#SCYWAWA

SUPERCHARGE YOUR WEB APPLICATION WITH AZURE

Introduction

Rick van den Bosch

@rickvdbosch

www.rickvandenbosch.net

Oscar van Tol

@oscarvantol

oscarvantol.wordpress.com

Todo list

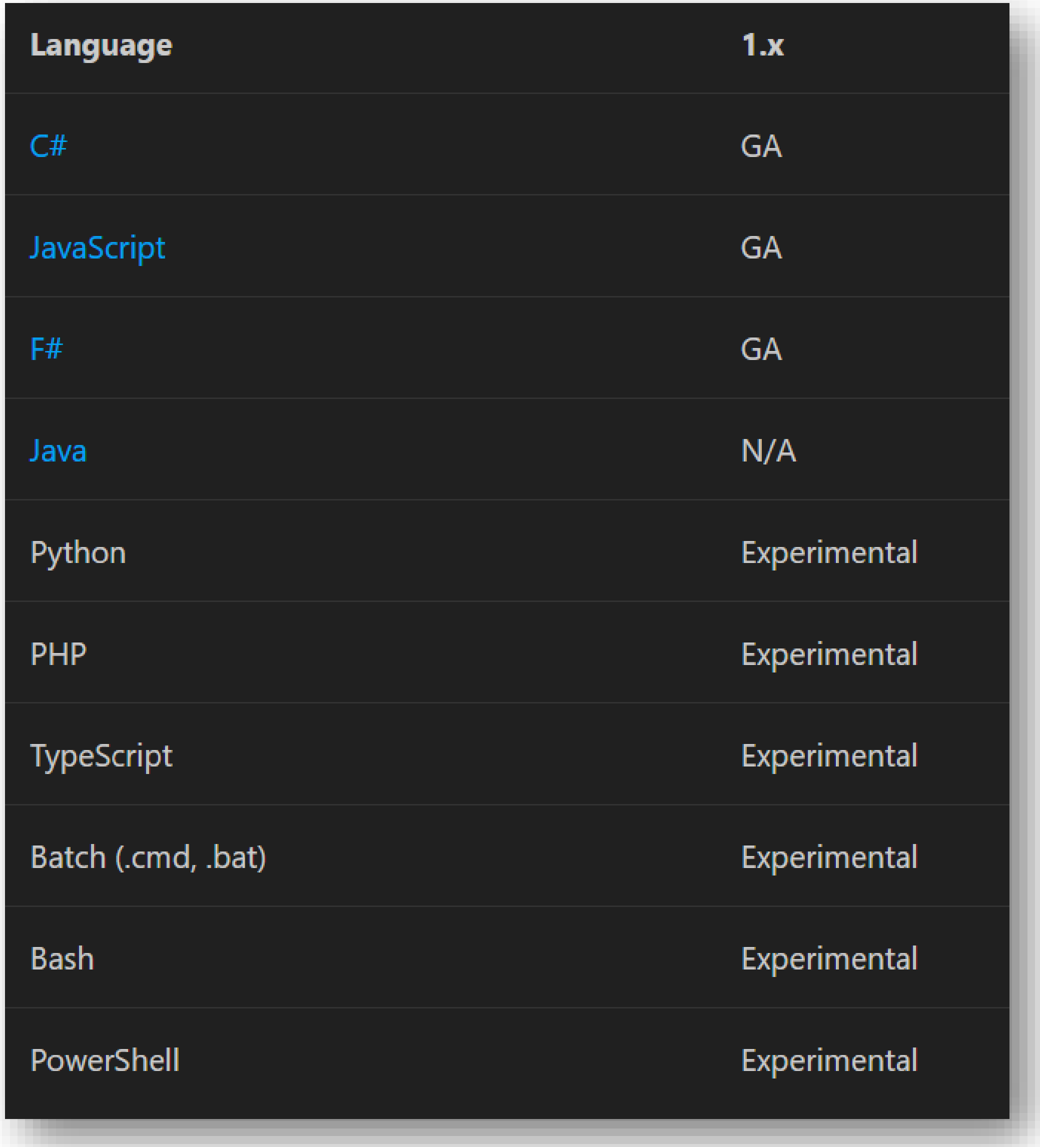
- ✓ Introduction
- ✓ Agenda
- ❑ Azure Functions, what’s up?
    - ❑ Durable Functions
    - ❑ Azure Functions 2.x (.NET Core)
- ❑ Azure CDN
- ❑ Azure Media Services
- ❑ ASP.NET Core 2.1 SignalR
- ❑ Azure SignalR Service
- ❑ Azure Functions + SignalR Service

Code
Events + data
Azure Functions

Azure Functions

Features

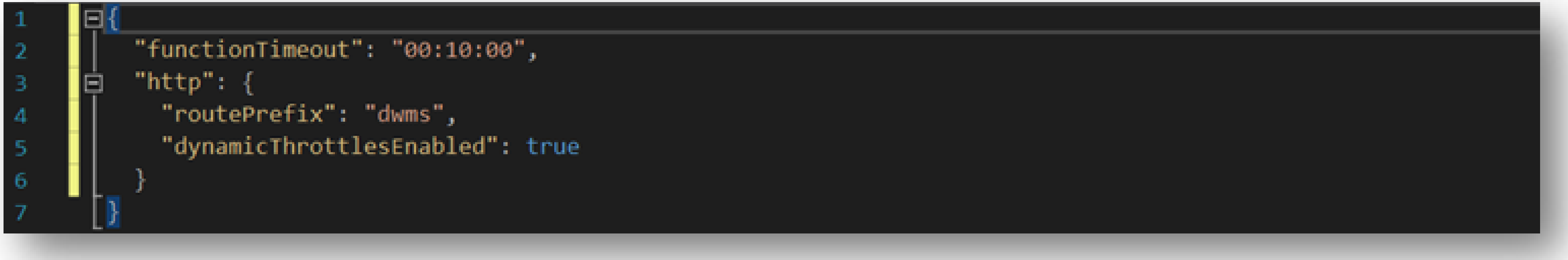
- Choice of language
- Pay-per-use pricing model
- Bring your own dependencies
- Integrated security
- Simplified integration
- Flexible development
- Open-source



Language	1.x
C#	GA
JavaScript	GA
F#	GA
Java	N/A
Python	Experimental
PHP	Experimental
TypeScript	Experimental
Batch (.cmd, .bat)	Experimental
Bash	Experimental
PowerShell	Experimental

Limitations

- Default time-out 5 minutes *
- Can be increased to 10 minutes



```
{
  "functionTimeout": "00:10:00",
  "http": {
    "routePrefix": "dwms",
    "dynamicThrottlesEnabled": true
  }
}
```

Azure Functions

{ Demo }

Durable Functions

Enables writing long-running,
stateful function orchestrations
in code in a serverless environment

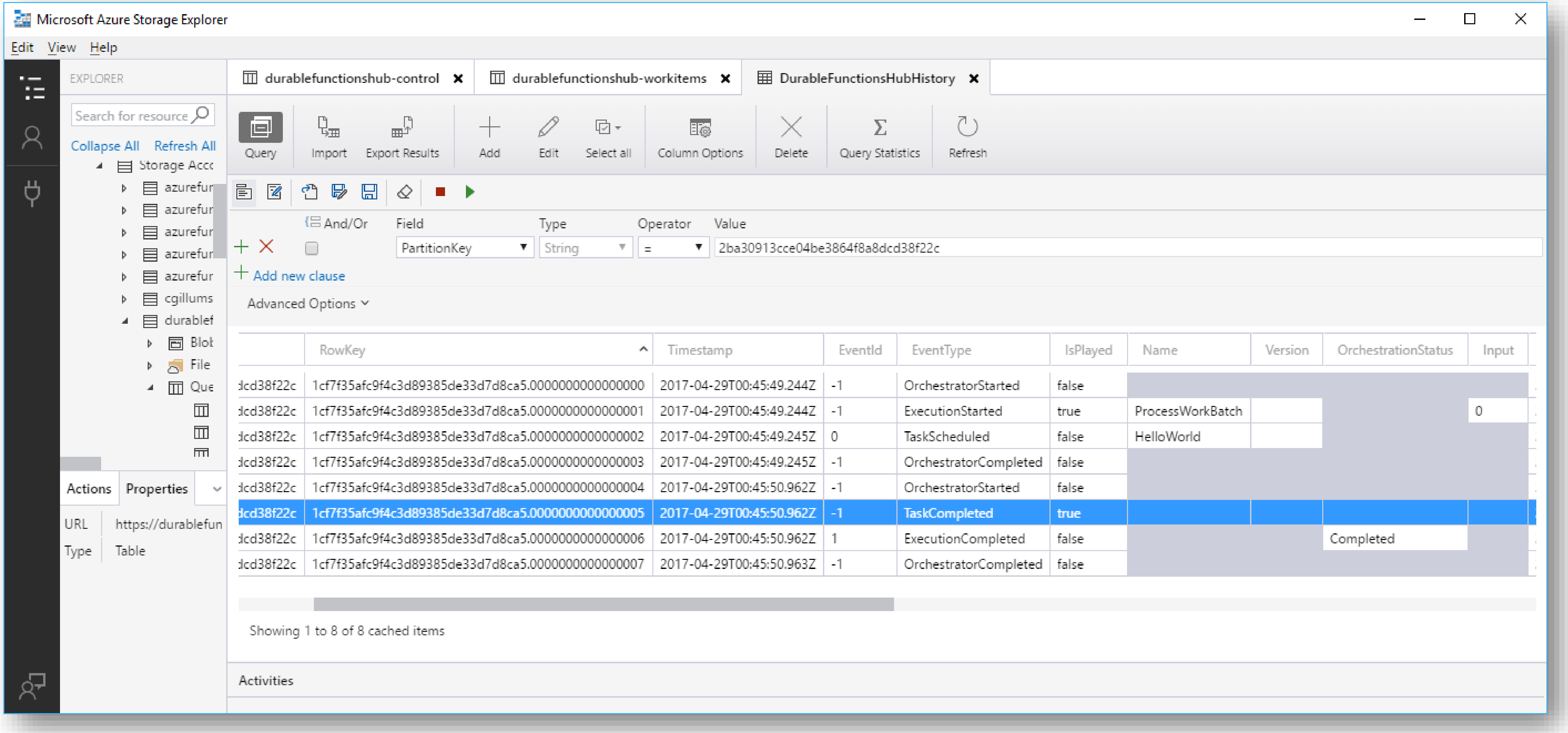
Behind the scenes

- Built on top of Durable Task Framework
- Maintain State: Event Sourcing
- Functions should be deterministic
- Duration timers: max 7 days

The process

- Await yields control back to dispatcher
- Dispatcher commits actions to storage (*execution history*)
- Adds messages to a queue to schedule the work
- Orchestrator can be unloaded
- Orchestrator wakes up and re-executes the entire function
- Check *execution history* for result:
  - Result found: replay result
  - No result found: do new work (or finish)

The process



URL https://durablefun

Type Table

And/Or	Field	Type	Operator	Value
□	PartitionKey	String	=	2ba30913cce04be3864f8a8dcd38f22c

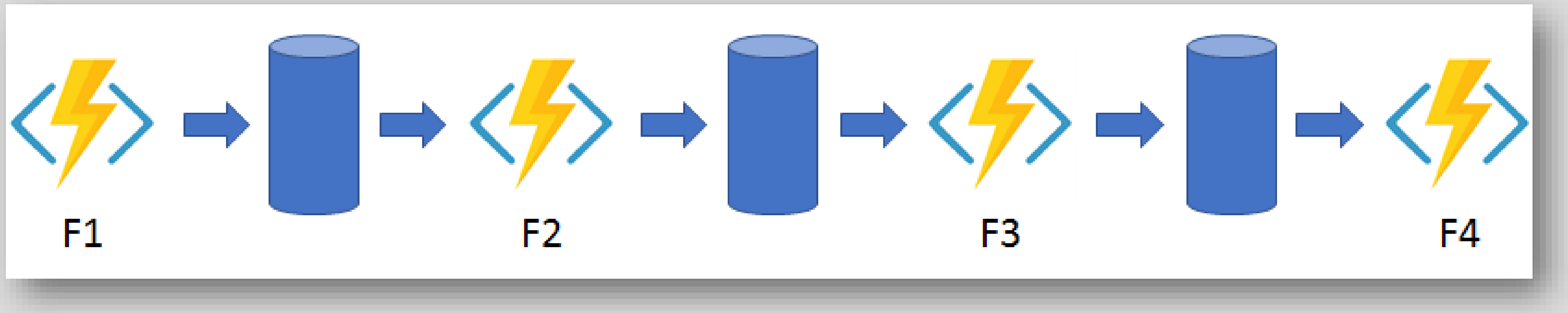
	RowKey	Timestamp	EventId	EventType	IsPlayed	Name	Version	OrchestrationStatus	Input
dcd38f22c	1cf7f35afc9f4c3d89385de33d7d8ca5.0000000000000000	2017-04-29T00:45:49.244Z	-1	OrchestratorStarted	false				
dcd38f22c	1cf7f35afc9f4c3d89385de33d7d8ca5.0000000000000001	2017-04-29T00:45:49.244Z	-1	ExecutionStarted	true	ProcessWorkBatch			0
dcd38f22c	1cf7f35afc9f4c3d89385de33d7d8ca5.0000000000000002	2017-04-29T00:45:49.245Z	0	TaskScheduled	false	HelloWorld			
dcd38f22c	1cf7f35afc9f4c3d89385de33d7d8ca5.0000000000000003	2017-04-29T00:45:49.245Z	-1	OrchestratorCompleted	false				
dcd38f22c	1cf7f35afc9f4c3d89385de33d7d8ca5.0000000000000004	2017-04-29T00:45:50.962Z	-1	OrchestratorStarted	false				
dcd38f22c	1cf7f35afc9f4c3d89385de33d7d8ca5.0000000000000005	2017-04-29T00:45:50.962Z	-1	TaskCompleted	true				
dcd38f22c	1cf7f35afc9f4c3d89385de33d7d8ca5.0000000000000006	2017-04-29T00:45:50.962Z	1	ExecutionCompleted	false			Completed	
dcd38f22c	1cf7f35afc9f4c3d89385de33d7d8ca5.0000000000000007	2017-04-29T00:45:50.963Z	-1	OrchestratorCompleted	false				

Showing 1 to 8 of 8 cached items

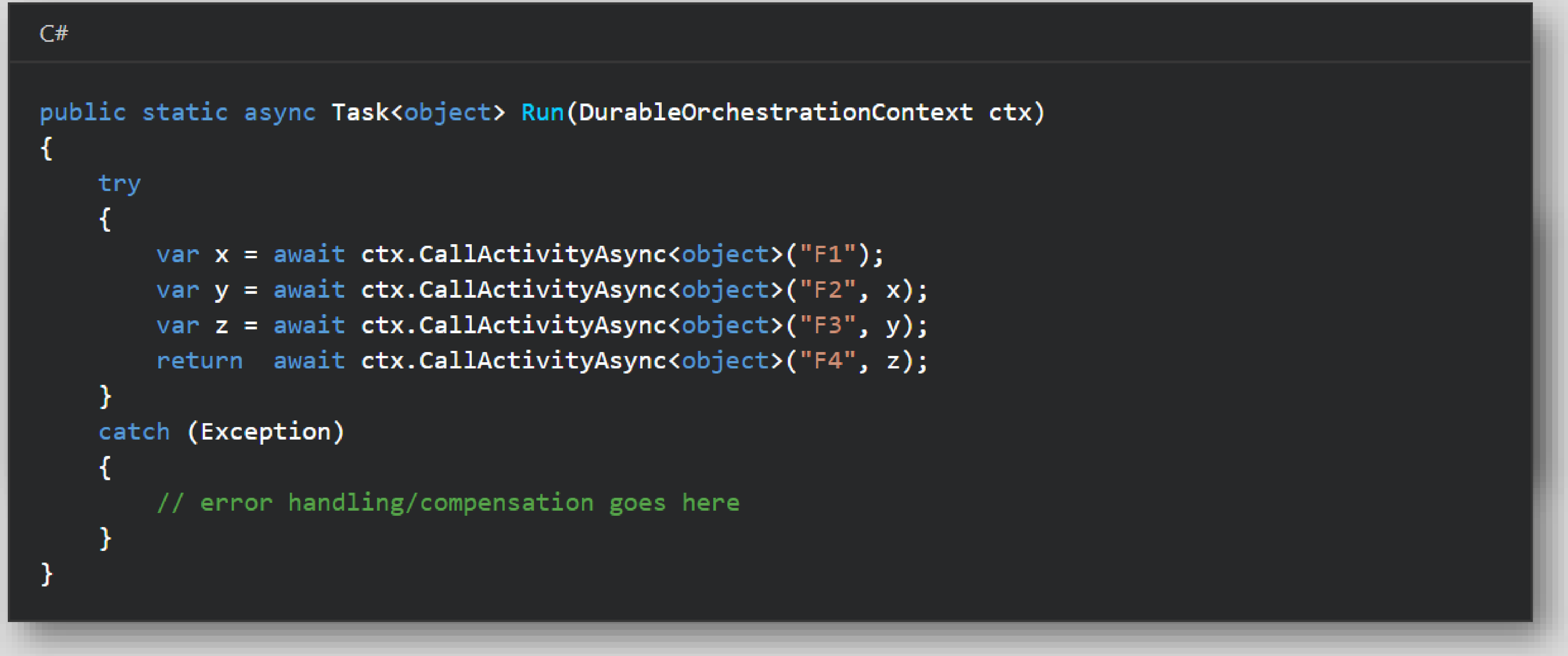
Patterns

- Function chaining
- Fan-out/fan-in
- Async HTTP APIs
- Monitoring
- Human interaction

Durable Functions



Durable Functions



C#

```
public static async Task<object> Run(DurableOrchestrationContext ctx)
{
    try
    {
        var x = await ctx.CallActivityAsync<object>("F1");
        var y = await ctx.CallActivityAsync<object>("F2", x);
        var z = await ctx.CallActivityAsync<object>("F3", y);
        return  await ctx.CallActivityAsync<object>("F4", z);
    }
    catch (Exception)
    {
        // error handling/compensation goes here
    }
}
```

Azure Functions 2.x

	1.x	2.x
Status	Generally Available (GA)	Preview
Development	Portal & Windows	Cross platform
Languages (GA)	C#, JavaScript, F#	-
Languages (experimental)	Python, PHP, TypeScript, Batch, Bash, PowerShell	-
Languages (preview)	-	C#, JavaScript, F#, Java
Bindings		New binding extensibility model

The Azure Content Delivery Network

Azure CDN

Azure CDN

- Designed to send static files faster and more reliably
- Using servers that are closest to the users
- Dramatically increases speed and availability
- Delivers significant user experience improvements

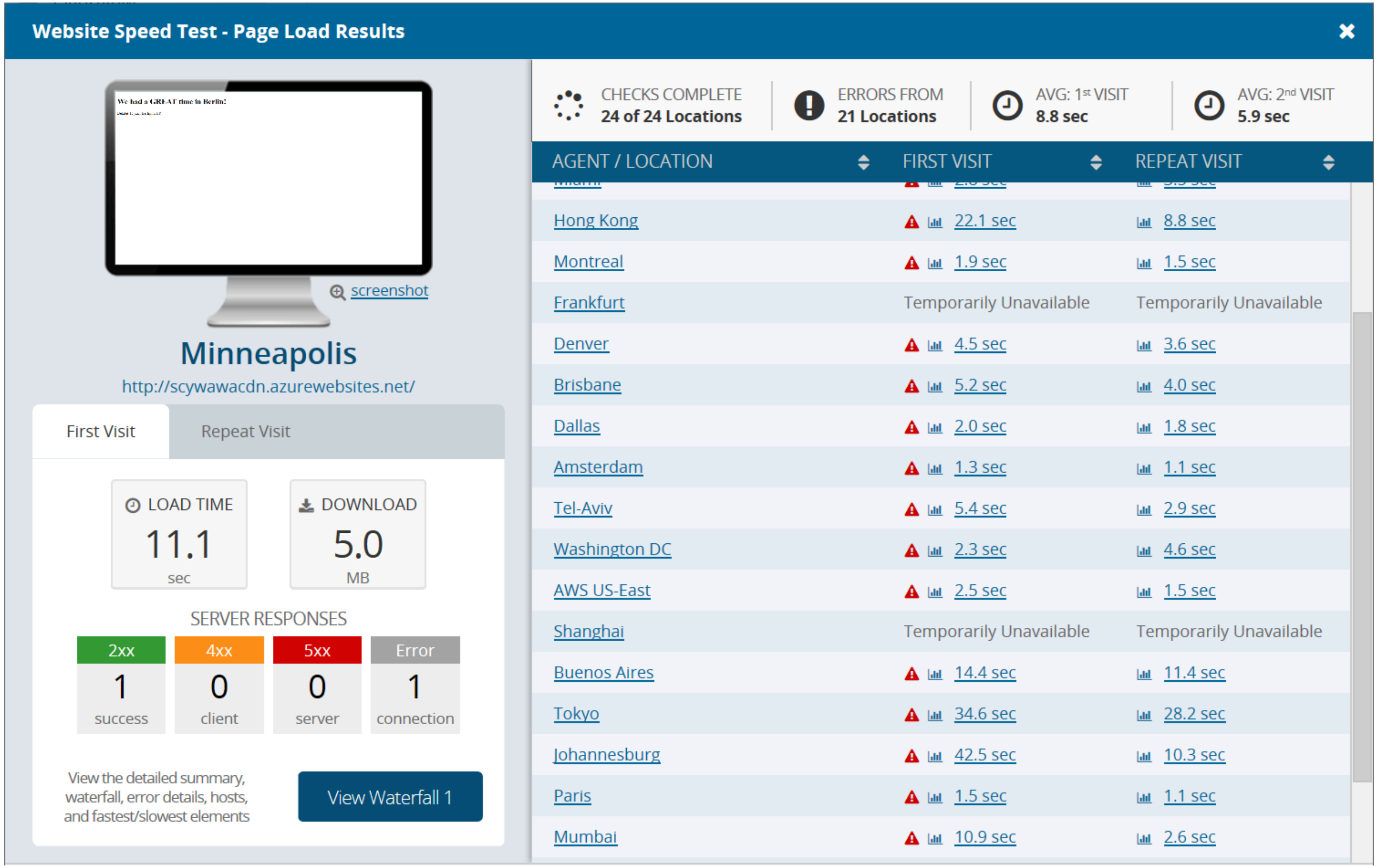
- Verizon
- Akamai
- Microsoft

Azure CDN Propagation

- It takes time for the registration to propagate
  - **Microsoft (Standard):** usually completes in 10 minutes
  - **Akamai (Standard):** usually completes within one minute
  - **Verizon (Standard):** usually completes within 90 minutes
  - **Verizon (Premium):** usually completes within 90 minutes

Azure CDN

{ Demo }



Website Speed Test - Page Load Results

screenshot

Minneapolis

http://scywawacdn.azurewebsites.net/

First Visit | Repeat Visit

LOAD TIME 11.1 sec

DOWNLOAD 5.0 MB

SERVER RESPONSES

2xx	4xx	5xx	Error
1 success	0 client	0 server	1 connection

View the detailed summary, waterfall, error details, hosts, and fastest/slowest elements

View Waterfall 1

CHECKS COMPLETE 24 of 24 Locations

ERRORS FROM 21 Locations

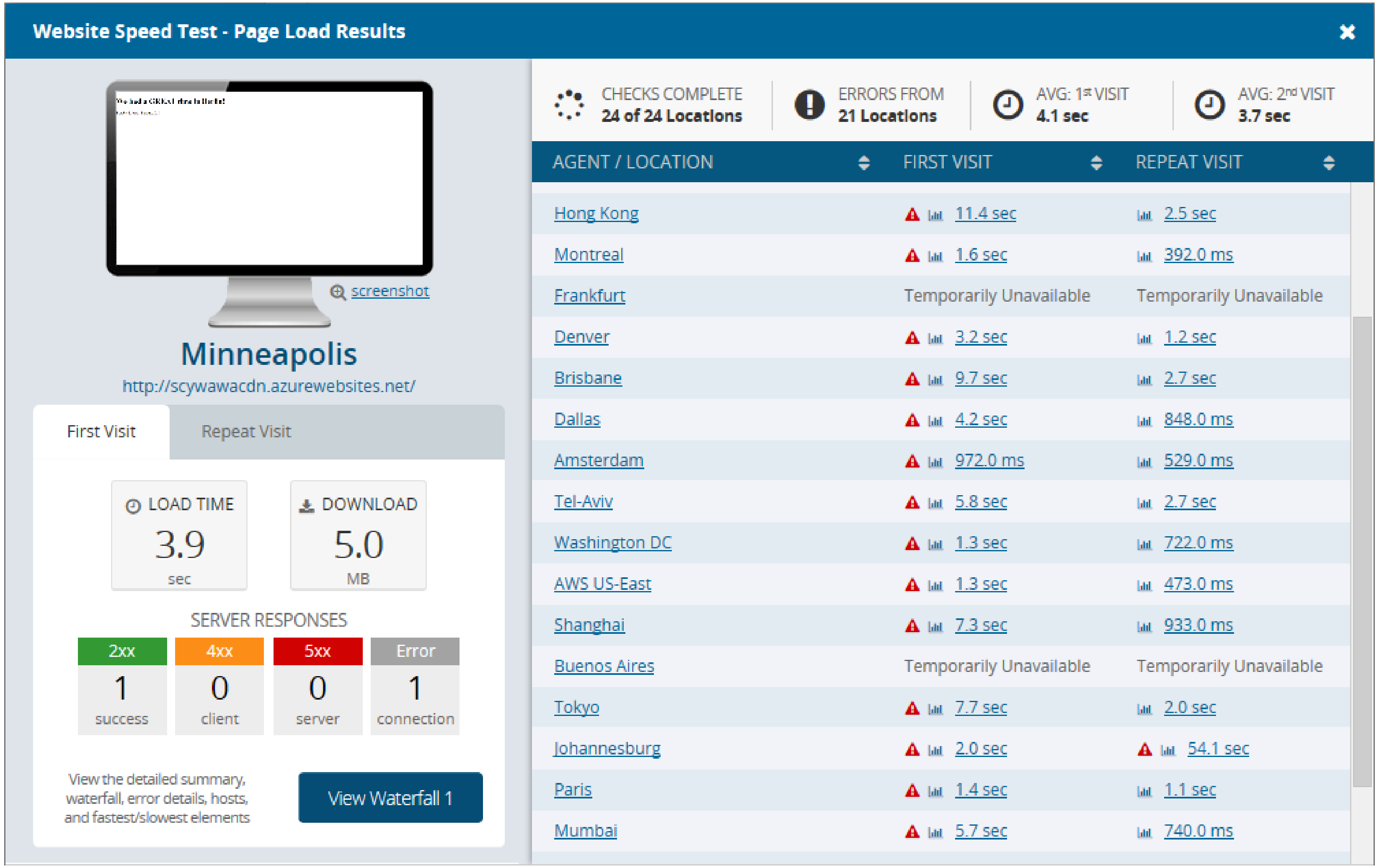
AVG: 1st VISIT 8.8 sec

AVG: 2nd VISIT 5.9 sec

AGENT / LOCATION	FIRST VISIT	REPEAT VISIT
Hong Kong	▲ 22.1 sec	8.8 sec
Montreal	▲ 1.9 sec	1.5 sec
Frankfurt	Temporarily Unavailable	Temporarily Unavailable
Denver	▲ 4.5 sec	3.6 sec
Brisbane	▲ 5.2 sec	4.0 sec
Dallas	▲ 2.0 sec	1.8 sec
Amsterdam	▲ 1.3 sec	1.1 sec
Tel-Aviv	▲ 5.4 sec	2.9 sec
Washington DC	▲ 2.3 sec	4.6 sec
AWS US-East	▲ 2.5 sec	1.5 sec
Shanghai	Temporarily Unavailable	Temporarily Unavailable
Buenos Aires	▲ 14.4 sec	11.4 sec
Tokyo	▲ 34.6 sec	28.2 sec
Johannesburg	▲ 42.5 sec	10.3 sec
Paris	▲ 1.5 sec	1.1 sec
Mumbai	▲ 10.9 sec	2.6 sec

Test results

Without CDN



Website Speed Test - Page Load Results

Minneapolis

http://scywawacdn.azurewebsites.net/

First Visit | Repeat Visit

LOAD TIME 3.9 sec

DOWNLOAD 5.0 MB

SERVER RESPONSES

2xx	4xx	5xx	Error
1	0	0	1
success	client	server	connection

View the detailed summary, waterfall, error details, hosts, and fastest/slowest elements

CHECKS COMPLETE 24 of 24 Locations

ERRORS FROM 21 Locations

AVG: 1st VISIT 4.1 sec

AVG: 2nd VISIT 3.7 sec

AGENT / LOCATION	FIRST VISIT	REPEAT VISIT
Hong Kong	▲ 11.4 sec	2.5 sec
Montreal	▲ 1.6 sec	392.0 ms
Frankfurt	Temporarily Unavailable	Temporarily Unavailable
Denver	▲ 3.2 sec	1.2 sec
Brisbane	▲ 9.7 sec	2.7 sec
Dallas	▲ 4.2 sec	848.0 ms
Amsterdam	▲ 972.0 ms	529.0 ms
Tel-Aviv	▲ 5.8 sec	2.7 sec
Washington DC	▲ 1.3 sec	722.0 ms
AWS US-East	▲ 1.3 sec	473.0 ms
Shanghai	▲ 7.3 sec	933.0 ms
Buenos Aires	Temporarily Unavailable	Temporarily Unavailable
Tokyo	▲ 7.7 sec	2.0 sec
Johannesburg	▲ 2.0 sec	▲ 54.1 sec
Paris	▲ 1.4 sec	1.1 sec
Mumbai	▲ 5.7 sec	740.0 ms

Test results

With CDN

ASP.NET Core 2.1 SignalR

`ASP.NET Core SignalR is a library that simplifies adding real-time web functionality to apps. Real-time web functionality enables server-side code to push content to clients instantly.`

ASP.NET Core 2.1 SignalR

- Server to client push: Global, Groups & Individual
- Stream Results via "Channel" Class
- TS/JS & .NET clients
- Integration with ASP.Net core
  - Dependency Injection
  - Routing
  - Auth
- No more jQuery dependency ;)

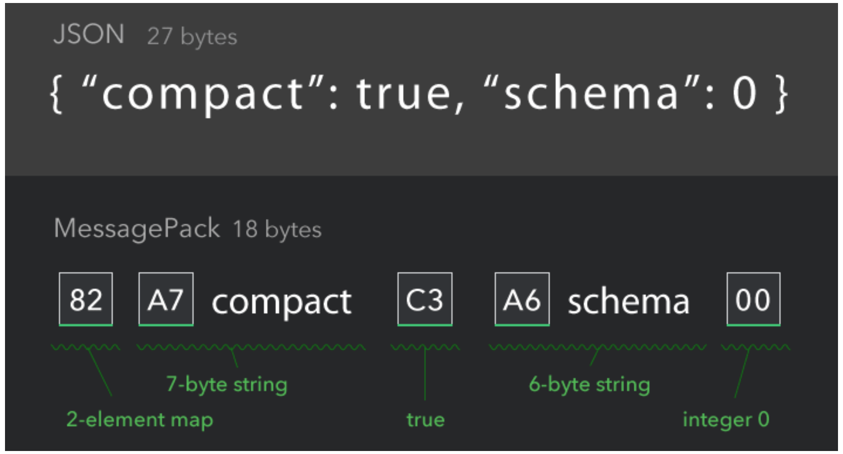
ASP.NET Core 2.1 SignalR

- Handles connection management automatically.
- Enables broadcasting messages to all connected clients simultaneously. For example, a chat room.
- Enables sending messages to specific clients or groups of clients.
- Is open-sourced at 'MS-' GitHub.

ASP.NET Core 2.1 SignalR

- Transport, auto detect
  - WebSockets
  - Server Send Events
  - Long Polling

- Protocols
  - Text based on JSON
  - Binary based on MessagePack



JSON 27 bytes

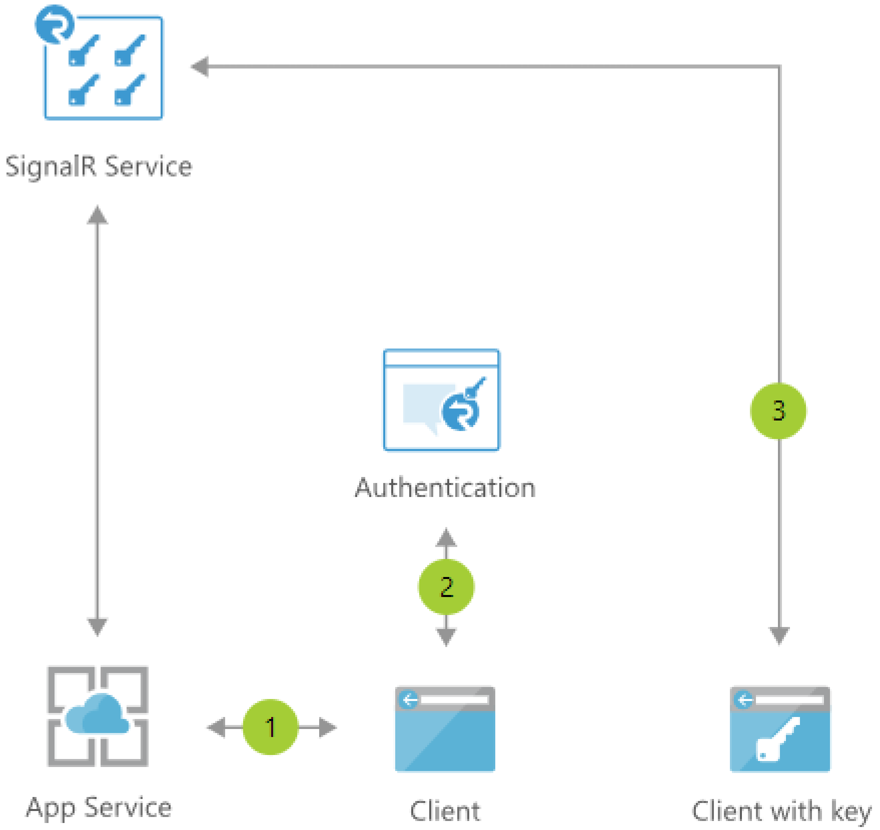
```
{ “compact”: true, “schema”: 0 }
```

MessagePack 18 bytes

Azure SignalR Service

`Because SignalR Service is a fully managed service, you can roll it out in a multiserver environment without worrying about hosting, scalability, load balancing, or authentication.`

Azure SignalR Service



ASP.NET Core 2.1 SignalR
Azure SignalR Service

{ Demo }

ASP.NET Core 2.1 SignalR

ASP.Net Core 2.1 App

1. Create solution
2. services.AddSignalR().AddAzureSignalR();
3. app.UseFileServer(); //for hosting angular in wwwroot
4. app.UseAzureSignalR(routes=> { routes.MapHub<HubClass:Hub>("/routeToHub"); });
5. Create HubClass: Hub
6. Add Methods to HubClass

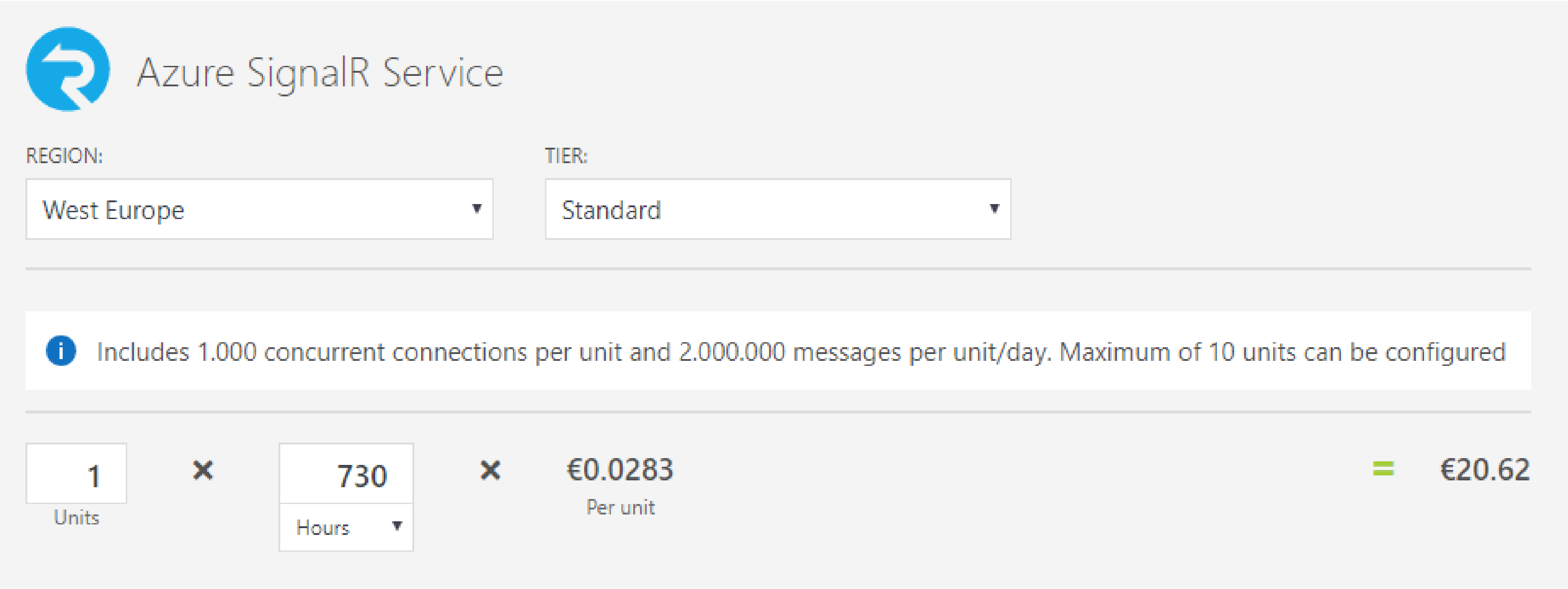
Angular app

1. Create app with cli: "ng new appname"
2. npm install @aspnet/signalr
3. let hubConn = new HubConnectionBuilder().withUrl("/routeToHub").build();
4. hubConn.start();
5. hubConn.on("sendmessage", (person: string, message: string) => { });
6. hubConn.send("methodInHub", "param1", "p2"…);

Azure SignalR Service - Preview

Current regions: West Europe, Southeast Azia, East US, West US

Dev/Test: Free



Azure SignalR Service

REGION: West Europe

TIER: Standard

Includes 1.000 concurrent connections per unit and 2.000.000 messages per unit/day. Maximum of 10 units can be configured

1 Units × 730 Hours × €0.0283 Per unit = €20.62

Azure SignalR Service + Azure Functions

Supported scenarios

- Allow clients to serverlessly connect to a SignalR Service hub without requiring an ASP.NET Core backend
- Use Azure Functions (any language supported by V2) to broadcast messages to all clients connected to a SignalR Service hub
- Example scenarios include: broadcast messages to a SignalR Service hub on HTTP requests and events from Cosmos DB change feed, Event Hub, Event Grid, etc

Current limitations

- Only supports broadcasting at this time, cannot invoke methods on a subset of connections, users, or groups
- Functions cannot be triggered by client invocation of server methods (clients need to call an HTTP endpoint or post messages to an Event Grid, etc, to trigger a function)

GitHub: anthonychu/**AzureAdvocates.WebJobs.Extensions.SignalRService**

Azure SignalR Service + Azure Functions

{ Demo }

Azure Media Services

Azure Media Services

- Supports most popular screens and devices
- Automatically chooses the best playback format
- Easily integrates into web and app solutions
- Lets you use familiar JavaScript API development
- Gives you integrated content protection

- Integrates with CDN

Azure Media Services

{ Demo }

TIP

- Kraken.io
- Image Optimizer (VS Tool, Mads Kristensen)

Resources

- https://github.com/oscarvantol/scywawa
- https://github.com/anthonychu/AzureAdvocates.WebJobs.Extensions.SignalRService

- http://azureinteractives.azurewebsites.net/CloudDesignPatterns/