GETTING STARTED WITH AZURE FUNCTIONS

Joe Plumb, Data and AI CSA

Agenda

- What are Azure Functions?
- How do they work?
- Getting Started Demo

So what are Azure Functions ?

WARNING

MARKETING CONTENT WITHIN

✨✨ MAGIC ✨✨

What is serverless?

Full abstraction of servers
Developers can just focus on their code—there are no distractions around server management, capacity planning, or availability.

Instant, event-driven scalability
Application components react to events and triggers in near real-time with virtually unlimited scalability; compute resources are used as needed.

Pay-per-use
Only pay for what you use: billing is typically calculated on the number of function calls, code execution time, and memory used.*

*Supporting services, like storage and networking, may be charged separately.

What are the benefits?

Focus

Solve business problems—not technology problems related to undifferentiated heavy lifting

Efficiency

Shorter time to market
Fixed costs converted to variable costs
Better service stability
Better development and testing management
Less waste

Flexibility

Simplified starting experience
Easier pivoting means more flexibility
Easier experimentation
Scale at your pace—don't bet the farm on Day 1
Natural fit for microservices

Full integration with Azure ecosystem

Functions is the center piece of the Serverless platform

Development

- IDE support
- Integrated DevOps
- Local development
- Monitoring
- Visual debug history

Platform

Event Grid

Manage all events that can trigger code or logic

Functions

Execute your code based on events you specify

Logic Apps

Design workflows and orchestrate processes

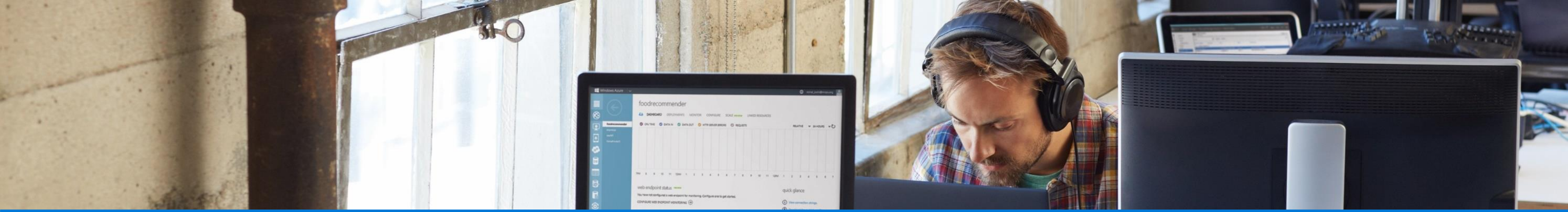
Database | Storage | Analytics | Intelligence | Security | IoT

So how do they work?

WARNING

MARKETING CONTENT WITHIN

(Come on they weren’t that bad)



Working with Azure Functions

Triggers

Use triggers to define how functions are invoked
Avoid hardcoding with preconfigured JSON files
Build serverless APIs using HTTP triggers

Bindings

Connect to data with input and output bindings
Bind to Azure solutions and third-party services
Use HTTP bindings in tandem with HTTP triggers

Proxies

Define one API surface for multiple function apps
Create endpoints as reverse proxies to other APIs
Condition proxies to use variables

Local debugging

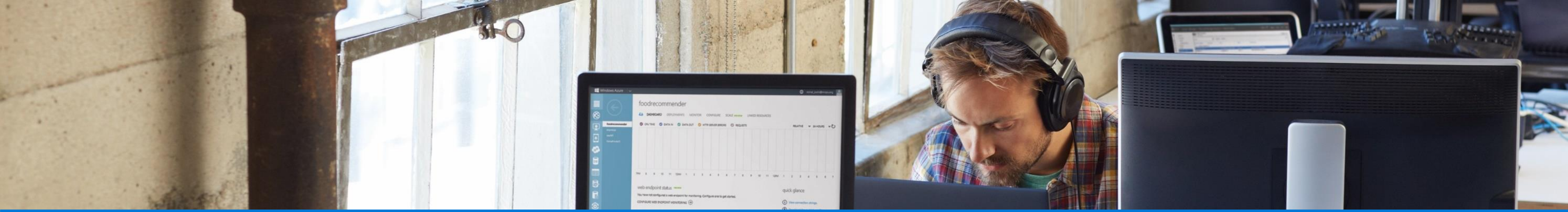
Debug C# and JavaScript functions locally
Use debugging tools in Azure portal, VS, and VS Code

CI/CD

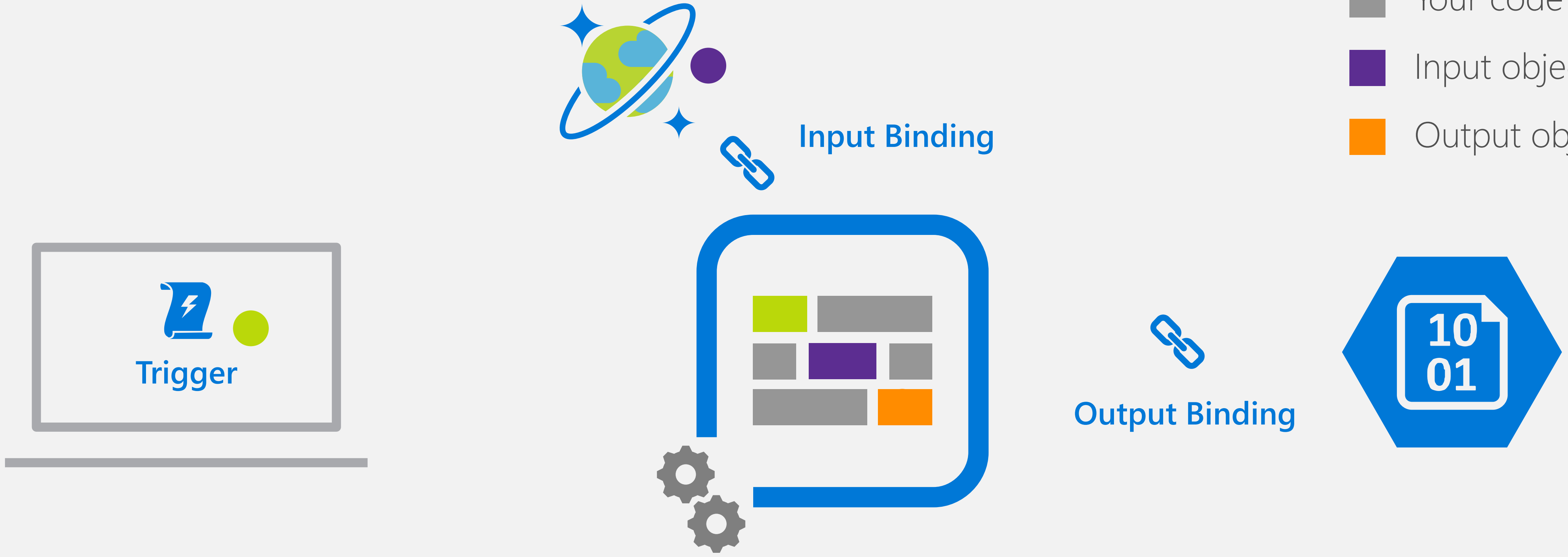
Save time with built-in DevOps
Deploy functions using App Service for CI
Leverage Microsoft, partner services for CD

Monitoring

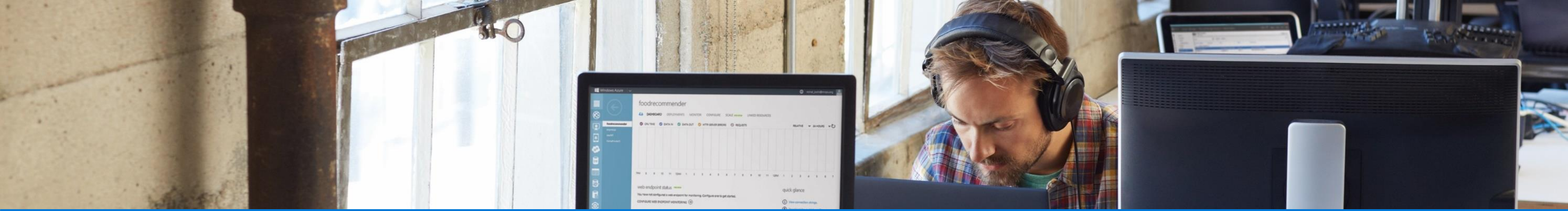
Integrate with Azure Application Insights
Get near real-time details about function apps
See metrics around failures, executions, etc.



Working with Azure Functions



🟩	Trigger object
⬜	Your code
🟪	Input object
🟧	Output object



Things you'll need

Visual Studio Code with Azure Functions Extension

Azure Functions Core Tools (for local emulation and deployment)

Optional: Azure CLI (for deployment of services)

DEMONSTRATION

Key takeaways

- Triggers make things happen
- Bindings define the inputs and outputs
- Az cli is great
- Azure functions – key files:
    - **local.settings.json** for your secrets
    - **function.json** to define triggers and bindings
    - **__init__.py** for your Python code (other languages are available)
- F5 for debug in VS Code
- Deploy straight to a Function app using the Extension

QUESTIONS?

THANKS!

Joe Plumb, Data and AI CSA