C# code

Standard code for version 1.0.0-beta.13

```
using Azure;
using Azure.AI.OpenAI;

// dotnet add package Azure.AI.OpenAI --version=1.0.0-beta.13

string Key = "";
string Endpoint = "";
string ModelName = "";

OpenAIClient client = new(new Uri(Endpoint), new AzureKeyCredential(Key));

var chatCompletionsRequest = new ChatCompletionsOptions()
{
    DeploymentName = ModelName,
    Messages =
    {
        new ChatRequestSystemMessage("You are helpful."),
        new ChatRequestUserMessage("Write  a slogan for a computer
programmer.")
    },
    MaxTokens = 200,
    Temperature = 0.8f,
    ChoiceCount = 2
};

ChatCompletions chatCompletionsResponse =
client.GetChatCompletions(chatCompletionsRequest);

Console.WriteLine(chatCompletionsResponse.Choices[0].Message.Content);
Console.WriteLine(chatCompletionsResponse.Choices[1].Message.Content);
```

Standard code for version 1.0.0-beta.9

```
using Azure;
using Azure.AI.OpenAI;

// dotnet add package Azure.AI.OpenAI --version=1.0.0-beta.9

string Key = "";
string Endpoint = "";
string ModelName = "";

OpenAIClient client = new(new Uri(Endpoint), new AzureKeyCredential(Key));

var chatCompletionsRequest = new ChatCompletionsOptions()
{
    DeploymentName = ModelName,
    Messages =
    {
        new ChatMessage("system", "You are helpful."),
        new ChatMessage("user", "Write  a slogan for a computer programmer.")
    },
    MaxTokens = 200,
    Temperature = 0.8f,
    ChoiceCount = 2
};

ChatCompletions chatCompletionsResponse =
client.GetChatCompletions(chatCompletionsRequest);

Console.WriteLine(chatCompletionsResponse.Choices[0].Message.Content);
Console.WriteLine(chatCompletionsResponse.Choices[1].Message.Content);
```

Standard code for version 2.0.0-beta.2

```
using Azure;
using Azure.AI.OpenAI;
using OpenAI.Assistants;
using OpenAI.Chat;

// dotnet add package Azure.AI.OpenAI --version=2.0.0-beta.2

string Key = "";
string Endpoint = "https://masopenai.openai.azure.com/";
string ModelName = "";

AzureOpenAIClient client = new AzureOpenAIClient(new Uri(Endpoint), new
AzureKeyCredential(Key));

ChatClient chatClient = client.GetChatClient(ModelName);
ChatCompletionOptions options = new()
{
    MaxTokens = 40,
    Temperature = 0.8f
};
ChatCompletion ChatCompletionsResponse = chatClient.CompleteChat(
[
        new SystemChatMessage("You are helpful"),
        new UserChatMessage("Write a slogan for a computer programmer"),
        new AssistantChatMessage("Unlocking the code to limitless
possibilities!"),
        new UserChatMessage("Can you translate it into Spanish.")
],
options: options

);

Console.WriteLine(ChatCompletionsResponse.Content[0].Text);
// Console.WriteLine(ChatCompletionsResponse.Content[1].Text);
```

C# code

DALL-E 2 code for version 1.0.0-beta.9

```
using System;
using System.IO;
using System.Threading.Tasks;
using Azure.AI.OpenAI;

// dotnet add package Azure.AI.OpenAI --version=1.0.0-beta.9

namespace Azure.AI.OpenAI.Tests.Samples
{
    public partial class GenerateImages
    {
        public static async Task Main(string[] args)
        {
            string endpoint = "";
            string key = "";

            OpenAIClient client = new(new Uri(endpoint), new
AzureKeyCredential(key));

            Response<ImageGenerations> imageGenerations = await
client.GetImageGenerationsAsync(
                new ImageGenerationOptions()
                {
                    Prompt = "A squirrel holding a photo of a motorcycle",
                    Size = ImageSize.Size1024x1024
                });

            Uri imageUri = imageGenerations.Value.Data[0].Url;

            Console.WriteLine(imageUri);
        }
    }
}
```

C# code

Own Data code for version 1.0.0-beta.13

```
using Azure;
using Azure.AI.OpenAI;

// dotnet add package Azure.AI.OpenAI --version=1.0.0-beta.13

string Key = "b1926da88dd0444787a314136bc37bce";
string Endpoint = "https://masopenai.openai.azure.com/";
string ModelName = "masgpt35";
string SearchEndpoint = "https://masopenaisearch.search.windows.net";
string SearchKey = "hpmaAfG81iJS3N19J1oC8yoKdbVu3XWLgDgiRRqobUAzSeBoL27A";
string SearchIndex= "masopenaisearchindex";

OpenAIClient client = new(new Uri(Endpoint), new AzureKeyCredential(Key));

AzureCognitiveSearchChatExtensionConfiguration ownData = new()
{
    SearchEndpoint = new Uri(SearchEndpoint),
    Authentication = new OnYourDataApiKeyAuthenticationOptions(SearchKey),
    IndexName = SearchIndex
};

var chatCompletionsRequest = new ChatCompletionsOptions()
{
    DeploymentName = ModelName,
    Messages =
    {
        new ChatRequestSystemMessage("You are helpful."),
        new ChatRequestUserMessage("What is the new name for Microsoft Power
Virtual Agents?")
    },
    MaxTokens = 200,
    Temperature = 0.8f,
    ChoiceCount = 1,
    AzureExtensionsOptions = new AzureChatExtensionsOptions()
    {
        Extensions = {ownData}
    }
};

ChatCompletions chatCompletionsResponse =
client.GetChatCompletions(chatCompletionsRequest);

Console.WriteLine(chatCompletionsResponse.Choices[0].Message.Content);
```

Python code

Standard code for version 1.x

- Create a .env file:

```
AZURE_OPENAI_ENDPOINT =
AZURE_OPENAI_KEY      =
AZURE_API_VERSION     =
AZURE_OPENAI_MODEL    =
```

- Create a Program.py file:

```
import os
from dotenv import load_dotenv
from openai import AzureOpenAI
load_dotenv()

client = AzureOpenAI(api_key = os.getenv("AZURE_OPENAI_KEY"),
                     api_version = os.getenv("AZURE_API_VERSION"),
                     azure_endpoint = os.getenv("AZURE_OPENAI_ENDPOINT"))

model_name = os.getenv("AZURE_OPENAI_MODEL")

response = client.chat.completions.create(
              model = model_name,
              messages = [
                  {"role": "system", "content": "You are being helpful"},
                  {"role": "user", "content": "Write a slogan for a computer
programmer."}
              ],
              max_tokens = 200,
              temperature=0.8,
              n=2
)

print (response.choices[0].message.content)
print (response.choices[1].message.content)
```

Python code

Standard code for version 0.28.1

- Create a .env file:

```
AZURE_OPENAI_KEY=
AZURE_OPENAI_ENDPOINT=
AZURE_API_VERSION=
AZURE_OPENAI_MODEL=
```

- Create a Program.py file:

```
import os
from dotenv import load_dotenv
import openai
load_dotenv()

openai.api_type = "azure"
openai.api_version = os.getenv("AZURE_API_VERSION")
openai.api_base = os.getenv("AZURE_OPENAI_ENDPOINT")
openai.api_key = os.getenv("AZURE_OPENAI_KEY")

model_name = os.getenv("AZURE_OPENAI_MODEL")

response = openai.ChatCompletion.create(
              engine = model_name,
              messages = [
                  {"role": "system", "content": "You are being helpful"},
                  {"role": "user", "content": "Write a slogan for a computer
programmer."}
              ],
              max_tokens = 200,
              temperature=0.8,
              n=2
)

print (response.choices[0].message.content)
print (response.choices[1].message.content)
```

Python code

DALL-E 2 code – version 0.28.1

```
#Note: The openai-python library support for Azure OpenAI is in preview.
import os
import openai
openai.api_type = "azure"
openai.api_base = ""
openai.api_version = ""
openai.api_key = ""

response = openai.Image.create(
    prompt='A squirrel on a motorbike.',
    size='1024x1024',
    n=1
)

image_url = response["data"][0]["url"]

print(image_url)
```

Python code

Own Data – version 1.x

- Create a .env file:

```
AZURE_OPENAI_ENDPOINT =
AZURE_OPENAI_KEY      =
AZURE_API_VERSION     =
AZURE_OPENAI_MODEL    =
SEARCH_ENDPOINT       =
SEARCH_KEY            =
SEARCH_INDEX          =
```

- Create a Program.py file:

```
import os
from dotenv import load_dotenv
from openai import AzureOpenAI
load_dotenv()

azure_search_endpoint = os.getenv("SEARCH_ENDPOINT")
azure_search_key = os.getenv("SEARCH_KEY")
azure_search_index = os.getenv("SEARCH_INDEX")

azure_oai_key = os.getenv("AZURE_OPENAI_KEY")
azure_oai_version = os.getenv("AZURE_API_VERSION")
azure_oai_endpoint = os.getenv("AZURE_OPENAI_ENDPOINT")
model_name = os.getenv("AZURE_OPENAI_MODEL")


client = AzureOpenAI(base_url =
f"{azure_oai_endpoint}/openai/deployments/{model_name}/extensions",
                     api_key = azure_oai_key,
                     api_version = azure_oai_version)

extra_config = dict(dataSources = [{"type":"AzureCognitiveSearch",
                                    "parameters": {
                                        "endpoint": azure_search_endpoint,
                                        "key": azure_search_key,
                                        "indexName": azure_search_index
                                    }
                                    }

])

response = client.chat.completions.create(
              model = model_name,
              messages = [
                  {"role": "system", "content": "You are being helpful"},
```

```
                {"role": "user", "content": "What is the new name for
Microsoft Power Virtual Agents?"}
            ],
            max_tokens = 200,
            temperature=0.8,
            n=1,
            extra_body=extra_config
)

print (response.choices[0].message.content)
```