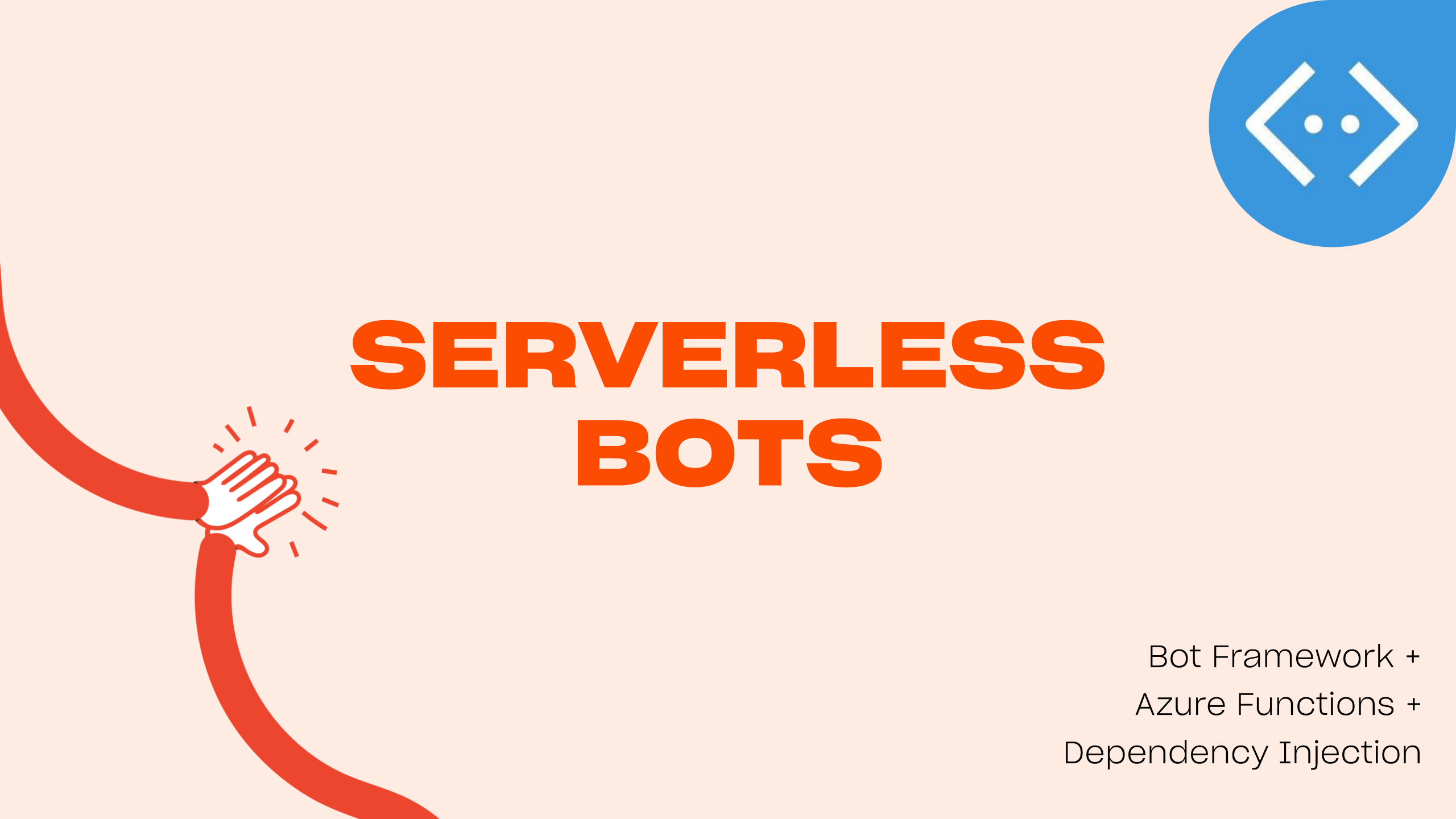


SERVERLESS BOTS

Bot Framework +
Azure Functions +
Dependency Injection

WHO?

ENZO CANO

@enz_cano

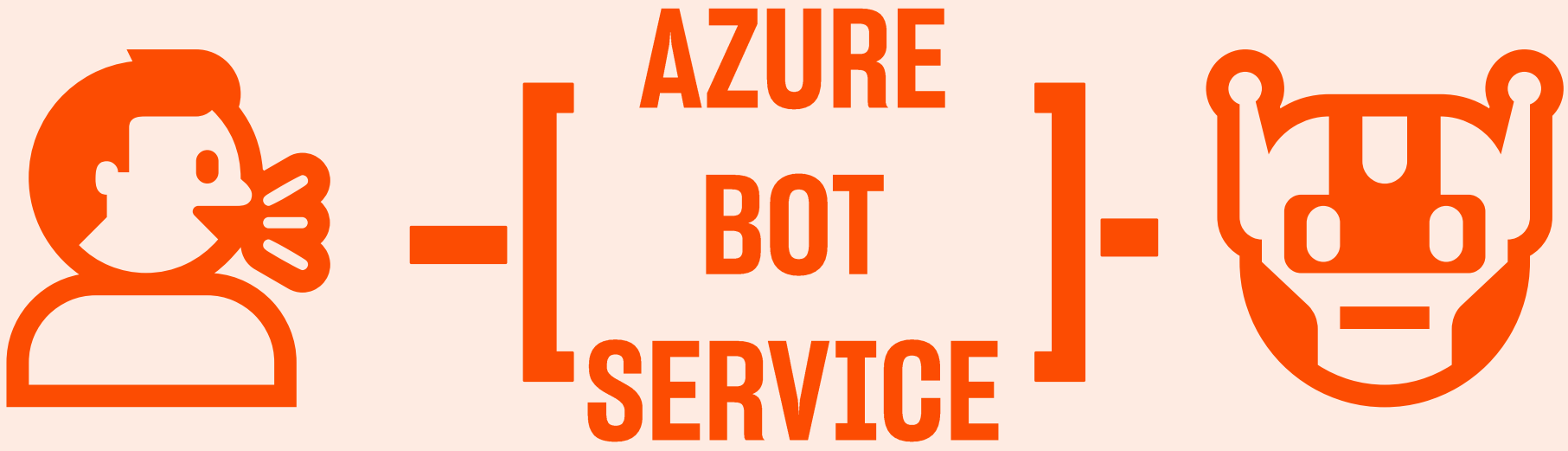
Lead Software Engineer - SOUTHWORKS

Learner on demand. Curious by default.

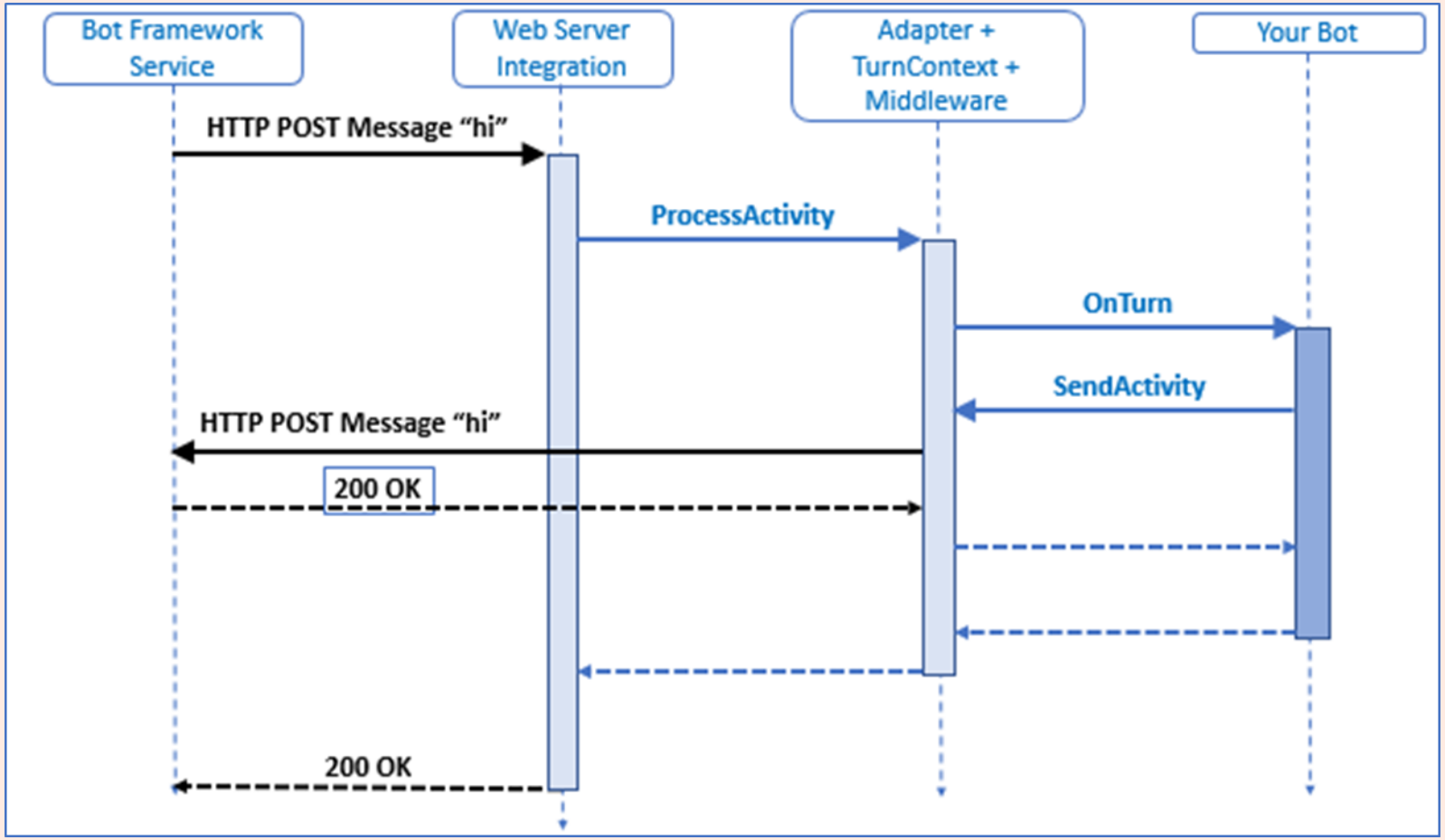
WHAT?

Technical Background!
How bots work.
How functions work.
Situation!
Implementation.
DEMO time!
?.

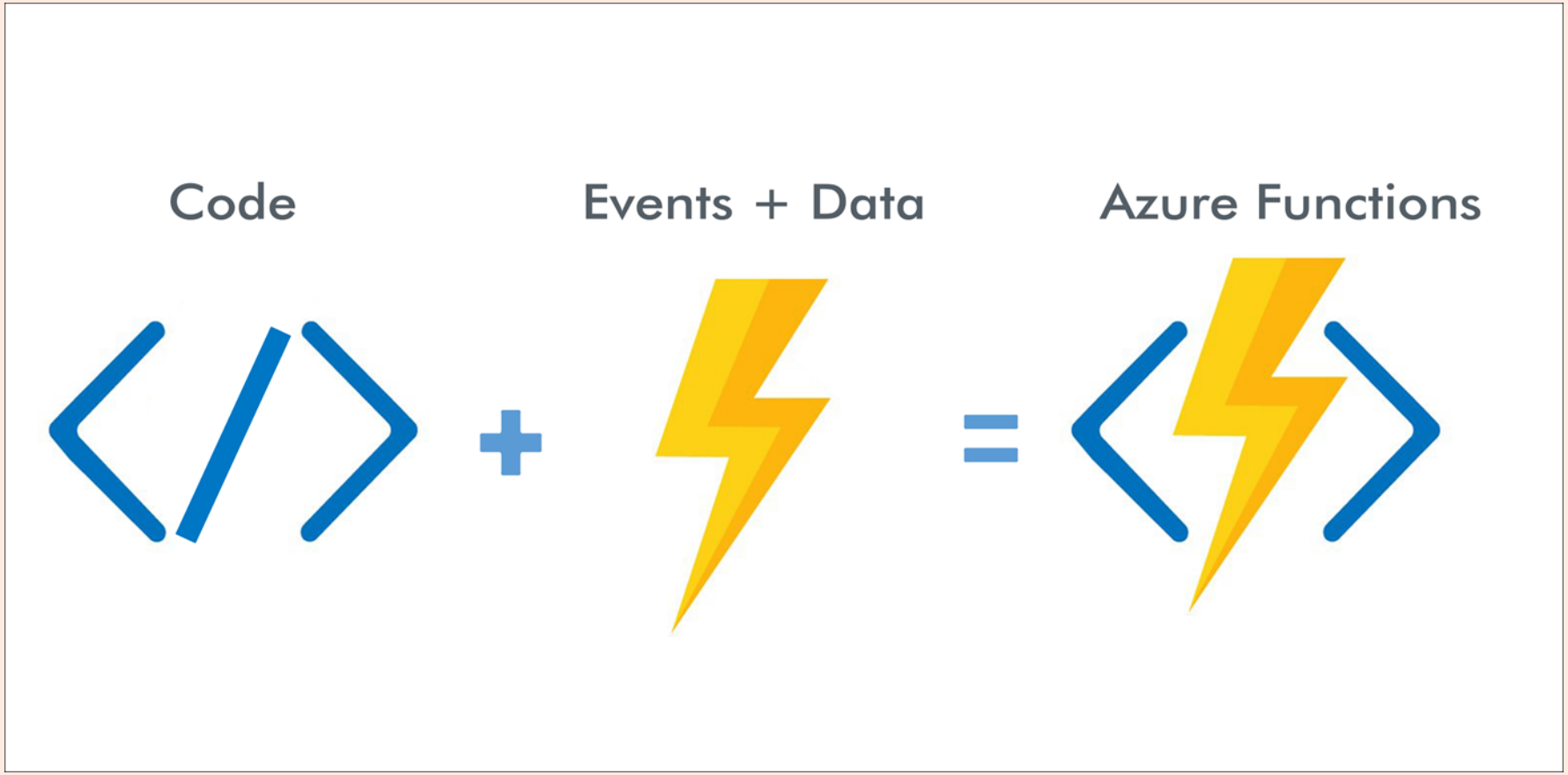
Bot architecture – 101



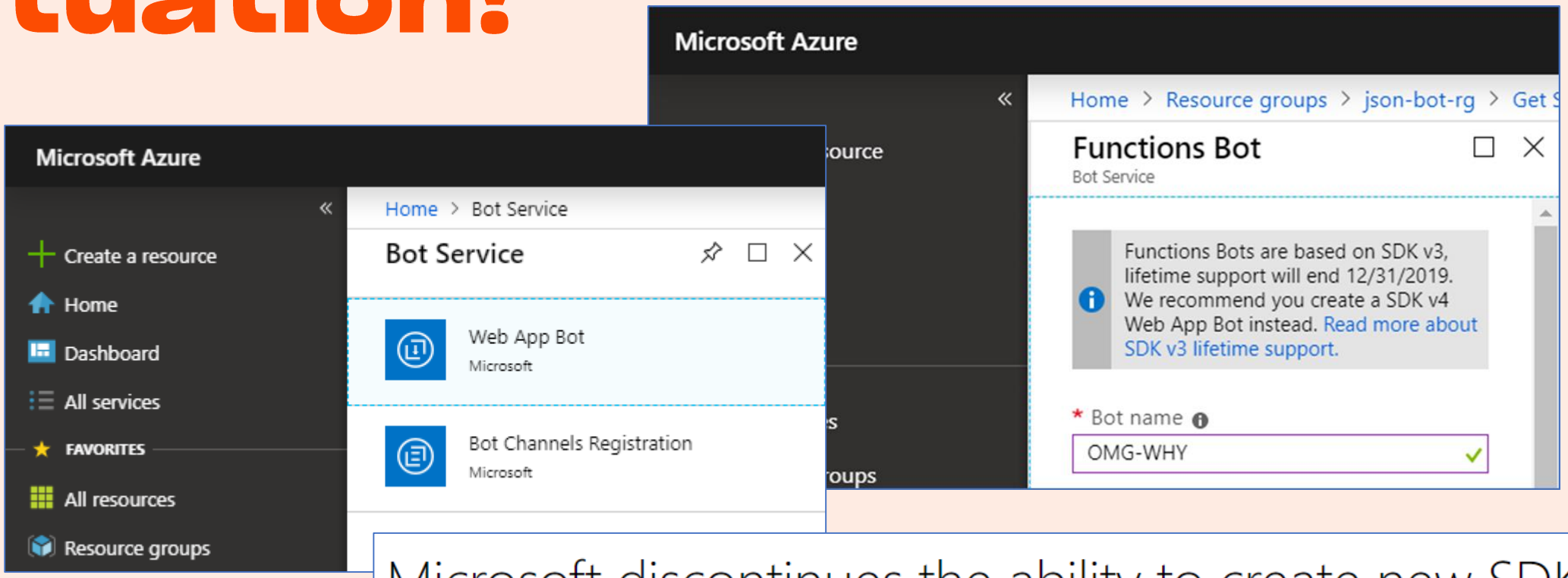
How bots work



How functions work



Situation!

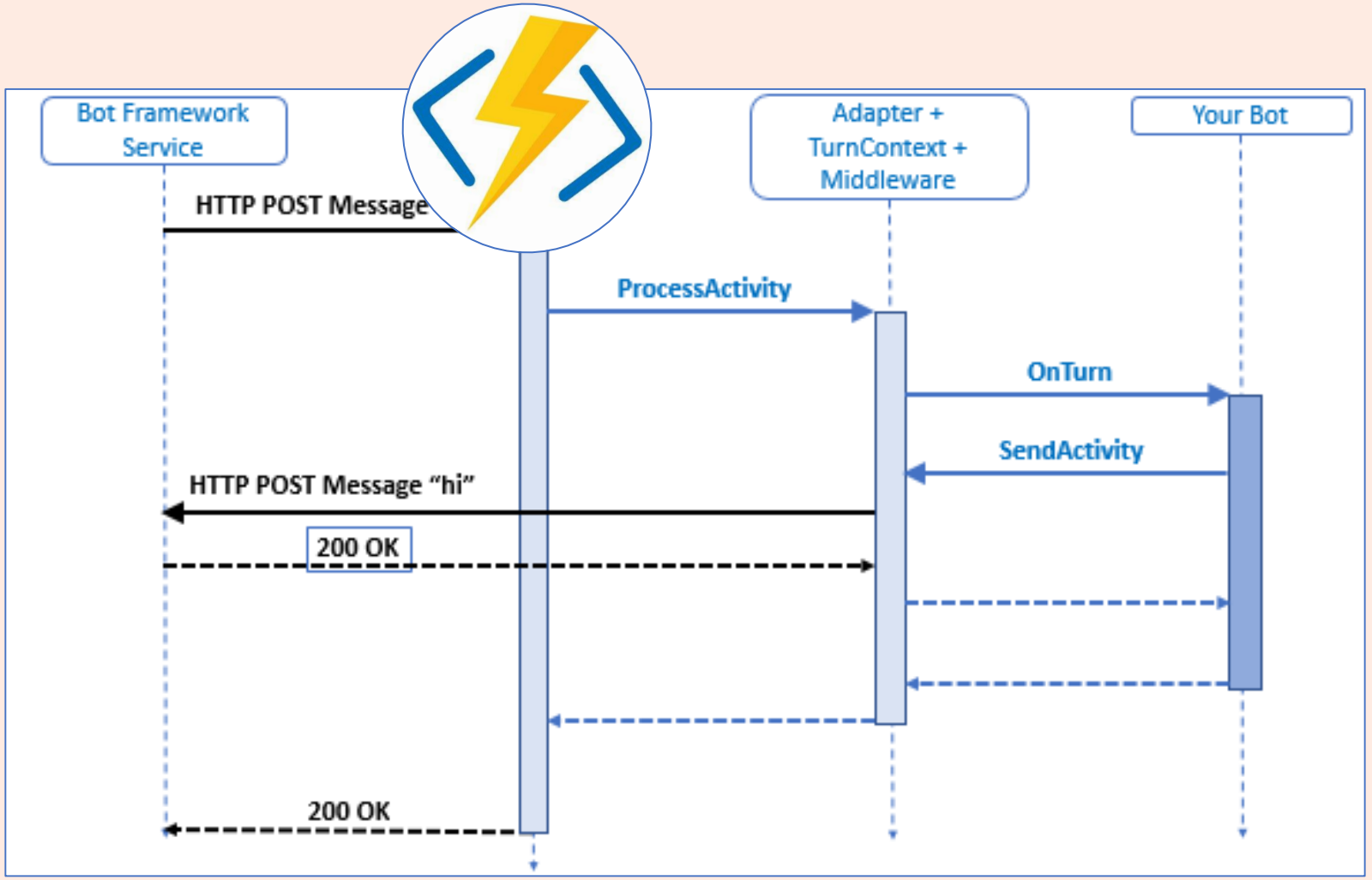


Microsoft discontinues the ability to create new SDK V3 Bots. Deployed V3 bots will continue to run without interruption.

Posted on June 7, 2019 by coryh

Share this post

How bots should work



Bot + Function App

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

        var appId = Environment.GetEnvironmentVariable("MicrosoftAppId", EnvironmentVariableTarget.Process);
        var appPassword = Environment.GetEnvironmentVariable("MicrosoftAppPassword", EnvironmentVariableTarget.Process);

        var credentialProvider = new SimpleCredentialProvider(appId, appPassword);
        var adapter = new BotFrameworkHttpAdapter(credentialProvider);
        var bot = new Bot();
        var res = new DefaultHttpResponse(req.HttpContext);
        await adapter.ProcessAsync(req, res, bot);

        return new StatusCodeResult(res.StatusCode);
    }
}
```

Bot + FA + Custom Adapter

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

        var appId = Environment.GetEnvironmentVariable("MicrosoftAppId", EnvironmentVariableTarget.Process);
        var appPassword = Environment.GetEnvironmentVariable("MicrosoftAppPassword", EnvironmentVariableTarget.Process);

        var credentialProvider = new SimpleCredentialProvider(appId, appPassword);
        var adapter = new BotFrameworkFunctionsAdapter(credentialProvider);
        var bot = new Bot();
        return await adapter.ProcessAsync(req, bot);
    }
}
```

Bot + FA + CA + DI

```
[assembly: FunctionsStartup(typeof(BotFunctionAppWithAdapterAndDI.Startup))]

namespace BotFunctionAppWithAdapterAndDI
{
    1 reference
    public class Startup : FunctionsStartup
    {
        0 references
        public override void Configure(IFunctionsHostBuilder builder)
        {
            var s = builder.Services;
            s.AddSingleton<ICredentialProvider, ConfigurationCredentialProvider>();
            s.AddSingleton<IBotFrameworkFunctionsAdapter, BotFrameworkFunctionsAdapter>();
            s.AddSingleton<IBot, Bot>();
            s.AddBotApplicationInsights();
        }
    }
}
```

```
0 references
public class Function
{
    2 references
    private readonly IBotFrameworkFunctionsAdapter adapter;
    2 references
    private readonly IBot bot;

    0 references
    public Function(IBotFrameworkFunctionsAdapter adapter, IBot bot)
    {
        this.adapter = adapter ?? throw new ArgumentNullException(nameof(adapter));
        this.bot = bot ?? throw new ArgumentNullException(nameof(bot));
    }

    [FunctionName("messages")]
    0 references
    public Task<IActionResult> Run(
        [HttpTrigger(AuthorizationLevel.Function, "post")] HttpRequest req)
    {
        return adapter.ProcessAsync(req, bot);
    }
}
```

DEMO time!

Questions?

WHO?

ENZO CANO

@enz_cano

Lead Software Engineer - SOUTHWORKS

Learner on demand. Curious by default.