Cloud Contraptions
TRAINING & CONSULTING

To discuss this course and customizations:
Call: +1 434-509-6890 or Email: sales@cloudcontraptions.com

Building AI Agents in .NET with Microsoft Agent Framework

Class Duration

21 hours of live training delivered over 3 days.

Student Prerequisites

- Professional C# development experience on modern .NET, including `async`/`await`, LINQ, and dependency injection (see *Modern C# Development with .NET*)
- Familiarity with LLM fundamentals: prompts, tool calling, and structured outputs
- Working knowledge of ASP.NET Core and the .NET generic host is helpful for the hosting modules
- Basic Azure experience (subscriptions, resource groups, Microsoft Entra ID)
- An Azure subscription with access to a Microsoft Foundry project and a model deployment

Target Audience

C# and .NET developers, architects, and platform teams building AI agents on Microsoft's stack, including teams with Semantic Kernel or AutoGen code to migrate. Copilot Studio and the wider Foundry platform are covered in *Building AI Agents with Microsoft Foundry and Copilot Studio*. For other stacks, see *Building AI Agents with Agent SDKs*.

Description

Microsoft Agent Framework is Microsoft's open-source SDK for AI agents and multi-agent workflows. It is the successor to Semantic Kernel and AutoGen, built by the same teams. This course works in C# from a first `ChatClientAgent` through deployment. Students add tools, MCP servers, memory, and middleware, then build checkpointed graph workflows and multi-agent orchestrations. Final modules cover A2A, observability, evaluation, hosting on Microsoft Foundry or Azure Container Apps, and migration.

Copyright 2026 Cloud Contraptions LLC – All Rights Reserved

To discuss this course and customizations:
Call: +1 434-509-6890 or Email: sales@cloudcontraptions.com

Learning Outcomes

- Place Agent Framework alongside Foundry, Copilot Studio, Semantic Kernel, and AutoGen.
- Choose between an agent, a workflow, or plain code for a task.
- Build `ChatClientAgent` agents on Foundry, Azure OpenAI, OpenAI, and other providers.
- Extend agents with function tools, hosted tools, approvals, and MCP servers.
- Manage multi-turn state with sessions, history providers, and context providers.
- Apply middleware for guardrails, logging, and compliance.
- Use the Agent Harness for long-running, multi-step tasks.
- Build graph workflows that checkpoint, resume, and pause for human input.
- Orchestrate agents with sequential, concurrent, handoff, group chat, and Magentic patterns.
- Call and host remote agents over A2A.
- Instrument, debug, and evaluate agents with OpenTelemetry, DevUI, and evaluators.
- Deploy agents to Foundry or Azure Container Apps, secured with Microsoft Entra ID.
- Plan a migration from Semantic Kernel and AutoGen.

Training Materials

Comprehensive courseware is distributed online at the start of class. All students receive a downloadable MP4 recording of the training.

Software Requirements

The .NET 10 SDK, Visual Studio or VS Code with the C# Dev Kit, an Azure subscription with a Microsoft Foundry project and model deployment, the Azure CLI, the Azure Developer CLI (`azd`) with the `azure.ai.agents` extension, Docker or Podman, and Git.

Training Topics

Agent Framework in the Microsoft AI Stack

- Successor to Semantic Kernel and AutoGen
- Agents, the harness, workflows, and integrations

Copyright 2026 Cloud Contraptions LLC – All Rights Reserved

- `IChatClient` and `AIFunction` foundations
- Shared message types
- Foundry, Copilot Studio, and Microsoft 365 Copilot
- Choosing an agent, a workflow, or plain code
- Stable and prerelease packages
- The Python and Go SDKs

Agents and Model Providers

- `AIAgent` and `ChatClientAgent`
- Creating agents with `AsAIAgent`
- Foundry, Azure OpenAI, and OpenAI clients
- Anthropic, Bedrock, Gemini, and Ollama
- `RunAsync` and `RunStreamingAsync`
- Structured output and multimodal input
- Background responses
- Remote agent services

Tools, Function Calling, and MCP

- Function tools with `AIFunctionFactory`
- `[Description]` metadata
- Code interpreter, file search, and web search
- Approval-required tools
- Handling tool approval requests
- MCP servers with the MCP C# SDK
- Hosted MCP tools
- Microsoft Foundry Toolbox
- Agents as tools and MCP servers

Conversation State and Memory

- Creating and resuming an `AgentSession`
- Serializing sessions
- Service-managed versus local chat history
- `ChatHistoryProvider`
- Context providers for memory and RAG
- Built-in providers such as Mem0 and Redis
- Compaction for long conversations
- Scoping conversation IDs to users and tenants

Copyright 2026 Cloud Contraptions LLC – All Rights Reserved

To discuss this course and customizations:
Call: +1 434-509-6890 or Email: sales@cloudcontraptions.com

Middleware and the Agent Pipeline

- How a run flows through the pipeline
- Agent run middleware
- Function calling middleware
- `IChatClient` middleware
- Registering middleware with `AsBuilder()`
- Guardrails and early termination
- Result overrides and exception handling
- Microsoft Purview middleware

The Agent Harness

- `HarnessAgent` and `AsHarnessAgent`
- Planning and todo tracking
- File memory and Agent Skills
- Standing tool approvals
- Built-in web search and telemetry
- File access, shell tools, and background agents
- Bounded looping
- Customizing defaults with `HarnessAgentOptions`

Graph-Based Workflows

- `WorkflowBuilder`, executors, and edges
- Conditional routing
- Fan-out and fan-in
- Supersteps and workflow events
- Streaming runs with `InProcessExecution`
- Shared state and sub-workflows
- Agents as executors
- Workflows as agents
- Declarative workflows in YAML

Checkpointing and Human-in-the-Loop

- Checkpoints with `CheckpointManager`
- Restoring and rehydrating workflows
- Stable identities for resumable workflows
- `RequestPort` and `RequestInfoEvent`
- Tool approvals inside workflows

Copyright 2026 Cloud Contraptions LLC – All Rights Reserved

- Checkpoint storage as a trust boundary

Multi-Agent Orchestration Patterns

- Sequential orchestration
- Concurrent orchestration
- Handoff orchestration with `AgentWorkflowBuilder`
- Group chat managers
- Magentic orchestration and progress ledgers
- Stall detection and plan review
- Agent-as-tool versus handoff

Agent-to-Agent (A2A) Interoperability

- Agent cards and discovery
- Calling remote agents with `A2AAgent`
- Resolving cards with `A2ACardResolver`
- Hosting agents with `AddA2AServer`
- JSON-RPC and HTTP+JSON endpoints
- Continuation tokens and stream reconnection
- Durable session and task stores

Observability, Debugging, and Evaluation

- OpenTelemetry with `UseOpenTelemetry`
- GenAI semantic conventions
- Application Insights and the Aspire dashboard
- Sensitive data in telemetry
- Local debugging with DevUI
- `LocalEvaluator` and `FoundryEvals`
- `Microsoft.Extensions.AI.Evaluation`
- Repetitions and expected outputs
- Evaluating workflows per agent

Hosting on Microsoft Foundry and Azure Container Apps

- Foundry hosted agents versus self-hosting
- The Responses and Invocations protocols
- `azd ai agent init` and `azd ai agent run`
- Deploying with `azd deploy`
- Self-hosting with `AddAIAgent` and `MapOpenAIResponses`
- Containers on Azure Container Apps

Copyright 2026 Cloud Contraptions LLC – All Rights Reserved

To discuss this course and customizations:
Call: +1 434-509-6890 or Email: sales@cloudcontraptions.com

- Durable agents on Azure Functions
- The Durable Task Scheduler

Identity and Security

- `ManagedIdentityCredential` in production
- Developer credentials locally
- Entra agent identities for hosted agents
- Per-user session isolation
- Authenticating MCP and A2A connections
- On-Behalf-Of passthrough
- Foundry guardrails and prompt-injection defenses
- Secrets in Azure Key Vault

Migrating from Semantic Kernel and AutoGen

- Semantic Kernel plugins to tools
- `ChatCompletionAgent` to `ChatClientAgent`
- `AgentThread` to `AgentSession`
- `InvokeAsync` to `RunAsync`
- Filters to middleware
- AutoGen teams and GraphFlow to workflows
- Planning an incremental migration

Copyright 2026 Cloud Contraptions LLC – All Rights Reserved