase>();
    // Configure input
    A.CallTo(() => context.GetInput<ParkingOrchestrationRequest>())
        .Returns(new ParkingOrchestrationRequest
        {
            ParkingGarageName = "Parking Garage 1",
            LicensePlateNumber = "ABC-123"
        });

    // Configure GetLicensePlateRegistration activity
    A.CallTo(() => context.CallActivityAsync<LicensePlateRegistration>(
        nameof(GetLicensePlateRegistration),
        A<string>._))
        .Returns(Task.FromResult(new LicensePlateRegistration {Type = LicensePlateType.Unknown}));

    // Configure DisplayMessage activity
    A.CallTo(() => context.CallActivityAsync(
        nameof(DisplayMessage),
        A<DisplayMessageRequest>._))
        .Returns(Task.CompletedTask);

    return context;
}
```

Testing - Azure

[Fact]

0 references | Marc Duiker, 4 days ago | 1 author, 1 change

```
public async Task GivenLicensePlateIsUnknown_WhenOrchestrationIsStarted_ThenGateOpenedShouldBeFalse()
{
    // Arrange
    var context = CreateFakeContextForUnkownLicensePlate();
    var logger = A.Fake<ILogger>();

    // Act
    var result = await ParkingGarageCarEntryOrchestration.Run(context, logger);

    // Assert
    result.GateOpened.Should().BeFalse();
}
```


Testing - Azure

The screenshot displays a REST client interface for configuring a POST request. The request URL is `http://localhost:7071/api/orchestration/ParkingGarageCarEntryOrchestration`. The request body is a JSON object with the following content:

```
1 {  
2   "licensePlateNumber": "123-ABC",  
3   "parkingGarageName": "Xebia Amsterdam"  
4 }
```

The interface includes tabs for Params, Authorization, Headers (1), Body (selected), Pre-request Script, and Tests. Below the tabs, there are radio buttons for content types: none, form-data, x-www-form-urlencoded, raw (selected), and binary. The content type is set to `JSON (application/json)`. A Beautify button is visible on the right side of the body editor.

Testing - GCP

```
baasie@baasie-Precision-5520:~/gitrepos/prive/serverless-showdown/gcloud$ functions start
Starting Google Cloud Functions Emulator...
Google Cloud Functions Emulator STARTED
```

| Status | Name | Trigger | Resource |
|--------|--------|---------|---|
| READY | handle | HTTP | http://localhost:8010/speeltuyn-kenny-baas/us-central1/handle |

```
baasie@baasie-Precision-5520:~/gitrepos/prive/serverless-showdown/gcloud$ functions call handle --data='{"number":"test"}'
ExecutionId: 63abd6a3-4e7c-4eb5-97c2-84eebf8e648e
Result: test
baasie@baasie-Precision-5520:~/gitrepos/prive/serverless-showdown/gcloud$ █
```

Testing

| | + | - |
|------------------------|--|---|
| AWS Lambda | <ul style="list-style-type: none"> ● Unit testing is easy ● SAM Local seems promising | <ul style="list-style-type: none"> ● SAM Local still in beta ● Mocking dynamo takes some time to understand |
| Azure Functions | <ul style="list-style-type: none"> ● Unit testing is easy ● Local runtime is available for running without cloud connection. | <ul style="list-style-type: none"> ● Local runtime depends on a separate storage emulator (not all versions are compatible). |
| GCP Functions | <ul style="list-style-type: none"> ● Node Emulator available for local testing. | <ul style="list-style-type: none"> ● Emulator is in alpha version ● Supports Node v6 only |

Monitoring - AWS

pg-requestcarentry-kb-local

Throttle Qualifiers Actions Select a test event. Test Save

Configuration **Monitoring**

CloudWatch metrics at a glance

View logs in CloudWatch View traces in X-Ray

1h 3h 12h 1d 3d 1w custom Refresh

Invocations

No data available. Try adjusting the dashboard time range.

12:30 13:00 13:30 14:00 14:30 15:00 15:30

Invocations

Duration

No data available. Try adjusting the dashboard time range.

12:30 13:00 13:30 14:00 14:30 15:00 15:30

Duration Minimum Duration Average Duration Maximum

Errors, Availability (%)

No data available. Try adjusting the dashboard time range.

12:30 13:00 13:30 14:00 14:30 15:00 15:30

Errors Availability (%)

Throttles

No data available. Try adjusting the dashboard time range.

12:30 13:00 13:30 14:00 14:30 15:00 15:30

Throttles

IteratorAge

No data available. Try adjusting the dashboard time range.

12:30 13:00 13:30 14:00 14:30 15:00 15:30

IteratorAge

DeadLetterErrors

No data available. Try adjusting the dashboard time range.

12:30 13:00 13:30 14:00 14:30 15:00 15:30

DeadLetterErrors

Monitoring - Azure

The screenshot displays the Azure Application Insights monitoring dashboard. On the left is a navigation sidebar with sections: Overview (selected), Activity log, Access control (IAM), Tags, Diagnose and solve problems, Investigate (Application map, Smart Detection, Live Metrics Stream, Metrics, Search, Availability, Failures, Performance, Servers, Browser, Troubleshooting guides, Workbooks). The main area contains four charts for Jan 23 (3 AM to 9 AM):

- Failed requests:** A bar chart showing 30 failed requests in total. The y-axis ranges from 0 to 8. A summary bar at the bottom shows 30.
- Server response time:** A line chart showing average response time. The y-axis ranges from 0ms to 4sec. A summary bar at the bottom shows 217.31 ms.
- Server requests:** A line chart showing the number of server requests. The y-axis ranges from 0 to 1.2k. A summary bar at the bottom shows 12.81 k.
- Availability:** A line chart showing availability percentage. The y-axis ranges from 0% to 100%. A summary bar at the bottom shows --.

Monitoring - GCP

request-car-entry

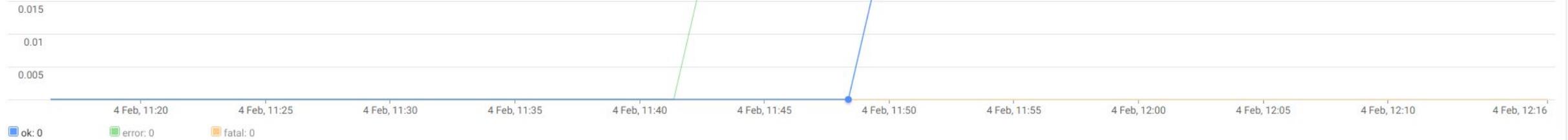
Version 1, deployed at 4 Feb 2019, 11:48:16

General Trigger Source Testing

Invocations

1 hour 6 hours 12 hours 1 day 2 days 4 days 7 days 14 days 30 days

Invocations/second



Last deployed

4 February 2019 at 11:48:16 UTC-5

Region

us-central1

Memory allocated

128 MB

Timeout

61 seconds

Service account

speeltuin-kenny-baas@appspot.gserviceaccount.com

Errors in the last 7 days

No error reported

Environment variables

Monitoring

| | + | - |
|------------------------|--|---|
| AWS Lambda | <ul style="list-style-type: none"> ● CloudWatch is mature, gives the right amount of stats | |
| Azure Functions | <ul style="list-style-type: none"> ● Basic logging OOTB and easy integration with Application Insights. | |
| GCP Functions | <ul style="list-style-type: none"> ● Basic monitoring OOTB | |

Key takeaways

Question: When do we use **AWS**?

Heuristic: When we want to use many different function runtimes.

Heuristic: When we want to use the latest runtimes.

Key takeaways

Question: When do we use **Azure**?

Heuristic: When we're familiar with the Microsoft ecosystem.

Heuristic: When we want to orchestrate functions.

Key takeaways

Question: When do we use **Google Cloud**?

Heuristic: When we want the best functions portal experience.

Key takeaways

Question: When should we use **Orchestrations**?

Heuristic: Use orchestrations within a bounded context.

Heuristic: Use orchestrations when having processes over multiple modules in your bounded context.

Heuristic: Use orchestrations for long running processes.

Sources

Pricing

<https://aws.amazon.com/lambda/pricing/>

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

<https://cloud.google.com/functions/pricing>

Comparison:

<https://docs.google.com/spreadsheets/d/1Q6vllVYe1CfHAK6MP6Uz6mTYPiU4DliwL8XN4E5FQdw/edit?usp=sharing>

Runtimes & languages

<https://docs.aws.amazon.com/lambda/latest/dg/lambda-runtimes.html>

<https://docs.microsoft.com/en-us/azure/azure-functions/supported-languages>

<https://cloud.google.com/functions/docs/writing/>

Cold Start

<https://mikhail.io/2018/08/serverless-cold-start-war/>

Github Repository with our demo code

<https://github.com/Baasie/serverless-showdown>

