

An abstract, complex geometric structure composed of numerous thin, white and light blue lines forming a dense, interconnected web of polygons and cubes. The structure is set against a dark, gradient background that transitions from black on the left to a soft purple and blue on the right. The lines create a sense of depth and complexity, resembling a digital or architectural framework.

AI for .NET developers

Gill Cleeren

CTO Xebia Belgium – Pluralsight Author



Gil Cleeren

CTO Xebia Belgium Microsoft Services

Pluralsight Author

@gillcleeren – gill.cleeren@xebia.com

My courses: gicl.me/mypscourses



Microsoft
Regional Director



Microsoft®
Most Valuable
Professional

+

•

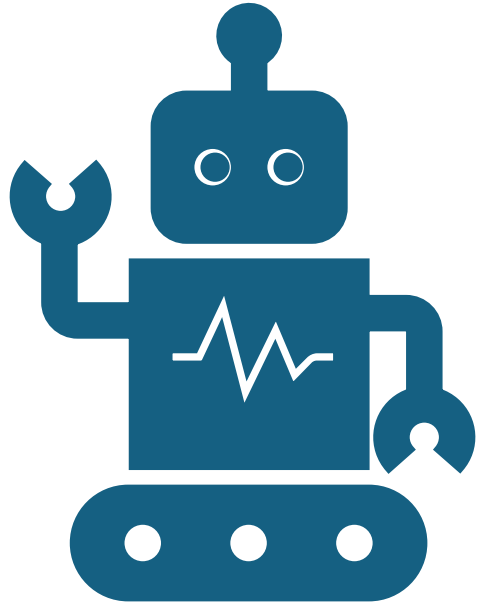
◦

Want to learn more?

Watch my Pluralsight courses!

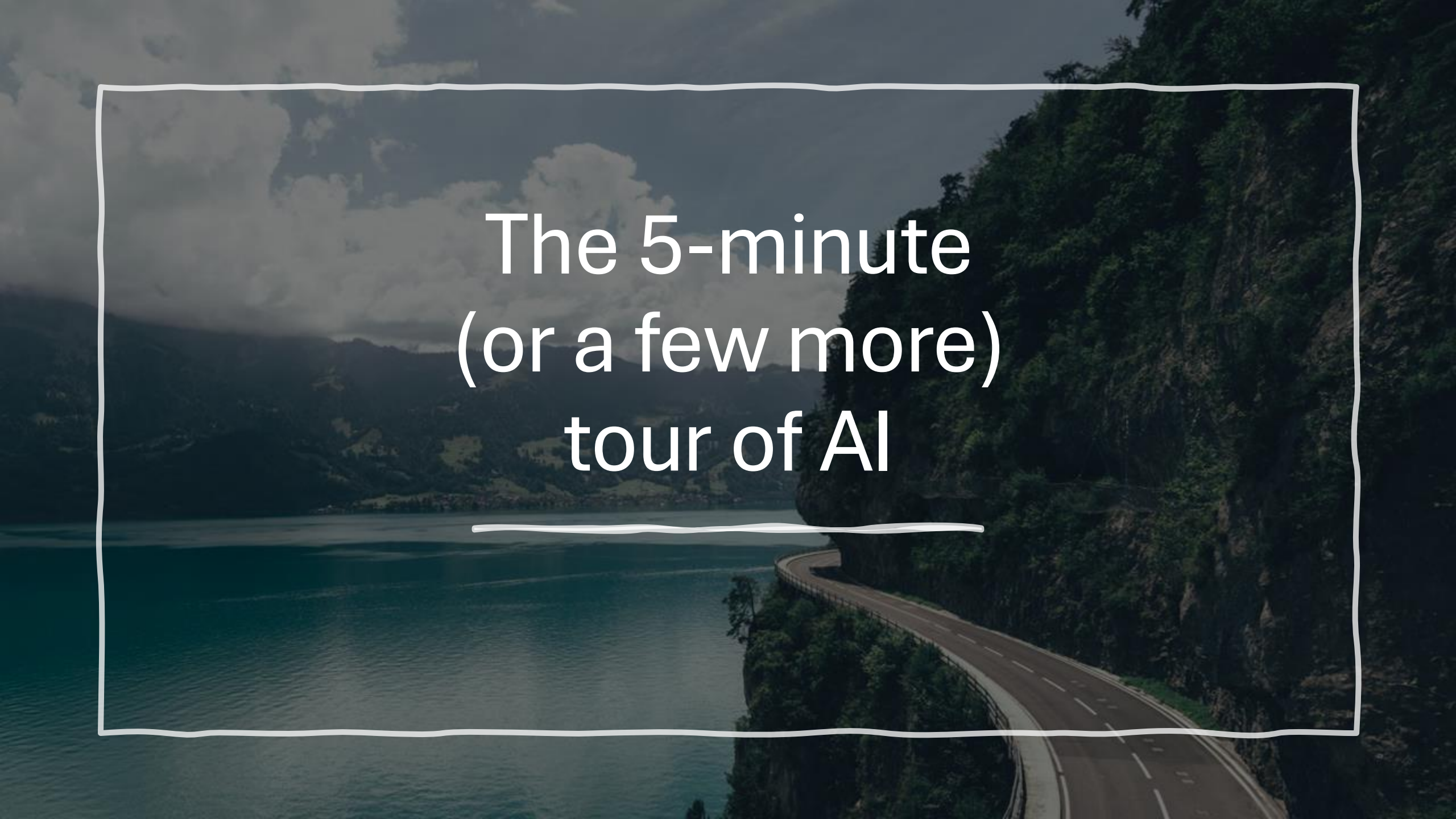
www.pluralsight.com/courses/building-ai-enabled-dot-net-applications

www.pluralsight.com/courses/semantic-kernel-c-sharp-building-ai-applications



Overview

- The 5-minute (or a few more) tour of AI
- Using OpenAI in .NET
- Working with Azure AI Services
- Understanding Semantic Kernel (incl agents)



The 5-minute (or a few more) tour of AI

- Computers are good at executing instructions...

But now they can also
think and learn!





Bringing AI into the mix

- Users can have a lot of benefits
 - We will get insights into large sets of data that weren't possible before
 - Spot trends that we can't on our own
- Developers can create smarter applications
- English (or another language) becomes a programming language





Some terminology


- AI: Ability for a machine to mimic human intelligence
 - GenAI:
 - Short for Generative AI
 - Branch of AI focused on generating new content (images, text...), based on patterns in data
 - LLMs
 - Large Language Models
 - Trained on large amounts of data
 - Capable of understanding and generating human-like content
 - Work internally with numbers which are transformed in a neural network
 - Typically have billions (and now trillions) of numbers: parameters
- 
- 

How LLMs work

- Once upon a...
- Once upon a time...
- Once upon a time, in...
- Once upon a time, in a...
- Once upon a time, in a land...



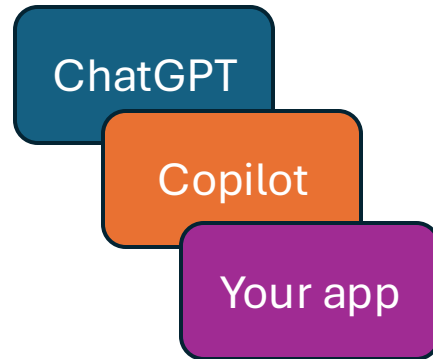
ChatGPT

- Most well-known model
 - G: Generative
 - Ability to generate new content, based on what it has learned
 - P: Pretrained
 - Trained on huge amounts of data
 - T: Transformer
 - Type of neural network that powers GPT
 - Focused on efficiently predicting the next word
- 

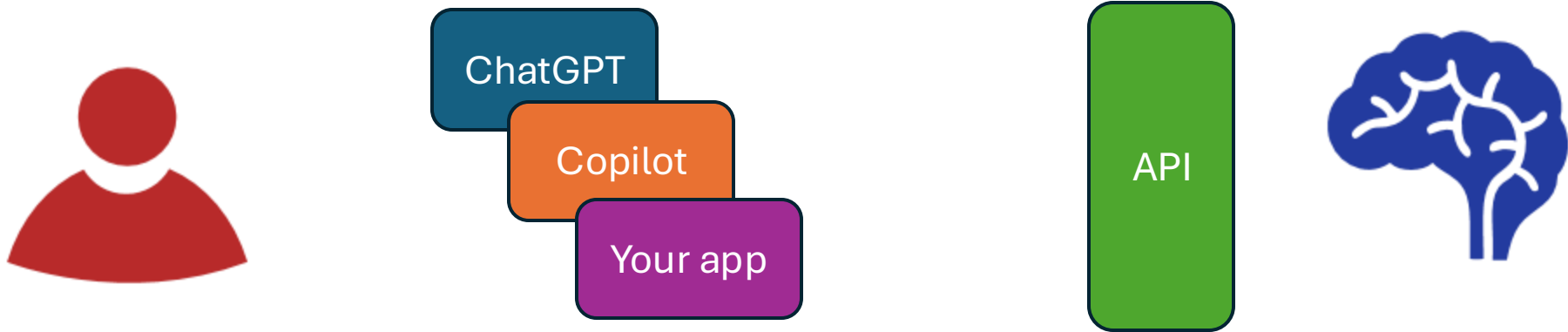
Types of models

- Text to text
 - Text to image
 - Text to video
 - Image to text
 - Speech to text
-
- Multimodal is becoming the norm nowadays

Interacting with models



Interacting with models

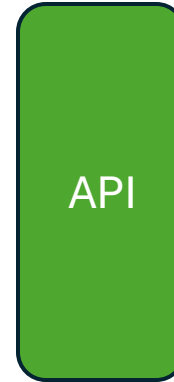


Interacting with models

- The prompt
 - Allows us to “ask” the model to do something
 - Outcome depends on how good the prompt is
- Bad prompt: *“Tell me about history”*
 - Too vague, broad, no context
- Good prompt: *“Tell me about the history of ancient Rome”*
 - Clear focus and topic, but still “open” for the AI to interpret the target
- Great prompt: *“Summarize the key political, cultural, and technological advancements of ancient Rome between 500 BC and 500 AD in less than 200 words”*
 - Very specific, time frame specified, length included

And how do we now use
all this in .NET apps?

Interacting with models



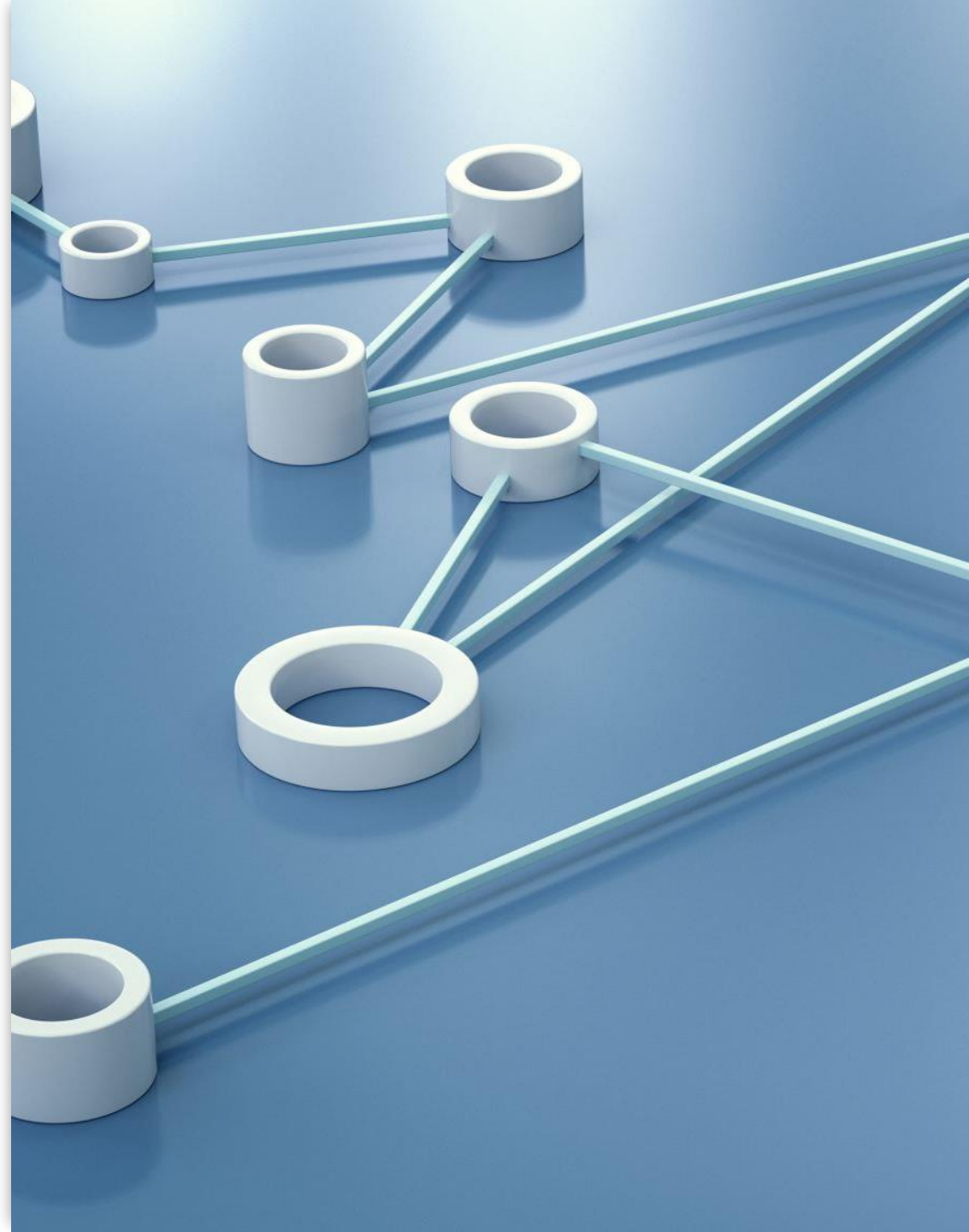
Options to interact with AI models from .NET

- OpenAI
 - OpenAI .NET API library
 - Wrapper around the OpenAI API
 - Exposes functionalities of OpenAI models
- Azure OpenAI Service
 - Managed version of OpenAI models in Azure
 - Adds many more features around security and privacy around public version of OpenAI models



Options to interact with AI models from .NET


- Azure AI Services
 - Set of managed AI services in Azure
 - Pre-built APIs for vision, language, speech
 - SDKs for nearly all services to integrate into apps
- Semantic Kernel
 - Lightweight SDK
 - Focused on simplifying integrating different models in .NET (and other languages)



Options to interact with AI models from .NET

- MCP C# SDK
 - Wrapper around the Model Context Protocol
 - Enables LLMs to invoke tools via structured API calls
- Azure AI Foundry
 - Enterprise-grade model development platform on Azure
 - Supports training, fine-tuning, evaluation, and deployment of custom LLMs
 - Strong support for agentic development

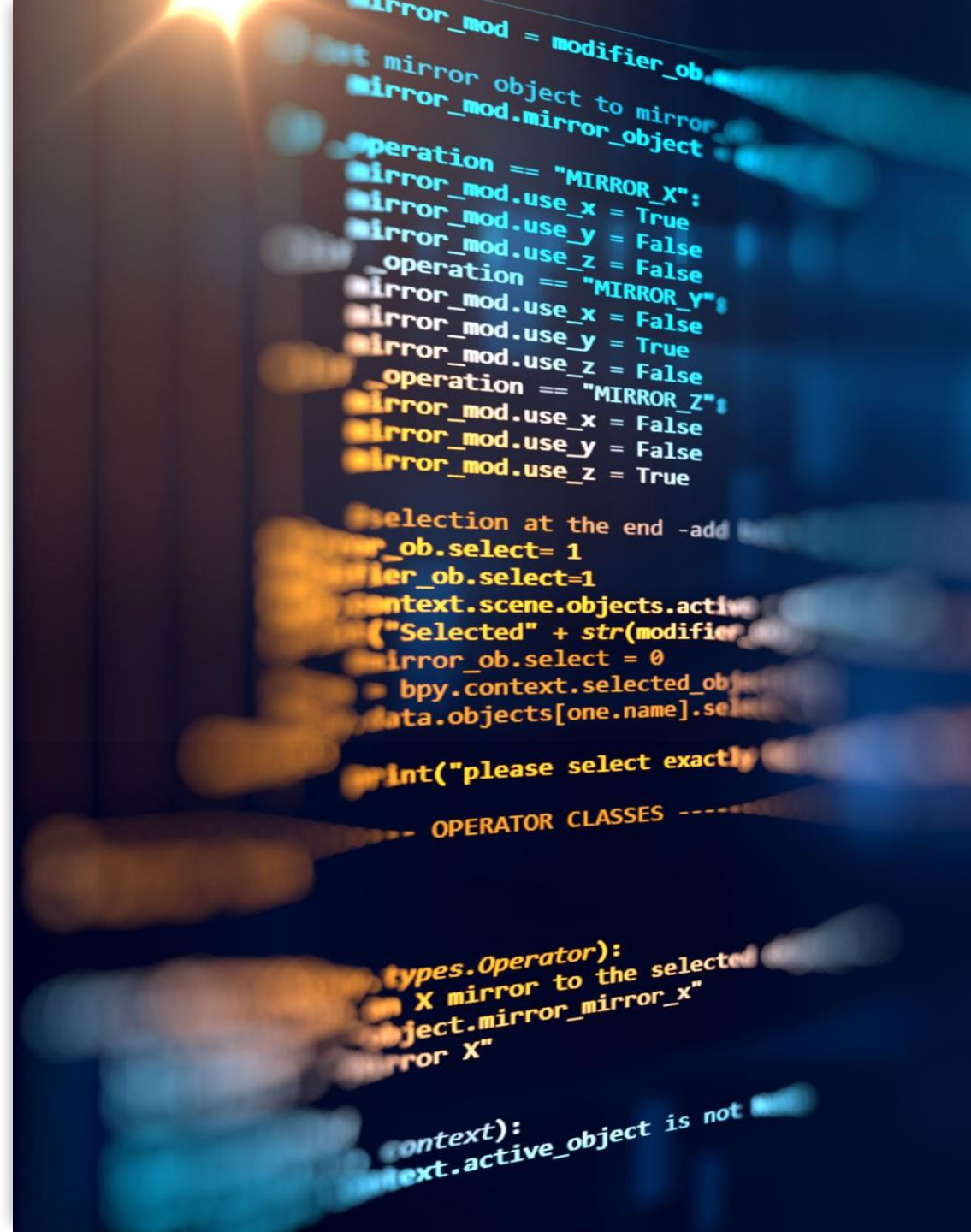




Using OpenAI in .NET

Using OpenAI

- Creators of ChatGPT, Dall-E and Whisper
- Expose functionalities of models through APIs
- We can use them without knowing AI expertise
- Solid integration and backing in Azure
- API allows for HTTP requests to any of their models
 - Works “everywhere”
- API usage requires API key
 - Paid, not combined with paid ChatGPT access



```
mirror_mod = modifier_ob.  
set mirror object to mirror.  
mirror_mod.mirror_object =  
operation == "MIRROR_X":  
mirror_mod.use_x = True  
mirror_mod.use_y = False  
mirror_mod.use_z = False  
operation == "MIRROR_Y":  
mirror_mod.use_x = False  
mirror_mod.use_y = True  
mirror_mod.use_z = False  
operation == "MIRROR_Z":  
mirror_mod.use_x = False  
mirror_mod.use_y = False  
mirror_mod.use_z = True  
selection at the end -add  
mirror_ob.select= 1  
modifier_ob.select=1  
context.scene.objects.active  
("Selected" + str(modifier_ob.  
mirror_ob.select = 0  
= bpy.context.selected_object  
data.objects[one.name].select  
print("please select exactly  
-- OPERATOR CLASSES --  
types.Operator):  
on X mirror to the selected  
object.mirror_mirror_x"  
mirror X"  
context):  
context.active_object is not
```

Interacting with the API

Authorization: Bearer YOUR_API_KEY

POST <https://api.openai.com/v1/chat/completions>

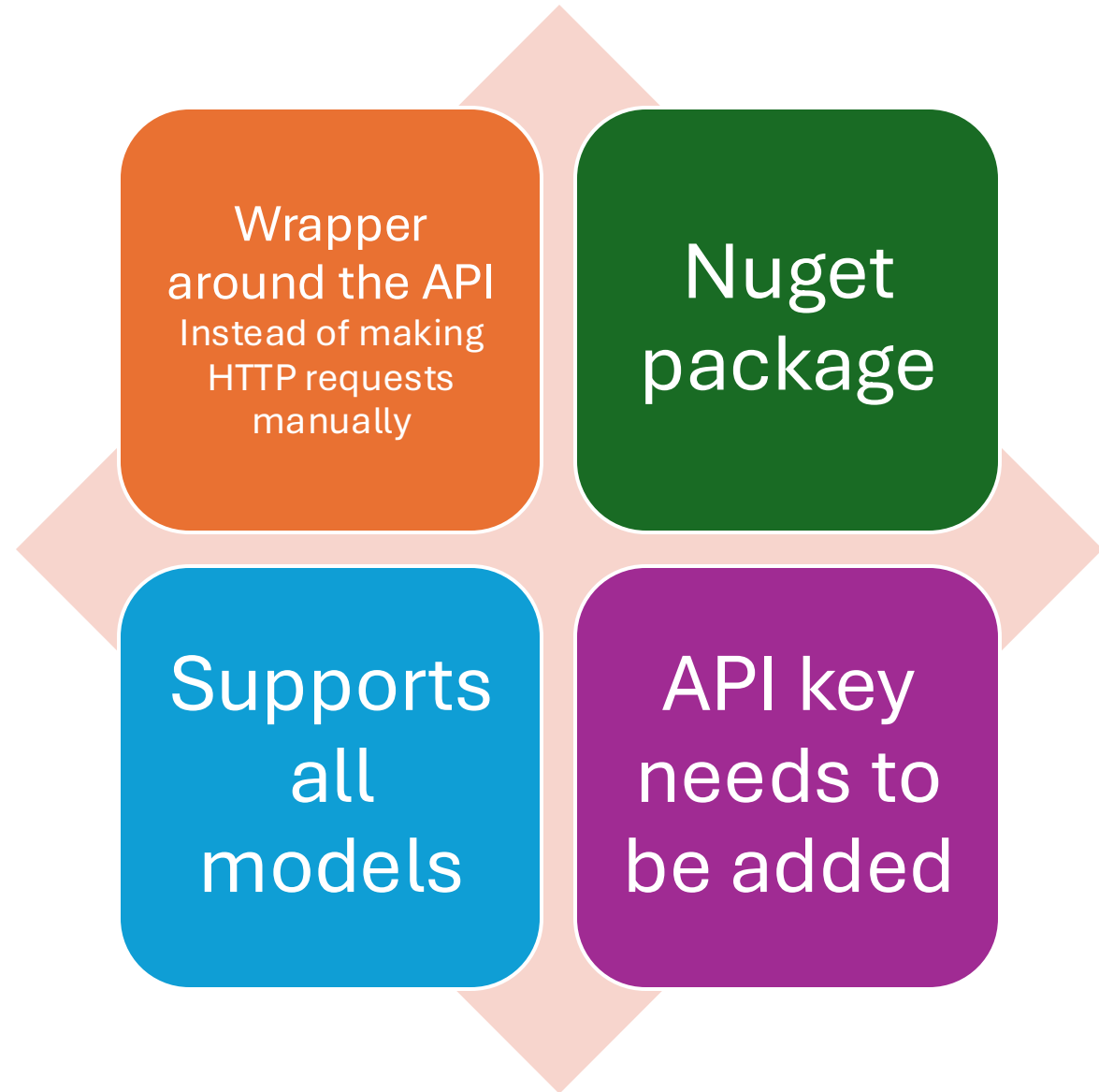
```
{  
  "model": "gpt-4o",  
  "messages": [{"role": "user",  
    "content": "Explain AI in simple  
    terms."}],  
  "max_tokens": 100,  
  "temperature": 0.7  
}
```


An isometric illustration of a staircase made of light gray rectangular blocks. Several stylized human figures are positioned on the blocks, each engaged in a different activity: some are standing and holding devices, some are sitting on the blocks, and one is sitting on a chair placed on a block. The staircase descends from the top left towards the bottom right. The background is a solid dark gray.

DEMO


Working with platform.openai.com

The OpenAI .NET API Library





Simple chat completion




```
ChatClient client = new(model: "gpt-4o",  
Environment.GetEnvironmentVariable("OPENAI_API_KEY"));
```

```
ChatCompletion completion = client.CompleteChat("Say  
'Hello AI world'");
```



Streaming content



```
AsyncCollectionResult<StreamingChatCompletionUpdate>
completionUpdates =
    chatClient.CompleteChatStreamingAsync(chatMessages);

await foreach (StreamingChatCompletionUpdate completionUpdate
in completionUpdates)
{
    foreach (ChatMessageContentPart contentPart
in completionUpdate.ContentUpdate)
    {
        Console.Write(contentPart.Text);
    }
}
```




DEMO

Using the OpenAI .NET API Library

Regular and streaming

History

The background is a complex, abstract composition. It features several concentric, slightly irregular circles or rings that create a tunnel-like effect, drawing the eye towards the center. The circles are dark, possibly black or very dark blue, with lighter, metallic-looking highlights along their edges. Scattered throughout the scene are numerous out-of-focus light points, or bokeh, in shades of blue, teal, and yellow. These lights vary in size and brightness, adding a sense of depth and movement. The overall color palette is cool, dominated by blues and teals, with some warmer yellow and orange highlights from the bokeh lights.

DEMO

Generating images



Adding function calling

- Model doesn't "know" everything
- Function calling can call into local functions in the app
- External system can be used to bring in this information
- Flow
 - "App" sends prompt and definition of functions
 - If needed, the model will ask to invoke local function, including parameters
 - Prompt result of function call back to model

The background of the slide is a black field filled with a grid of small, semi-transparent dots. These dots are arranged in a pattern that forms a large, roughly circular shape on the left side of the frame. The dots themselves are multi-colored, with hues of red, orange, yellow, green, and blue, giving the impression of a digital or particle-based visualization.

DEMO

Function calling

DEMO

Adding useful AI into a real application



Working with Azure AI Services

What is Azure AI Services?



Set of cloud-based AI tools by Microsoft




Pre-built services on already trained models



Build advanced features with limited (or none at all) knowledge of AI or ML



Accessible using APIs and nearly always using SDKs (including C#)



Available services in Azure AI Services

Language

Translator

Speech

Vision

Search

And many more!

Using Azure AI Language

NLP features for analyzing and understanding text

Many different features

- Language detection
- Sentiment analysis
- Summarization
- And much more!

Can be used with pre-built or custom-trained models

Available to use from our apps using SDK or HTTP requests



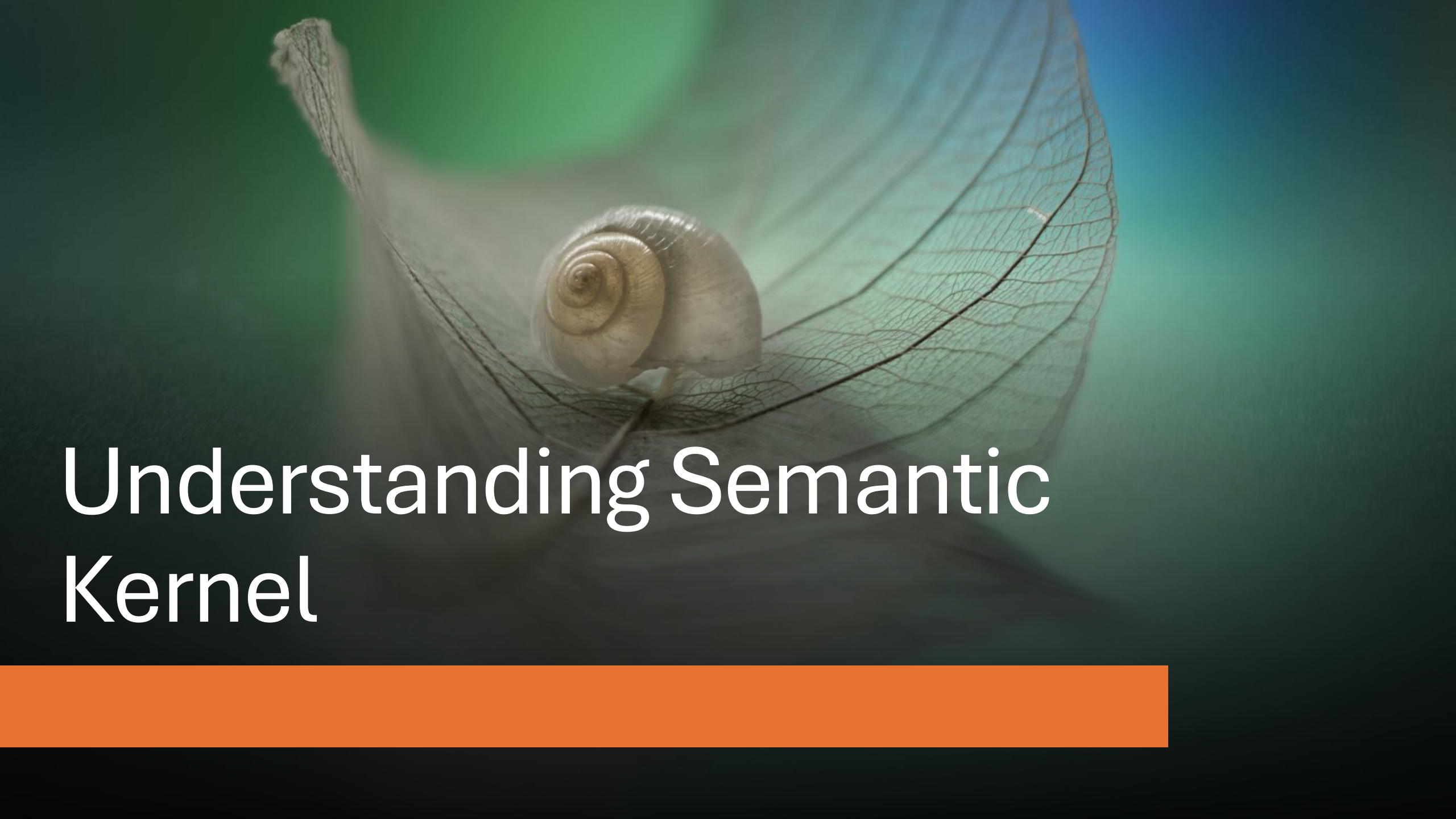
DEMO

Using Azure AI Language



DEMO

Adding Azure AI Services to a real application



Understanding Semantic Kernel



Understanding Semantic Kernel



.NET library, created by Microsoft



Goal: make AI integration into apps easier



Support for many tasks

Text generation, image generation, chat...



Abstraction to work with different AI models

Standardize model interactions



Use from different languages including C#, Java and Python

Getting Started with Semantic Kernel

01

Add in the
correct Nuget
package

02

Connect with
a model

03

Bring in one
or more API
keys

The kernel in Semantic Kernel



Central component



Orchestrator between app and AI models



Acts as DI container to make AI services available for use in app



Manages services and plugins



Services:

AI and other services




Plugins

Components to bring in other functionality




Adding chat completion using SK



```
Kernel kernel = Kernel.CreateBuilder()  
    .AddOpenAIChatCompletion(  
        modelId: modelName,  
        apiKey:  
            Environment.GetEnvironmentVariable("OPENAI_API_KEY"))  
    .Build();
```



Creating a simple chat app using SK



```
string response = string.Empty;

while (response != "quit")
{
    Console.WriteLine("Enter your message:");
    response = Console.ReadLine();
    Console.WriteLine(await kernel.InvokePromptAsync(response));
}
```

A close-up photograph of a wooden abacus, a traditional counting device. It features a light-colored wooden board with several circular holes and a thick black line drawn across it. A small, polished metal sphere is resting on the board, partially inside one of the holes. The numbers '17' and '18' are visible on the board, along with some other faint markings. The lighting is warm and focused on the sphere.

DEMO

First steps with Semantic Kernel

Plugins in Semantic Kernel

- Remember Function Calling
 - Works but is through the OpenAI quite verbose
- Much simpler from SK using Plugins

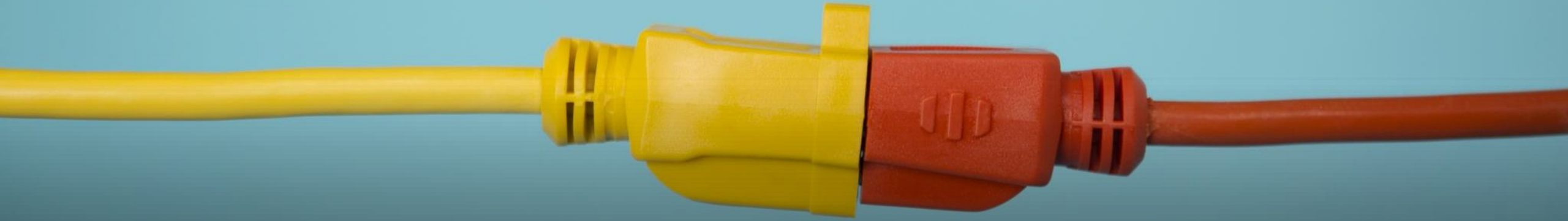
```
public class TimePlugin
{
    [KernelFunction]
    [Description("Gets the current date and time in UTC")]
    public string GetCurrentDateAndTime()
    {
        return DateTime.UtcNow.ToString("R");
    }
}
```



Bringing in Plugins

```
kernel.ImportPluginFromType<TimePlugin>();
```

```
OpenAIPromptExecutionSettings settings =  
    new() { ToolCallBehavior =  
        ToolCallBehavior.AutoInvokeKernelFunctions };
```

DEMO

Using plugins

DEMO

Using Semantic Kernel from our real application

So far...



Our app was “single-agent”



One single “app” (or agent) that does everything

A large orange circle is positioned on the left side of the slide, partially cut off by the edge. It serves as a background for the main title.

Introducing AI Agents

Autonomous system

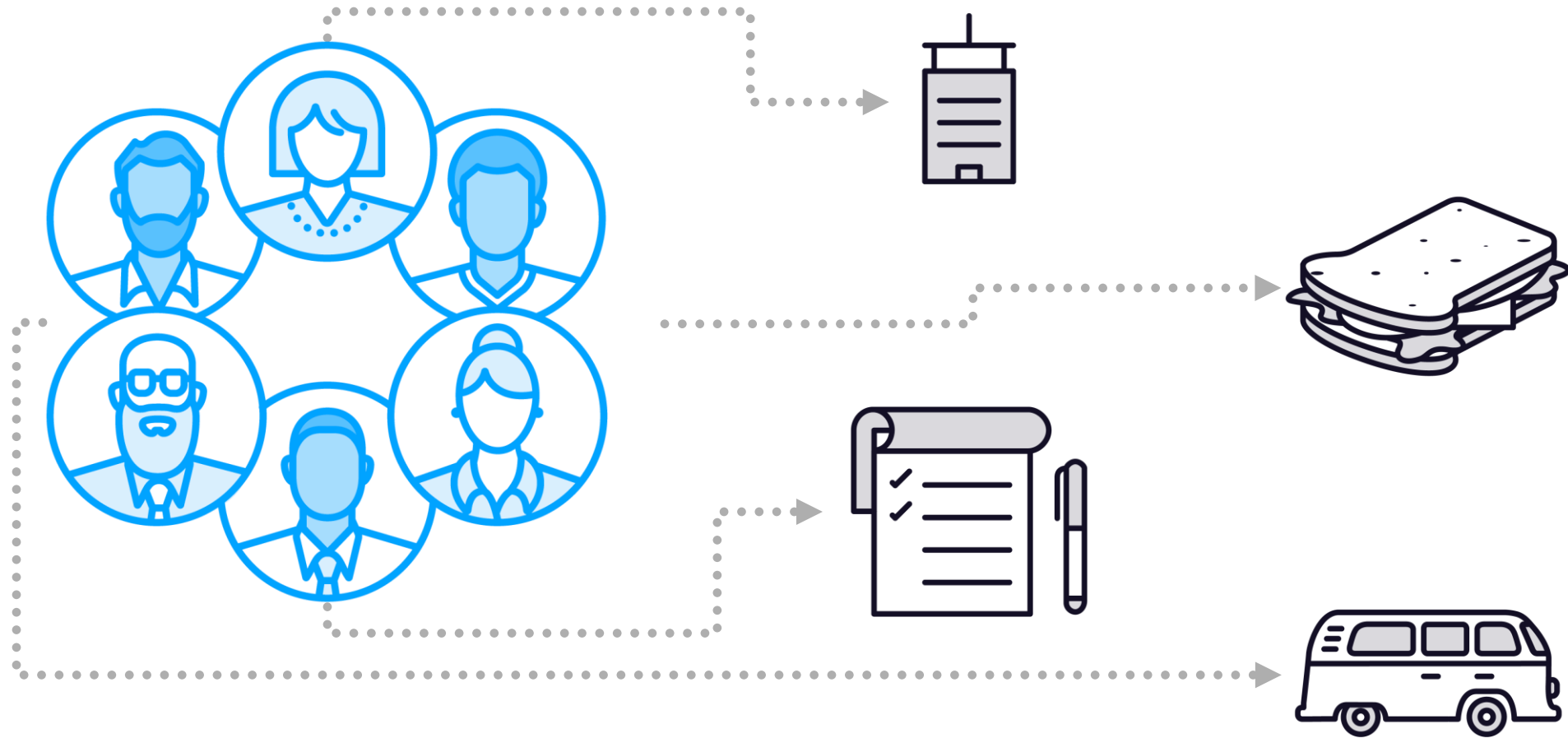
Leverages AI models,
memory, and plugins

Designed to work
collaboratively with each
other

Organizing a Large Event as a “Single Agent”



Using a Team




Using AI Agents



Assign tasks to specialized agents



Break down complexity into smaller parts

A large orange circle is positioned on the left side of the slide, partially cut off by the edge.

Extra
packages
will be
required

Microsoft.SemanticKernel.Agents.Abstractions

Microsoft.SemanticKernel.Agents.Core

Microsoft.SemanticKernel.Agents.OpenAI



Creating an AI Agent

```
ChatCompletionAgent venueAgent = new()  
{  
    Instructions = venueAgentInstructions,  
    Name = "VenueAgent",  
    Kernel = kernel  
};
```


Important concepts when working with agents



GROUP CHAT



SELECTION STRATEGY



TERMINATION
STRATEGY



Demo: AI Agents with Semantic Kernel



Summary

- .NET can definitely be used to create AI-enabled applications
- OpenAI provides a very solid API to access all their models
- Azure AI Services is an easy way to integrate AI-related functionality to any app
- Semantic Kernel provides a wrapper and abstracts away the differences



Questions?



Thank you!

