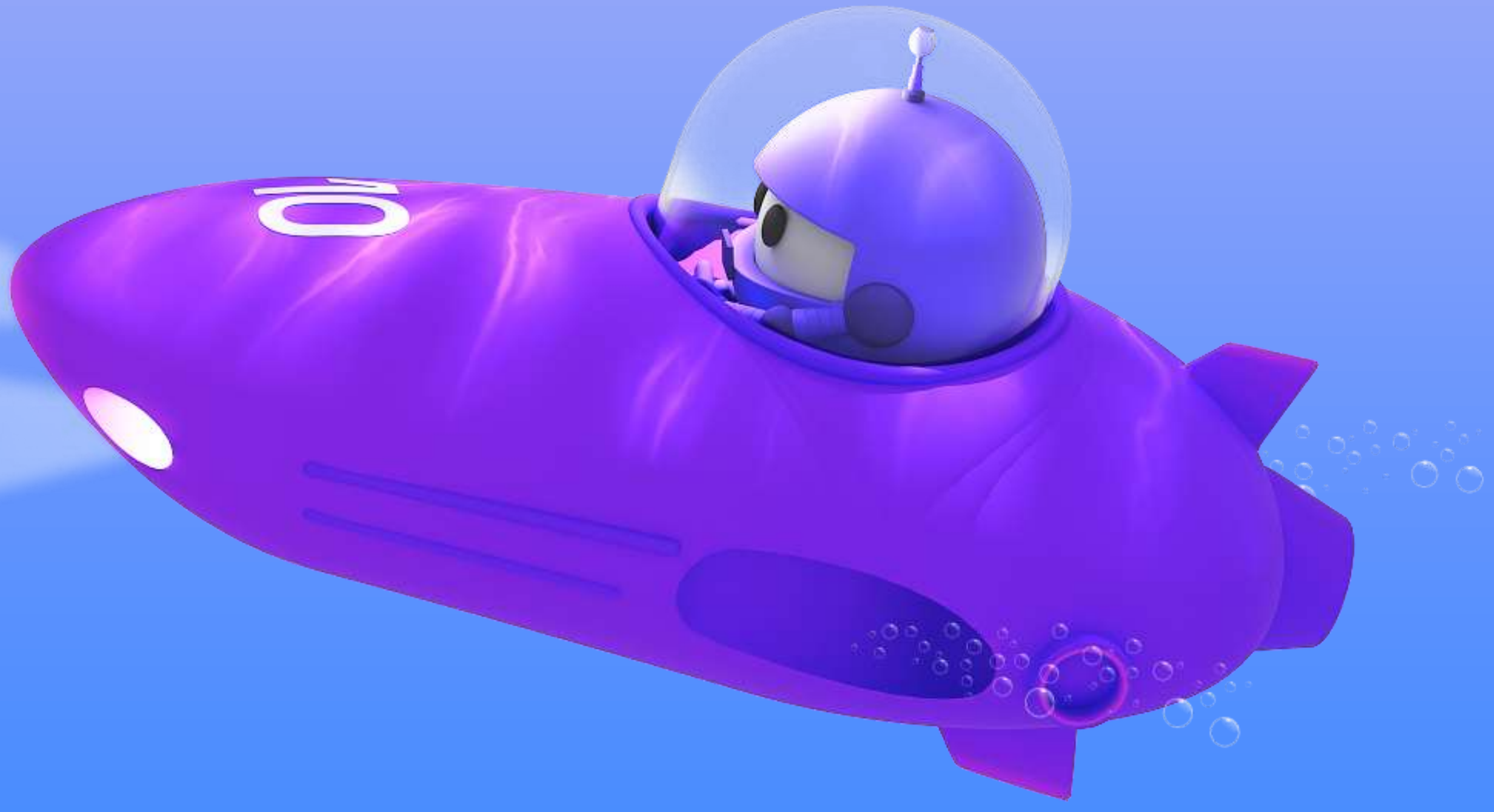
.NET 💖 Artificial Intelligence

.NET



Rodrigo Liberoff

Azure Fundamentals
Azure AI Fundamentals
Power Platform Fundamentals

Azure AI Engineer Associate
Azure Administrator Associate
Azure Developer Associate

Azure & GitHub DevOps Engineer Expert
Azure Solutions Architect Expert

¿Quién soy?

Soy Senior Principal Architect for AI Platform Engineering, y uno de los principales referentes en Inteligencia Artificial generativa en ENCAMINA, con una hiper-especialización en tecnologías Microsoft.

¡Soy un veterano de las tecnologías Microsoft, con más de 25 años de experiencia!

Graduado de Ingeniero en Ciencias de la Computación
Universidad Simón Bolívar - Venezuela

@rliberoff

https://github.com/rliberoff

https://www.linkedin.com/in/rliberoff/

Azurebrains

.NET AI Libraries

Microsoft.Extensions.AI and Microsoft.Extensions.VectorData

Streamline AI integration with our unified APIs

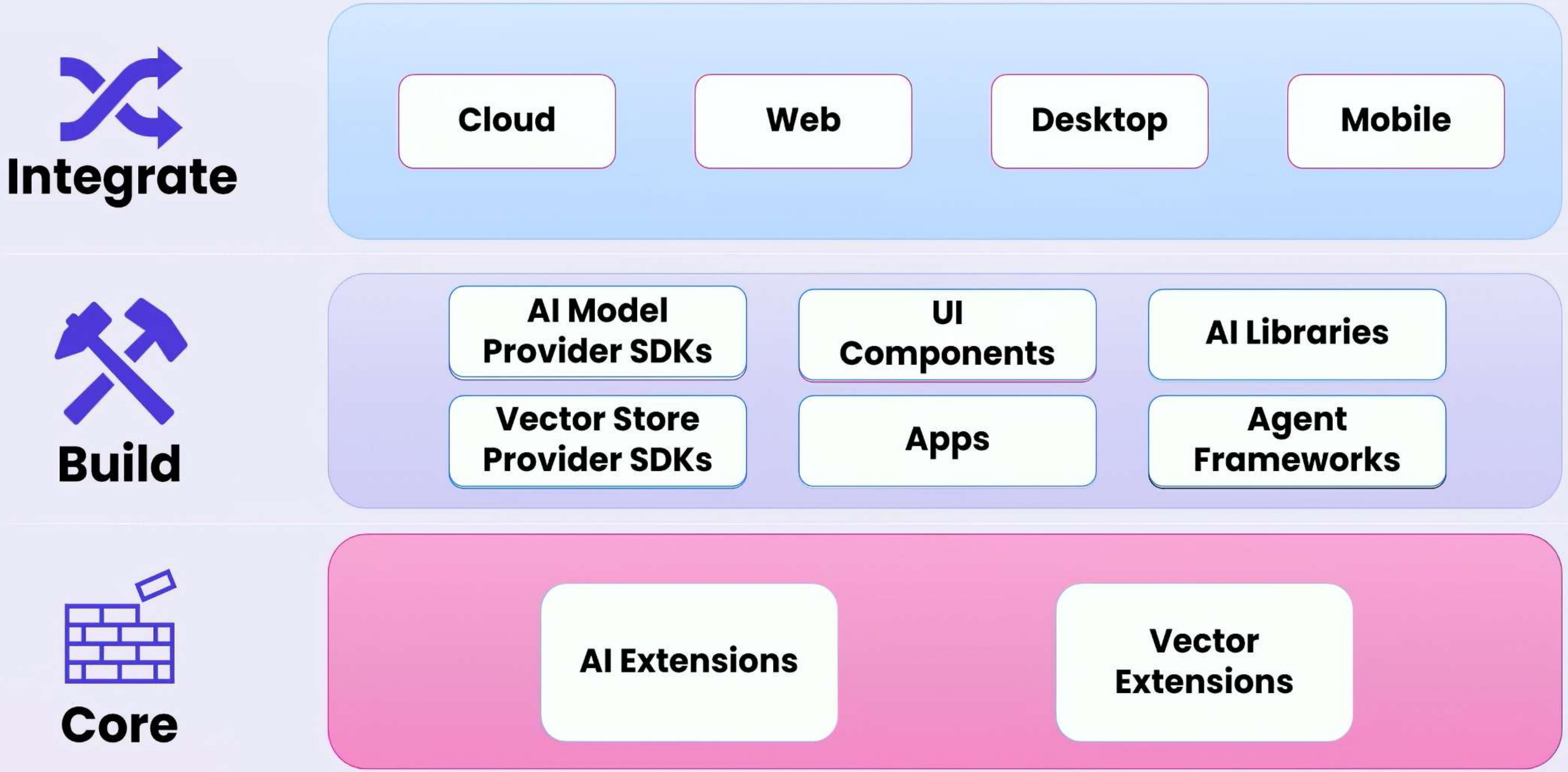
Common AI abstractions

Standard middleware

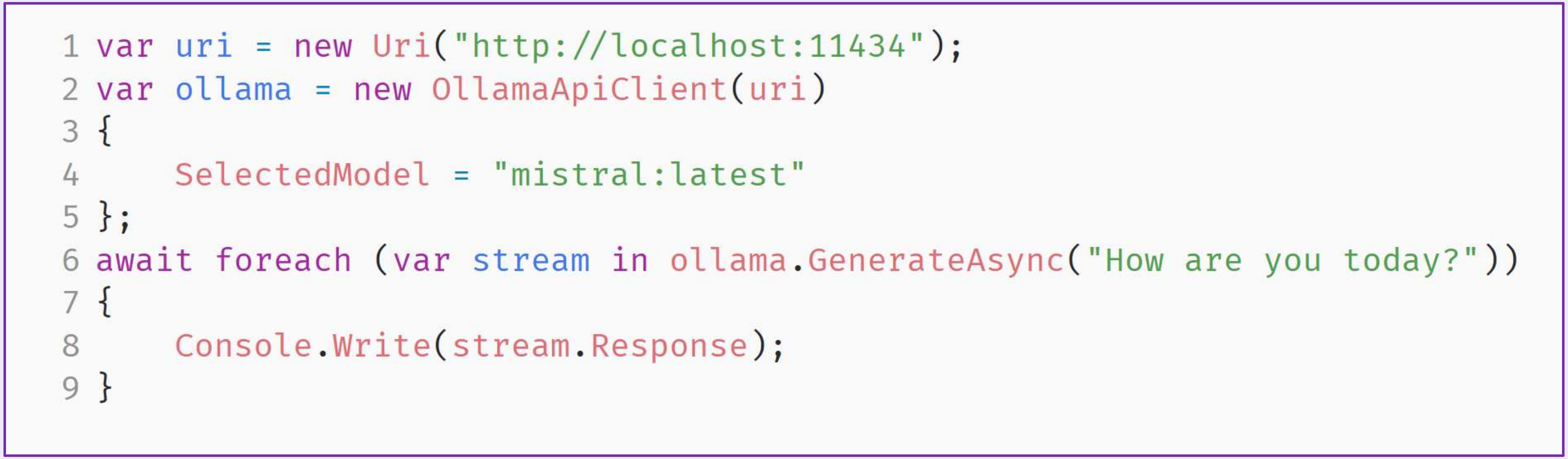
Vector store operations and abstractions

Interoperability and extensibility

AI and Vector Data
extensions



Una API unificada para múltiples proveedores

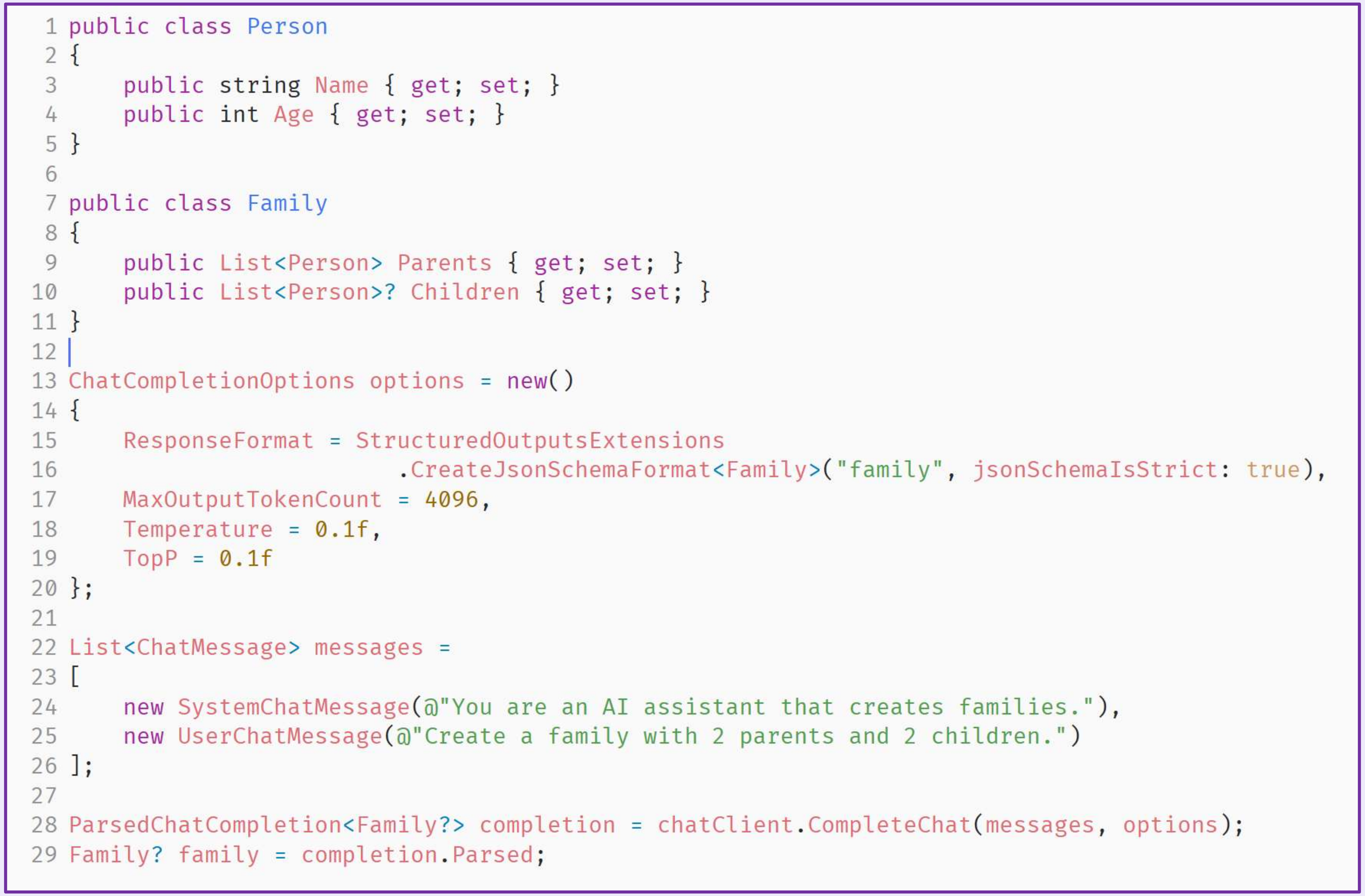


```csharp
var uri = new Uri("http://localhost:11434");
var ollama = new OllamaApiClient(uri)
{
    SelectedModel = "mistral:latest"
};
await foreach (var stream in ollama.GenerateAsync("How are you today?"))
{
    Console.Write(stream.Response);
}
```

Una API unificada para múltiples proveedores

```csharp
OpenAIResponseClient client = new(@"o3-mini",
                                  Environment.GetEnvironmentVariable(@"OPENAI_API_KEY"));

OpenAIResponse response =
    await client.CreateResponseAsync([ResponseItem.CreateUserMessageItem(@"How are you today?")]);

foreach (ResponseItem outputItem in response.OutputItems)
{
    if (outputItem is MessageResponseItem message)
    {

        Console.WriteLine($@"{message.Content.FirstOrDefault()?.Text}");
    }
}

// — — —

IChatClient client = new OpenAIClient(key).GetChatClient(@"o3-mini").AsIChatClient();

await foreach (ChatResponseUpdate update in client.GetStreamingResponseAsync(@"How are you today?"))
{
    Console.Write(update);
}
```

Más allá de la mera conveniencia → Antes...

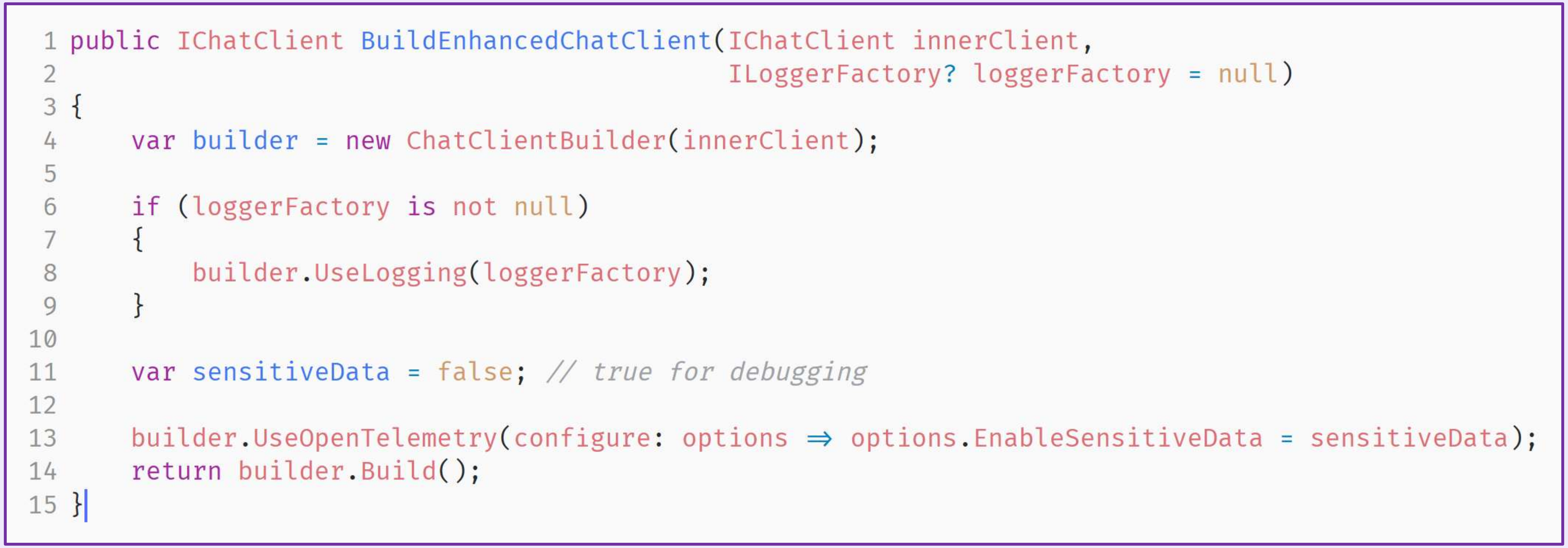


```csharp
public class Person
{
    public string Name { get; set; }
    public int Age { get; set; }
}

public class Family
{
    public List<Person> Parents { get; set; }
    public List<Person>? Children { get; set; }
}

ChatCompletionOptions options = new()
{
    ResponseFormat = StructuredOutputsExtensions
                        .CreateJsonSchemaFormat<Family>("family", jsonSchemaIsStrict: true),
    MaxOutputTokenCount = 4096,
    Temperature = 0.1f,
    TopP = 0.1f
};

List<ChatMessage> messages =
[
    new SystemChatMessage(@"You are an AI assistant that creates families."),
    new UserChatMessage(@"Create a family with 2 parents and 2 children.")
];

ParsedChatCompletion<Family?> completion = chatClient.CompleteChat(messages, options);
Family? family = completion.Parsed;
```

Más allá de la mera conveniencia ← Ahora...

```
public class Person
{
    public string Name { get; set; }
    public int Age { get; set; }
}

public class Family
{
    public List<Person> Parents { get; set; }
    public List<Person>? Children { get; set; }
}

var family = await client.GetResponseAsync<Family>(
 [
    new ChatMessage(ChatRole.System, @"You are an AI assistant that creates families."),
    new ChatMessage(ChatRole.User, @"Create a family with 2 parents and 2 children.")
 ]);
```

La experiencia de middlewares que ya conoces



```csharp
public IChatClient BuildEnhancedChatClient(IChatClient innerClient,
                                           ILoggerFactory? loggerFactory = null)
{
    var builder = new ChatClientBuilder(innerClient);

    if (loggerFactory is not null)
    {
        builder.UseLogging(loggerFactory);
    }

    var sensitiveData = false; // true for debugging

    builder.UseOpenTelemetry(configure: options => options.EnableSensitiveData = sensitiveData);
    return builder.Build();
}
```

Los eventos de OpenTelemetry pueden enviarse a un servicio en la nube como Application Insights o, si estás usando Aspire, a tu panel de Aspire.

Multimodal por diseño 🖼️ 🔈 💬

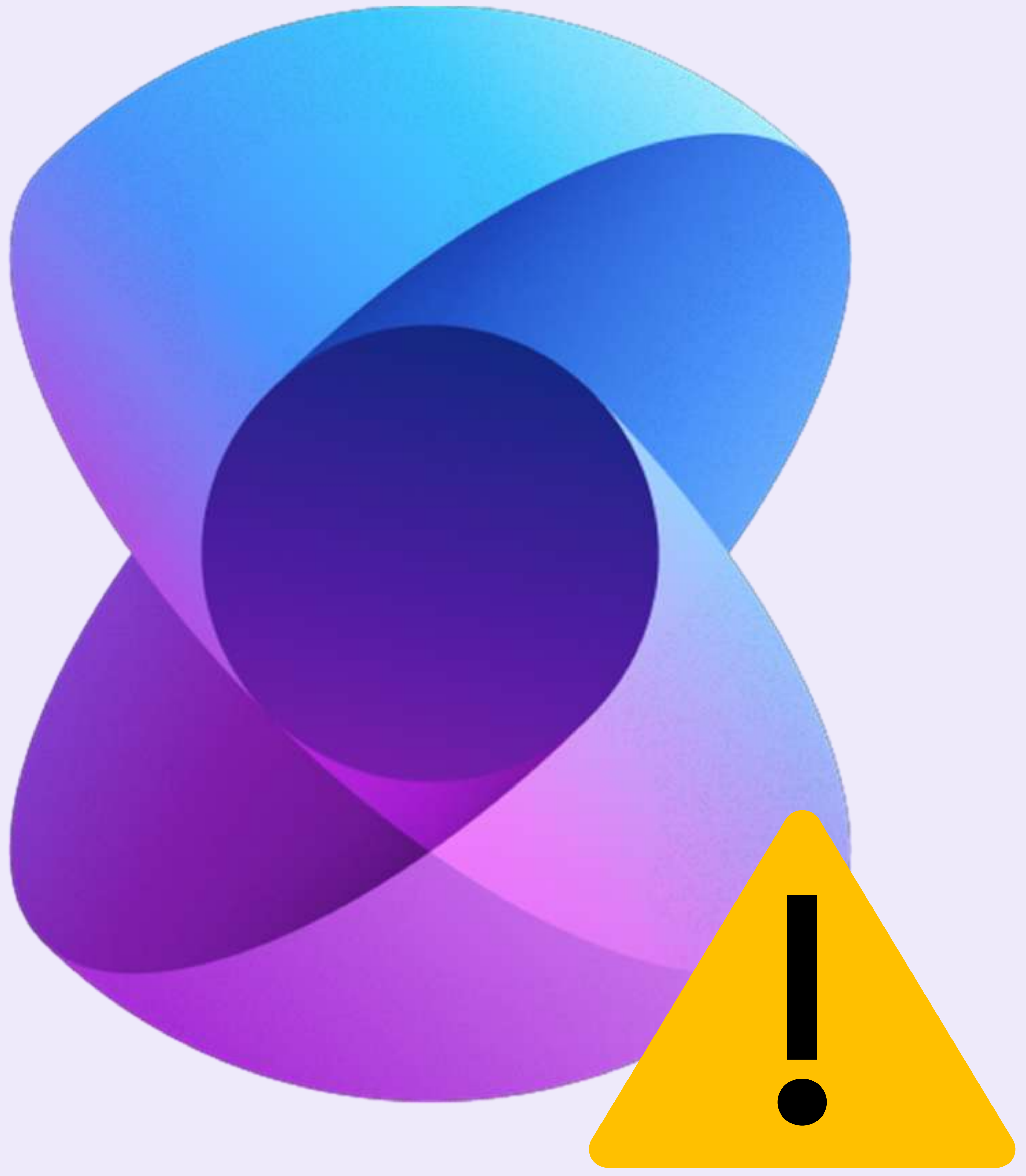
Los modelos de IA generativa actuales hacen mucho más que simplemente intercambiar texto. Cada vez se publican más modelos multimodales que pueden aceptar datos en una variedad de formatos, incluyendo imágenes y sonidos, y devolver recursos similares.

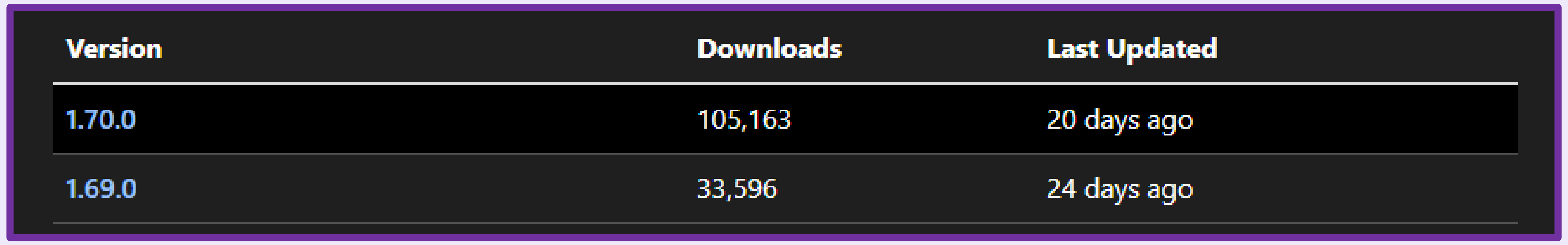
Existen varios tipos de contenido integrados, todos basados en `AIContent`, y el que probablemente usarás con más frecuencia es `DataContent`, que puede representar prácticamente cualquier tipo de medio (audio, vídeo, imagen). Es simplemente un array de bytes con un tipo de medio.

Multimodal por diseño 🖼️ 🔈 💬

```csharp
var instructions = @"You are a photo analyst able to extract the utmost detail from a photograph and provide a description so thorough and accurate that another LLM could generate almost the same image just from your description.";

var prompt = new TextContent(@"What's this photo all about? Please provide a detailed description along with tags.");

var image = new DataContent(File.ReadAllBytes(@"c:\photo.jpg"), @"image/jpeg");

var messages = new List<ChatMessage>
            {
                new(ChatRole.System, instructions),
                new(ChatRole.User, [prompt, image])
            };

record ImageAnalysis(string Description, string[] tags);

var analysis = await chatClient.GetResponseAsyn<ImageAnalysis>(messages);
```

El futuro de las librerías...





Version	Downloads	Last Updated
1.70.0	105,163	20 days ago
1.69.0	33,596	24 days ago

Public Preview

Microsoft Agent Framework

Open-source engine for building and orchestrating intelligent AI agents

Open Standards & Interoperability	Pipeline for Research	Community-Driven & Extensible by Design	Ready for Production

aka.ms/AgentFramework

Azure AI Foundry

AI Agent Ecosystem

Authoring Tools

- Visual Studio
- GitHub

Knowledge

- Microsoft Fabric
- Azure Cosmos DB
- SharePoint
- Microsoft Graph
- Azure AI Search
- Microsoft Bing

Enterprise Trust

- VNet deployments
- OBO Auth
- BYO-resources
- Managed Identities

Agent catalog

Ready-made agents to kickstart your agent workforce

Microsoft Agent Framework

Client SDKs for enterprise and production agentic systems

- Agent tools
- Orchestration
- Channels

Foundry Agent Service

Deploy and manage agents with fully-managed runtime

- Multi-agent workflows
- Built-in threads
- Long-term memory
- Evaluation
- Tracing & Monitoring
- Governance + Safety

Foundry Models

OpenAI, Llama, Grok, Flux, Mistral, Phi, Cohere, Paige, stability.ai Stability, Hugging Face, Industry Models

Foundry Labs

- Muse
- Magentic-One
- Aurora
- OmniParser

Open Ecosystem

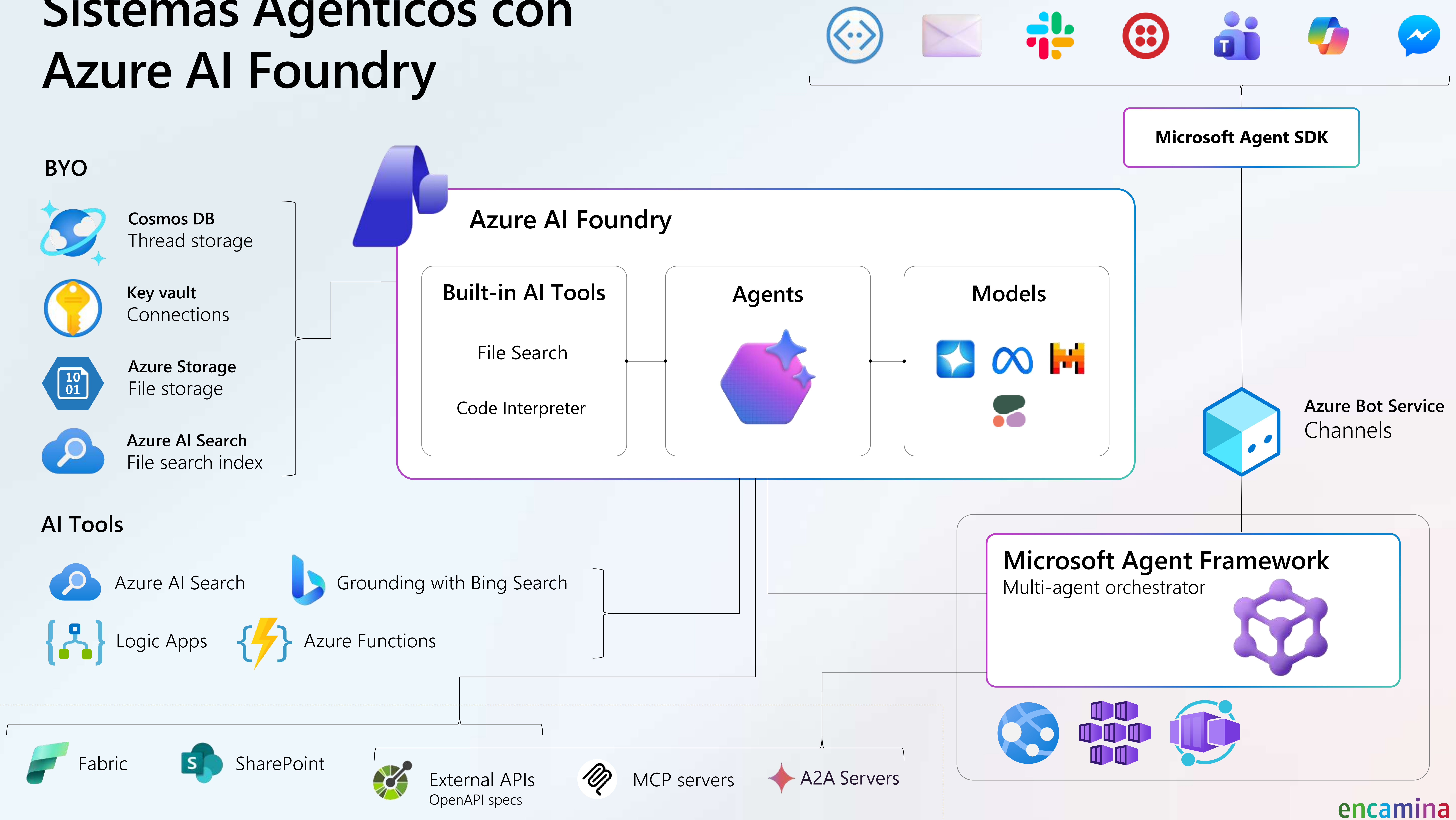
Protocols

A2A, MCP, OpenAPI

Connectors

- Google VertexAI
- Amazon Bedrock
- CrewAI
- LangChain
- ElasticSearh
- Pinecone
- Logic Apps
- Azure Functions

Sistemas Agénticos con Azure AI Foundry



encamina

MCP 🛠️

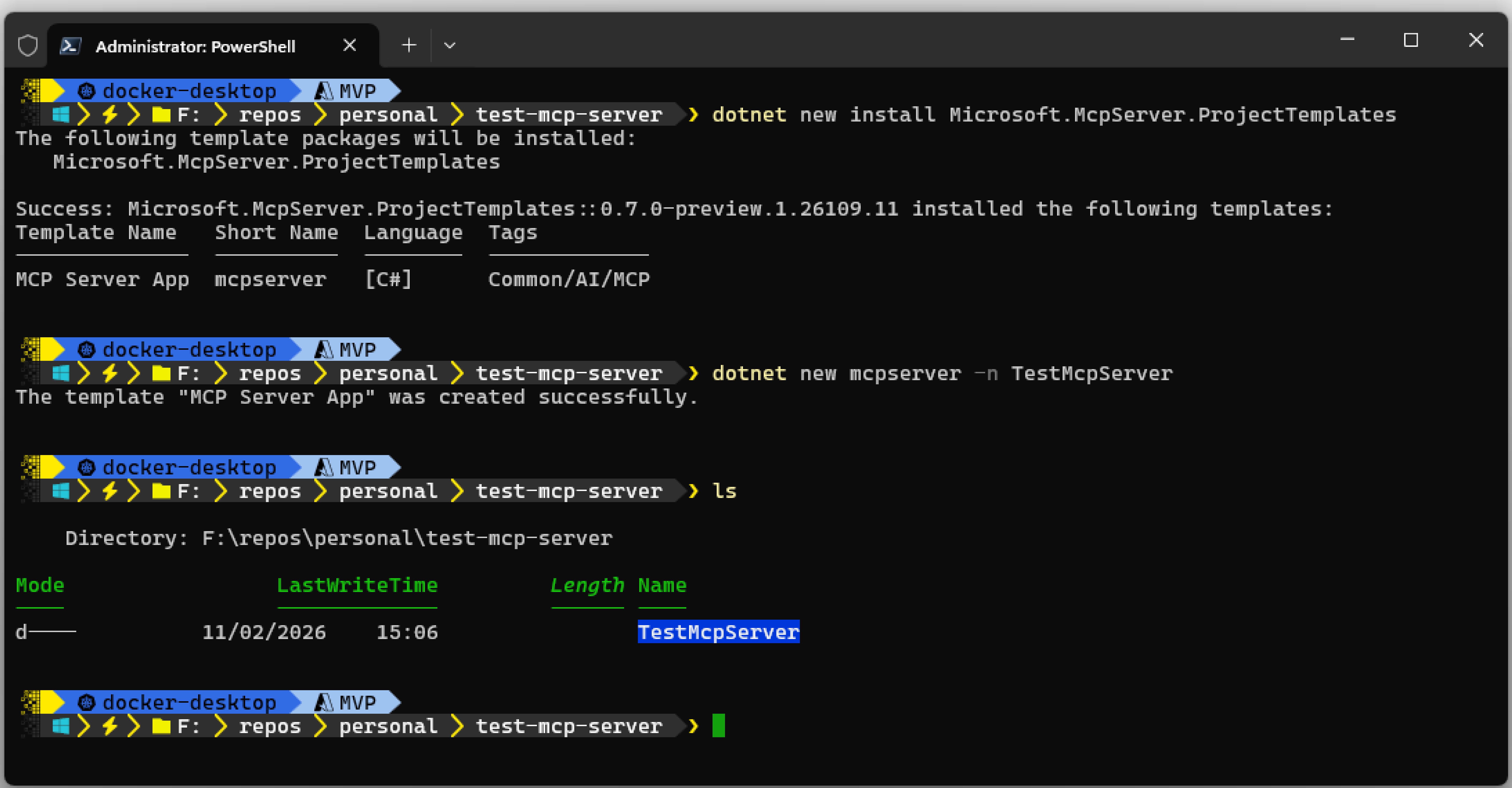
Microsoft junto a Anthropic están creando el SDK MCP «oficial» →

- Código: https://aka.ms/mcp-cs-sdk
- Documentación: https://modelcontextprotocol.github.io/csharp-sdk

Actualmente en la versión 0.8.0-preview.1, evolucionando constante y rápidamente, con cambios ocasionales importantes

El objetivo es implementar fielmente la especificación MCP

Lo utilizan muchos productos de Microsoft que están añadiendo compatibilidad con servidores MCP



Administrator: PowerShell

```
 docker-desktop  MVP
 F:  repos  personal  test-mcp-server  dotnet new install Microsoft.McpServer.ProjectTemplates
The following template packages will be installed:
   Microsoft.McpServer.ProjectTemplates

Success: Microsoft.McpServer.ProjectTemplates::0.7.0-preview.1.26109.11 installed the following templates:
Template Name   Short Name  Language  Tags
-------------   ----------  --------  -------------
MCP Server App  mcpserver   [C#]      Common/AI/MCP


 docker-desktop  MVP
 F:  repos  personal  test-mcp-server  dotnet new mcpserver -n TestMcpServer
The template "MCP Server App" was created successfully.


 docker-desktop  MVP
 F:  repos  personal  test-mcp-server  ls

    Directory: F:\repos\personal\test-mcp-server

Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d----          11/02/2026    15:06                TestMcpServer


 docker-desktop  MVP
 F:  repos  personal  test-mcp-server 
```

.NET is “all-in” for AI

Microsoft.Extensions.AI

AI primitives and building blocks for .NET

VectorData extensions

Data, semantic search, embeddings

MCP server

Inter-agent tools and discovery

AI templates

Out-of-the-box guidance with code ready to go

Agent Framework

Multi-agentic workflows and orchestration

Model eval

Model evaluations and scoring across multiple dimensions

¡GRACIAS!

@rliberoff

https://github.com/rliberoff

https://www.linkedin.com/in/rliberoff/

https://www.youtube.com/@codertectura

https://www.codertectura.com/