

Agentic UI with **Angular**

Architecting Agentic AI with Open Standards



Manfred Steyer

Agentic UI with Angular

Architecting Agentic AI with Open Standards

Manfred Steyer

v1.0.0, August 17, 2026

Contents

Preface	1
Whom This Book Is For	1
How to Read This Book	2
Structure of This Book	2
Source Code	2
What You Need	3
API Keys	3
Libraries and Frameworks	4
Versions Used in This Book	4
Skills for Agentic Development	4
Acknowledgements	5
About the Author	6
Trainings and Consultancy	7
Feedback	7
1 From Agents to Agentic UI	8
How Agents Work	8
The Agentic Loop	8
System Prompt	10
Conversation with Tool Calling	12
Instruction Hierarchy	14
The Importance of Examples in the Prompt	15
Prompt Injection	15
What We Do Differently Today	15
What Production Systems Add	16
Forms of Agentic UI	17
Natural Language Queries	17
Extracting Data from Images and Documents	18
Workflows with Structured and Unstructured Data	19
Co-Creation	19
Chat with Dynamic Components and Actions	20
Implicit Intent Recognition	21
Autonomous Behavior	21
Summary	22
2 AG-UI: The Standard for Agentic User Interfaces	23
Understanding AG-UI	23
What Is Agentic AI in the Frontend?	23

What is AG-UI?	26
Message Types in AG-UI	27
AG-UI in Practice: The SDK for TypeScript	28
What Does the AG-UI SDK Provide?	29
Using the TypeScript SDK on the Server	29
Using the TypeScript SDK in the Browser	31
Server-Side Tool Calling with the AG-UI SDK	32
Client-Side Tool Calling with the AG-UI SDK	33
AG-UI, Message History, and Client Tools	36
Reading Client Tool Information on the Server	36
AG-UI End to End: Connecting Server and Client	36
An Agent with Mastra	37
The AG-UI Client	43
Testing the Application	45
An Angular Client with CopilotKit	46
Implementing AG-UI with Angular and CopilotKit	49
Integrating CopilotKit into Angular	49
Connecting Agents through an Agent Store	50
The Helper Function <code>initAgentStore</code>	51
Defining Client-Side Tools	52
The Inner Workings of <code>createFrontendTool</code>	54
Widgets Are Tools with a Component	55
Sending Requests to the Agent	56
Presenting the Chat History in Angular Templates	57
Trying Out the Solution	58
Summary	59
3 A2UI: How AI Generates Dynamic UIs at Runtime	60
A2UI Fundamentals	60
What is A2UI?	60
A2UI in Action: Examples of Dynamically Generated UIs	61
What Components Does the A2UI Basic Catalog Contain?	61
What Functions Does A2UI Provide?	63
How Do A2UI Messages and the Renderer Work?	63
Integrating the A2UI Renderer in Angular	64
Example: A Passenger Card with A2UI in Angular	64
Integrating A2UI with AG-UI and CopilotKit	71
How Do You Transmit A2UI Messages over AG-UI?	71
The Building Blocks: A2UI Renderer and CopilotKit	73
Client-Side Integration with CopilotKit	73
Custom Catalogs: Your Own Components for AI-Generated UIs	82
Custom Catalogs with the A2UI Renderer	82
Custom Catalogs with CopilotKit and AG-UI	88
A2UI with a DSL: Controllable and Optimized for Performance	99
The Problem: Why A2UI Generation Was Slow and Expensive	99

The Solution: an Application-Specific DSL Instead of A2UI Markup	100
Consequences: Pros and Cons of the DSL Approach	103
Even More Performance: Caching the A2UI Markup	103
The Result: 300 Times Faster, 30 Times Fewer Tokens	105
Summary	107
4 MCP Apps: Tool Results as Interactive Widgets	108
MCP and MCP Apps	108
MCP at a Glance	108
MCP Apps at a Glance	109
First Demo Application	110
MCP Apps SDK	111
Providing a Host	112
Providing an App	115
Cleaning Up App and Host	115
Integrating an MCP Server with MCP Apps	116
Providing an MCP Server	117
Direct Access with the MCP Inspector	120
Integrating MCP Apps into the Backend	120
MCP Apps via AG-UI	121
Displaying MCP Apps with CopilotKit	123
Accessing the MCP Server via a Proxy	124
Rendering Activities	125
Summary	128
5 Human-in-the-Loop (HITL) Patterns	129
Transparency	129
Approvals and Interrupts	131
A look at the protocol	132
The interrupt in the client	135
Interrupts: the server’s view	136
Red lines: actions the agent never executes	138
Offering options with interrupts	139
Emergency stop button	141
Action Cards	142
Co-Planning	147
A toolbox instead of a master key	148
Plan and execution mode in the client	150
The Execute button	150
Plan and execution mode on the server	152
Co-Creation	153
The artifact lives in the client	154
Generating and refining are two different beasts	156
Summary	156

6	Guardrails	157
	Guardrails Come in Many Forms	157
	Architecture as a Guardrail in the Broader Sense	160
	Input and Output Checks	162
	Deterministic: the Blocklist	163
	Model-based: the Off-Topic Check	164
	From Tripwire to Chat Message	165
	The Limits of the Threshold	167
	Detecting Prompt Injections	168
	Sandboxes	170
	Why Doesn't the Model Just Do the Math?	170
	QuickJS as Jailhouse	171
	The Interplay	171
	The Path to the Chart in the Client	174
	JSONata as an Alternative	176
	Summary	177
7	Sub-Agents and Workflows	178
	Sub-Agents	178
	The Bridge to the UI: AG-UI	181
	Workflows: The Travel Planner	182
	The Criteria: A Rough Plan as a Schema	182
	The Main Agent: Criteria Instead of Answers	183
	The Deterministic Steps	184
	The Finalizer: Judgment Where It Counts	184
	The Progress Indicator: Making Steps Visible	186
	A Look Behind the Scenes: The Detail View	187
	The Finished Plan	189
	Integrating External Agents with A2A	189
	Summary	191
8	State Management with AG-UI	192
	State and AG-UI	192
	Using State Management on the Client	194
	State Management on the Server	198
	Result	200
	Summary	201
9	Testing with AG-UI	202
	Seams for Testing	202
	Vitest in Browser Mode	203
	From Unit Tests to Component Tests	205
	Agent Store with a Mocked Agent	205
	Simulating the Backend via AG-UI	209
	aimock: A Ready-Made AG-UI Mock Server	212
	Frontend Tools as Ordinary Functions	217

What the Tests Cover – and What They Don’t	220
Summary	221
10 Multimodality	222
Documents and Images	222
Working with Documents	222
Generating Documents	232
Deterministic Alternatives	234
Voice	235
Browser APIs	235
Voice Models	239
Summary	242
Trainings and Consulting	243
Imprint & Legal Notice	244