

WHITE PAPER

LIVE Q/A WEBINAR · LOGIC APPS CLI

Why the command line matters in an AI world

What we learned when Microsoft's Wagner Silveira and our CTO Ahmed Bayoumy answered your questions about the new Logic Apps CLI.

```
$ az extension add --name logicapp
  --allow-preview true
```



Wagner Silveira
Microsoft



Ahmed Bayoumy
CTO, Contica



Q/A MODERATOR
Robin Wilde
Contica

Contica
Ahmed Bayoumy and Robin Wilde

Webinar held 4 November 2026
Microsoft Teams

AT A GLANCE

Five things to take from the webinar

On 4 November 2026 we held a live Q/A about the Logic Apps CLI, which Microsoft released as a public preview on 28 October 2026. Wagner Silveira from the Logic Apps product group and our CTO Ahmed Bayoumy answered every question that came in.

1. **The CLI adds the deterministic part.**
   The model interprets the requirement and writes the workflow. The CLI discovers connectors, validates and runs tests, the same way every time.

2. **It covers the whole life cycle, not only scaffolding.**
   Discovery, validation, local runtime and tests in one command surface, outside any single editor.

3. **Your role moves up, it does not disappear.**
   Wagner compares it to becoming the maestro. Ahmed says the integration developer and the architect become even more crucial.

4. **Keep what works and add the CLI where your skills stop.**
   It is another tool in the toolbox, not a replacement for everything.

5. **Your team's patterns belong in your own skills.**
   The CLI handles what is deterministic. Code review and team standards stay in separate skills.

HOW THE AUDIENCE BUILDS LOGIC APPS TODAY



26 registrants answered before the webinar. Several answers were possible.

BACKGROUND

What is the Logic Apps CLI?

Until now, the local development loop for Logic Apps Standard has lived inside the VS Code extension. The Logic Apps CLI opens that loop to the terminal, to scripts and pipelines, and to coding agents.

Before the CLI: the local loop was locked in one editor

TODAY'S CLI · CLOUD ONLY

Manage what is deployed

`create · settings · scale · zip deploy`

LOCAL LOOP · VS CODE EXTENSION ONLY

Build on your machine

`scaffold · repair · connectors · runtime · tests`

↓

A local CLI opens the workflow life cycle to other tools, including coding agents.

Adapted from Wagner Silveira, "Evolving the Logic Apps Developer Experience", Nordic Integration Summit 2026, slide 13 ("Today Logic Apps dev loop is locked in one editor").

STATUS

Public preview

Released 28 October 2026

COMMANDS

59

Azure CLI extension logicapp 1.0.0b1

REQUIRES

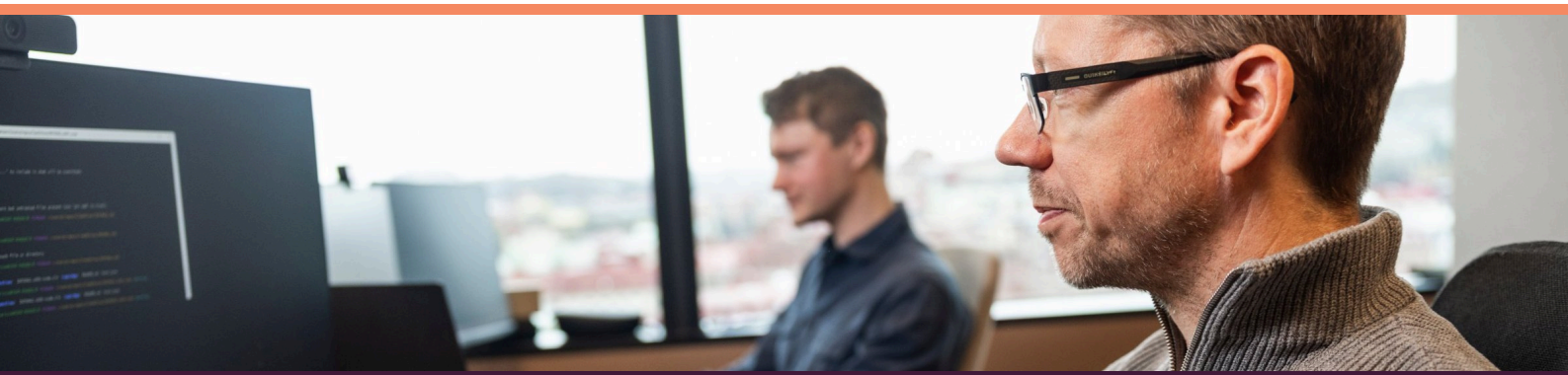
Azure CLI 2.67.0+

and Python 3.10 or later

INSTALL

```
$ az extension add --name logicapp --allow-preview true
```

A predefined skill for coding agents, azure-logic-apps-standard-cli, is available on Awesome Copilot.



WHY A CLI

Adding the deterministic part to AI assisted development

Coding agents make building Logic Apps easier, but Wagner described today's result as costly and hit and miss. What the agents lack is the discovery the portal and the VS Code extension do well: which actions and connectors exist, and how to use them.

THE MODEL

Interprets and writes

- Reads the business requirement
- Turns it into a workflow definition, or into C# for codeful
- Should decide as little as possible: the less you leave to the model, the better

⇄

THE CLI

Discovers, validates, runs

- Finds the right connector and action in the catalog
- Validates a whole project or a single workflow
- Starts the runtime and runs tests, the same way every time

FROM WAGNER'S ARTICLE

"Plausible isn't authoritative." The operation, its inputs and its authentication should come from the connector catalog, not from the agent's memory of a similar example.

Two places it helps

1. **Repeatable processes.** If you always create your workspace the same way or always run the same tests, a script can do it without manual clicks.
2. **Coding agents.** The agent gets deterministic steps for the same things you would do by hand: create, discover, connect, validate.

> "Adding the determinism where it is needed makes the process more robust, because it is repeatable, and also cheaper."



Wagner Silveira, Microsoft

Ahmed's view from a month of heavy use: own skills, rules and guardrails were not enough in a bigger team. Validation became the first gate, and unit tests plug in after it, in the same session and without VS Code.

Q THE OPENING QUESTION

If an agent can build the Logic App, what is left for the integration developer and the architect?



Answered together by **Wagner Silveira**, Microsoft, and **Ahmed Bayoumy**, Contica. They agreed on the answer.

THE ANSWER IN FIVE POINTS

1. **The agent can build the workflow. Someone still has to own it.**
   Owning its quality and its context remains the job of the integration developer and the integration architect.

2. **The role becomes more important, not less.**
   Without someone who understands the business process, five developers give five different Logic Apps, and giant workflows grow into a spider web.

3. **The integration developer owns the quality.**
   Understand the business process and the problem first, validate what the agent produced, and correct it along the way.

4. **The integration architect owns the context.**
   Define the boundaries and the patterns, check that the solution fits everything else, and set the rules when it spans many Logic Apps.

5. **It is not autopilot.**
   Experience goes into rules and skills that the agent builds from, and deterministic tools like the CLI check that they are followed.

Summary of the webinar, 00:21:08 to 00:27:37, with Ahmed at 00:51:51 and Wagner at 00:53:12. Each of them in their own words on the next page.

THE OPENING QUESTION · IN THEIR OWN WORDS



Wagner Silveira
Microsoft

Give the same requirement to five developers with different experience and you get five different Logic Apps. The real job is to define the boundaries and the patterns, and to ask whether you are solving the problem or described it wrong.

> **"You become the course corrector and the maestro of the whole thing."**



Ahmed Bayoumy
Contica

Understanding the business process is what integration work has always been about. Without it you get a spider web of workflows. When a solution spans 10 or 15 Logic Apps, the architect sets the rules and the CLI validates against them.

> **"You need to know how to validate. This is about validation, not autopilot."**

Why Wagner talked about tailors

Wagner explained the shift with an analogy from Nir Mashkowski's keynote at Nordic Integration Summit 2026. It describes what happens to a craft when production is suddenly automated, and where the people who know the craft end up.

THEN

Tailors in 1760 London

We build software by hand.

FOR US

Workflows built by hand in the designer or in JSON, one at a time.

→

NOW

The industrial loom

Production speeds up and gets cheaper, and arguably lower in quality.

FOR US

Agents generate workflows fast. Five developers, five different Logic Apps, and spider webs if nobody steers.

→

OUR ROLE

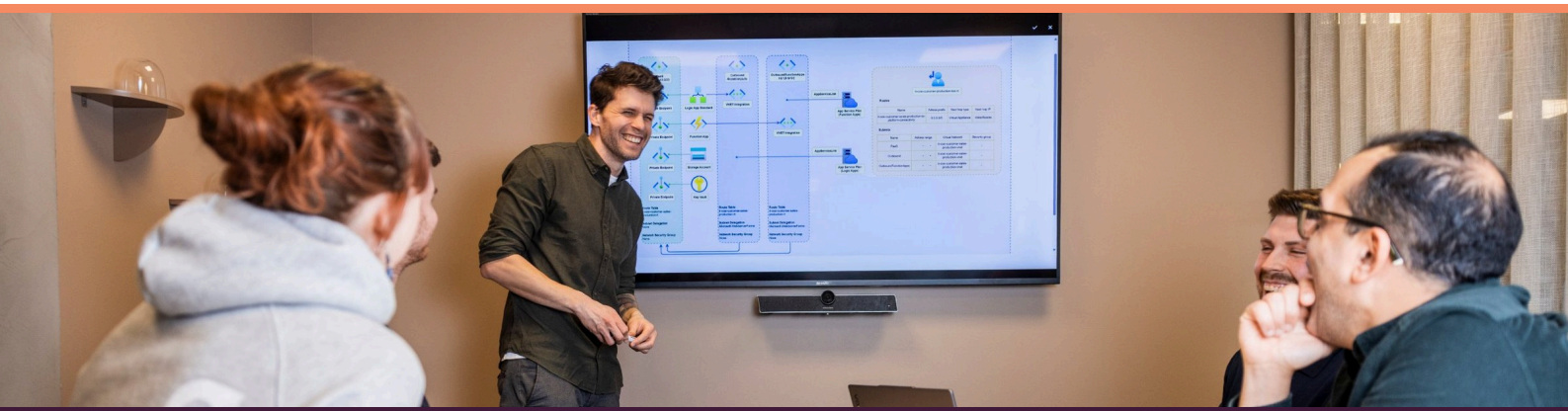
Fashion designers

Figure out trends, apply judgment and good taste to find the right solution.

FOR US

The developer owns the quality and the architect owns the context. The loom does the weaving.

Analogy: Nir Mashkowski, "The Next Era of Integration: Powering AI, Agents, and Agentic Automation", Nordic Integration Summit 2026, slides 4 to 6 and speaker notes. "For us" is Contica's reading of how it applies to the answers in the webinar.



1 FROM THE Q&A

My AI skill already works. What does the CLI add?

THE QUESTION

Webinar 00:27:53

We already use an AI skill with the VS Code extension to create workspaces, workflows and connections, and it works first time. What extra benefit does the Logic Apps CLI bring if the output is the same?

Scaffolding is only one step. **The CLI covers discovery, validation and testing as well**, and it does so without loading all of that knowledge into the agent's context.

A SKILL ON ITS OWN

Knowledge lives in the context

- Creates the workspace, workflow and connections
- Every rule and connector detail goes into the prompt, so the context grows
- Small Python, Node.js or PowerShell scripts fill the gaps

SKILL PLUS THE CLI

Deterministic steps on call

- Finds the right connector for the business process without hardcoding it
- Validates a workspace or a single workflow
- Starts and stops the local runtime and runs tests
- Scripts become one line commands over time

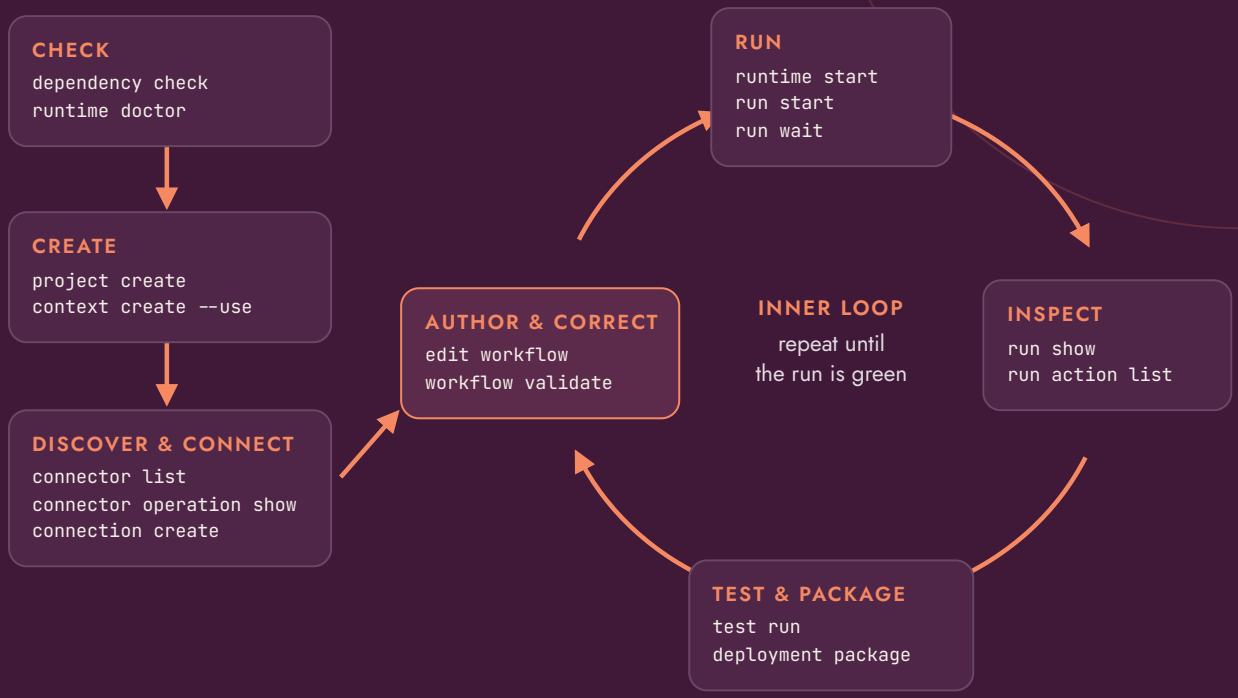
> "We're not thinking about just the scaffolding. We're thinking about the whole development life cycle."



Wagner Silveira, Microsoft

THE WHOLE LIFE CYCLE

One command surface, from first check to green run



Adapted from Wagner Silveira, "Evolving the Logic Apps Developer Experience", Nordic Integration Summit 2026, slide 15 ("The CLI closes the inner development loop"). Command names updated to the public preview, per Microsoft's article of 28 October 2026. Shown without the az logicapp prefix.

WAGNER'S ADVICE

1. **Keep what works.** If you already have a process that works, continue using it and augment it with the CLI.
2. **Start where your skills stop.** Look at what your skills do not do yet, and bring in the CLI there.

> "It's not going to solve everything. It's a piece I put in the middle that makes the quality a little bit higher."
>
>  **Ahmed Bayoumy**, Contica

TAKE WITH YOU

Another tool in your toolbox. No process yet? Start with the CLI and the predefined skill for it on Awesome Copilot.

2 FROM THE Q&A

Codeful C# or the JSON workflow definition?

THE QUESTION

Webinar 00:32:01

When the CLI creates a workflow or adds steps, does it use the new codeful C# authoring or the JSON workflow definition?

Today the CLI targets **JSON workflows**, because that is what is mature and generally available. But the CLI does not write your workflow. The model does.

Model
writes workflow.json or the .cs file

→

CLI
looks up definitions, connections and how to run

→

Same for both
codeful and codeless projects

When the CLI creates a project, the template is a workflow.json today. With the next codeful version, you will say which type the project is, and the CLI will create either a C# file or a workflow.json based on that context.

The lookups are the same either way: what the definition of a connection is, how to create it, how to run it. Microsoft wanted those things to work alike, so most of what the CLI offers applies whether the project is codeful or codeless.

> **"I use the CLI as a supervisor. It ensures that what I want to do lands in the right context, in the right templates and in the right order."**



Ahmed Bayoumy, Contica

TAKE WITH YOU

Pick the commands you need most. The CLI is a list of tools: validation of workflows and workspaces for some, discovery or automated runs and results for others.

3 FROM THE Q&A

Managed identity, and what to know before adopting

TWO QUESTIONS, ANSWERED TOGETHER — Webinar 00:39:00

Does it support managed identity for all applicable connectors, or is there any limitation?
What limitations should developers be aware of before adopting the CLI?

Managed identity works locally. The local runtime uses the default credentials pattern, so when you test, it runs as you. You need permissions on the resources you connect to, and you create the connection once and reuse connections.json.

Microsoft's article of 28 October 2026 lists the requirements for local managed identity: the Standard VS Code extension 5.961.19 or later and workflow bundle 1.165.52 or later. It also notes that commands and outputs can change during the preview.

WAGNER · STILL IN PROGRESS

- ! A bug in **operations list** and **operation show** is being fixed
- ! **Testing** is still limited in the preview and is being improved
- \+ More tools are coming, for example a **Bicep template** for a managed connector
- \+ Open source and an extension, so changes ship faster than in CLI core

AHMED · BE AWARE

1. It validates the **structure**, not the correctness of the workflow
2. It **packages a zip** but does not deploy. Use your own pipelines and deployment templates
3. It is for **Logic Apps Standard**, not Consumption

TWO CLIS, TWO HOSTING MODELS

az logic	az logicapp	Deploy
Consumption	Standard · this extension adds the local parts	already part of Azure CLI core

Wagner also mentioned that an MVP's pull request for an azd Logic App extension has been merged, and that Microsoft is working with the azd team so it can be installed as an extension.

4 FROM THE Q&A

One install, one command?

ASKED TWICE, IN TWO VERSIONS — Webinar 00:46:38

Will the CLI eventually include the runtime to run Logic Apps locally, so you install one thing and run one command instead of setting up several extensions?

Not in the foreseeable future. Logic Apps runs as a layer on top of the Azure Functions runtime. Shipping the engine inside the CLI would make the extension large and move that dependency into it.

What the CLI does instead

TODAY

Check and install

`dependency check · dependency install`

The CLI checks if your machine is ready and installs what is missing. Wagner called it the closest thing to installing one thing and getting everything in place.

BEING EXPLORED · NO TIMELINE

Validate without a running runtime

Microsoft is looking at how much of what the designer does today really needs the runtime. The idea is to do everything up to validation with the CLI alone, and use the runtime only to run the Logic App.

Wagner was clear that the second idea is at the ideation phase, shared for the sake of transparency. According to Microsoft's article, dependency install does not install the .NET SDK.

TAKE WITH YOU

Let the CLI prepare your machine. Use dependency check before you start, in your own setup and in a pipeline.

5 FROM THE Q&A

The CLI for tests, your skills for code review?

THE QUESTION

Webinar 00:49:26

Is the CLI meant for creating, validating and running automated tests for Logic Apps, while other skills still handle things like code review?

Yes. The CLI does the deterministic steps: discover, validate and run. How a workflow is built and reviewed is a matter of taste and team patterns, and belongs in separate skills.

DETERMINISTIC · THE CLI

The same every time

- Discover
- Validate
- Run and test

+

TASTE · YOUR SKILLS

How your team builds

- Where the scope and the response go
- Exception handling with logging
- Parameters for what should be configurable

The tech lead cycle, with a new kind of spike

Discover talk to the customer → **Plan** → **Spike** now an MD file, a skill → **Expand** agents build the same way

> "It is still the same tech lead cycle. The difference is that the spike is now an MD file, instead of a Logic App you copy and paste."



Wagner Silveira, Microsoft

Ahmed added that every team and every customer has its own standards. Microsoft provides the components, and the architect's job is to use them in the right way for the scenario and the team.



HOW WE WORK WITH IT AT CONTICA

The CLI as a supervisor in the middle of our toolbox

Ahmed Bayoumy, CTO at Contica and Microsoft MVP, had used the CLI heavily for about a month before the webinar. This is how he described putting it to work.

WHAT WE ALREADY HAD	Our own templates library, developed by colleagues over years, our own test tooling, and skills, rules and guardrails for working with AI.
WHAT THE CLI ADDED	A supervisor that checks that what our tools and skills produce lands in the right context, in the right templates and in the right order.
HOW IT FITS	Validation is the first gate. Unit tests plug in after it, in C#, in the same session and without VS Code. Deployment still runs through our own pipelines and templates.

What changed for us

Before, Ahmed was working out how to write down a structure for the whole team and keep maintaining it, so that every project would follow it. Now the CLI keeps the standard structure, and the product team maintains it. That leaves the team to think about the logic and how the workflow should look.

> **"It removed a fear I had that every project would end up like a jungle, with five or six different structures. This helped us standardize things, and now we have less to worry about."**
>
>  **Ahmed Bayoumy**, Contica

NEXT STEPS

Get started, and be brutally honest

Wagner's one call to action: install the CLI, try it and give feedback. Microsoft can only improve it if you tell them what you love and what you hate.

1. **Install the extension**

   ```
   az extension add --name logicapp --allow-preview true
   ```

2. **Check your machine**

   Run dependency check and runtime doctor, and install missing tools with dependency install.

3. **Add the skill to GitHub Copilot**

   The skill azure-logic-apps-standard-cli is on Awesome Copilot. With GitHub CLI 2.90.0 or later:

   ```
   gh skill install github/awesome-copilot azure-logic-apps-standard-cli --agent github-copilot --scope user
   ```

4. **Start where your skills stop**

   Use the CLI for discovery, validation and tests, and keep the process that already works.

5. **Give feedback**

   What is missing and what breaks decides how fast the CLI matures.

LINKS

Wagner's article	**Introducing the Logic Apps local CLI: develop Standard workflows from your terminal** ↗. Microsoft Tech Community, 28 October 2026
Our first test	**What we saw when we tested the CLI** ↗ (in Swedish)
The recording	**Watch the webinar on contica.se** ↗
Community	**Logic Apps Aviators** ↗



ABOUT CONTICA

Integration is not part of what we do. It is all we do.

Contica has built Microsoft integrations from Gothenburg since 2010. +30 specialists, on BizTalk and on Azure. An agentic approach is part of how we work, first in our work for customers and in our own processes too.



Questions about the Logic Apps CLI?
Ahmed Bayoumy, CTO · ahmed.bayoumy@contica.se



Questions about the webinar or this white paper?
Robin Wilde, Business Developer Integration · robin.wilde@contica.se

contica

contica.se